IRISA

INSTITUT DE RECHERCHE
EN INFORMATIQUE
ET SYSTÈMES ALÉATOIRES

Campus Universitaire de Beaulieu
35042 RENNES CEDEX FRANCE
Tél.: 02 99 84 71 00 - Télex : UNIRISA 950 473 F
Télécopie : 02 99 84 71 71

Test Specification

RIPng Conformance Test Suite

V 1.2.3

Table Of Contents

1 Introduction

The RIPng Mechanism is described in RFC 2080 **[RFC2080]** (RIPng for IPv6, January 1997).

Even though test suite was verified, there can still be a few mistakes. All suggestions are welcome and can be send to:

- Frédéric Roudaut (Frederic.Roudaut@irisa.fr)
- Anthony Baire (Anthony.baire@irisa.fr)
- Annie Floch (Annie.floch@irisa.fr)
- César Viho (Cesar.Viho@irisa.fr)

2 Test Case Description

Each test case of this RIPng Conformance Test suite is described using the following sections:

Purpose:	The goal of this section is to describe briefly in a general way the main aim of the test case.
References:	This section is used to give references in the standards concerning the test purpose.
Resource Requirement:	It describes test equipments needed to perform the test case. It can be hardware or software need.
Test Requirement:	It is used to describe specific configurations for the node to test or for the tester in order to perform the test case. This section is in fact the initialization part of the test case.
Discussion:	This section gives more specificaly, the aim of the test according our current test architecture. It can also describe what the awaited problems are …
Procedure:	The test instructions to carry out the test are described step-by-step in this part.
Observable Results:	This section emphasizes the Observable Results to check by the tester to verify that the NUT is operating as specified in the standards.
Test Sequence:	The test sequence describes packet exchanges at the IP Layer level between entities available in the test case. Packet exchange labeled by a number references the corresponding test instruction of the “Procedure” section. When the Packet Exchange is not labeled, it means that it is an observable result.
Postamble:	This section gives the current state of the Node Under Test after the test execution. This part is used to have the Node Under Test in its initial state after a run of the test case.
Possible Problem:	This section gives a description of possible problems that an execution of the test case could encounter. It discusses know issues which could lead to a “FAIL” verdict.

3 Terminology

Acronyms:

The following acronyms will be used in this document:

TRi: Test Router I (as emulated by the TN)

TN: Test Node

NUT: Node Under Test.

RTE: Route Table Entry

RUT： Router Under Test (here: NUT = RUT)

4 Topology and Test Configuration

Test Architecture:

The RUT and the Tester are connected with 2 links (eg.: Link1 and Link2) in which no other communications are allowed. The Tester sends and receive IPv6 packets on the links, and provides a tool for packet analysis (sequence and packet fields).

The logical environment of the RUT (topology simulated by the Tester) is shown on Figure 1.

Each Link is an RJ45 crossed cable. This topology is the commun topology of all test cases of this RIPng test suite.

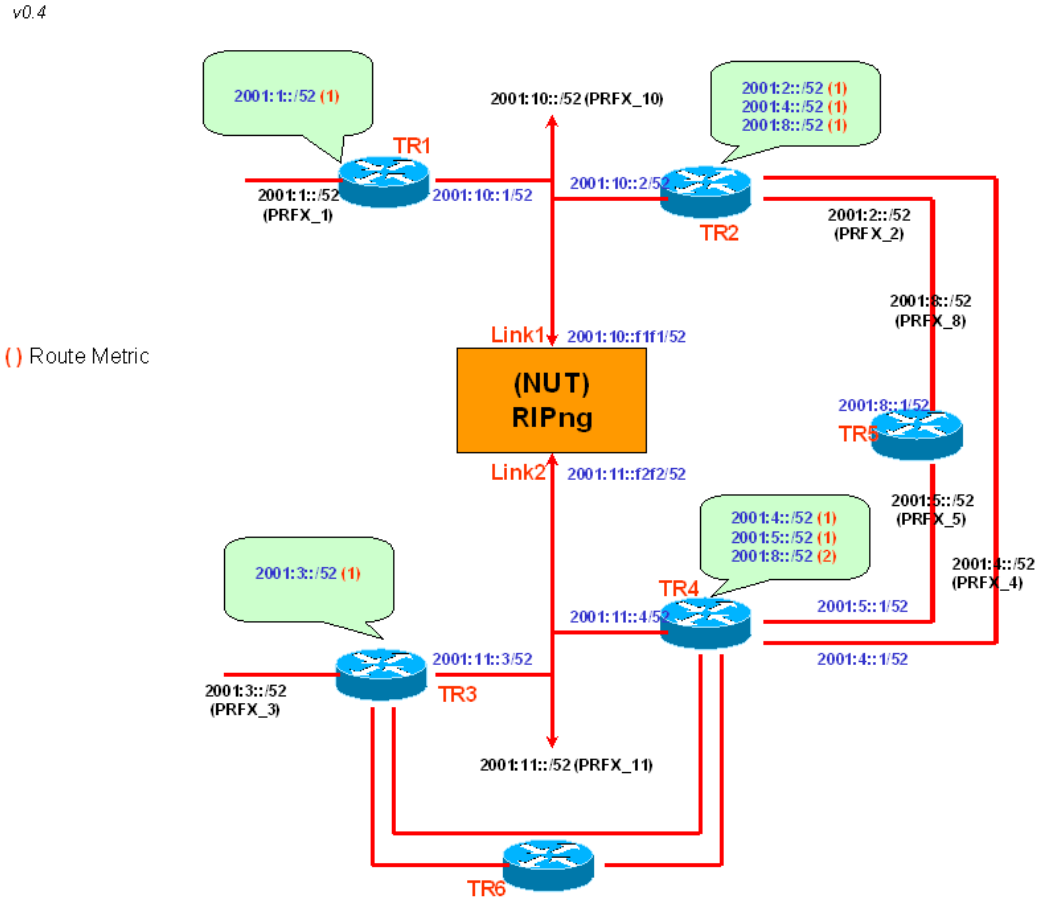


Figure 1: Logical Test Environment for RIPng Conformance Testing

Addressing:

The following defined values are just an exemple of needed values for this test suite:

- **TR1:**

  MAC address: `00:00:00:00:a1:a1`

  Link local address: `fe80::200:ff:fe00:a1a1`

  Global Address: `2001:10::1/52`

  Link local address on Internal Network: `fe80::200:ff:fe01:a1a1`

  Multicast Address Solicited Node of TR1 on Internal Network: `ff02::1:ff00:0001:a1a1`

- **TR2:**

  MAC address: `00:00:00:00:a2:a2`

  Link local address: `fe80::200:ff:fe00:a2a2`

  Global Address: `2001:10::2/52`

- **TR3:**

  MAC address: `00:00:00:00:a3:a3`

  Link local address: `fe80::200:ff:fe00:a3a3`

  Global Address: `2001:11::3/52`

- **TR4:**

  MAC address: `00:00:00:00:a4:a4`

  Link local address: `fe80::200:ff:fe00:a4a4`

  Global Address: `2001:11::4/52`

In the following we will use "PRFX_**i**" for the prefix `2001:i::/52` and "PRFX_6Bone" means `3ffe::/16`, "PRFX_6to4" means `2002::/16`. Moreover, in the following we will use the term Global Address to describe the global address of a TR on main link (ie Link1 or Link2). Futhermore, MAC address will be Mac Address on main Link if not explicitely precised.

Two IPv6 Global Addresses have to be configured on the RUT:

Global Address Link1: `2001:10::`f1f1`/52` (PRFX_10::f1f1)

Global Address Link2: `2001:11::`f2f2`/52` (PRFX_11::f2f2)

Link-locals addresses of the RUT will be autoconfigurated. When needed a second IPv6 Link-local address will be to add to the RUT on Link1 and eventually Link2. These addresses will be `fe80::`f1f1 and `fe80::`f2f2.

On RUT, MTUs will be set to 1500 when not explicitaly precised. We will consider that medias are Ethernet Links. As defined in **[RFC2464]**, The default MTU size for IPv6 packets on an Ethernet is 1500 octets.

Moreover for best tests analyze, it could be required to desactivate Router Advertisement from the RUT.

Execution of test cases:

Each test case described here should begin when the RUT is at its initial state. We tried to limit the test link effect so that the test cases can be executed in sequence. However, some tests will have more reliable verdicts in a lone run. As no procedure ensures perfect clean-up when a RUT is found to be non conformant, a reinitialization of the RUT should follow every test case that leads to a verdict « FAIL ».

Link-local addresses resolution:

As part of the Neighbor Discovery protocol **[RFC2461]**, the RUT should send Neighbor Solicitation packets to discover or probe the link-local address of a node at any time. Thus, in most cases, the Tester node should respond by a Neighbor Advertisement Packet.

Packets consideration:

- Although the Ripng Metric is not fixed by the protocol, in the following we will consider a value of 1.
- When not explicitely precised, the Route Tag field in Response packet will be set to 0.

5 Test suite for RIPng Routers

5.1 RTEs Creation

5.1.1 Basic RTE Creation

5.1.1.1 Reachable/Unreachable Basic Routes

Purpose: Check the correct creation of a RTE in the RUT

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

The goal of this test is to check the correct creation of RTEs in the RUT. If Entries Metrics are set to 15 or 16, these entries MUST not be taken into account.

Packet:

- *RTEs sent by TR1 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_1	1	0
PRFX_15	15	0
PRFX_16	16	0

Procedure:

1. **TR1** sends a **RIPng Response** to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. 521).
   The source address is the Link-local address of TR1 and the destination address is the RIPng Multicast group FF02::9.
   The Hop limit has to be set to 255.
2. **TR2** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP port source is set to `777` and destination is the RIPng port (ie. `521`).
   The source address is the global address of TR2 and the destination address is the global address of the RUT on Link1.
   The request is done for PRFX_1, PRFX_10, PRFX_11, PRFX_15, PRFX_16.
3. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP port source is set to `777` and destination is the RIPng port (ie. `521`).
   The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2.
   The request is done for PRFX_1, PRFX_10, PRFX_11, PRFX_15, PRFX_16.

Observable Results:

- **Step 1:** Regular Routing Updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent. Route Entries with metric 15 or 16 have not to be taken into account by the RUT.
- **Step 2:** A Response has to be generated by the RUT on link1.
  The Response source address is the RUT Global address of Link 1, the destination address is TR2 global address, the source port is the RIPng port source, the destination port is `777`.
  The Response should contain the following fields:

IPv6 Prefix	Metric	Route Tag
PRFX_1	2	0
PRFX_10	1	0
PRFX_11	1	0
PRFX_15	16	0
PRFX_16	16	0

- **Step 3:** A Response has to be generated by the RUT on link2.
  The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address, the source port is the RIPng port source and the destination port is `777`.
  The Response should contain the following fields:

IPv6 Prefix	Metric	Route Tag
PRFX_1	2	0
PRFX_10	1	0
PRFX_11	1	0
PRFX_15	16	0
PRFX_16	16	0

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
	RIPng Response			
TR1	---------->	1		
	RIPng Request			
TR2	(PRFX_i; i=1,10,11,15,16)			
	---------->	2		
	RIPng Response			
TR2	<----------	step2		
			RIPng Request	
			(PRFX_i; i=1,10,11,15,16)	TR3
		3	<----------	
			RIPng Response	
		Step3	---------->	TR3

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT: the Tester sends a **RIPng Response** from **TR1** in which all Metrics are set to infinity (`16`).

5.1.1.2 Creation of as much RTE as possible to fill a Response Packet

Purpose: Check the correct creation of RTEs in the RUT. The number of RTEs to create is choosen in order to fill the corresponding Response packet.

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

The goal of this test is to check the correct creation of RTEs in the RUT when the corresponding Response packet size is equal to the MTU. Indeed the maximum datagram size is limited by the MTU of the medium over which the protocol is being used. Since an unsolicited RIPng update is never propagated across a router, there is no danger of an MTU mismatch. The determination of the number of RTEs which may be put into a given message is a function of the medium's MTU, the number of octets of header information preceeding the RIPng message, the size of the RIPng header, and the size of an RTE. The formula is:

$$\#RTEs = INT\left[\frac{MTU - sizeof(IPv6\_hrds) - UDP\_hdrlen - RIPng\_hdrlen}{RTE\_size}\right]$$

As defined in **[RFC2464]**, the default MTU size for IPv6 packets on an Ethernet is 1500 octets. It means that:

#RTE = 72

Packets:

- *RTEs sent by TR1 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
2001:0:i:/52	5	0

$1 \le i \le 72 \quad (0x48)$

Procedure:

1. **TR1** sends a **RIPng Response** to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. 521).
   The source address is the Link-local address of TR1 and the destination address is the RIPng Multicast group FF02::9.
   The Hop limit has to be set to 255.
2. **TR3** sends a **RIPng Request** to get the whole routing table of the RUT.
   The UDP ports source and destination are the RIPng port (ie. 521).
   The source address is the Link-local address of TR3 and the destination address is the RIPng Multicast group FF02::9.
   There is exactly one entry in the request with a destination prefix of zero, a prefix length of zero and a metric of 16 (Infinity).

Observable Results:

- **Step 1:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.

- **Step 2:** A Response has to be generated by the RUT on link2 giving its whole routing table. The packet is the same as Regular routing updates generated every 30s. The Response should be divided into 2 packets. The Response should contain:

IPv6 Prefix	Metric
PRFX10	1
All prefixes defined in the previous Response	6

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
	RIPng Response			
TR1	---------->	1		
			RIPng Request	
		2	<----------	TR3
			RIPng Response	
		Step2	---------->	TR3

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT by sending a **Ripng Response** packet from **TR1** in which all Metrics are set to infinity(16).

5.1.1.3 Route Tag handling

Purpose: Check the correct handling of the Route Tag Field

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

The goal of this test is to check the correct handling of the Route Tag Field. The Route Tag field is an attribute assigned to a route which must be preserved and readvertised.

Packet:

- *RTEs sent by TR1 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_6Bone	5	1
PRFX_6to4	6	65535
PRFX_1	1	0

Procedure:

1. **TR1** sends a **RIPng Response** to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. 521).
   The source address is the Link-local address of TR1 and the destination address is the RIPng Multicast group FF02::9.
   The Hop Limit has to be set to 255.
   The Route Tag field of PRFX_6Bone is `1` and the Route Tag Field of PRFX_3 is `65535`
2. **TR2** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to `777` and destination is the RIPng port (ie. `521`).
   The source address is the global address of TR2 and the destination address is the global address of the RUT on Link1.
   The request is done for PRFX_6Bone, PRFX_6to4, PRFX_1, PRFX_10 and PRFX_11.
3. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to `777` and destination is the RIPng port (ie. `521`).
   The source address is the global address of TR3 and the destination address is the global address 1 of the RUT on Link2.
   The request is done for PRFX_6Bone, PRFX_6to4, PRFX_1, PRFX_10 and PRFX_11.

Observable Results:

- **Step 1:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent. Route Entries with metric 15 or 16 have not to be taken into account by the RUT.
- **Step 2:** A Response has to be generated by the RUT on link1.
  The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address, the source port is the RIPng port source, the destination port is `777`.
  The Response should contain the following fields:

IPv6 Prefix	Metric	Route Tag
PRFX_6Bone	6	1
PRFX_6to4	7	65535
PRFX_1	2	0
PRFX_10	1	0
PRFX_11	1	0

- **Step 3:** A Response has to be generated by the RUT on link2.
  The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address, the source port is the RIPng port source and the destination port is `777`.
  The Response should the following fields:

IPv6 Prefix	Metric	Route Tag
PRFX_6Bone	6	1
PRFX_6to4	7	65535
PRFX_1	2	0
PRFX_10	1	0
PRFX_11	1	0

Test Sequence:

Tester (Tri)	Link1	RUT	Link2	Tester (TRi)
TR1	RIPng Response ---------->	1		
TR2	RIPng Request (PRFX_i with i=6Bone, 6to4, 1, 10, 11) ---------->	2		
TR2	RIPng Response <----------	step2		
		3	RIPng Request (PRFX_i with i=6Bone, 6to4, 1, 10, 11) <----------	TR3
		Step3	RIPng Response ---------->	TR3

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT: the Tester sends a **Ripng Response** from **TR1** in which all Metrics are set to infinity (16).

5.1.1.4 Default Route & Next Hop Setting

Purpose: The goal of this test is to check that entries are created in the routing table of the RUT with a correct Next Hop field.

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

The goal of this test is to check that entries are created in the routing table of the RUT with a correct Next Hop field. Moreover it checks the correct establishment of the default route on the RUT. Any prefix with a prefix length of zero is used to designate a default route. It is suggested that the prefix 0:0:0:0:0:0:0:0 be used when specifying the default route, though the prefix is essentially ignored.

Packet:

- *RTEs sent by TR1 in RIPng Response are the following :*

IPv6 Prefix	Metric	Route Tag
2001:11::f2f2/0	1	0
PRFX_1	1	0

Procedure:

1. **TR1** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521).
   The source address is the Link-local address of TR1 and the destination address is the RIPng Multicast group FF02::9.
   The Hop limit has to be set to 255. This response contains the route 2001:11::f2f2/0 which is the global address of the RUT on Link2 with a prefix length of 0. It must be considered as the default route since the prefix length is 0.
2. **TR2** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521).
   The source address is the global address of TR2 and the destination address is the global address of the RUT on Link1.
   The Hop limit has to be set to 255.
   The request is done for PRFX_1, PRFX_10, PRFX_11, the Default Route (0:0:0:0:0:0:0:0 with a prefix length of 0) and another time the Default Route 2001:10::f1f1/0 (global address of the RUT on Link1 with a prefix length of 0).
3. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521).
   The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2.
   The request is done for PRFX_1, PRFX_10, PRFX_11, the Default Route (0:0:0:0:0:0:0:0 with a prefix length of 0) and another time the Default Route 2001:10::f1f1/0 (global address of the RUT on Link1 with a prefix length of 0).
4. **TR2** sends an **Echo Request** through the RUT to:

   (a) PRFX_1::1

(b) An Unreachable host (PRFX_FFFF::1)

The source Address is TR2 Global Address.

5. **TR3** sends an **Echo Request** through the RUT to:

(a) PRFX_1::1

(b) An Unreachable host (PRFX_FFFF::1)

The source Address is TR3 Global Address.

Observable Results:

- **Step 1:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent. Route Entries with metric 15 or 16 have not to be taken into account by the RUT.
- **Step 2:** A Response has to be generated by the RUT on link1.
  The Response source address is the RUT Global address of Link 1, the destination address is TR2 global address, the source port is the RIPng port source, the destination port is 777.
  The Response should contain the following fields:

IPv6 Prefix	Metric	Route Tag
PRFX_1	2	0
PRFX_10	1	0
PRFX_11	1	0
0:0:0:0:0:0:0:0/0	2	0
2001:10::f1f1/0 (global address of the RUT on Link1)	2	0

- **Step 3:** A Response has to be generated by the RUT on link2.
  The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address, the source port is the RIPng port source and the destination port is 777.
  The Response should contain the following fields:

IPv6 Prefix	Metric	Route Tag
PRFX_1	2	0
PRFX_10	1	0
PRFX_11	1	0
0:0:0:0:0:0:0:0/0	2	0
2001:10::f1f1/0 (global address of the RUT on Link1)	2	0

- **Step 4:** The **two Echo Requests** for PRFX_1::1 and PRFX_FFFF::1 MUST be forwarded to **TR1**.
- **Step 5:** The **two Echo Requests** for PRFX_1::1 and PRFX_FFFF::1 MUST be forwarded to **TR1**.

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
TR1	RIPng Response ---------->	1		
TR2	RIPng Request (PRFX_i with i=1,10,11, the Default Route (0:0:0:0:0:0:0:0/0) and the Default Route 2001:10::f1f1/0 ---------->	2		
TR2	<---------- RIPng Response	step2		
		3	<---------- RIPng Request (PRFX_i with i=1,10,11, the Default Route (0:0:0:0:0:0:0:0/0) and the Default Route 2001:10::f1f1/0	TR3
		Step3	----------> RIPng Response	TR3
TR2	Echo Request (PRFX_1::1) ---------->	4(a)		
TR1	Echo Request (PRFX_1::1) <----------	Step4(a)		
TR2	Echo Request (PRFX_FFFF::1) ---------->	4(b)		
TR1	Echo Request (PRFX_FFFF::1) <----------	Step4(b)		
		5(a)	Echo Request (PRFX_1::1) <----------	TR3
TR1	Echo Request (PRFX_1::1) <----------	Step5(a)		
		5(b)	Echo Request (PRFX_FFFF::1) <----------	TR3
TR1	Echo Request (PRFX_FFFF::1) <----------	Step5(b)		

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT: the Tester sends a **Response packet** from **TR1** in which all Metrics are set to infinity (16).

5.1.2 Next Hop RTE

5.1.2.1 Next Hop Route Table Entry

Purpose: Check the correct Handling of the Next Hop RTE. Check also that Route Tags are correctly handled even when a NEXT HOP RTE is present.

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

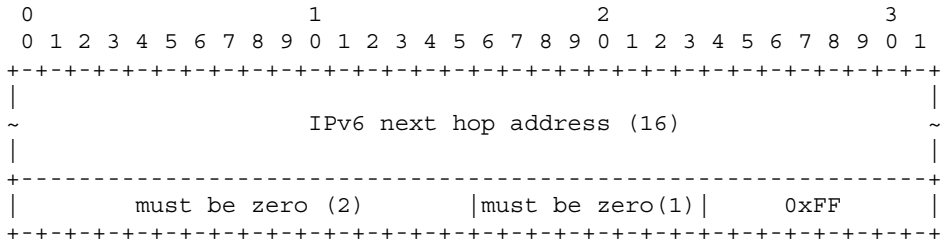
- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

RIPng provides the ability to specify the immediate next hop IPv6 address to which packets to a destination specified by a route table entry (RTE) should be forwarded in much the same way as RIP-2. In RIP-2, each route table entry has a next hop field. Including a next hop field for each RTE in RIPng would nearly double the size of the RTE. Therefore, in RIPng, the next hop is specified by a special RTE and applies to all of the address RTEs following the next hop RTE until the end of the message or until another next hop RTE is encountered.

A next hop RTE is identified by a value of 0xFF in the metric field of an RTE. The prefix field specifies the Ipv6 address of the next hop. The route tag and prefix length in the next hop RTE must be set to zero on sending and ignored on reception.

The next hop Route Table Entry (RTE) has the following format:



Packet:

- *RTEs sent by TR3 in Ripng Response are the following in the following order: :*

Next Hop	TR4 Link-local Address	
Ipv6 Prefix	*Metric*	*Route Tag*
PRFX_4	1	0x4
PRFX_5	1	0x5
Next Hop	TR4 Global Address	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_3	3	0x3
Next Hop	RUT Link-local address Link1	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_11	1	0
Next Hop	TR4 Link-local Address	
IPv6 Prefix	*Metric*	*Route Tag*

PRFX_8	2	0x8
Next Hop	0:0:0:0:0:0:0:0	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_6Bone	7	0xFFFF

Procedure:

1. **TR3** sends a **RIPng Response** to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. 521).
   The source address is the Link-local address of TR3 and the destination address is the RIPng Multicast group FF02::9.
   The Hop limit has to be set to 255.
2. **TR1** sends an **Echo Request** through the RUT to:
   (a) PRFX_3::1
   (b) PRFX_4::1
   (c) PRFX_5::1
   (d) PRFX_8::1
   (e) PRFX_11::f2f2
   (f) PRFX_6Bone::1

   The Source Address is TR1 Global Address.
3. **TR3** sends an **Echo Request** through the RUT to:
   (a) PRFX_3::1
   (b) PRFX_4::1
   (c) PRFX_5::1
   (d) PRFX_8::1
   (e) PRFX_11:: f2f2
   (f) PRFX_6Bone::1

   The Source Address is TR3 Global Address.
4. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to `777` and destination is the RIPng port (ie. `521`).
   The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1.
   The Hop limit has to be set to `255`
   The request is done for PRFX_6Bone, PRFX_3, PRFX_4, PRFX_5 and PRFX_8.
5. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to `777` and destination is the RIPng port (ie. `521`).
   The source address is the global address of TR3 and the destination address is the global address 1 of the RUT on Link2.
   The Hop limit has to be set to `255`.
   The request is done for PRFX_6Bone, PRFX_3, PRFX_4, PRFX_5 and PRFX_8.

Observable Results:

- **Step 1:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.
- **Step 2:** The three Echo Requests for PRFX_4::1 (2.b), PRFX_5::1 (2.c) and PRFX_8::1(2.d) MUST be forwarded to TR4
  Echo Request for PRFX_3::1 (2.a), PRFX_6Bone::1 (2.f) MUST be forwarded to TR3.
  An Echo Reply From the RUT Global Address Link 2 has to be generated by the RUT and send to TR1 (2.e)
- **Step 3:** The three Echo Requests for PRFX_4::1 (3.b), PRFX_5::1 (3.c) and PRFX_8::1(3.d) MUST be forwarded to TR4
  Echo Request for PRFX_3::1 (3.a), PRFX_6Bone::1 (3.f) MUST be forwarded to TR3.
  An Echo Reply From the RUT Global Address Link 2 has to be generated by the RUT and send to TR3 (3.e)

- **Step 4:** A Response has to be generated by the RUT on link1.
  The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address, the source port is the RIPng port source, the destination port is 777.
  The Response should contain the following fields:

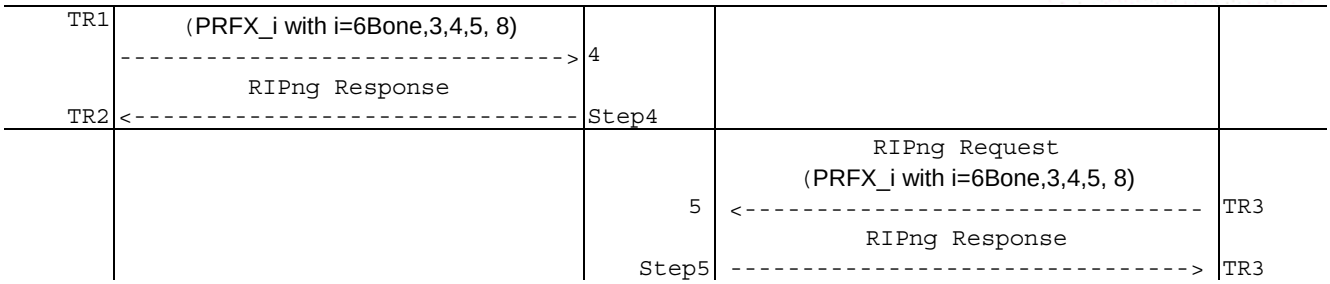
IPv6 Prefix	Metric	Route Tag
PRFX_6Bone	8	0xFFFF
PRFX_3	4	0x3
PRFX_4	2	0x4
PRFX_5	2	0x5
PRFX_8	3	0x8

- **Step 5:** A Response has to be generated by the RUT on link2.
  The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address, the source port is the RIPng port source and the destination port is 777.
  The Response should contain the following fields:

IPv6 Prefix	Metric	Route Tag
PRFX_6Bone	8	0xFFFF
PRFX_3	4	0x3
PRFX_4	2	0x4
PRFX_5	2	0x5
PRFX_8	3	0x8

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response <----------	TR3
TR1	Echo Request (PRFX_3::1) ---------->	2(a)		
		Step2(a)	Echo Request (PRFX_3::1) ---------->	TR3
TR1	Echo Request (PRFX_4::1) ---------->	2(b)		
		Step2(b)	Echo Request (PRFX_4::1) ---------->	TR4
TR1	Echo Request (PRFX_5::1) ---------->	2(c)		
		Step2(c)	Echo Request (PRFX_5::1) ---------->	TR4
TR1	Echo Request (PRFX_8::1) ---------->	2(d)		
		Step2(d)	Echo Request (PRFX_8::1) ---------->	TR4
TR1	Echo Request (PRFX_11::1) ---------->	2(e)		
TR1	Echo Reply <----------	Step2(e)		
TR1	Echo Request (PRFX_6Bone::1) ---------->	2(f)		
		Step2(f)	Echo Request (PRFX_6Bone::1) ---------->	TR3
		3(a)	Echo Request (PRFX_3::1) <----------	TR3
		Step3(a)	Echo Request (PRFX_3::1) ---------->	TR3
		3(b)	Echo Request (PRFX_4::1) <----------	TR3
		Step3(b)	Echo Request (PRFX_4::1) ---------->	TR4
		3(c)	Echo Request (PRFX_5::1) <----------	TR3
		Step3(c)	Echo Request (PRFX_5::1) ---------->	TR4
		3(d)	Echo Request (PRFX_8::1) <----------	TR3
		Step3(d)	Echo Request (PRFX_8::1) ---------->	TR4
		3(e)	Echo Request (PRFX_11::1) <----------	TR3
		Step3(e)	Echo Reply ---------->	TR3
		3(f)	Echo Request (PRFX_6Bone::1) <----------	TR3
		Step3(f)	Echo Request (PRFX_6Bone::1) ---------->	TR3
	RIPng Request			



Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT: the Tester sends a **RIPng Response packet** from **TR3** and **TR4** in which all Metrics are set to infinity (16).

5.1.2.2 Next Hop Route Table Entry with Invalid Route Tag and Prefix Len field

Purpose: Check the correct Handling of the Next Hop RTE even when Route Tag Field and Prefix Len Field are not zero.

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

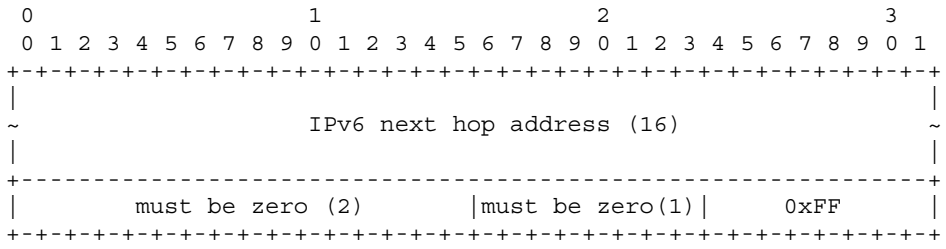
- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

RIPng provides the ability to specify the immediate next hop IPv6 address to which packets to a destination specified by a route table entry (RTE) should be forwarded in much the same way as RIP-2. In RIP-2, each route table entry has a next hop field. Including a next hop field for each RTE in RIPng would nearly double the size of the RTE. Therefore, in RIPng, the next hop is specified by a special RTE and applies to all of the address RTEs following the next hop RTE until the end of the message or until another next hop RTE is encountered.

A next hop RTE is identified by a value of 0xFF in the metric field of an RTE. The prefix field specifies the IPv6 address of the next hop. The route tag and prefix length in the next hop RTE must be set to zero on sending and ignored on receiption.

The next hop Route Table Entry (RTE) has the following format:



This test is the same that the one described in 5.1.2.1 except that Route Tag and Prefix Len field are not zero in the Next Hop RTE.

Packet:

- *RTEs sent by TR3 in Ripng Response are the following in the following order: :*

Next Hop	TR4 Link-local Address	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_4	1	0
PRFX_5	1	0
Next Hop	TR4 Global Address	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_3	3	0
Next Hop	RUT Link-local address Link1	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_11	1	0
Next Hop	TR4 Link-local Address	

IPv6 Prefix	*Metric*	*Route Tag*
PRFX_8	2	0
Next Hop	0:0:0:0:0:0:0:0	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_6Bone	7	0

Each **Route Tag** field and **Prefix Len** field of the different Next Hop RTEs are set to "**`0xFFFF`**" and "**`0xFF`**"

Procedure:

1 **TR3** sends a **RIPng Response** to give its RIPng routing table to the RUT.
The UDP ports source and destination are the RIPng port (ie. 521).
The source address is the Link-local address of TR3 and the destination address is the RIPng Multicast group FF02::9.
The Hop limit has to be set to 255.

2 **TR1** sends an **Echo Request** through the RUT to:

(a) PRFX_3::1

(b) PRFX_4::1

(c) PRFX_5::1

(d) PRFX_8::1

(e) PRFX_11::f2f2

(f) PRFX_6Bone::1

The Source Address is TR1 Global Address.

3 **TR3** sends an **Echo Request** through the RUT to:

(a) PRFX_3::1

(b) PRFX_4::1

(c) PRFX_5::1

(d) PRFX_8::1

(e) PRFX_11:: f2f2

(f) PRFX_6Bone::1

The Source Address is TR3 Global Address.

Observable Results:

- **Step 1:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.
- **Step 2:** The three Echo Requests for PRFX_4:1 (2.b), PRFX_5:1 (2.c) and PRFX_8:1(2.d) MUST be forwarded to TR4.
  Echo Request for PRFX_3:1 (2.a), PRFX_6Bone (2.f) MUST be forwarded to TR3.
  An Echo Reply From the RUT Global Address Link 2 has to be generated by the RUT and send to TR1 (2.e)
- **Step 3:** The three Echo Requests for PRFX_4:1 (3.b), PRFX_5:1 (3.c) and PRFX_8:1(3.d) MUST be forwarded to TR4.
  Echo Request for PRFX_3:1 (3.a), PRFX_6Bone (3.f) MUST be forwarded to TR3.
  An Echo Reply From the RUT Global Address Link 2 has to be generated by the RUT and send to TR3 (3.5)

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response <----------	TR3
TR1	Echo Request (PRFX_3::1) ---------->	2(a)		
		Step2(a)	Echo Request (PRFX_3::1) ---------->	TR3
TR1	Echo Request (PRFX_4::1) ---------->	2(b)		
		Step2(b)	Echo Request (PRFX_4::1) ---------->	TR4
TR1	Echo Request (PRFX_5::1) ---------->	2(c)		
		Step2(c)	Echo Request (PRFX_5::1) ---------->	TR4
TR1	Echo Request (PRFX_8::1) ---------->	2(d)		
		Step2(d)	Echo Request (PRFX_8::1) ---------->	TR4
TR1	Echo Request (PRFX_11::1) ---------->	2(e)		
TR1	Echo Reply <----------	Step2(e)		
TR1	Echo Request (PRFX_6Bone::1) ---------->	2(f)		
		Step2(f)	Echo Request (PRFX_6Bone::1) ---------->	TR3
		3(a)	Echo Request (PRFX_3::1) <----------	TR3
		Step3(a)	Echo Request (PRFX_3::1) ---------->	TR3
		3(b)	Echo Request (PRFX_4::1) <----------	TR3
		Step3(b)	Echo Request (PRFX_4::1) ---------->	TR4
		3(c)	Echo Request (PRFX_5::1) <----------	TR3
		Step3(c)	Echo Request (PRFX_5::1) ---------->	TR4
		3(d)	Echo Request (PRFX_8::1) <----------	TR3
		Step3(d)	Echo Request (PRFX_8::1) ---------->	TR4
		3(e)	Echo Request (PRFX_11::1) <----------	TR3
		Step3(e)	Echo Reply ---------->	TR3
		3(f)	Echo Request (PRFX_6Bone::1) <----------	TR3
		Step3(f)	Echo Request (PRFX_6Bone::1) ---------->	TR3

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT: the Tester sends a **Ripng Response** packet from **TR3** and **TR4** in which all Metrics are set to infinity (16).

5.1.3 Invalid RTEs

5.1.3.1 Incorrect Prefix, Prefix Length, Metric in RTEs

Purpose: Check that routes with Incorrect Prefix, Prefix Length, Metric in RTEs are silently discarded

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

The goal of this test is to check that routes with Incorrect Prefix, Prefix Length, Metric in RTEs are silently discarded. Valid entries are neverteless take into account.

Incorrect metrics and other format errors usually indicate misbehaving neighbors and should probably be brought to the administrator's attention. For example, if the metric is greater than infinity, ignore the entry but log the event.

The basic validation tests are:

- is the destination prefix valid (e.g., not a multicast prefix and not a link-local address)? A link-local address should never be present in an RTE.

- is the prefix length valid (i.e., between 0 and 128, inclusive)?

- is the metric valid (i.e., between 1 and 16, inclusive)?

If any check fails, ignore that entry and proceed to the next.

Packet:

- *RTEs sent by TR1 in RIPng Response are the following :*

IPv6 Prefix	Metric	Route Tag
ff02::1:ff00:0001:a1a1/128	1	0
fe80::200:ff:fe01:a1a1/128	1	0
ff02::1:ff00:0000:a1a1/128	1	0
fe80::200:ff:fe00:a1a1/128	1	0
PRFX_2/129	1	0
PRFX_2/255	1	0
PRFX_17	17	0
PRFX_254	254	0
PRFX_4	0	0
PRFX_1	1	0

Procedure:

1. **TR1** sends a **RIPng Response** to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. 521).
   The source address is the Link-local address of TR1 and the destination address is the RIPng Multicast group FF02::9.
   The Hop limit has to be set to 255.

   RTEs included in the RIPng response Packets are the following:

   - `ff02::1:FF00:0001:a1a1/128` (Metric:1): Multicast Address
   - `fe80::200:ff:fe01:a1a1/128` (Metric:1): Link Local address
   - `ff02::1:FF00:0000:a1a1/128` (Metric:1): Multicast Address
   - `fe80::200:ff:fe00:a1a1/128` (Metric:1): Link Local address
   - `PRFX_2::1/129` (Metric:1): Incorrect Prefix Length
   - `PRFX_2::1/255` (Metric:1): Incorrect Prefix Length
   - `PRFX_17` (Metric: 17): Incorrect Metric
   - `PRFX_254` (Metric: 254): Incorrect Metric
   - `PRFX_4` (Metric: 0): Incorrect Metric
   - `PRFX_1` (Metric: 1): Valid RTE

2. **TR2** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR2 and the destination address is the global address of the RUT on Link1. Moreover the request is done for:

   - `ff02::1:FF00:0001:a1a1/128`: Multicast Address
   - `fe80::200:ff:fe01:a1a1/128`: Link Local address
   - `ff02::1:FF00:0000:a1a1/128` (Metric:1): Multicast Address
   - `fe80::200:ff:fe00:a1a1/128` (Metric:1): Link Local address
   - `PRFX_2::1/129`: Incorrect Prefix Length
   - `PRFX_2::1/255`: Incorrect Prefix Length
   - `PRFX_17`
   - `PRFX_254`
   - `PRFX_4`
   - `PRFX_1`: Valid RTE

3. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for:

   - `ff02::1:FF00:0001:a1a1/128`: Multicast Address
   - `fe80::200:ff:fe01:a1a1/128`: Link Local address
   - `ff02::1:FF00:0000:a1a1/128` (Metric:1): Multicast Address
   - `fe80::200:ff:fe00:a1a1/128` (Metric:1): Link Local address
   - `PRFX_2::1/129`: Incorrect Prefix Length
   - `PRFX_2::1/255`: Incorrect Prefix Length
   - `PRFX_17`
   - `PRFX_254`
   - `PRFX_4`
   - `PRFX_1`: Valid RTE

Observable Results:

- **Step 1:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.

- **Step 2:** A Response has to be generated by the RUT on link1.
  The Response source address is the RUT Global address of Link 1, the destination address is TR2 global address, the source port is the RIPng port source, the destination port is 777.
  The Response should contain the following fields:

IPv6 Prefix	Metric	Route Tag
ff02::1:ff00:0001:a1a1/128	16	0
fe80::200:ff:fe01:a1a1/128	16	0
ff02::1:ff00:0000:a1a1/128	16	0
fe80::200:ff:fe00:a1a1/128	16	0
PRFX_2/129	16	0
PRFX_2/255	16	0
PRFX_17	16	0
PRFX_254	16	0
PRFX_4	16	0
PRFX_1	2	0

- **Step 3:** A Response has to be generated by the RUT on link2.
  The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address, the source port is the RIPng port source, the destination port is 777.
  The Response should contain the following fields:

IPv6 Prefix	Metric	Route Tag
ff02::1:ff00:0001:a1a1/128	16	0
fe80::200:ff:fe01:a1a1/128	16	0
ff02::1:ff00:0000:a1a1/128	16	0
fe80::200:ff:fe00:a1a1/128	16	0
PRFX_2/129	16	0
PRFX_2/255	16	0
PRFX_17	16	0
PRFX_254	16	0
PRFX_4	16	0
PRFX_1	2	0

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
TR1	RIPng Response ---------->	1		
TR2	RIPng Request - ff02::1:ff00:0001:a1a1/128 - fe80::200:ff:fe01:a1a1/128 - ff02::1:ff00:0000:a1a1/128 - fe80::200:ff:fe00:a1a1/128 - PRFX_2/129 - PRFX_2/255 - PRFX_17 - PRFX_254 - PRFX_4 - PRFX_1 ---------->	2		
TR2	<---------- RIPng Response	step2		
			RIPng Request - ff02::1:ff00:0001:a1a1/128 - fe80::200:ff:fe01:a1a1/128 - ff02::1:ff00:0000:a1a1/128 - fe80::200:ff:fe00:a1a1/128 - PRFX_2/129 - PRFX_2/255 - PRFX_17 - PRFX_254 - PRFX_4 - PRFX_1 <----------	TR3
		Step3	RIPng Response ---------->	TR3

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT: the Tester sends a **Ripng Response** packet from **TR1** in which all Metrics are set to infinity (16).

5.1.3.2 Incorrect Response Packet

Purpose: Check the incorrect Response packet do not update the Routing Table on the NUT

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.
- Add a second Link-local address on NUT's Link1

Discussion:

The goal of this test is to check that incorrect Response packet do not update the Routing Table on the NUT. Because processing of a Response may update the router's routing table, the Response must be checked carefully for validity. The Response must be ignored if it is not from the RIPng port.

The datagram's IPv6 source address should be checked to see whether the datagram is from a valid neighbor; the source of the datagram must be a link-local address.

It is also worth checking to see whether the response is from one of the router's own addresses. Interfaces on broadcast networks may receive copies of their own multicasts immediately. If a router processes its own output as new input, confusion is likely, and such datagrams must be ignored.

As an additional check, periodic advertisements must have their hop counts set to 255, and inbound, multicast packets sent from the RIPng port (i.e. periodic advertisement or triggered update packets) must be examined to ensure that the hop count is 255. This absolutely guarantees that a packet is from a neighbor, because any intermediate node would have decremented the hop count. Queries and their responses may still cross intermediate nodes and therefore do not require the hop count test to be done.

Packet:

- *RTEs sent by TR1 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_1	1	0
PRFX_6Bone	5	0

Procedure:

1. Run the whole test case using each following **RIPng Response** Packet:

   (A) Incorrect Hop Limit

   TR1 sends a **RIPng Response** to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. `521`). The source address is the Link-local address of TR1 and the destination address is the RIPng Multicast group `FF02::9`.
   The Hop limit has to be set to **254**.*Incorrect UDP Source Port.*

   (B) Incorrect UDP Source Port

   TR1 sends a **RIPng Response** to give its RIPng routing table to the RUT.
   The UDP source port is **777** and the destination port is RIPng port (ie. `521`). The source address is the Link-local address of TR1 and the destination address is the RIPng Multicast group `FF02::9`.
   The Hop limit has to be set to 255.

(C) Incorrect Source Address

TR1 sends a RIPng Response to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. `521`). The destination address is the RIPng Multicast group `FF02::9`. Moreover the Hop limit has to be set to `255`. The source address is :

C.1. the **Global Address of TR1**

C.2. the **Multicast Solicited Address of TR1**

C.3. **`FF02::1` (Multicast Address All hosts on the link)**

C.4. the **Global Address of The RUT on link1**

C.5. the **Link-local Address of The RUT on link1**

C.6. the **Global Address of The RUT on link2**

2. **TR2** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR2 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_1 and PRFX_6Bone.
3. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_1 and PRFX_6Bone.
4. **TR1** sends a **RIPng Response** to the RUT to clear the route table entries for PRFX_1 and PRFX_6Bone. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the same as in Step 1 and the destination address is the RIPng Multicast group FF02::9. The Hop limit is set to **255** and the metric is set to **16** for each routing entry.

Observable Results:

- **Step 1:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. When routes are created, an update has also to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent. Because all Response Packet are invalid, the RUT has to silently discard them. No Triggered Update Should Be sent on Link1 and Link2 during the whole test.
- **Step 2:** A Response has to be generated by the RUT on Link1. The Response source address is the RUT Global address of Link1, the destination address is TR2 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	**Route Tag**
PRFX_1	16	0
PRFX_6Bone	16	0

- **Step 3:** A Response has to be generated by the RUT on Link2. The Response source address is the RUT Global address of Link3, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	**Route Tag**
PRFX_1	16	0
PRFX_6Bone	16	0

- **Step 4:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.

Test Sequence:

Tester (TRi)	**Link1**	**RUT**	**Link2**	**Tester (TRi)**
TR1	RIPng Response ---------->	A-1		
TR2	RIPng Request ---------->	A-2		
TR2	RIPng Response <----------	StepA-2		

		A-3	RIPng Request <---------- RIPng Response	TR3
		StepA-3	---------->	TR3
TR1	RIPng Response ---------->	A-4		
TR1	RIPng Response ---------->	B-1		
TR2	RIPng Request ---------->	B-2		
TR2	RIPng Response <----------	StepB-2		
		B-3	RIPng Request <---------- RIPng Response	TR3
		StepB-3	---------->	TR3
TR1	RIPng Response ---------->	B-4		
TR1	RIPng Response ---------->	C1-1		
TR1	RIPng Response ---------->	C2-1		
TR1	RIPng Response ---------->	C3-1		
TR1	RIPng Response ---------->	C4-1		
TR1	RIPng Response ---------->	C5-1		
TR1	RIPng Response ---------->	C6-1		
TR1	RIPng Response ---------->	C7-1		
TR2	RIPng Request ---------->	C-2		
TR2	RIPng Response <----------	StepC-2		
		C-3	RIPng Request <---------- RIPng Response	TR3
		StepC-3	---------->	TR3
TR1	RIPng Response ---------->	C-4		

Postamble:

Because the Routing Table of the RUT has not been modified, no route should be present in the RUT except route from RUT's Networks.

5.2 RTEs Update

5.2.1 Update of route characteristics by routers involved in the route

5.2.1.1 Metric Update

Purpose: Check the update of Metrics from the same router as specified in the Next Hop address of the route.

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

If there is an existing route, compare the next hop address to the address of the router from which the datagram came. If this datagram is from the same router as the existing route, reinitialize the timeout. Next, compare the metrics. If the datagram is from the same router as the existing route, and the new metric is different than the old one; or, if the new metric is lower than the old one; do the following actions:

- Adopt the route from the datagram. That is, put the new metric in, and adjust the next hop address (if necessary).
- Set the route change flag and signal the output process to trigger an update.
- If the new metric is infinity, start the deletion process; otherwise, re-initialize the timeout.

If the new metric is infinity, the deletion process begins for the route, which is no longer used for routing packets. Note that the deletion process is started only when the metric is first set to infinity. If the metric was already infinity, then a new deletion process is not started.

If the new metric is the same as the old one, it is simplest to do nothing further (beyond reinitializing the timeout, as specified above); but, there is a heuristic which could be applied. Normally, it is senseless to replace a route if the new route has the same metric as the existing route; this would cause the route to bounce back and forth, which would generate an intolerable number of triggered updates. However, if the existing route is showing signs of timing out, it may be better to switch to an equally-good alternative route immediately, rather than waiting for the timeout to happen. Therefore, if the new metric is the same as the old one, examine the timeout for the existing route. If it is at least halfway to the expiration point, switch to the new route. This heuristic is optional, but highly recommended.

Procedure:

1. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255. The RTEs included in the Response are the following:

IPv6 Prefix	Metric
PRFX_4	1
PRFX_5	2
PRFX_8	2

- *Part A: Lower Metric*

2. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255. The RTEs included in the Response are the following:

IPv6 Prefix	Metric	**Route Tag**
PRFX_4	1	0
PRFX_5	**1**	0
PRFX_8	2	0

3. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.

4. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.

- *Part B: Greater Metric*

5. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255. The RTEs included in the Response are the following:

IPv6 Prefix	Metric	**Route Tag**
PRFX_4	1	0
PRFX_5	**2**	0
PRFX_8	2	0

6. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.

7. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.

Observable Results:

- **Step 1, 2 & 5:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.
- **Step 3:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	**Route Tag**
PRFX_4	2	0
PRFX_5	**2**	0
PRFX_8	3	0

- **Step 4:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	**Route Tag**

PRFX_4	2	0
PRFX_5	**2**	0
PRFX_8	3	0

- **Step 6:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	**Route Tag**
PRFX_4	2	0
PRFX_5	**3**	0
PRFX_8	3	0

- **Step 7:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	**Route Tag**
PRFX_4	2	0
PRFX_5	**3**	0
PRFX_8	3	0

Test Sequence:

Tester (TRi)	**Link1**	**RUT**	**Link2**	**Tester (TRi)**
		1	RIPng Response <----------	TR4
		2	RIPng Response <----------	TR4
TR1	RIPng Request (PRFX_i with i=4,5,8) ---------->	3		
TR1	RIPng Response <----------	Step3		
		4	RIPng Request (PRFX_i with i=4,5,8) <----------	TR3
		Step4	RIPng Response ---------->	TR3
		5	RIPng Response <----------	TR4
TR1	RIPng Request ---------->	6		
TR1	RIPng Response <----------	Step6		
		7	RIPng Request <----------	TR3
		Step7	RIPng Response ---------->	TR3

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT by sending a **Ripng Response** packet from **TR4** in which all Metrics are set to infinity(16).

5.2.1.2 Next Hop Update

Purpose: Check the update of the Next Hop field from the same router as specified in the Next Hop address of the route. Check that Link-local address of the TR, global address of any node, special address "0:0:0:0:0:0:0:0"use as the next Hop address are considered as if the next Hop was TR.

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

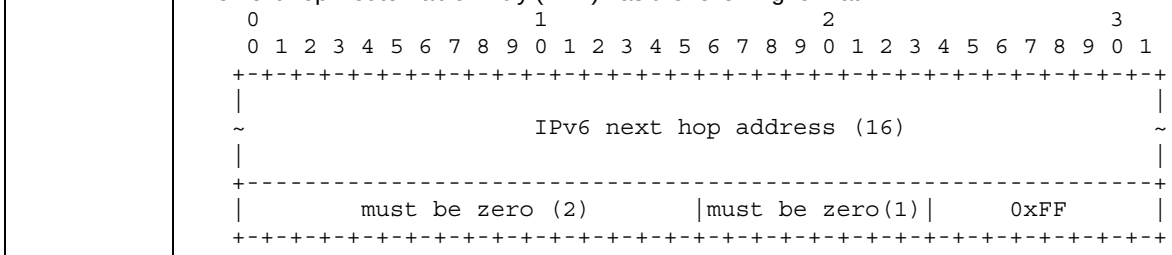
- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

RIPng provides the ability to specify the immediate next hop IPv6 address to which packets to a destination specified by a route table entry (RTE) should be forwarded in much the same way as RIP-2. In RIP-2, each route table entry has a next hop field. Including a next hop field for each RTE in RIPng would nearly double the size of the RTE. Therefore, in RIPng, the next hop is specified by a special RTE and applies to all of the address RTEs following the next hop RTE until the end of the message or until another next hop RTE is encountered.

A next hop RTE is identified by a value of 0xFF in the metric field of an RTE. The prefix field specifies the IPv6 address of the next hop. The route tag and prefix length in the next hop RTE must be set to zero on sending and ignored on receiption.

The next hop Route Table Entry (RTE) has the following format:



Procedure:

1. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR3 and the destination address is the RIPng Multicast group FF02::9.

IPv6 Prefix	Metric	Route Tag
PRFX_4	**1**	0
PRFX_5	**2**	0
PRFX_8	**3**	0
Next Hop	*TR3 Link-local Address*	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_3	**1**	0
PRFX_6Bone	**7**	0

2. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR3 and the destination address is the RIPng Multicast group FF02::9.

Next Hop	*TR3 Link-local Address*	
Ipv6 Prefix	*Metric*	*Route Tag*
PRFX_4	**2**	0
PRFX_5	**2**	0
Next Hop	TR4 Global Address	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_3	1	0
Next Hop	RUT Link-local address Link2	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_11	1	0
Next Hop	*TR3 Link-local Address*	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_8	**2**	0
Next Hop	0:0:0:0:0:0:0:0	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_6Bone	7	0

3. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_3, PRFX_4, PRFX_5, PRFX_8 and PRFX_6Bone.

4. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_3, PRFX_4, PRFX_5 PRFX_8 and PRFX_6Bone.

5. **TR1** sends an **Echo Request** through the RUT to:
   (a) PRFX_3::1
   (b) PRFX_4::1
   (c) PRFX_5::1
   (d) PRFX_8::1
   (e) PRFX_6Bone::1

   The Source Address is TR1 Global Address.

6. **TR3** sends an **Echo Request** through the RUT to:
   (a) PRFX_3::1
   (b) PRFX_4::1
   (c) PRFX_5::1
   (d) PRFX_8::1
   (e) PRFX_6Bone::1

   The Source Address is TR3 Global Address.

Observable Results:

- **Step 1:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent. The RUT should now have the following routing fields:

IPv6 Prefix	Metric	Next HOP	Route Tag
PRFX_4	2	TR4	0
PRFX_5	3	TR4	0
PRFX_8	4	TR4	0
PRFX_3	2	TR3	0
PRFX_10	1	Direct	0
PRFX_11	1	Direct	0
PRFX_6Bone	8	TR3	0

- **Step 2:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent. The RUT should now have the following routing fields:

IPv6 Prefix	Metric	Next HOP	Route Tag
PRFX_4	3	TR3	0
PRFX_5	3	TR3	0
PRFX_8	3	TR3	0
PRFX_3	2	TR4	0
PRFX_10	1	Direct	0
PRFX_11	1	Direct	0
PRFX_6Bone	8	TR4	0

- **Step 3:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_3	2	0
PRFX_4	3	0
PRFX_5	3	0
PRFX_8	3	0

- **Step 4:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_3	2	0
PRFX_4	3	0
PRFX_5	3	0
PRFX_8	3	0

- **Step 5**: The Echo Request for PRFX_4::1, PRFX_5::1 and PRFX_8::1 have to be forwarded to TR3. The Echo Request for PRFX_3::1, PRFX_6Bone::1 have to be forwarded to TR4.

- **Step 6**: The Echo Request for PRFX_4::1, PRFX_5::1 and PRFX_8::1 have to be forwarded to TR3. The Echo Request for PRFX_3::1, PRFX_6Bone::1 have to be forwarded to TR4.

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response <----------	TR3
		2	RIPng Response <----------	TR3
TR1	RIPng Request (PRFX_i with i=3,4,5,8, 6Bone) ---------->	3		
	RIPng Response <----------	Step3		
		4	RIPng Request (PRFX_i with i=3, 4,5,8, 6Bone) <----------	TR4
		Step4	RIPng Response ---------->	TR4
TR1	Echo Request (PRFX_3::1) ---------->	5(a)		
		Step5(a)	Echo Request (PRFX_3::1) ---------->	TR4
TR1	Echo Request (PRFX_4::1) ---------->	5(b)		
		Step5(b)	Echo Request (PRFX_4::1) ---------->	TR3
TR1	Echo Request (PRFX_5::1) ---------->	5(c)		
		Step5(c)	Echo Request (PRFX_5::1) ---------->	TR3
TR1	Echo Request (PRFX_8::1) ---------->	5(d)		
		Step5(d)	Echo Request (PRFX_8::1) ---------->	TR3
TR1	Echo Request (PRFX_6Bone::1) ---------->	5(e)		
		Step5(e)	Echo Request (PRFX_6Bone::1) ---------->	TR4
		6(a)	Echo Request (PRFX_3::1) <----------	TR3
		Step6(a)	Echo Request (PRFX_3::1) ---------->	TR4
		6(b)	Echo Request (PRFX_4::1) <----------	TR3
		Step6(b)	Echo Request (PRFX_4::1) ---------->	TR3
		6(c)	Echo Request (PRFX_5::1) <----------	TR3
		Step6(c)	Echo Request (PRFX_5::1) ---------->	TR3
		6(d)	Echo Request (PRFX_8::1) <----------	TR3
		Step6(d)	Echo Request (PRFX_8::1) ---------->	TR3
		6(e)	Echo Request (PRFX_6Bone::1) <----------	TR3
		Step6(e)	Echo Request (PRFX_6Bone::1) ---------->	TR4

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT by sending a **RIPng Response** packet from **TR3** and **TR4** in which all Metrics are set to infinity(16).

5.2.1.3 Route Tag Update

Purpose: Check the update of Route Tag from the same router as specified in the Next Hop address of the route.

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

If there is an existing route, compare the next hop address to the address of the router from which the datagram came. If this datagram is from the same router as the existing route, reinitialize the timeout. Next, compare the metrics. If the datagram is from the same router as the existing route, and the new metric is different than the old one; or, if the new metric is lower than the old one; do the following actions:

- Adopt the route from the datagram. That is, put the new metric in, and adjust the next hop address (if necessary).
- Set the route change flag and signal the output process to trigger an update.
- If the new metric is infinity, start the deletion process; otherwise, re-initialize the timeout.

If the new metric is infinity, the deletion process begins for the route, which is no longer used for routing packets. Note that the deletion process is started only when the metric is first set to infinity. If the metric was already infinity, then a new deletion process is not started.

If the new metric is the same as the old one, it is simplest to do nothing further (beyond reinitializing the timeout, as specified above); but, there is a heuristic which could be applied. Normally, it is senseless to replace a route if the new route has the same metric as the existing route; this would cause the route to bounce back and forth, which would generate an intolerable number of triggered updates. However, if the existing route is showing signs of timing out, it may be better to switch to an equally-good alternative route immediately, rather than waiting for the timeout to happen. Therefore, if the new metric is the same as the old one, examine the timeout for the existing route. If it is at least halfway to the expiration point, switch to the new route. This heuristic is optional, but highly recommended.

The route tag field is an attribute assigned to a route which must be preserved and readvertised with a route. The intended use of the route tag is to provide a method of separating "internal" RIPng routes (routes for networks within the RIPng routing domain) from "external" RIPng routes, which may have been imported from an EGP or another IGP.

Procedure:

1. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255. The RTEs included in the Response are the following:

IPv6 Prefix	Metric	Route Tag
PRFX_4	1	**0**
PRFX_5	1	**0**
PRFX_8	2	**0**
PRFX_6Bone	7	**150**

2. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255. The RTEs included in the Response are the following:

IPv6 Prefix	Metric	Route Tag
PRFX_4	1	**1**
PRFX_5	1	**255**
PRFX_8	1	**128**
PRFX_6Bone	8	**151**

3. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5, PRFX_8 and PRFX_6Bone.

4. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_4, PRFX_5, PRFX_8 and PRFX_6Bone.

5. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255. The RTEs included in the Response are the following:

IPv6 Prefix	Metric	Route Tag
PRFX_4	1	**0**
PRFX_5	1	**0**
PRFX_8	2	**0**
PRFX_6Bone	7	**150**

6. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.

7. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.

Observable Results:

- **Step 1, 2 & 5:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.
- **Step 3:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	2	**1**
PRFX_5	2	**255**
PRFX_8	2	**128**
PRFX_6Bone	9	**151**

- **Step 4:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	2	**1**
PRFX_5	2	**255**
PRFX_8	2	**128**
PRFX_6Bone	9	**151**

- **Step 6:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	2	**0**
PRFX_5	2	**0**
PRFX_8	3	**0**
PRFX_6Bone	8	**150**

- **Step 7:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	2	**0**
PRFX_5	2	**0**
PRFX_8	3	**0**
PRFX_6Bone	8	**150**

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response <----------	TR4
		2	RIPng Response <----------	TR4
TR1	RIPng Request (PRFX_i with i=4,5,8, 6Bone) ---------->	3		
TR1	RIPng Response <----------	Step3		
		4	RIPng Request (PRFX_i with i=4,5,8, 6Bone) <----------	TR3
		Step4	RIPng Response ---------->	TR3
		5	RIPng Response <----------	TR4
TR1	RIPng Request (PRFX_i with i=4,5,8, 6Bone) ---------->	6		
TR1	RIPng Response <----------	Step6		
		7	RIPng Request (PRFX_i with i=4,5,8, 6Bone) <----------	TR3
		Step7	RIPng Response ---------->	TR3

Postamble:

- Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT by sending a **RIPng Response** packet from **TR4** in which all Metrics are set to infinity(16).

5.2.1.4 Route Tag Update with Next Hop RTE Field

Purpose: Check the update of Route Tag even when a Next Hop field is present in the Response. Check that Link-local address of the TR, global address of any node, special address "0:0:0:0:0:0:0:0"use as the next Hop address are considered as if the next Hop was TR.

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

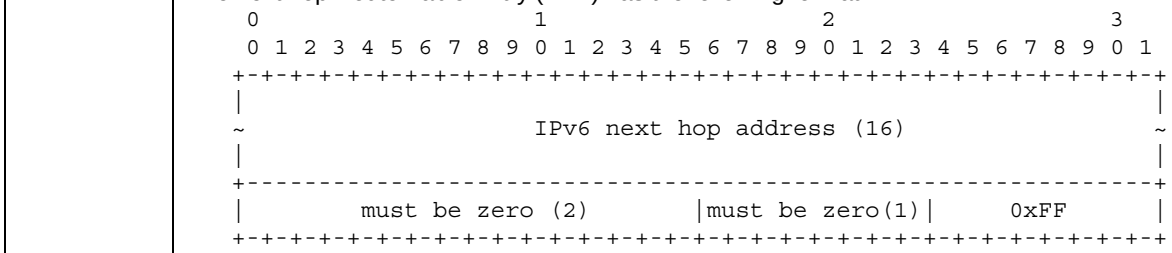
- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

RIPng provides the ability to specify the immediate next hop IPv6 address to which packets to a destination specified by a route table entry (RTE) should be forwarded in much the same way as RIP-2. In RIP-2, each route table entry has a next hop field. Including a next hop field for each RTE in RIPng would nearly double the size of the RTE. Therefore, in RIPng, the next hop is specified by a special RTE and applies to all of the address RTEs following the next hop RTE until the end of the message or until another next hop RTE is encountered.

A next hop RTE is identified by a value of 0xFF in the metric field of an RTE. The prefix field specifies the IPv6 address of the next hop. The route tag and prefix length in the next hop RTE must be set to zero on sending and ignored on receiption.

The next hop Route Table Entry (RTE) has the following format:



Procedure:

1. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR3 and the destination address is the RIPng Multicast group FF02::9.

IPv6 Prefix	Metric	Route Tag
PRFX_4	**1**	0x41
PRFX_5	**2**	0x41
PRFX_8	**3**	0x41
Next Hop	*TR3 Link-local Address*	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_3	**1**	0x41
PRFX_6Bone	**7**	0x41

2. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR3 and the destination address is the RIPng Multicast group FF02::9.

Next Hop	*TR3 Link-local Address*	
Ipv6 Prefix	*Metric*	*Route Tag*
PRFX_4	**2**	**0x42**
PRFX_5	**2**	**0x42**
Next Hop	TR4 Global Address	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_3	1	**0x42**
Next Hop	RUT Link-local address Link2	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_11	1	**0x42**
Next Hop	*TR3 Link-local Address*	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_8	**2**	**0x42**
Next Hop	0:0:0:0:0:0:0:0	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_6Bone	7	**0x42**

3. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_3, PRFX_4, PRFX_5, PRFX_8 and PRFX_6Bone.

4. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_3, PRFX_4, PRFX_5 PRFX_8 and PRFX_6Bone.

5. **TR1** sends an **Echo Request** through the RUT to:

   (f) PRFX_3::1

   (g) PRFX_4::1

   (h) PRFX_5::1

   (i) PRFX_8::1

   (j) PRFX_6Bone::1

   The Source Address is TR1 Global Address.

6. **TR3** sends an **Echo Request** through the RUT to:

   (a) PRFX_3::1

   (b) PRFX_4::1

   (c) PRFX_5::1

   (d) PRFX_8::1

   (e) PRFX_6Bone::1

   The Source Address is TR3 Global Address.

Observable Results:

- **Step 1:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent. The RUT should now have the following routing fields:

IPv6 Prefix	Metric	Next HOP	Route Tag
PRFX_4	2	TR4	0x41
PRFX_5	3	TR4	0x41
PRFX_8	4	TR4	0x41
PRFX_3	2	TR3	0x41
PRFX_10	1	Direct	0
PRFX_11	1	Direct	0
PRFX_6Bone	8	TR3	0x41

- **Step 2:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent. The RUT should now have the following routing fields:

IPv6 Prefix	Metric	Next HOP	Route Tag
PRFX_4	3	TR3	0x42
PRFX_5	3	TR3	0x42
PRFX_8	3	TR3	0x42
PRFX_3	2	TR4	0x42
PRFX_10	1	Direct	0
PRFX_11	1	Direct	0
PRFX_6Bone	8	TR4	0x42

- **Step 3:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_3	2	0x42
PRFX_4	3	0x42
PRFX_5	3	0x42
PRFX_8	3	0x42

- **Step 4:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_3	2	0x42
PRFX_4	3	0x42
PRFX_5	3	0x42
PRFX_8	3	0x42

- **Step 5**: The Echo Request for PRFX_4::1, PRFX_5::1 and PRFX_8::1 have to be forwarded to TR3. The Echo Request for PRFX_3::1, PRFX_6Bone::1 have to be forwarded to TR4.

- **Step 6**: The Echo Request for PRFX_4::1, PRFX_5::1 and PRFX_8::1 have to be forwarded to TR3. The Echo Request for PRFX_3::1, PRFX_6Bone::1 have to be forwarded to TR4.

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response <----------	TR3
		2	RIPng Response <----------	TR3
TR1	RIPng Request (PRFX_i with i=3,4,5,8, 6Bone) ---------->	3		
	RIPng Response <----------	Step3		
		4	RIPng Request (PRFX_i with i=3, 4,5,8, 6Bone) <----------	TR4
		Step4	RIPng Response ---------->	TR4
TR1	Echo Request (PRFX_3::1) ---------->	5(a)		
		Step5(a)	Echo Request (PRFX_3::1) ---------->	TR4
TR1	Echo Request (PRFX_4::1) ---------->	5(b)		
		Step5(b)	Echo Request (PRFX_4::1) ---------->	TR3
TR1	Echo Request (PRFX_5::1) ---------->	5(c)		
		Step5(c)	Echo Request (PRFX_5::1) ---------->	TR3
TR1	Echo Request (PRFX_8::1) ---------->	5(d)		
		Step5(d)	Echo Request (PRFX_8::1) ---------->	TR3
TR1	Echo Request (PRFX_6Bone::1) ---------->	5(e)		
		Step5(e)	Echo Request (PRFX_6Bone::1) ---------->	TR4
		6(a)	Echo Request (PRFX_3::1) <----------	TR3
		Step6(a)	Echo Request (PRFX_3::1) ---------->	TR4
		6(b)	Echo Request (PRFX_4::1) <----------	TR3
		Step6(b)	Echo Request (PRFX_4::1) ---------->	TR3
		6(c)	Echo Request (PRFX_5::1) <----------	TR3
		Step6(c)	Echo Request (PRFX_5::1) ---------->	TR3
		6(d)	Echo Request (PRFX_8::1) <----------	TR3
		Step6(d)	Echo Request (PRFX_8::1) ---------->	TR3
		6(e)	Echo Request (PRFX_6Bone::1) <----------	TR3
		Step6(e)	Echo Request (PRFX_6Bone::1) ---------->	TR4

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT by sending a **RIPng Response** packet from **TR3** and **TR4** in which all Metrics are set to infinity(16).

5.2.2 Update with better route from Same Link

5.2.2.1 Metric Update

Purpose: Check the update of route if a better route is found

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

If there is an existing route, compare the next hop address to the address of the router from which the datagram came. If this datagram is from the same router as the existing route, reinitialize the timeout. Next, compare the metrics. If the datagram is from the same router as the existing route, and the new metric is different than the old one; or, if the new metric is lower than the old one; do the following actions:

- Adopt the route from the datagram. That is, put the new metric in, and adjust the next hop address (if necessary).
- Set the route change flag and signal the output process to trigger an update.
- If the new metric is infinity, start the deletion process; otherwise, re-initialize the timeout.

If the new metric is infinity, the deletion process begins for the route, which is no longer used for routing packets. Note that the deletion process is started only when the metric is first set to infinity. If the metric was already infinity, then a new deletion process is not started.

If the new metric is the same as the old one, it is simplest to do nothing further (beyond reinitializing the timeout, as specified above); but, there is a heuristic which could be applied. Normally, it is senseless to replace a route if the new route has the same metric as the existing route; this would cause the route to bounce back and forth, which would generate an intolerable number of triggered updates. However, if the existing route is showing signs of timing out, it may be better to switch to an equally-good alternative route immediately, rather than waiting for the timeout to happen. Therefore, if the new metric is the same as the old one, examine the timeout for the existing route. If it is at least halfway to the expiration point, switch to the new route. This heuristic is optional, but highly recommended.

Packets:

- *RTEs sent by TR3 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_3	1	0
PRFX_4	1	0
PRFX_5	2	0
PRFX_8	3	0

- *RTEs sent by TR4 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_3	2	0

PRFX_4	1	0
PRFX_5	1	0
PRFX_8	2	0

Procedure:

1. **TR3** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR3 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255
2. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255.
3. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_3, PRFX_4, PRFX_5 and PRFX_8.
4. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_3, PRFX_4, PRFX_5 and PRFX_8.
5. **TR1** sends an **Echo Request** through the RUT to:
   (a) PRFX_3::1
   (b) PRFX_4::1
   (c) PRFX_5::1
   (d) PRFX_8::1

   The Source Address is TR1 Global Address.
6. **TR3** sends an **Echo Request** through the RUT to:
   (a) PRFX_3::1
   (b) PRFX_4::1
   (c) PRFX_5::1
   (d) PRFX_8::1

   The Source Address is TR3 Global Address.

Observable Results:

- **Step 1, 2:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.
- **Step 3:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 2, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_3	2	0
PRFX_4	2	0
PRFX_5	2	0
PRFX_8	3	0

- **Step 4:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_3	2	0
PRFX_4	2	0

PRFX_5	2	0
PRFX_8	3	0

- **Step 5**: The Echo Request for PRFX_5::1 and PRFX_8::1 have to be forwarded to TR4. The Echo Request for PRFX_3::1 has to be forwarded to TR3. The Echo Request for PRFX_4::1 SHOULD be forwarded to TR3(recommended behaviour, if not it MUST be TR4).**Step 6**: The Echo Request for PRFX_5::1 and PRFX_8::1 have to be forwarded to TR4. The Echo Request for PRFX_3::1 has to be forwarded to TR3. The Echo Request for PRFX_4::1 SHOULD be forwarded to TR3(recommended behaviour, if not it MUST be TR4).

Test Sequence:

Tester (Tri)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response <----------	TR3
		2	RIPng Response <----------	TR4
TR1	RIPng Request (PRFX_i with i=3,4,5,8) ---------->	3		
TR1	RIPng Response <----------	Step3		
		4	RIPng Request (PRFX_i with i=3,4,5,8) <----------	TR3
		Step4	RIPng Response ---------->	TR3
TR1	Echo Request (PRFX_3::1) ---------->	5(a)		
		Step5(a)	Echo Request (PRFX_3::1) ---------->	TR3
TR1	Echo Request (PRFX_4::1) ---------->	5(b)		
		Step5(b)	Echo Request (PRFX_4::1) ---------->	TR3|TR4
TR1	Echo Request (PRFX_5::1) ---------->	5(c)		
		Step5(c)	Echo Request (PRFX_5::1) ---------->	TR4
TR1	Echo Request (PRFX_8::1) ---------->	5(d)		
		Step5(d)	Echo Request (PRFX_8::1) ---------->	TR4
		6(a)	Echo Request (PRFX_3::1) <----------	TR3
		Step6(a)	Echo Request (PRFX_3::1) ---------->	TR3
		6(b)	Echo Request (PRFX_4::1) <----------	TR3
		Step6(b)	Echo Request (PRFX_4::1) ---------->	TR3|TR4
		6(c)	Echo Request (PRFX_5::1) <----------	TR3
		Step6(c)	Echo Request (PRFX_5::1) ---------->	TR4
		6(d)	Echo Request (PRFX_8::1) <----------	TR3
		Step6(d)	Echo Request (PRFX_8::1) ---------->	TR4

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT by sending a **RIPng Response** packet from **TR4** and **TR3** in which all Metrics are set to infinity(16).

5.2.2.2 Metric Update with Next Hop RTE field

Purpose: Check that receiving a route in a RTE field with a metric better than the one existing update the Next Hop and the metric in the routing table.

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

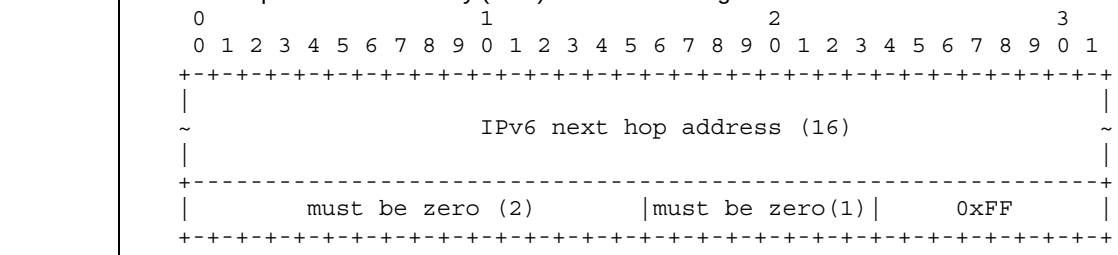
- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

RIPng provides the ability to specify the immediate next hop IPv6 address to which packets to a destination specified by a route table entry (RTE) should be forwarded in much the same way as RIP-2. In RIP-2, each route table entry has a next hop field. Including a next hop field for each RTE in RIPng would nearly double the size of the RTE. Therefore, in RIPng, the next hop is specified by a special RTE and applies to all of the address RTEs following the next hop RTE until the end of the message or until another next hop RTE is encountered.

A next hop RTE is identified by a value of 0xFF in the metric field of an RTE. The prefix field specifies the IPv6 address of the next hop. The route tag and prefix length in the next hop RTE must be set to zero on sending and ignored on receiption.

The next hop Route Table Entry (RTE) has the following format:



Packets:

- *RTEs sent by TR3 in Ripng Response are the following in the following order: :*

Next Hop	*TR3 Link-local Address*	
Ipv6 Prefix	*Metric*	*Route Tag*
PRFX_4	**2**	0
PRFX_5	**2**	0
Next Hop	TR4 Global Address	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_3	1	0
Next Hop	RUT Link-local address Link1	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_11	1	0
Next Hop	*TR3 Link-local Address*	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_8	**2**	0

Next Hop	0:0:0:0:0:0:0:0	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_6Bone	7	0

- *RTEs sent by TR4 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_4	**1**	0
PRFX_5	**2**	0
PRFX_8	**3**	0

Procedure:

1. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9.
2. **TR3** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR3 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255.
3. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.
4. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.
5. **TR1** sends an **Echo Request** to:
   (a) PRFX_4::1
   (b) PRFX_5::1
   (c) PRFX_8::1

Observable Results:

- **Step 1:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent. The RUT should now have the following routing fields:

IPv6 Prefix	Metric	Next HOP	Route Tag
PRFX_4	**2**	TR4	0
PRFX_5	**3**	TR4	0
PRFX_8	**4**	TR4	0

- **Step 2:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent. The RUT should now have the following routing fields:

IPv6 Prefix	Metric	Next HOP	Route Tag
PRFX_4	**2**	TR4	0
PRFX_5	**3**	TR4 (MAY TR3)	0
PRFX_8	**3**	**TR3**	0
PRFX_3	2	TR3	0
PRFX_10	1	Direct	0
PRFX_11	1	Direct	0

PRFX_6Bone	8	TR3	0

- **Step 3:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	**2**	0
PRFX_5	**3**	0
PRFX_8	**3**	0

- **Step 4:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	**2**	0
PRFX_5	**3**	0
PRFX_8	**3**	0

- **Step 5**: The Echo Request for PRFX_4::1have to be forwarded to TR4. The Echo Request for PRFX_8::1 has to be forwarded to TR2. The Echo Request for PRFX_5_SHOULD be forwarded to TR4 (recommended behaviour, if not it MUST be TR3).

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response <----------	TR4
		2	RIPng Response <----------	TR3
TR1	RIPng Request (PRFX_i with i=4,5,8) ---------->	3		
TR1	RIPng Response <----------	Step3		
		4	RIPng Request (PRFX_i with i=4,5,8) <----------	TR3
		Step4	RIPng Response ---------->	TR3
TR1	Echo Request PRFX4::1 ---------->	5(a)		
		Step5(a)	Echo Request PRFX4::1 ---------->	TR4
TR1	Echo Request PRFX5::1 ---------->	5(b)		
		Step5(b)	Echo Request PRFX5::1 ---------->	TR4 | TR3
TR1	Echo Request PRFX8::1 ---------->	5(c)		
		Step5(c)	Echo Request PRFX8::1 ---------->	TR3

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT by sending a **RIPng Response** packet from **TR3** and **TR4** in which all Metrics are set to infinity(16).

5.2.2.3 Route Tag Update

Purpose: Check the update of Route Tag if a better route is found

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

If there is an existing route, compare the next hop address to the address of the router from which the datagram came. If this datagram is from the same router as the existing route, reinitialize the timeout. Next, compare the metrics. If the datagram is from the same router as the existing route, and the new metric is different than the old one; or, if the new metric is lower than the old one; do the following actions:

- Adopt the route from the datagram. That is, put the new metric in, and adjust the next hop address (if necessary).
- Set the route change flag and signal the output process to trigger an update.
- If the new metric is infinity, start the deletion process; otherwise, re-initialize the timeout.

If the new metric is infinity, the deletion process begins for the route, which is no longer used for routing packets. Note that the deletion process is started only when the metric is first set to infinity. If the metric was already infinity, then a new deletion process is not started.

If the new metric is the same as the old one, it is simplest to do nothing further (beyond reinitializing the timeout, as specified above); but, there is a heuristic which could be applied. Normally, it is senseless to replace a route if the new route has the same metric as the existing route; this would cause the route to bounce back and forth, which would generate an intolerable number of triggered updates. However, if the existing route is showing signs of timing out, it may be better to switch to an equally-good alternative route immediately, rather than waiting for the timeout to happen. Therefore, if the new metric is the same as the old one, examine the timeout for the existing route. If it is at least halfway to the expiration point, switch to the new route. This heuristic is optional, but highly recommended.

The route tag field is an attribute assigned to a route which must be preserved and readvertised with a route. The intended use of the route tag is to provide a method of separating "internal" RIPng routes (routes for networks within the RIPng routing domain) from "external" RIPng routes, which may have been imported from an EGP or another IGP.

Packets:

- *RTEs sent by TR3 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_3	1	0x3
PRFX_4	1	0x3
PRFX_5	2	0x3
PRFX_8	3	0x3

- *RTEs sent by TR4 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_3	2	0x4
PRFX_4	1	0x4
PRFX_5	1	0x4
PRFX_8	2	0x4

Procedure:

1. **TR3** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR3 and the destination address is the RIPng Multicast group FF02::9.
2. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255.
3. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_3, PRFX_4, PRFX_5 and PRFX_8.
4. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_3, PRFX_4, PRFX_5 and PRFX_8.
5. **TR3** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR3 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255.
6. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_3, PRFX_4, PRFX_5 and PRFX_8.
7. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_3, PRFX_4, PRFX_5 and PRFX_8.

Observable Results:

- **Step 1, 2 & 5:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.
- **Step 3:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_3	2	0x3
PRFX_4	2	0x3
PRFX_5	2	0x4
PRFX_8	3	0x4

- **Step 4:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_3	2	0x3
PRFX_4	2	0x3
PRFX_5	2	0x4

PRFX_8	3	0x4

- **Step 6:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777; The response should be the same than in Step 3 since nothing should have been modified in the routing table of the RUT.

IPv6 Prefix	Metric	Route Tag
PRFX_3	2	0x3
PRFX_4	2	0x3
PRFX_5	2	0x4
PRFX_8	3	0x4

- **Step 7:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777: The response should be the same than in Step 3 since nothing should have been modified in the routing table of the RUT.

IPv6 Prefix	Metric	Route Tag
PRFX_3	2	0x3
PRFX_4	2	0x3
PRFX_5	2	0x4
PRFX_8	3	0x4

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response <----------	TR3
		2	RIPng Response <----------	TR4
TR1	RIPng Request (PRFX_i with i=3,4,5,8) ---------->	3		
TR1	RIPng Response <----------	Step3		
		4	RIPng Request (PRFX_i with i=3,4,5,8) <----------	TR3
		Step4	RIPng Response ---------->	TR3
		5	RIPng Response <----------	TR3
TR1	RIPng Request (PRFX_i with i=3,4,5,8) ---------->	6		
TR1	RIPng Response <----------	Step6		
		7	RIPng Request (PRFX_i with i=3,4,5,8) <----------	TR3
		Step7	RIPng Response ---------->	TR3

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT by sending a **RIPng Response** packet from **TR4** and **TR3** in which all Metrics are set to infinity(16).

5.2.2.4 Route Tag Update with Next Hop RTE field

Purpose: Check that receiving a route in a RTE field with a metric better than the one existing update the Next Hop and the metric in the routing table.

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

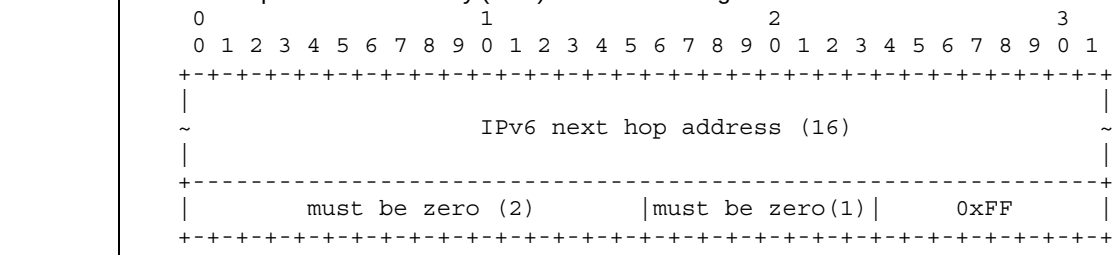
- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

RIPng provides the ability to specify the immediate next hop IPv6 address to which packets to a destination specified by a route table entry (RTE) should be forwarded in much the same way as RIP-2. In RIP-2, each route table entry has a next hop field. Including a next hop field for each RTE in RIPng would nearly double the size of the RTE. Therefore, in RIPng, the next hop is specified by a special RTE and applies to all of the address RTEs following the next hop RTE until the end of the message or until another next hop RTE is encountered.

A next hop RTE is identified by a value of 0xFF in the metric field of an RTE. The prefix field specifies the IPv6 address of the next hop. The route tag and prefix length in the next hop RTE must be set to zero on sending and ignored on receiption.

The next hop Route Table Entry (RTE) has the following format:



Packets:

- *RTEs sent by TR3 in Ripng Response are the following in the following order: :*

Next Hop	*TR3 Link-local Address*	
Ipv6 Prefix	*Metric*	*Route Tag*
PRFX_4	**2**	0x3
PRFX_5	**2**	0x3
Next Hop	TR4 Global Address	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_3	1	0x3
Next Hop	RUT Link-local address Link1	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_11	1	0x3
Next Hop	*TR3 Link-local Address*	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_8	**2**	0x3

Next Hop	0:0:0:0:0:0:0:0	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_6Bone	7	0x3

- *RTEs sent by TR4 in Ripng Response are the following :*

IPv6 Prefix	**Metric**	**Route Tag**
PRFX_4	**1**	0x4
PRFX_5	**2**	0x4
PRFX_8	**3**	0x4

Procedure:

1. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9.
2. **TR3** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR3 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255.
3. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.
4. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.
5. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9.
6. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.
7. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.
8. **TR1** sends an **Echo Request** to:
   (a) PRFX_4::1
   (b) PRFX_5::1
   (c) PRFX_8::1

Observable Results:

- **Step 1:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent. The RUT should now have the following routing fields:

IPv6 Prefix	Metric	Next HOP	Route Tag
PRFX_4	**2**	TR4	**0x4**
PRFX_5	**3**	TR4	**0x4**
PRFX_8	**4**	TR4	**0x4**

- **Step 2:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent. The RUT should now have the following routing fields:

IPv6 Prefix	Metric	Next HOP	Route Tag
PRFX_4	2	TR4	0x4
PRFX_5	3	TR4	0x4
PRFX_8	**3**	**TR3**	**0x3**
PRFX_3	2	TR3	0x3
PRFX_10	1	Direct	0
PRFX_11	1	Direct	0
PRFX_6Bone	8	TR3	0x3

- **Step 3:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	2	0x4
PRFX_5	3	0x4
PRFX_8	**3**	**0x3**

- **Step 4:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	2	0x4
PRFX_5	3	0x4
PRFX_8	**3**	**0x1**

- **Step 6:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	2	0x4
PRFX_5	3	0x4
PRFX_8	**3**	**0x3**

- **Step 7:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	2	0x4
PRFX_5	3	0x4
PRFX_8	**3**	**0x3**

- **Step 8**: The Echo Request for PRFX_4::1, PRFX_5::1 have to be forwarded to TR4. The Echo Request for PRFX_8::1 has to be forwarded to TR3.

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response <----------	TR4
		2	RIPng Response <----------	TR3
TR1	RIPng Request (PRFX_i with i=4,5,8) ---------->	3		
TR1	RIPng Response <----------	Step3		
		4	RIPng Request (PRFX_i with i=4,5,8) <----------	TR3
		Step4	RIPng Response ---------->	TR3
		5	RIPng Response <----------	TR4
TR1	RIPng Request (PRFX_i with i=4,5,8) ---------->	6		
TR1	RIPng Response <----------	Step6		
		7	RIPng Request (PRFX_i with i=4,5,8) <----------	TR3
		Step7	RIPng Response ---------->	TR3
TR1	Echo Request PRFX4::1 ---------->	8(a)		
		Step8(a)	Echo Request PRFX4::1 ---------->	TR4
TR1	Echo Request PRFX5::1 ---------->	8(b)		
		Step8(b)	Echo Request PRFX5::1 ---------->	TR4
TR1	Echo Request PRFX8::1 ---------->	8(c)		
		Step8(c)	Echo Request PRFX8::1 ---------->	TR3

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT by sending a **RIPng Response** packet from **TR3** and **TR4** in which all Metrics are set to infinity(16).

5.2.3 Update with better route from a different Link

5.2.3.1 Metric Update

Purpose: Check the update of route if a better route is found

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

If there is an existing route, compare the next hop address to the address of the router from which the datagram came. If this datagram is from the same router as the existing route, reinitialize the timeout. Next, compare the metrics. If the datagram is from the same router as the existing route, and the new metric is different than the old one; or, if the new metric is lower than the old one; do the following actions:

- Adopt the route from the datagram. That is, put the new metric in, and adjust the next hop address (if necessary).
- Set the route change flag and signal the output process to trigger an update.
- If the new metric is infinity, start the deletion process; otherwise, re-initialize the timeout.

If the new metric is infinity, the deletion process begins for the route, which is no longer used for routing packets. Note that the deletion process is started only when the metric is first set to infinity. If the metric was already infinity, then a new deletion process is not started.

If the new metric is the same as the old one, it is simplest to do nothing further (beyond reinitializing the timeout, as specified above); but, there is a heuristic which could be applied. Normally, it is senseless to replace a route if the new route has the same metric as the existing route; this would cause the route to bounce back and forth, which would generate an intolerable number of triggered updates. However, if the existing route is showing signs of timing out, it may be better to switch to an equally-good alternative route immediately, rather than waiting for the timeout to happen. Therefore, if the new metric is the same as the old one, examine the timeout for the existing route. If it is at least halfway to the expiration point, switch to the new route. This heuristic is optional, but highly recommended.

Packets:

- *RTEs sent by TR2 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_2	1	0
PRFX_4	1	0
PRFX_5	2	0
PRFX_8	1	0

- *RTEs sent by TR4 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_4	1	0

PRFX_5	1	0
PRFX_8	2	0

Procedure:

1. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255
2. **TR2** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR2 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255.
3. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_2, PRFX_4, PRFX_5 and PRFX_8.
4. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_2, PRFX_4, PRFX_5 and PRFX_8.
5. **TR1** sends an **Echo Request** to:
   (a) PRFX_2::1
   (b) PRFX_4::1
   (c) PRFX_5::1
   (d) PRFX_8::1
6. **TR3** sends an **Echo Request** to:
   (a) PRFX_2::1
   (b) PRFX_4::1
   (c) PRFX_5::1
   (d) PRFX_8::1
7. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255.
8. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_2, PRFX_4, PRFX_5 and PRFX_8.
9. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_2, PRFX_4, PRFX_5 and PRFX_8.
10. **TR1** sends an **Echo Request** to:
    (a) PRFX_2::1
    (b) PRFX_4::1
    (c) PRFX_5::1
    (d) PRFX_8::1
11. **TR3** sends an **Echo Request** to:
    (a) PRFX_2::1
    (b) PRFX_4::1
    (c) PRFX_5::1
    (d) PRFX_8::1

Observable Results:

- **Step 1, 2 & 7:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.
- **Step 3:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_2	**2**	0
PRFX_4	**2**	0
PRFX_5	**2**	0
PRFX_8	**2**	0

- **Step 4:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_2	**2**	0
PRFX_4	**2**	0
PRFX_5	**2**	0
PRFX_8	**2**	0

- **Step 5:** The Echo Request for PRFX_4::1 has to be forwarded to TR4. The Echo Request for PRFX_2::1 and PRFX_8::1 has to be forwarded to TR2, The Echo Request for PRFX_5::1 SHOULD be forwarded to TR4 (recommended behaviour, if not it MUST be TR2).
- **Step 6:** The Echo Request for PRFX_4::1 has to be forwarded to TR4. The Echo Request for PRFX_2::1 and PRFX_8::1 has to be forwarded to TR2, The Echo Request for PRFX_5::1 SHOULD be forwarded to TR4 (recommended behaviour, if not it MUST be TR2).**Step 8:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777; The response should be the same than in Step 3 since nothing should have been modified in the routing table of the RUT.

IPv6 Prefix	Metric	Route Tag
PRFX_2	2	0
PRFX_4	2	0
PRFX_5	2	0
PRFX_8	2	0

- **Step 9:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777: The response should be the same than in Step 3 since nothing should have been modified in the routing table of the RUT.

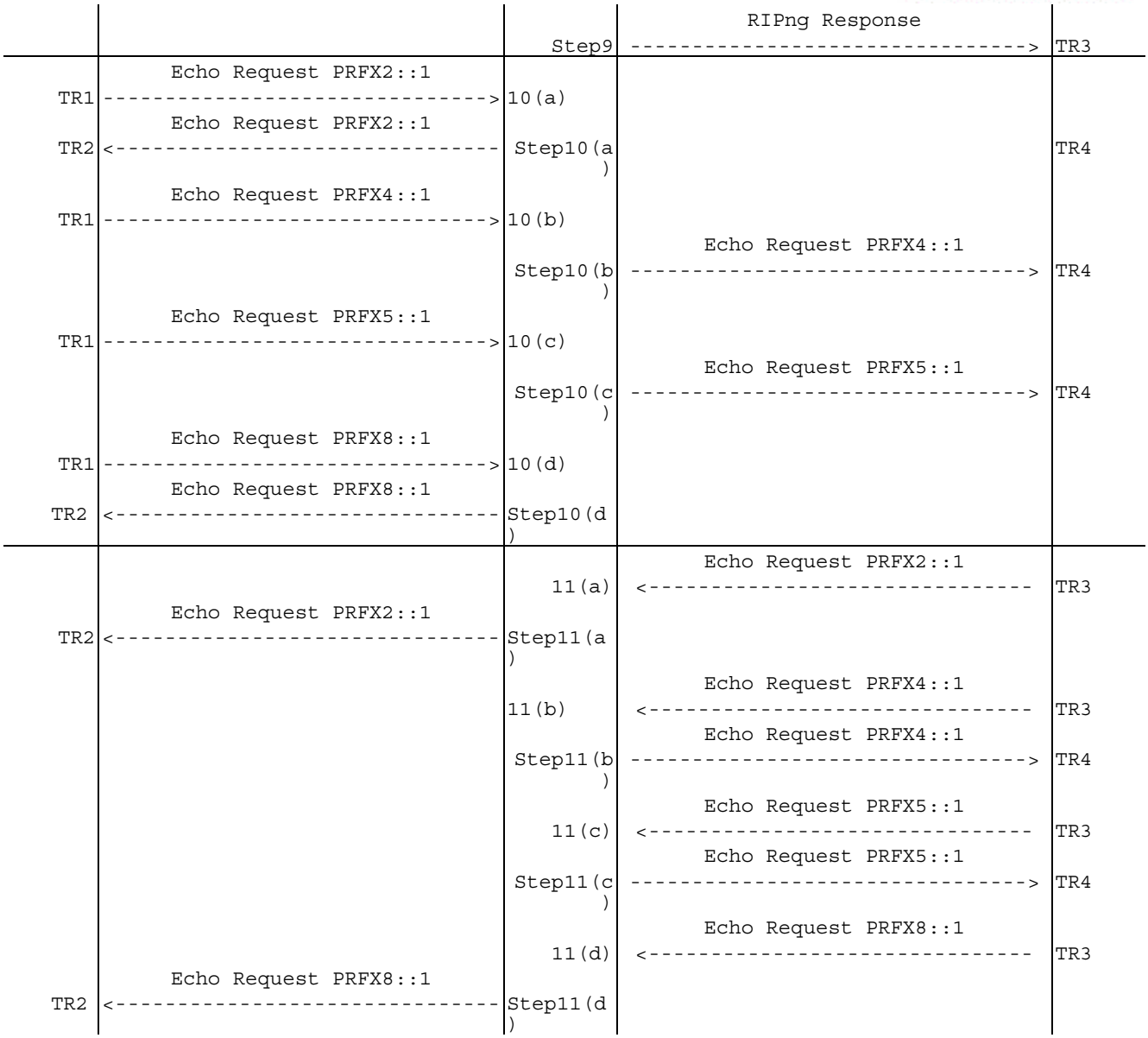
IPv6 Prefix	Metric	Route Tag
PRFX_2	2	0
PRFX_4	2	0
PRFX_5	2	0
PRFX_8	2	0

- **Step 10**: The Echo Request for PRFX_4::1, PRFX_5::1 have to be forwarded to TR4. The Echo Request for PRFX_2::1 and PRFX_8::1 has to be forwarded to TR2.

- **Step 11**: The Echo Request for PRFX_4::1, PRFX_5::1 have to be forwarded to TR4. The Echo Request for PRFX_2::1 and PRFX_8::1 has to be forwarded to TR2.

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response <----------	TR4
TR2	RIPng Response ---------->	2		
TR1	RIPng Request (PRFX_i with i=2,4,5,8) ---------->	3		
TR1	RIPng Response <----------	Step3		
		4	RIPng Request (PRFX_i with i=2,4,5,8) <----------	TR3
		Step4	RIPng Response ---------->	TR3
TR1	Echo Request PRFX2::1 ---------->	5(a)		
TR2	Echo Request PRFX2::1 <----------	Step5(a)		TR4
TR1	Echo Request PRFX4::1 ---------->	5(b)		
		Step5(b)	Echo Request PRFX4::1 ---------->	TR4
TR1	Echo Request PRFX5::1 ---------->	5(c)		
		Step5(c)	Echo Request PRFX5::1 ---------->	TR4
TR1	Echo Request PRFX8::1 ---------->	5(d)		
TR2	Echo Request PRFX8::1 <----------	Step5(d)		
		6(a)	Echo Request PRFX2::1 <----------	TR3
TR2	Echo Request PRFX2::1 <----------	Step6(a)		
		6(b)	Echo Request PRFX4::1 <----------	TR3
		Step6(b)	Echo Request PRFX4::1 ---------->	TR4
		6(c)	Echo Request PRFX5::1 <----------	TR3
		Step6(c)	Echo Request PRFX5::1 ---------->	TR4
		6(d)	Echo Request PRFX8::1 <----------	TR3
TR2	Echo Request PRFX8::1 <----------	Step6(d)		
		7	RIPng Response <----------	TR4
TR1	RIPng Request (PRFX_i with i=2,4,5,8) ---------->	8		
TR1	RIPng Response <----------	Step8		
		9	RIPng Request (PRFX_i with i=4,5,8) <----------	TR3



Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT by sending a **RIPng Response** packet from **TR4** and **TR2** in which all Metrics are set to infinity(16).

5.2.3.2 Route Tag Update

Purpose: Check the update of Route Tag if a better route is found

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

If there is an existing route, compare the next hop address to the address of the router from which the datagram came. If this datagram is from the same router as the existing route, reinitialize the timeout. Next, compare the metrics. If the datagram is from the same router as the existing route, and the new metric is different than the old one; or, if the new metric is lower than the old one; do the following actions:

- Adopt the route from the datagram. That is, put the new metric in, and adjust the next hop address (if necessary).
- Set the route change flag and signal the output process to trigger an update.
- If the new metric is infinity, start the deletion process; otherwise, re-initialize the timeout.

If the new metric is infinity, the deletion process begins for the route, which is no longer used for routing packets. Note that the deletion process is started only when the metric is first set to infinity. If the metric was already infinity, then a new deletion process is not started.

If the new metric is the same as the old one, it is simplest to do nothing further (beyond reinitializing the timeout, as specified above); but, there is a heuristic which could be applied. Normally, it is senseless to replace a route if the new route has the same metric as the existing route; this would cause the route to bounce back and forth, which would generate an intolerable number of triggered updates. However, if the existing route is showing signs of timing out, it may be better to switch to an equally-good alternative route immediately, rather than waiting for the timeout to happen. Therefore, if the new metric is the same as the old one, examine the timeout for the existing route. If it is at least halfway to the expiration point, switch to the new route. This heuristic is optional, but highly recommended.

The route tag field is an attribute assigned to a route which must be preserved and readvertised with a route. The intended use of the route tag is to provide a method of separating "internal" RIPng routes (routes for networks within the RIPng routing domain) from "external" RIPng routes, which may have been imported from an EGP or another IGP.

Packets:

- *RTEs sent by TR2 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_2	1	0x2
PRFX_4	1	0x2
PRFX_5	2	0x2
PRFX_8	1	0x2

- *RTEs sent by TR4 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_4	1	0x4
PRFX_5	1	0x4
PRFX_8	2	0x4

Procedure:

1. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255.
2. **TR2** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR2 and the destination address is the RIPng Multicast group FF02::9.
3. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_2, PRFX_4, PRFX_5 and PRFX_8.
4. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_2, PRFX_4, PRFX_5 and PRFX_8.
5. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255.
6. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_2, PRFX_4, PRFX_5 and PRFX_8.
7. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_2, PRFX_4, PRFX_5 and PRFX_8.

Observable Results:

- **Step 1, 2 & 5:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.
- **Step 3:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_2	2	**0x2**
PRFX_4	2	**0x4**
PRFX_5	2	**0x4**
PRFX_8	2	**0x2**

- **Step 4:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_2	2	**0x2**
PRFX_4	2	**0x4**
PRFX_5	2	**0x4**
PRFX_8	2	**0x2**

- **Step 6:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777; The response should be the same than in Step 3 since nothing should have been modified in the routing table of the RUT.

IPv6 Prefix	Metric	Route Tag
PRFX_2	2	0x2
PRFX_4	2	0x4
PRFX_5	2	0x4
PRFX_8	2	0x2

- **Step 7:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777: The response should be the same than in Step 3 since nothing should have been modified in the routing table of the RUT.

IPv6 Prefix	Metric	Route Tag
PRFX_2	2	0x2
PRFX_4	2	0x4
PRFX_5	2	0x4
PRFX_8	2	0x2

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response <----------	TR4
TR2	RIPng Response ---------->	2		
TR1	RIPng Request (PRFX_i with i=2,4,5,8) ---------->	3		
TR1	RIPng Response <----------	Step3		
		4	RIPng Request (PRFX_i with i=2,4,5,8) <----------	TR3
		Step4	RIPng Response ---------->	TR3
		5	RIPng Response <----------	TR4
TR1	RIPng Request (PRFX_i with i=2,4,5,8) ---------->	6		
TR1	RIPng Response <----------	Step6		
		7	RIPng Request (PRFX_i with i=2,4,5,8) <----------	TR3
		Step7	RIPng Response ---------->	TR3

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT by sending a **RIPng Response** packet from **TR4** and **TR2** in which all Metrics are set to infinity(16).

5.2.4 Update for a tierce router

5.2.4.1 Metric Update with Next Hop RTE field

Purpose: Check that route with greater or equal metrics are not taken into account if the Next Hop field in the Routing Table is different from the Sender of this Response and even if the Next Hop field in this Response is the same as in the Routing Table.

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

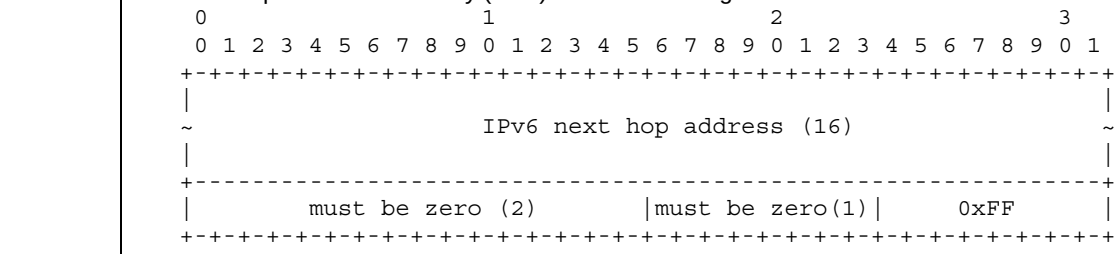
- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

RIPng provides the ability to specify the immediate next hop IPv6 address to which packets to a destination specified by a route table entry (RTE) should be forwarded in much the same way as RIP-2. In RIP-2, each route table entry has a next hop field. Including a next hop field for each RTE in RIPng would nearly double the size of the RTE. Therefore, in RIPng, the next hop is specified by a special RTE and applies to all of the address RTEs following the next hop RTE until the end of the message or until another next hop RTE is encountered.

A next hop RTE is identified by a value of 0xFF in the metric field of an RTE. The prefix field specifies the IPv6 address of the next hop. The route tag and prefix length in the next hop RTE must be set to zero on sending and ignored on receiption.

The next hop Route Table Entry (RTE) has the following format:



According to **[RFC2080]**, If a RIPng router receives a Response from a router for an existing route, it has to compare the next hop address to the address of the router from which the datagram came. If this datagram is not from the same router as the existing route, it has to compare the metrics. The route has to be updated only if the new metric is lower than the old one. As a consequence if a tierce router (TR3) uses the Next Hop field to update a route for a certain router TR4, this update will be taken into account only if the metric is lower than the old one provided by this routeur.

Procedure:

1. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR3 and the destination address is the RIPng Multicast group FF02::9.

IPv6 Prefix	Metric	Route Tag
PRFX_4	**1**	0
PRFX_5	**2**	0
PRFX_8	**3**	0

2. **TR3** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR3 and the destination address is the RIPng Multicast group FF02::9.

Next Hop	*TR4 Link-local Address*	
Ipv6 Prefix	*Metric*	*Route Tag*
PRFX_4	**2**	0
PRFX_5	**2**	0
Next Hop	TR4 Global Address	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_3	1	0
Next Hop	RUT Link-local address Link2	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_11	1	0
Next Hop	*TR4 Link-local Address*	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_8	**2**	0
Next Hop	0:0:0:0:0:0:0:0	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_6Bone	7	0

3. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.

4. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.

5. **TR1** sends an **Echo Request** through the RUT to:
   (a) PRFX_4::1
   (b) PRFX_5::1
   (c) PRFX_8::1

   The Source Address is TR1 Global Address.

6. **TR3** sends an **Echo Request** through the RUT to:
   (a) PRFX_4::1
   (b) PRFX_5::1
   (c) PRFX_8::1

   The Source Address is TR3 Global Address.

Observable Results:

- **Step 1:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent. The RUT should now have the following routing fields:

IPv6 Prefix	Metric	Next HOP	Route Tag
PRFX_4	**2**	TR4	0
PRFX_5	**3**	TR4	0
PRFX_8	**4**	TR4	0

- **Step 2:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent. The RUT should now have the following routing fields:

IPv6 Prefix	Metric	Next HOP	Route Tag
PRFX_4	**2**	TR4	0
PRFX_5	**3**	TR4	0
PRFX_8	**3**	TR4	0
PRFX_3	2	TR3	0
PRFX_10	1	Direct	0
PRFX_11	1	Direct	0
PRFX_6Bone	8	TR3	0

- **Step 3:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	**2**	0
PRFX_5	**3**	0
PRFX_8	**3**	0

- **Step 4:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	**2**	0
PRFX_5	**3**	0
PRFX_8	**3**	0

- **Step 5**: The Echo Request for PRFX_4::1, PRFX_5::1 and PRFX_8::1 have to be forwarded to TR4.
- **Step 6**: The Echo Request for PRFX_4::1, PRFX_5::1 and PRFX_8::1 have to be forwarded to TR4.

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response <----------	TR3
		2	RIPng Response <----------	TR3
TR1	RIPng Request (PRFX_i with i=4,5,8) ---------->	3		
	RIPng Response <----------	Step3		
		4	RIPng Request (PRFX_i with i=4,5,8) <----------	TR4
		Step4	RIPng Response ---------->	TR4
TR1	Echo Request (PRFX_3::1) ---------->	5(a)		
		Step5(a)	Echo Request (PRFX_3::1) ---------->	TR4
TR1	Echo Request (PRFX_5::1) ---------->	5(b)		
		Step5(b)	Echo Request (PRFX_5::1) ---------->	TR4
TR1	Echo Request (PRFX_8::1) ---------->	5(c)		
		Step5(c)	Echo Request (PRFX_8::1) ---------->	TR4
		6(a)	Echo Request (PRFX_3::1) <----------	TR3
		Step6(a)	Echo Request (PRFX_3::1) ---------->	TR4
		6(b)	Echo Request (PRFX_5::1) <----------	TR3
		Step6(b)	Echo Request (PRFX_5::1) ---------->	TR4
		6(c)	Echo Request (PRFX_8::1) <----------	TR3
		Step6(c)	Echo Request (PRFX_8::1) ---------->	TR4

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT by sending a **RIPng Response** packet from **TR3** and **TR4** in which all Metrics are set to infinity(16).

5.2.4.2 Tag Update with Next Hop RTE field

Purpose: Check that route (mainly the TAG) with greater or equal metrics are not taken into account if the Next Hop field in the Routing Table is different from the Sender of this Response and even if the Next Hop field in this Response is the same as in the Routing Table.

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

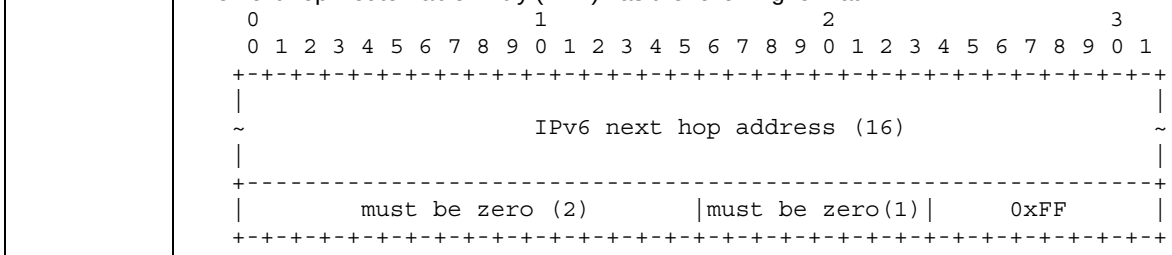
- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

RIPng provides the ability to specify the immediate next hop IPv6 address to which packets to a destination specified by a route table entry (RTE) should be forwarded in much the same way as RIP-2. In RIP-2, each route table entry has a next hop field. Including a next hop field for each RTE in RIPng would nearly double the size of the RTE. Therefore, in RIPng, the next hop is specified by a special RTE and applies to all of the address RTEs following the next hop RTE until the end of the message or until another next hop RTE is encountered.

A next hop RTE is identified by a value of 0xFF in the metric field of an RTE. The prefix field specifies the IPv6 address of the next hop. The route tag and prefix length in the next hop RTE must be set to zero on sending and ignored on receiption.

The next hop Route Table Entry (RTE) has the following format:



According to **[RFC2080]**, If a RIPng router receives a Response from a router for an existing route, it has to compare the next hop address to the address of the router from which the datagram came. If this datagram is not from the same router as the existing route, it has to compare the metrics. The route has to be updated only if the new metric is lower than the old one. As a consequence if a tierce router (TR3) uses the Next Hop field to update a route for a certain router TR4, this update will be taken into account only if the metric is lower than the old one provided by this routeur.

Procedure:

1. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR3 and the destination address is the RIPng Multicast group FF02::9.

IPv6 Prefix	Metric	Route Tag
PRFX_4	**1**	0x41
PRFX_5	**2**	0x41
PRFX_8	**3**	0x41

2. **TR3** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR3 and the destination address is the RIPng Multicast group FF02::9.

Next Hop	*TR4 Link-local Address*	
Ipv6 Prefix	*Metric*	*Route Tag*
PRFX_4	**2**	0x31
PRFX_5	**2**	0x31
Next Hop	TR4 Global Address	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_3	3	0x31
Next Hop	RUT Link-local address Link1	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_11	1	0
Next Hop	*TR4 Link-local Address*	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_8	**2**	0x31
Next Hop	0:0:0:0:0:0:0:0	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_6Bone	7	0x31

3. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.
4. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.

Observable Results:

- **Step 1:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent. The RUT should now have the following routing fields:

IPv6 Prefix	Metric	Next HOP	Route Tag
PRFX_4	**2**	TR4	**0x41**
PRFX_5	**3**	TR4	**0x41**
PRFX_8	**4**	TR4	**0x41**

- **Step 2:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent. The RUT should now have the following routing fields:

IPv6 Prefix	Metric	Next HOP	Route Tag
PRFX_4	**2**	TR4	**0x41**
PRFX_5	**3**	TR4	**0x41**
PRFX_8	**3**	TR4	**0x31**
PRFX_3	4	TR3	0x31
PRFX_10	1	Direct	0
PRFX_11	1	Direct	0
PRFX_6Bone	5	TR3	0x31

- **Step 3:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	**2**	**0x41**
PRFX_5	**3**	**0x41**
PRFX_8	**3**	**0x31**

- **Step 4:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	**2**	**0x41**
PRFX_5	**3**	**0x41**
PRFX_8	**3**	**0x31**

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response <----------	TR3
		2	RIPng Response <----------	TR3
TR1	RIPng Request (PRFX_i with i=4,5,8) ---------->	3		
	RIPng Response <----------	Step3		
		4	RIPng Request (PRFX_i with i=4,5,8) <----------	TR3
		Step4	RIPng Response ---------->	TR3

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT by sending a **RIPng Response** packet from **TR3** and **TR4** in which all Metrics are set to infinity(16).

5.2.5 Invalid RTEs

5.2.5.1 Incorrect Metric in RTEs

Purpose: Check that routes with Incorrect Metric in RTEs are silently discarded

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

The goal of this test is to check that routes with Incorrect Metric in RTEs are silently discarded. Valid entries are neverteless taken into account.

Incorrect metrics and other format errors usually indicate misbehaving neighbors and should probably be brought to the administrator's attention. For example, if the metric is greater than infinity, ignore the entry but log the event. The basic validation tests are:

- is the destination prefix valid (e.g., not a multicast prefix and not a link-local address) A link-local address should never be present in an RTE.
- is the prefix length valid (i.e., between 0 and 128, inclusive)
- is the metric valid (i.e., between 1 and 16, inclusive)

If any check fails, ignore that entry and proceed to the next.

Packets:

1. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. 521).
   The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9.
   The Hop limit has to be set to 255.

IPv6 Prefix	Metric	Route Tag
PRFX_4	1	0x41
PRFX_5	1	0x41
PRFX_8	2	0x41

2. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. 521).
   The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9.
   The Hop limit has to be set to 255.

IPv6 Prefix	Metric	Route Tag
PRFX_4	17	0x42
PRFX_5	255	0x42

PRFX_8	0	0x42

3. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.
4. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.

Observable Results:

- **Step 1:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.
- **Step 2:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because the check of all the other RTEs fails, no triggered update should be generated
- **Step 3:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	2	0x41
PRFX_5	2	0x41
PRFX_8	3	0x41

- **Step 4:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	2	**0x41**
PRFX_5	2	**0x41**
PRFX_8	3	**0x41**

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response <----------	TR4
		2	RIPng Response <----------	TR4
TR1	RIPng Request (PRFX_i with i=4,5,8) ---------->	3		
TR1	RIPng Response <----------	Step3		
		4	RIPng Request (PRFX_i with i=4,5,8) <----------	TR3
		Step4	RIPng Response ---------->	TR3

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT by sending a **RIPng Response** packet from **TR4** in which all Metrics are set to infinity(16).

5.2.5.2 Incorrect Response Packet

Purpose: Check that incorrect Response packet do not update the Routing Table on the NUT

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.
- Add a second Link-local address on NUT's Link1

Discussion:

The goal of this test is to check that incorrect Response packet do not update the Routing Table on the NUT. Incorrect metrics and other format errors usually indicate misbehaving neighbors and should probably be brought to the administrator's attention. For example, if the metric is greater than infinity, ignore the entry but log the event. The basic validation tests are:

- is the destination prefix valid (e.g., not a multicast prefix and not a link-local address) A link-local address should never be present in an RTE.
- is the prefix length valid (i.e., between 0 and 128, inclusive)
- is the metric valid (i.e., between 1 and 16, inclusive)

If any check fails, ignore that entry and proceed to the next. Again, logging the error is probably a good idea.

Packets:

- *RTEs sent by TR4 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_4	1	0x4
PRFX_5	1	0x4
PRFX_8	2	0x4

- *RTEs sent by TR2 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_4	1	0x2
PRFX_5	2	0x2
PRFX_8	1	0x2

Procedure:

1. TR4 sends a RIPng Response to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. 521).
   The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9.
   The Hop limit has to be set to 255.
2. Send each of the following invalid Response Packet from TR2:

- *Part A: Incorrect Hop Limit*

  TR2 sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR2 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to **254**.

- *Part B: Incorrect UDP Source Port*

  TR2 sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP port source is 777 and the destination port is RIPng port (ie. 521). The source address is the Link-local address of TR2 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255.

- *Part C: Incorrect UDP Destination Port*

  TR2 sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP port source is the RIPng Port (ie. 521) and the destination port is 777. The source address is the Link-local address of TR2 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255.

- *Part D: Incorrect Source Address*

  TR2 sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255. The source address is :

  (1) the **Global Address of TR2**

  (2) the **Multicast Solicited Address of TR2**

  (3) **FF02::1 (Multicast Address All hosts on the link)**

  (4) **The Global Address of the RUT on link1**

  (5) **The Link-local Address of the RUT on link1**

  (6) **The second Link-local Address of the RUT on link1**

  (7) **The Global Address of the RUT on link2**

3. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.
4. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.
5. **TR4** sends a **RIPng Response** to the RUT to clear the route table entries for PRFX_4, PRFX_5 and PRFX_8. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the same as in Step 1 and the destination address is the RIPng Multicast group FF02::9. The Hop limit is set to **255** and the metric is set to **16** for each routing entry.

Observable Results:

- **Step 1:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.
- **Step 2:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. When routes are created, an update has also to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent. Because all Response Packet are invalid the RUT has to silently discard them. As a consequence no Triggered Update Should Be sent on Link1 and Link2.
- **Step 3:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	2	0x4
PRFX_5	2	0x4

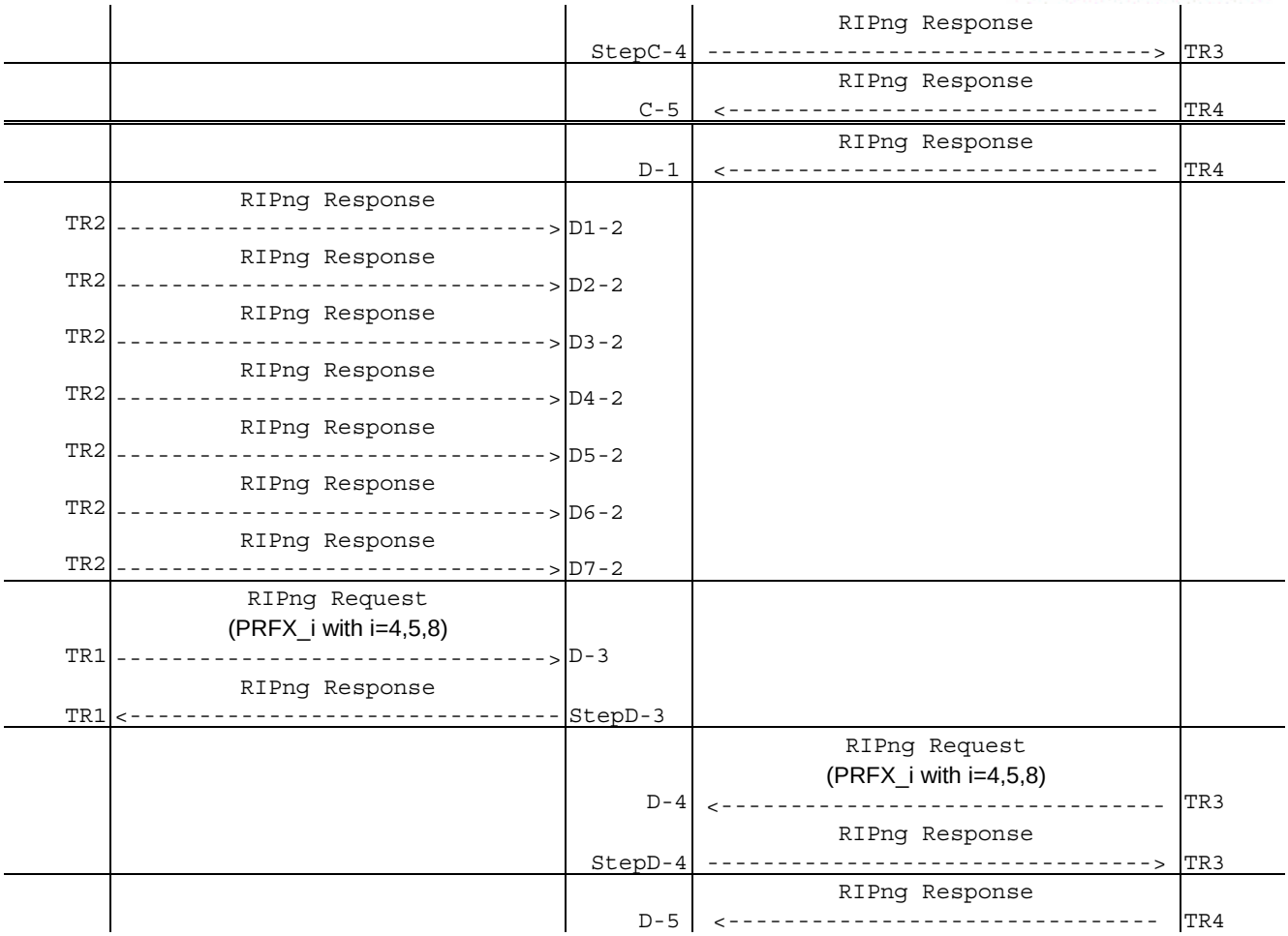
PRFX_8	3	0x4

- **Step 4:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	2	0x4
PRFX_5	2	0x4
PRFX_8	3	0x4

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		A-1	RIPng Response <----------	TR4
TR2	RIPng Response ---------->	A-2		
TR1	RIPng Request (PRFX_i with i=4,5,8) ---------->	A-3		
TR1	RIPng Response <----------	StepA-3		
		A-4	RIPng Request (PRFX_i with i=4,5,8) <----------	TR3
		StepA-4	RIPng Response ---------->	TR3
		A-5	RIPng Response <----------	TR4
		B-1	RIPng Response <----------	TR4
TR2	RIPng Response ---------->	B-2		
TR1	RIPng Request (PRFX_i with i=4,5,8) ---------->	B-3		
TR1	RIPng Response <----------	StepB-3		
		B-4	RIPng Request (PRFX_i with i=4,5,8) <----------	TR3
		StepB-4	RIPng Response ---------->	TR3
		B-5	RIPng Response <----------	TR4
		C-1	RIPng Response <----------	TR4
TR2	RIPng Response ---------->	C-2		
TR2	ICMPv6 Error Port Unreachable ---------->	StepC-2		
TR1	RIPng Request (PRFX_i with i=4,5,8) ---------->	C-3		
TR1	RIPng Response <----------	StepC-3		
		C-4	RIPng Request (PRFX_i with i=4,5,8) <----------	TR3



Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT by sending a **RIPng Response** packet from **TR4** in which all Metrics are set to infinity(16).

5.2.6 Bouncing Back and Force Route

5.2.6.1 Route Update with same metric

Purpose: Check that a new route with the same metric as the old one is updated if it is at least halfway to the expiration point. This heuristic is optional, but highly recommended. Otherwise the route do not have to be updated.

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

If there is an existing route, compare the next hop address to the address of the router from which the datagram came. If this datagram is from the same router as the existing route, reinitialize the timeout. Next, compare the metrics. If the datagram is from the same router as the existing route, and the new metric is different than the old one; or, if the new metric is lower than the old one; do the following actions:

- Adopt the route from the datagram. That is, put the new metric in, and adjust the next hop address (if necessary).
- Set the route change flag and signal the output process to trigger an update.
- If the new metric is infinity, start the deletion process; otherwise, re-initialize the timeout.

If the new metric is infinity, the deletion process begins for the route, which is no longer used for routing packets. Note that the deletion process is started only when the metric is first set to infinity. If the metric was already infinity, then a new deletion process is not started.

If the new metric is the same as the old one, it is simplest to do nothing further (beyond reinitializing the timeout, as specified above); but, there is a heuristic which could be applied. Normally, it is senseless to replace a route if the new route has the same metric as the existing route; this would cause the route to bounce back and forth, which would generate an intolerable number of triggered updates. However, if the existing route is showing signs of timing out, it may be better to switch to an equally-good alternative route immediately, rather than waiting for the timeout to happen. Therefore, if the new metric is the same as the old one, examine the timeout for the existing route. If it is at least halfway to the expiration point, switch to the new route. This heuristic is optional, but highly recommended.

Packets:

- *RTEs sent by TR3 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_3	1	0x3

- *RTEs sent by TR4 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_3	1	0x4

Procedure:

1. **TR3** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR3 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255
2. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255.
3. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_3.
4. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_3.
5. **TR1** sends an **Echo Request** through the RUT to PRFX_3::1. The Source Address is TR1 Global Address.
6. **TR3** sends an **Echo Request** through the RUT to PRFX_3::1. The Source Address is TR3 Global Address.
7. **TR3** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR3 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255
8. **Wait 90 seconds.**
9. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255.
10. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
    The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_3.
11. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
    The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_3.
12. **TR1** sends an **Echo Request** through the RUT to PRFX_3::1. The Source Address is TR1 Global Address.
13. **TR3** sends an **Echo Request** through the RUT to PRFX_3::1. The Source Address is TR3 Global Address.

Observable Results:

- **Step 1, 2, 7, 9:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.
- **Step 3:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 2, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_3	2	0x3

  Indeed, because the Metric was the same in both responses sent by TR3 and TR4, only the first update sent by TR3 has to be taken into account.

- **Step 4:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_3	2	0x3

  Indeed, because the Metric was the same in both responses sent by TR3 and TR4, only the first update sent by TR3 has to be taken into account.

- **Step 5**: The Echo Request for PRFX_3::1 SHOULD be forwarded to TR3.
- **Step 6**: The Echo Request for PRFX_3::1 SHOULD be forwarded to TR3.
- **Step 8:** Because the timeout is 180 seconds. After 90 seconds, If it is at least halfway to the expiration point; as a consequence it is recommended to switch to a new route announced if this one has the same Metric.
- **Step 10:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 2, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_3	2	0x4

Indeed, because the Metric was the same in both responses sent by TR3 and TR4, only the first update sent by TR3 has to be taken into account.

- **Step 11:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_3	2	0x4

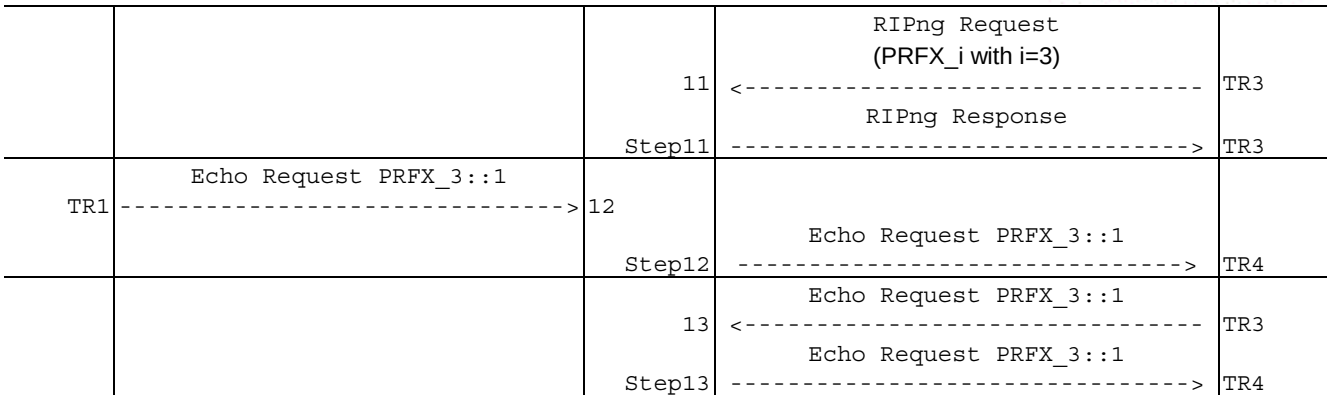
- **Step 12**: The Echo Request for PRFX_3::1 SHOULD be forwarded to TR4.
- **Step 13**: The Echo Request for PRFX_3::1 SHOULD be forwarded to TR4.

Test Sequence:

Tester (Tri)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response <----------	TR3
		2	RIPng Response <----------	TR4
TR1	RIPng Request (PRFX_i with i=3) ---------->	3		
TR1	RIPng Response <----------	Step3		
		4	RIPng Request (PRFX_i with i=3) <----------	TR3
		Step4	RIPng Response ---------->	TR3
TR1	Echo Request PRFX_3::1 ---------->	5		
		Step5	Echo Request PRFX_3::1 ---------->	TR3
		6	Echo Request PRFX_3::1 <----------	TR3
		Step6	Echo Request PRFX_3::1 ---------->	TR3
		7	RIPng Response <----------	TR3

8: Wait 90 seconds

		9	RIPng Response <----------	TR4
TR1	RIPng Request (PRFX_i with i=3) ---------->	10		
TR1	RIPng Response <----------	Step10		


RIPng Request
(PRFX_i with i=3)
11
TR3
RIPng Response
Step11
TR3
Echo Request PRFX_3::1
TR1
12
Echo Request PRFX_3::1
Step12
TR4
Echo Request PRFX_3::1
13
TR3
Echo Request PRFX_3::1
Step13
TR4

5.3 RTEs Deletion

5.3.1 Infinite Metric

Purpose: Check that RTEs are deleted when metric are set to infinite

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

If there is an existing route, compare the next hop address to the address of the router from which the datagram came. If this datagram is from the same router as the existing route, reinitialize the timeout. Next, compare the metrics. If the datagram is from the same router as the existing route, and the new metric is different than the old one; or, if the new metric is lower than the old one; do the following actions:

- Adopt the route from the datagram. That is, put the new metric in, and adjust the next hop address (if necessary).
- Set the route change flag and signal the output process to trigger an update.
- If the new metric is infinity, start the deletion process; otherwise, re-initialize the timeout.

If the new metric is infinity, the deletion process begins for the route, which is no longer used for routing packets. Note that the deletion process is started only when the metric is first set to infinity. If the metric was already infinity, then a new deletion process is not started.

Procedure:

1. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255. The RTEs included in the Response are the following:

IPv6 Prefix	Metric
PRFX_4	1
PRFX_5	1
PRFX_8	2

2. **TR1** sends an **Echo Request** to:
   (a) PRFX_4::1
   (b) PRFX_5::1
   (c) PRFX_8::1
3. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255. The RTEs included in the Response are the following:

IPv6 Prefix	Metric	Route Tag
PRFX_4	2	0
PRFX_5	**15**	0
PRFX_8	**16**	0

4. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.
5. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.
6. **TR1** sends an **Echo Request** to:
   (a) PRFX_4::1
   (b) PRFX_5::1
   (c) PRFX_8::1
7. **Wait 120 seconds. The Garbage-collection timer will have expired.**
8. **TR1** sends an **Echo Request** to:
   (a) PRFX_4::1
   (b) PRFX_5::1
   (c) PRFX_8::1

Observable Results:

- **Step 1 & 3:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.
- **Step 2**: The Echo Request for PRFX_4::1, PRFX_5::1 and PRFX_8::1 have to be forwarded to TR4.
- **Step 4:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	3	0
PRFX_5	**16**	0
PRFX_8	**16**	0

- **Step 5:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	3	0
PRFX_5	**16**	0
PRFX_8	**16**	0

- **Step 6**: The Echo Request for PRFX_4::1, PRFX_5::1 and PRFX_8::1 have to be forwarded to TR4 since the garbage-collection timer will have not expired yet

- **Step 7**: The garbage-collection timer will have completely expired but not the timeout for PRFX_4::1

- **Step 8**: The Echo Request for PRFX_4::1 has to be forwarded to TR4. The RUT MUST also send two ICMPv6 Route Unreachable packets containing the Echo requests for PRFX_5::1 and for PRFX_8::1.

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response <----------	TR4
TR1	Echo Request (PRFX_4::1) ---------->	2(a)		
		Step2(a)	Echo Request (PRFX_4::1) ---------->	TR4
TR1	Echo Request (PRFX_5::1) ---------->	2(b)		
		Step2(b)	Echo Request (PRFX_5::1) ---------->	TR4
TR1	Echo Request (PRFX_8::1) ---------->	2(c)		
		Step2(c)	Echo Request (PRFX_8::1) ---------->	TR4
		3	RIPng Response <----------	TR4
TR1	RIPng Request (PRFX_i with i=4,5,8) ---------->	4		
TR1	RIPng Response <----------	Step4		
		5	RIPng Request (PRFX_i with i=4,5,8) <----------	TR3
		Step5	RIPng Response ---------->	TR3
TR1	Echo Request (PRFX_4::1) ---------->	6(a)		
		Step6(a)	Echo Request (PRFX_4::1) ---------->	TR4
TR1	Echo Request (PRFX_5::1) ---------->	6(b)		
		Step6(b)	Echo Request (PRFX_5::1) ---------->	TR4
TR1	Echo Request (PRFX_8::1) ---------->	6(c)		
		Step6(c)	Echo Request (PRFX_8::1) ---------->	TR4

7：Wait 120 seconds. The Garbage-collection timer will have expired.

TR1	Echo Request (PRFX_4::1) ---------->	8(a)		
		Step8(a)	Echo Request (PRFX_4::1) ---------->	TR4
TR1	Echo Request (PRFX_5::1) ---------->	8(b)		
TR1	ICMPv6 Route Unreachable <----------	Step8(b)		
TR1	Echo Request (PRFX_8::1) ---------->	8(c)		
TR1	ICMPv6 Route Unreachable <----------	Step8(c)		

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT by sending a **Ripng Response** packet from **TR4** in which all Metrics are set to infinity(16).

5.3.2 Infinite Metric in Next Hop RTE sent from same router

Purpose: Check that RTEs are deleted when metric are set to infinite using RIPng response with Next HOP RTE Field. This response is sent by the router involved in the routing of these particular addresses

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

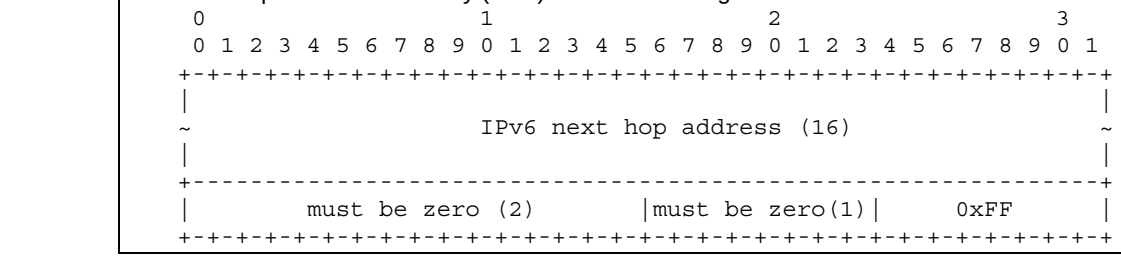
- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

RIPng provides the ability to specify the immediate next hop IPv6 address to which packets to a destination specified by a route table entry (RTE) should be forwarded in much the same way as RIP-2. In RIP-2, each route table entry has a next hop field. Including a next hop field for each RTE in RIPng would nearly double the size of the RTE. Therefore, in RIPng, the next hop is specified by a special RTE and applies to all of the address RTEs following the next hop RTE until the end of the message or until another next hop RTE is encountered.

A next hop RTE is identified by a value of 0xFF in the metric field of an RTE. The prefix field specifies the IPv6 address of the next hop. The route tag and prefix length in the next hop RTE must be set to zero on sending and ignored on receiption.

The next hop Route Table Entry (RTE) has the following format:



Procedure:

1. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. RTEs sent by TR4 are:

IPv6 Prefix	Metric	Route Tag
PRFX_4	**1**	0
PRFX_5	**1**	0
PRFX_8	**2**	0

2. **TR1** sends an **Echo Request** to:
   (a) PRFX_4::1
   (b) PRFX_5::1
   (c) PRFX_8::1
3. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255. RTEs sent by TR4 in Ripng Response are the following in the following order:

Next Hop	*TR3 Link-local Address*	
Ipv6 Prefix	*Metric*	*Route Tag*
PRFX_4	**2**	0
PRFX_5	**15**	0
Next Hop	TR4 Global Address	
Ipv6 Prefix	*Metric*	*Route Tag*
PRFX_3	1	0
Next Hop	RUT Link-local address Link1	
Ipv6 Prefix	*Metric*	*Route Tag*
PRFX_11	1	0
Next Hop	*TR4 Link-local Address*	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_8	**16**	0
Next Hop	0:0:0:0:0:0:0:0	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_6Bone	7	0

4. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.
5. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.
6. **TR1** sends an **Echo Request** to:
   (a) PRFX_4::1
   (b) PRFX_5::1
   (c) PRFX_8::1
7. **Wait 120 seconds. The Garbage-collection timer will have expired.**
8. **TR1** sends an **Echo Request** to:
   (a) PRFX_4::1
   (b) PRFX_5::1
   (c) PRFX_8::1

Observable Results:

- **Step 1 & 3:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.
- **Step 2**: The Echo Request for PRFX_4::1, PRFX_5::1 and PRFX_8::1 have to be forwarded to TR4.
- **Step 4:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	**Route Tag**
PRFX_4	2	0
PRFX_5	**16**	0
PRFX_8	**16**	0

- **Step 5:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	2	0
PRFX_5	**16**	0
PRFX_8	**16**	0

- **Step 6**: The Echo Request for PRFX_4::1, PRFX_5::1 and PRFX_8::1 have to be forwarded to TR4 since the garbage-collection timer will have not expired yet

- **Step 7**: The garbage-collection timer will have completely expired but not the timeout for PRFX_4::1

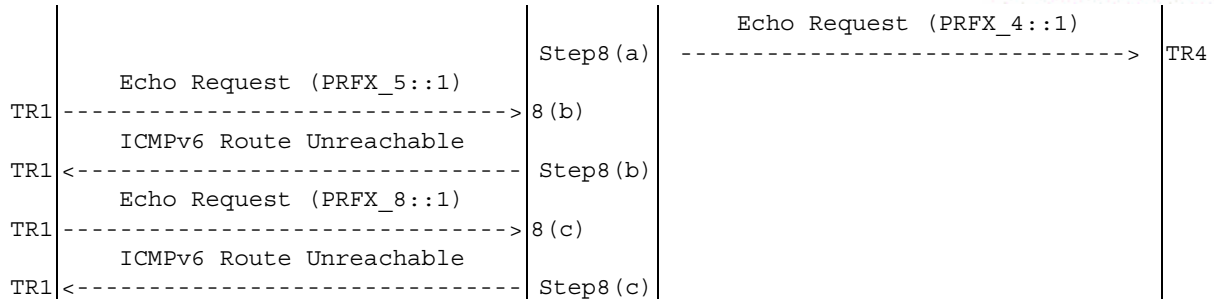
- **Step 8**: The Echo Request for PRFX_4::1 has to be forwarded to TR4. The RUT MUST also send two ICMPv6 Route Unreachable packets containing the Echo requests for PRFX_5::1 and for PRFX_8::1.

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response <----------	TR4
TR1	Echo Request (PRFX_4::1) ---------->	2(a)		
		Step2(a)	Echo Request (PRFX_4::1) ---------->	TR4
TR1	Echo Request (PRFX_5::1) ---------->	2(b)		
		Step2(b)	Echo Request (PRFX_5::1) ---------->	TR4
TR1	Echo Request (PRFX_8::1) ---------->	2(c)		
		Step2(c)	Echo Request (PRFX_8::1) ---------->	TR4
		3	RIPng Response <----------	TR4
TR1	RIPng Request (PRFX_i with i=4,5,8) ---------->	4		
TR1	RIPng Response <----------	Step4		
		5	RIPng Request (PRFX_i with i=4,5,8) <----------	TR3
		Step5	RIPng Response ---------->	TR3
TR1	Echo Request (PRFX_4::1) ---------->	6(a)		
		Step6(a)	Echo Request (PRFX_4::1) ---------->	TR4
TR1	Echo Request (PRFX_5::1) ---------->	6(b)		
		Step6(b)	Echo Request (PRFX_5::1) ---------->	TR4
TR1	Echo Request (PRFX_8::1) ---------->	6(c)		
		Step6(c)	Echo Request (PRFX_8::1) ---------->	TR4

7： Wait 120 seconds. The Garbage-collection timer will have expired.

TR1	Echo Request (PRFX_4::1) ---------->	8(a)		



Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT by sending a **Ripng Response** packet from **TR4** in which all Metrics are set to infinity(16).

5.3.3 Infinite Metric in Next Hop RTE sent from a tierce router

Purpose: Check that RTEs are deleted when metric are set to infinite using RIPng response with Next HOP RTE Field.

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

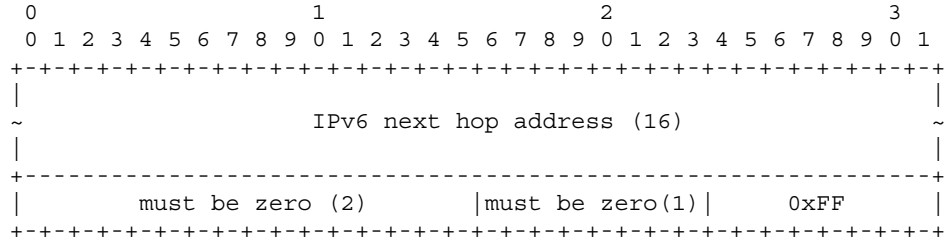
- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

RIPng provides the ability to specify the immediate next hop IPv6 address to which packets to a destination specified by a route table entry (RTE) should be forwarded in much the same way as RIP-2. In RIP-2, each route table entry has a next hop field. Including a next hop field for each RTE in RIPng would nearly double the size of the RTE. Therefore, in RIPng, the next hop is specified by a special RTE and applies to all of the address RTEs following the next hop RTE until the end of the message or until another next hop RTE is encountered.

A next hop RTE is identified by a value of 0xFF in the metric field of an RTE. The prefix field specifies the IPv6 address of the next hop. The route tag and prefix length in the next hop RTE must be set to zero on sending and ignored on receiption.

The next hop Route Table Entry (RTE) has the following format:



Packets:

- *RTEs sent by TR3 in Ripng Response are the following in the following order: :*

Next Hop	*TR4 Link-local Address*	
Ipv6 Prefix	*Metric*	*Route Tag*
PRFX_4	**2**	0
PRFX_5	**15**	0
Next Hop	TR4 Global Address	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_3	1	0
Next Hop	RUT Link-local address Link1	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_11	1	0
Next Hop	*TR4 Link-local Address*	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_8	**16**	0

Next Hop	0:0:0:0:0:0:0:0	
IPv6 Prefix	*Metric*	*Route Tag*
PRFX_6Bone	7	0x3

- *RTEs sent by TR4 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_4	**1**	0
PRFX_5	**1**	0
PRFX_8	**2**	0

Procedure:

1. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9.
2. **TR1** sends an **Echo Request** to:
   (a) PRFX_4::1
   (b) PRFX_5::1
   (c) PRFX_8::1
3. **TR3** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR3 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255.
4. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.
5. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.
6. **TR1** sends an **Echo Request** to:
   (a) PRFX_4::1
   (b) PRFX_5::1
   (c) PRFX_8::1
7. **Wait 120 seconds. The Garbage-collection timer will have expired.**
8. **TR1** sends an **Echo Request** to:
   (a) PRFX_4::1
   (b) PRFX_5::1
   (c) PRFX_8::1

Observable Results:

- **Step 1 & 3:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.
- **Step 2**: The Echo Request for PRFX_4::1, PRFX_5::1 and PRFX_8::1 have to be forwarded to TR4.
- **Step 4:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	3	0
PRFX_5	**16**	0
PRFX_8	**16**	0

- **Step 5:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	**Route Tag**
PRFX_4	3	0
PRFX_5	**16**	0
PRFX_8	**16**	0

- **Step 6**: The Echo Request for PRFX_4::1, PRFX_5::1 and PRFX_8::1 have to be forwarded to TR4 since the garbage-collection timer will have not expired yet

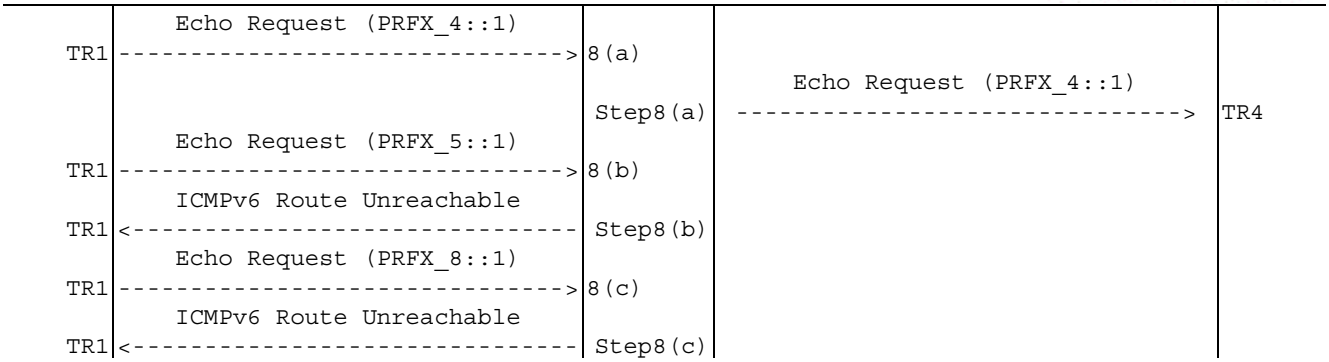
- **Step 7**: The garbage-collection timer will have completely expired but not the timeout for PRFX_4::1

- **Step 8**: The Echo Request for PRFX_4::1 has to be forwarded to TR4. The RUT MUST also send two ICMPv6 Route Unreachable packets containing the Echo requests for PRFX_5::1 and for PRFX_8::1.

Test Sequence:

Tester (TRi)	**Link1**	**RUT**	**Link2**	**Tester (TRi)**
		1	RIPng Response <----------	TR4
TR1	Echo Request (PRFX_4::1) ---------->	2(a) Step2(a)	Echo Request (PRFX_4::1) ---------->	TR4
TR1	Echo Request (PRFX_5::1) ---------->	2(b) Step2(b)	Echo Request (PRFX_5::1) ---------->	TR4
TR1	Echo Request (PRFX_8::1) ---------->	2(c) Step2(c)	Echo Request (PRFX_8::1) ---------->	TR4
		3	RIPng Response <----------	TR3
TR1	RIPng Request (PRFX_i with i=4,5,8) ---------->	4 Step4		
TR1	RIPng Response <----------			
		5	RIPng Request (PRFX_i with i=4,5,8) <----------	TR3
		Step5	RIPng Response ---------->	TR3
TR1	Echo Request (PRFX_4::1) ---------->	6(a) Step6(a)	Echo Request (PRFX_4::1) ---------->	TR4
TR1	Echo Request (PRFX_5::1) ---------->	6(b) Step6(b)	Echo Request (PRFX_5::1) ---------->	TR4
TR1	Echo Request (PRFX_8::1) ---------->	6(c) Step6(c)	Echo Request (PRFX_8::1) ---------->	TR4

7： Wait 120 seconds. The Garbage-collection timer will have expired.



Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT by sending a **Ripng Response** packet from **TR4** in which all Metrics are set to infinity(16).

5.3.4 Route Lifetime

Purpose: Check that the route lifetime is 180 seconds

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

There are two timers associated with each route, a "timeout" and a "garbage-collection time." Upon expiration of the timeout, the route is no longer valid; however, it is retained in the routing table for a short time so that neighbors can be notified that the route has been dropped. Upon expiration of the garbage-collection timer, the route is finally removed from the routing table.

The timeout is initialized when a route is established, and any time an update message is received for the route. If 180 seconds elapse from the last time the timeout was initialized, the route is considered to have expired, and the deletion process described below begins for that route.

Deletions can occur for one of two reasons: the timeout expires, or the metric is set to 16 because of an update received from the current router (see section 2.4.2 for a discussion of processing updates from other routers). In either case, the following events happen:

- The garbage-collection timer is set for 120 seconds.

- The metric for the route is set to 16 (infinity). This causes the route to be removed from service.

- The route change flag is to indicate that this entry has been changed

- The output process is signalled to trigger a response.

Until the garbage-collection timer expires, the route is included in all updates sent by this router. When the garbage-collection timer expires, the route is deleted from the routing table.

Should a new route to this network be established while the garbage-collection timer is running, the new route will replace the one that is about to be deleted. In this case the garbage-collection timer must be cleared

Packets:

- *RTEs sent by TR4 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_4	1	0
PRFX_5	1	0
PRFX_8	2	0

Procedure:

1. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255.
2. **TR1** sends an **Echo Request** through the RUT to:

(a) PRFX_4::1

(b) PRFX_5::1

(c) PRFX_8::1

The Source Address is TR1 Global Address.

3. **Wait 180 seconds**
4. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.
5. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.
6. **TR1** sends an **Echo Request** through the RUT to:

   (a) PRFX_4::1

   (b) PRFX_5::1

   (c) PRFX_8::1

   The Source Address is TR1 Global Address.
7. **Wait 120 seconds. The Garbage-collection timer will have expired.**
8. **TR1** sends an **Echo Request** to:

   (a) PRFX_4::1

   (b) PRFX_5::1

   (c) PRFX_8::1

Observable Results:

- **Step 1:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.
- **Step 2**: The Echo Request for PRFX_4::1, PRFX_5::1 and PRFX_8::1 have to be forwarded to TR4.
- **Step 3**: because 180 seconds have been elapsed from the last time the timeout was initialized, the route is considered to have expired
- **Step 4:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 2, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	**16**	0
PRFX_5	**16**	0
PRFX_8	**16**	0

- **Step 5:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	**16**	0
PRFX_5	**16**	0
PRFX_8	**16**	0

- **Step 6**: The Echo Request for PRFX_4::1, PRFX_5::1 and PRFX_8::1 have to be forwarded to TR4 since the garbage-collection timer will have not expired yet

- **Step 7**: The garbage-collection timer will have completely expired.

- **Step 8**: The RUT MUST send three ICMPv6 Route Unreachable packets containing the Echo requests for PRFX_4::1, PRFX_5::1 and for PRFX_8::1.

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response <----------	TR4
TR1	Echo Request (PRFX_4::1) ---------->	2(a)		
		Step2(a)	Echo Request (PRFX_4::1) ---------->	TR4
TR1	Echo Request (PRFX_5::1) ---------->	2(b)		
		Step2(b)	Echo Request (PRFX_5::1) ---------->	TR4
TR1	Echo Request (PRFX_8::1) ---------->	2(c)		
		Step2(c)	Echo Request (PRFX_8::1) ---------->	TR4

3 : WAIT 180 Seconds

TR1	RIPng Request (PRFX_i with i=4,5,8) ---------->	4		
TR1	RIPng Response <----------	Step4		
		5	RIPng Request (PRFX_i with i=4,5,8) <----------	TR3
		Step5	RIPng Response ---------->	TR3
TR1	Echo Request (PRFX_4::1) ---------->	6(a)		
		Step6(a)	Echo Request (PRFX_4::1) ---------->	TR4
TR1	Echo Request (PRFX_5::1) ---------->	6(b)		
		Step6(b)	Echo Request (PRFX_5::1) ---------->	TR4
TR1	Echo Request (PRFX_8::1) ---------->	6(c)		
		Step6(c)	Echo Request (PRFX_8::1) ---------->	TR4

7： Wait 120 seconds. The Garbage-collection timer will have expired.

TR1	Echo Request (PRFX_4::1) ---------->	7(a)		
TR1	ICMPv6 Route Unreachable <----------	Step7(a)		
TR1	Echo Request (PRFX_5::1) ---------->	7(b)		
TR1	ICMPv6 Route Unreachable <----------	Step7(b)		
TR1	Echo Request (PRFX_8::1) ---------->	7(c)		
TR1	ICMPv6 Route Unreachable <----------	Step7(c)		

Postamble:

Because the Routing Table of the RUT is clear nothing should be done in this postamble. Nevertheless it could be required to clear all remaining routes in the RUT by sending a **RIPng Response** packet from **TR4** in which all Metrics are set to infinity(16).

5.3.5 Route Lifetime Re-initialisation

Purpose: Check that route are correctly reinitialized during the garbage-collection time. Ie. The timeout is set again for 120 seconds.

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

There are two timers associated with each route, a "timeout" and a "garbage-collection time." Upon expiration of the timeout, the route is no longer valid; however, it is retained in the routing table for a short time so that neighbors can be notified that the route has been dropped. Upon expiration of the garbage-collection timer, the route is finally removed from the routing table.

The timeout is initialized when a route is established, and any time an update message is received for the route. If 180 seconds elapse from the last time the timeout was initialized, the route is considered to have expired, and the deletion process described below begins for that route.

Deletions can occur for one of two reasons: the timeout expires, or the metric is set to 16 because of an update received from the current router (see section 2.4.2 for a discussion of processing updates from other routers). In either case, the following events happen:

- The garbage-collection timer is set for 120 seconds.

- The metric for the route is set to 16 (infinity). This causes the route to be removed from service.

- The route change flag is to indicate that this entry has been changed

- The output process is signalled to trigger a response.

Until the garbage-collection timer expires, the route is included in all updates sent by this router. When the garbage-collection timer expires, the route is deleted from the routing table.

Should a new route to this network be established while the garbage-collection timer is running, the new route will replace the one that is about to be deleted. In this case the garbage-collection timer must be cleared

Procedure:

1. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255. RTEs sent by TR4 in Ripng Response are the following :

IPv6 Prefix	Metric	Route Tag
PRFX_4	1	0x41
PRFX_5	1	0x41

2. **TR1** sends an **Echo Request** through the RUT to:
   (a) PRFX_4::1
   (b) PRFX_5::1

   The Source Address is TR1 Global Address.

3. **Wait 180 seconds: The timeout should have expired**

4. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255. RTEs sent by TR4 in Ripng Response are the following :

IPv6 Prefix	Metric	Route Tag
PRFX_5	2	0x42
PRFX_8	2	0x42

5. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.

6. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.

7. **TR1** sends an **Echo Request** through the RUT to:

   (a) PRFX_4::1

   (b) PRFX_5::1

   (c) PRFX_8::1

   The Source Address is TR1 Global Address.

8. **Wait 120 seconds. The Garbage-collection timer will have expired for PRFX_4 but not the Timeout for PRFX_5 and PRFX_8**

9. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.

10. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.

11. **TR1** sends an **Echo Request** to:

   (a) PRFX_4::1

   (b) PRFX_5::1

   (c) PRFX_8::1

12. **Wait 60 seconds. The Timeout will have expired for PRFX_5 and PRFX_8**

13. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.

14. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.

15. **TR1** sends an **Echo Request** to:

   (a) PRFX_4::1

   (b) PRFX_5::1

   (c) PRFX_8::1

16. **Wait 120 seconds. The Garbage-collection timer will have expired for PRFX_5 and PRFX_8**

17. **TR1** sends an **Echo Request** to:

   (a) PRFX_5::1

   (b) PRFX_8::1

Observable Results:

- **Step 1 & 4:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.

- **Step 2**: The Echo Request for PRFX_4::1 and PRFX_5::1 have to be forwarded to TR4.

- **Step 3**: because 180 seconds have been elapsed from the last time the timeout was initialized, the route is considered to have expired

- **Step 5:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 2, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	16	0
PRFX_5	3	0x42
PRFX_8	3	0x42

- **Step 6:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	16	0
PRFX_5	3	0x42
PRFX_8	3	0x42

- **Step 7:** The Echo Request for PRFX_4::1, PRFX_5::1 and PRFX_8::1 have to be forwarded to TR4 since the garbage-collection timer will have not expired yet

- **Step 8:** The garbage-collection timer will have completely expired for PRFX_4 but not the Timeout for PRFX_5 and PRFX_8

- **Step 9:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 2, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	16	0
PRFX_5	3	0x42
PRFX_8	3	0x42

- **Step 10:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	16	0
PRFX_5	3	0x42
PRFX_8	3	0x42

- **Step 11:** The Echo Request for PRFX_5::1 and PRFX_8::1 have to be forwarded to TR4 since the garbage-collection timer will have not expired yet. Moreover, the RUT MUST send an ICMPv6 Route Unreachable packets containing the Echo request for PRFX_4::1.

- **Step 12:** the Timeout will have completely expired for PRFX_5 and PRFX_8 and the garbage-collection timer should be running.

- **Step 13:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 2, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	16	0
PRFX_5	16	0
PRFX_8	16	0

- **Step 14:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	16	0
PRFX_5	16	0
PRFX_8	16	0

- **Step 15:** The Echo Request for PRFX_5::1 and PRFX_8::1 have to be forwarded to TR4 since the garbage-collection timer will have not expired yet. Moreover, the RUT MUST send an ICMPv6 Route Unreachable packets containing the Echo request for PRFX_4::1.

- **Step 16:** the garbage-collection timer should have completely expired for PRFX_5 and PRFX_8

- **Step 17:**::The RUT MUST send two ICMPv6 Route Unreachable packets containing the Echo requests for PRFX_5::1 and for PRFX_8::1.

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response <----------	TR4
TR1	Echo Request (PRFX_4::1) ---------->	2(a)		
		Step2(a)	Echo Request (PRFX_4::1) ---------->	TR4
TR1	Echo Request (PRFX_5::1) ---------->	2(b)		
		Step2(b)	Echo Request (PRFX_5::1) ---------->	TR4

3 : WAIT 180 Seconds. The Timeout should have expired

		4	RIPng Response <----------	TR4
TR1	RIPng Request (PRFX_i with i=4,5,8) ---------->	5		
TR1	RIPng Response <----------	Step5		
		6	RIPng Request (PRFX_i with i=4,5,8) <----------	TR3
		Step6	RIPng Response ---------->	TR3
TR1	Echo Request (PRFX_4::1) ---------->	7(a)		
		Step7(a)	Echo Request (PRFX_4::1) ---------->	TR4
TR1	Echo Request (PRFX_5::1) ---------->	7(b)		
		Step7(b)	Echo Request (PRFX_5::1) ---------->	TR4
TR1	Echo Request (PRFX_8::1) ---------->	7(c)		
			Echo Request (PRFX_8::1)	

		Step7(c)	---------->	TR4

8：Wait 120 seconds. The Garbage-collection timer will have expired for PRFX_4 but not the timeout for PRFX_5, PRFX_8.

	RIPng Request (PRFX_i with i=4,5,8)			
TR1	---------->	9		
	RIPng Response			
TR1	<----------	Step9		
			RIPng Request (PRFX_i with i=4,5,8)	
		10	<----------	TR3
			RIPng Response	
		Step10	---------->	TR3
	Echo Request (PRFX_4::1)			
TR1	---------->	11(a)		
	ICMPv6 Route Unreachable			
TR1	<----------	Step11(a)		
	Echo Request (PRFX_5::1)			
TR1	---------->	11(b)		
			Echo Request (PRFX_5::1)	
		Step11(b)	---------->	TR4
	Echo Request (PRFX_8::1)			
TR1	---------->	11(c)		
			Echo Request (PRFX_8::1)	
		Step11(c)	---------->	TR4

12 : Wait 60 seconds. The Timeout will have expired for PRFX_5 and PRFX_8

	RIPng Request (PRFX_i with i=4,5,8)			
TR1	---------->	13		
	RIPng Response			
TR1	<----------	Step13		
			RIPng Request (PRFX_i with i=4,5,8)	
		14	<----------	TR3
			RIPng Response	
		Step14	---------->	TR3
	Echo Request (PRFX_4::1)			
TR1	---------->	15(a)		
	ICMPv6 Route Unreachable			
TR1	<----------	Step15(a)		
	Echo Request (PRFX_5::1)			
TR1	---------->	15(b)		
			Echo Request (PRFX_5::1)	
		Step15(b)	---------->	TR4
	Echo Request (PRFX_8::1)			
TR1	---------->	15(c)		
			Echo Request (PRFX_8::1)	
		Step15(c)	---------->	TR4

16：Wait 120 seconds. The Garbage-collection timer will have expired for PRFX_5, PRFX_8.

	Echo Request (PRFX_5::1)			
TR1	---------->	17(a)		
	ICMPv6 Route Unreachable			
TR1	<----------	Step17(a)		
	Echo Request (PRFX_8::1)			
TR1	---------->	17(b)		
	ICMPv6 Route Unreachable			
TR1	<----------	Step17(b)		

Postamble:

Because the Routing Table of the RUT is clear nothing should be done in this postamble. Nevertheless it could be required to clear all remaining routes in the RUT by sending a **RIPng Response** packet from **TR4** in which all Metrics are set to infinity(16).

5.3.6 Update of route during the Garbage-collection Time.

Purpose: Check that route are correctly updated during the garbage-collection time.

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

There are two timers associated with each route, a "timeout" and a "garbage-collection time." Upon expiration of the timeout, the route is no longer valid; however, it is retained in the routing table for a short time so that neighbors can be notified that the route has been dropped. Upon expiration of the garbage-collection timer, the route is finally removed from the routing table.

The timeout is initialized when a route is established, and any time an update message is received for the route. If 180 seconds elapse from the last time the timeout was initialized, the route is considered to have expired, and the deletion process described below begins for that route.

Deletions can occur for one of two reasons: the timeout expires, or the metric is set to 16 because of an update received from the current router (see section 2.4.2 for a discussion of processing updates from other routers). In either case, the following events happen:

- The garbage-collection timer is set for 120 seconds.

- The metric for the route is set to 16 (infinity). This causes the route to be removed from service.

- The route change flag is to indicate that this entry has been changed

- The output process is signalled to trigger a response.

Until the garbage-collection timer expires, the route is included in all updates sent by this router. When the garbage-collection timer expires, the route is deleted from the routing table.

Should a new route to this network be established while the garbage-collection timer is running, the new route will replace the one that is about to be deleted. In this case the garbage-collection timer must be cleared

Procedure:

1. **TR2** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR2 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255. RTEs sent by TR2 in Ripng Response are the following :

IPv6 Prefix	Metric	Route Tag
PRFX_4	1	0x2
PRFX_5	2	0x2
PRFX_8	1	0x2

2. **TR1** sends an **Echo Request** through the RUT to:
   (a) PRFX_4::1
   (b) PRFX_5::1
   (c) PRFX_8::1
   The Source Address is TR1 Global Address.
3. **Wait 180 seconds: The timeout should have expired**

4. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255. RTEs sent by TR4 in Ripng Response are the following :

IPv6 Prefix	Metric	Route Tag
PRFX_4	1	0x4
PRFX_5	1	0x4
PRFX_8	2	0x4

5. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.

6. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.

7. **TR1** sends an **Echo Request** through the RUT to:

   (a) PRFX_4::1

   (b) PRFX_5::1

   (c) PRFX_8::1

   The Source Address is TR1 Global Address.

8. **Wait 120 seconds. The Garbage-collection timer if running will have expired but not the Timeout for PRFX_4, PRFX_5 and PRFX_8.**

9. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.

10. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.

11. **TR1** sends an **Echo Request** to:

   (a) PRFX_4::1

   (b) PRFX_5::1

   (c) PRFX_8::1

12. **Wait 60 seconds. The Timeout will have expired for PRFX_4, PRFX_5 and PRFX_8.**

13. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.

14. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.

15. **TR1** sends an **Echo Request** to:

   (a) PRFX_4::1

   (b) PRFX_5::1

   (c) PRFX_8::1

16. **Wait 120 seconds. The Garbage-collection timer will have expired for PRFX_4, PRFX_5 and PRFX_8**

17. **TR1** sends an **Echo Request** to:

   (a) PRFX_4::1

   (b) PRFX_5::1

(c) PRFX_8::1

Observable Results:

- **Step 1 & 4:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.
- **Step 2**: The Echo Request for PRFX_4::1, PRFX_5::1 and PRFX_8::1 have to be forwarded to TR2.
- **Step 3**: because 180 seconds have been elapsed from the last time the timeout was initialized, the route is considered to have expired
- **Step 5:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 2, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	**2**	**0x4**
PRFX_5	**2**	**0x4**
PRFX_8	**3**	**0x4**

- **Step 6:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	**2**	**0x4**
PRFX_5	**2**	**0x4**
PRFX_8	**3**	**0x4**

- **Step 7:** The Echo Request for PRFX_4::1, PRFX_5::1 and PRFX_8::1 have to be forwarded to TR4.
- **Step 8:** The garbage-collection timer will have completely expired if running.
- **Step 9:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 2, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	**2**	**0x4**
PRFX_5	**2**	**0x4**
PRFX_8	**3**	**0x4**

- **Step 10:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	**2**	**0x4**
PRFX_5	**2**	**0x4**
PRFX_8	**3**	**0x4**

- **Step 11:** The Echo Request for PRFX_4::1, PRFX_5::1 and PRFX_8::1 have to be forwarded to TR4.
- **Step 12**: because 180 seconds have been elapsed from the last time the timeout was initialized, the routes are considered to have expired
- **Step 13:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 2, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	16	0
PRFX_5	16	0
PRFX_8	16	0

- **Step 14:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	16	0
PRFX_5	16	0
PRFX_8	16	0

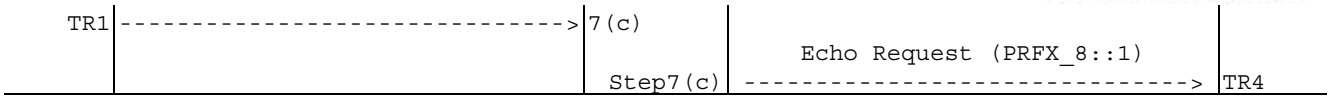
- **Step 15:** The Echo Request for PRFX_4::1, PRFX_5::1 and PRFX_8::1 have to be forwarded to TR4.
- **Step 16:** The garbage-collection timer will have completely expired PRFX_4::1, PRFX_5::1 and PRFX_8::1.
- **Step 17:**::The RUT MUST send three ICMPv6 Route Unreachable packets containing the Echo requests for PRFX_4::1, PRFX_5::1 and for PRFX_8::1.

Test Sequence:

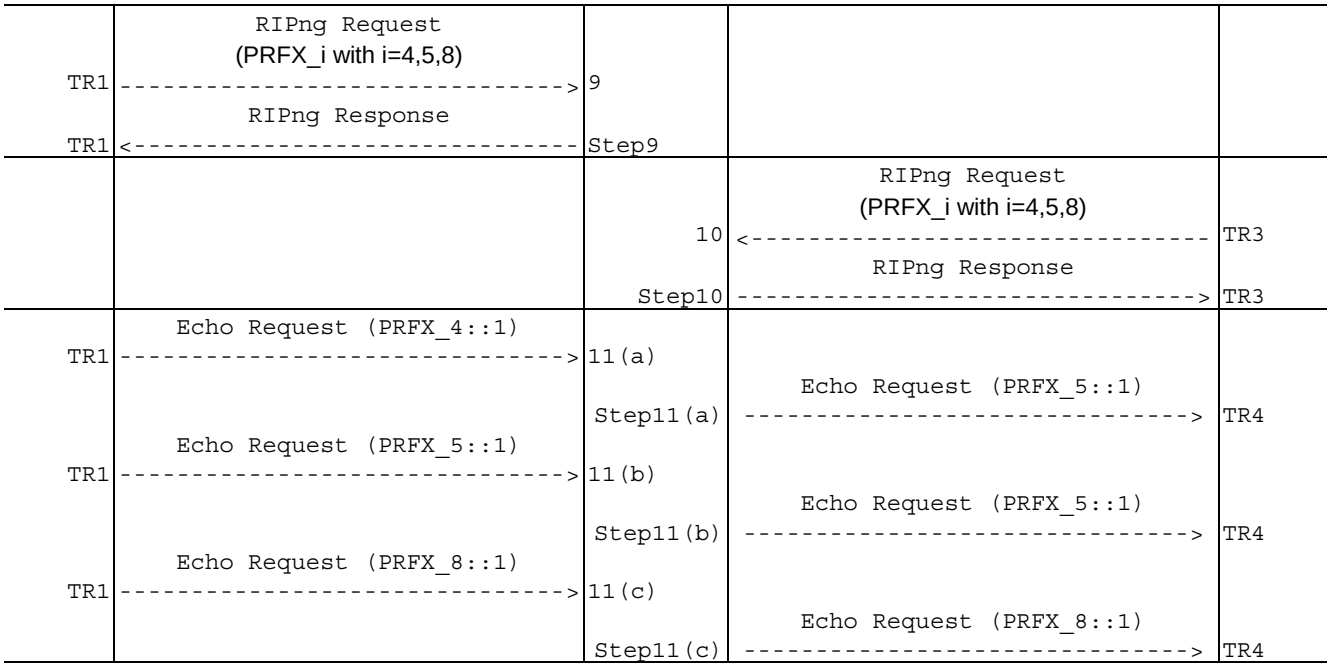
Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response <----------	TR2
TR1	Echo Request (PRFX_4::1) ---------->	2(a)		
		Step2(a)	Echo Request (PRFX_4::1) ---------->	TR4
TR1	Echo Request (PRFX_5::1) ---------->	2(b)		
		Step2(b)	Echo Request (PRFX_5::1) ---------->	TR4
TR1	Echo Request (PRFX_8::1) ---------->	2(c)		
		Step2(c)	Echo Request (PRFX_8::1) ---------->	TR4

3 : WAIT 180 Second. The Timeout should have expired

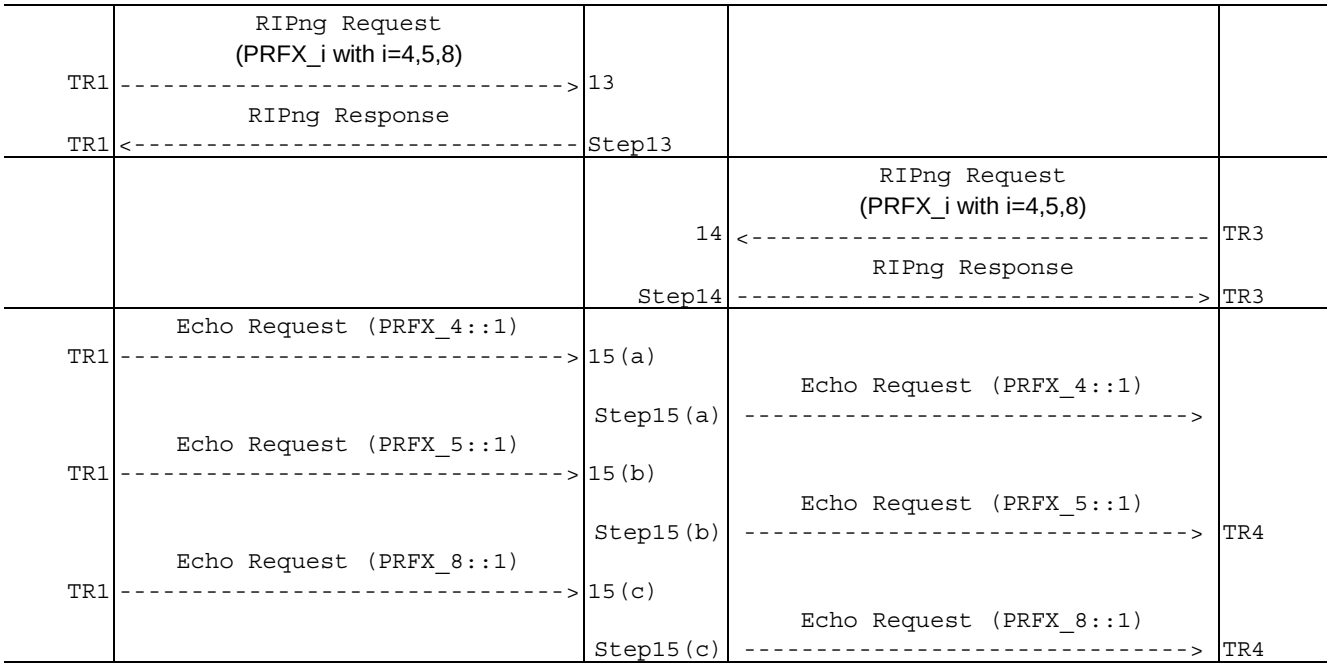
		4	RIPng Response <----------	TR4
TR1	RIPng Request (PRFX_i with i=4,5,8) ---------->	5		
TR1	RIPng Response <----------	Step5		
		6	RIPng Request (PRFX_i with i=4,5,8) <----------	TR3
		Step6	RIPng Response ---------->	TR3
TR1	Echo Request (PRFX_4::1) ---------->	7(a)		
		Step7(a)	Echo Request (PRFX_4::1) ---------->	TR4
TR1	Echo Request (PRFX_5::1) ---------->	7(b)		
		Step7(b)	Echo Request (PRFX_5::1) ---------->	TR4
	Echo Request (PRFX_8::1)			



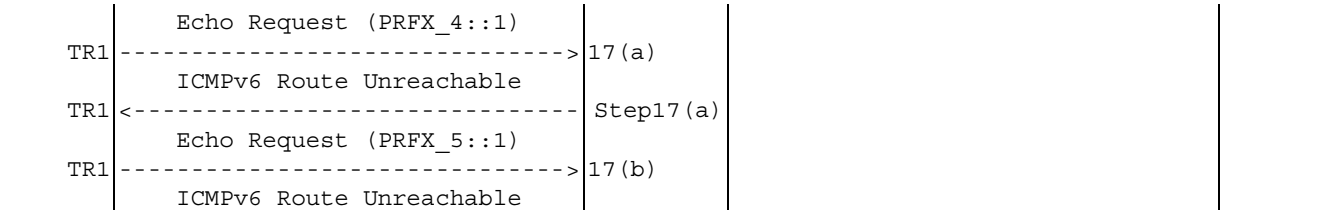
8：Wait 120 seconds. The Garbage-collection timer will have completely expired if running but not the timeout for for PRFX_4, PRFX_5, PRFX_8.

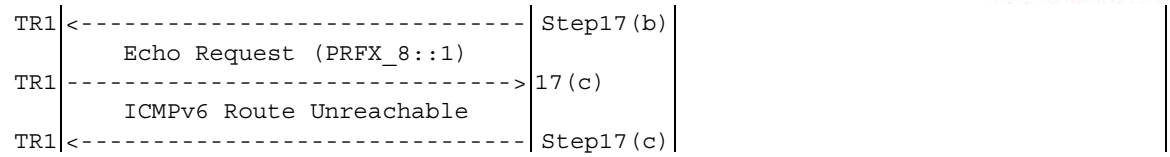


12 : Wait 60 seconds. The Timeout will have expired for PRFX_4, PRFX_5 and PRFX_8



16：Wait 120 seconds. The Garbage-collection timer will have expired for PRFX_4, PRFX_5, PRFX_8.





Postamble:

Because the Routing Table of the RUT is clear nothing should be done in this postamble. Nevertheless it could be required to clear all remaining routes in the RUT by sending a **RIPng Response** packet from **TR4** in which all Metrics are set to infinity(16).

5.3.7 Incorrect Deletion Packet

Purpose: Check that incorrect Response packet with a metric set to 16 do not delete the Routing Table on the NUT

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.
- Add a second Link-local address on NUT's Link1

Discussion:

The goal of this test is to check that incorrect Response packet with a Metric set to 16 do not update the Routing Table on the NUT. Incorrect metrics and other format errors usually indicate misbehaving neighbors and should probably be brought to the administrator's attention. For example, if the metric is greater than infinity, ignore the entry but log the event. The basic validation tests are:

- is the destination prefix valid (e.g., not a multicast prefix and not a link-local address) A link-local address should never be present in an RTE.
- is the prefix length valid (i.e., between 0 and 128, inclusive)
- is the metric valid (i.e., between 1 and 16, inclusive)

If any check fails, ignore that entry and proceed to the next. Again, logging the error is probably a good idea.

Packets:

- *RTEs sent by TR2 in Ripng Response (except the first one) are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_4	16	0x22
PRFX_5	16	0x22
PRFX_8	16	0x22

Procedure:

1. TR2 sends a RIPng Response to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. 521).
   The source address is the Link-local address of TR2 and the destination address is the RIPng Multicast group FF02::9.
   The Hop limit has to be set to 255.

   The RIPng Response is the following:

IPv6 Prefix	Metric	Route Tag
PRFX_4	1	0x21
PRFX_5	2	0x21
PRFX_8	1	0x21

2. Send each of the following invalid Response Packet from TR2:

- *Part A: Incorrect Hop Limit*

  TR2 sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR2 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to **254**.

- *Part B: Incorrect UDP Source Port*

  TR2 sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP port source is 777 and the destination port is RIPng port (ie. 521). The source address is the Link-local address of TR2 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255.

- *Part C: Incorrect Source Address*

  TR2 sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255. The source address is :

  (1) the **Global Address of TR2**

  (2) the **Multicast Solicited Address of TR2**

  (3) **FF02::1 (Multicast Address All hosts on the link)**

  (4) **The Global Address of the RUT on link1**

  (5) **The Link-local Address of the RUT on link1**

  (6) **The Global Address of the RUT on link2**

3. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.
4. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is done for PRFX_4, PRFX_5 and PRFX_8.
5. **TR2** sends a **RIPng Response** to the RUT to clear the route table entries for PRFX_4, PRFX_5 and PRFX_8. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the same as in Step 1 and the destination address is the RIPng Multicast group FF02::9. The Hop limit is set to **255** and the metric is set to **16** for each routing entry.

Observable Results:

- **Step 1:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.
- **Step 2:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. When routes are created, an update has also to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent. Because all Response Packet are invalid the RUT has to silently discard them. As a consequence no Triggered Update Should Be sent on Link1 and Link2.
- **Step 3:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	2	0x21
PRFX_5	3	0x21
PRFX_8	2	0x21

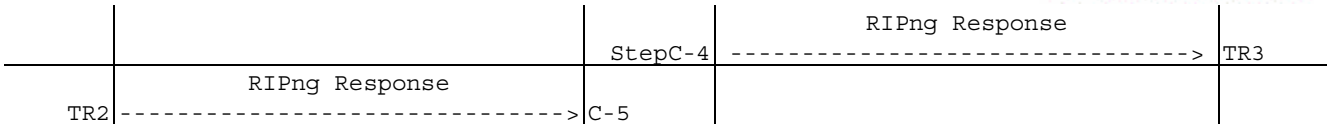
- **Step 4:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag

PRFX_4	2	0x21
PRFX_5	3	0x21
PRFX_8	2	0x21

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
TR2	RIPng Response ---------->	A-1		
TR2	RIPng Response ---------->	A-2		
TR1	RIPng Request (PRFX_i with i=4,5,8) ---------->	A-3		
TR1	RIPng Response <----------	StepA-3		
		A-4	RIPng Request (PRFX_i with i=4,5,8) <----------	TR3
		StepA-4	RIPng Response ---------->	TR3
TR2	RIPng Response ---------->	A-5		
TR2	RIPng Response ---------->	B-1		
TR2	RIPng Response ---------->	B-2		
TR1	RIPng Request (PRFX_i with i=4,5,8) ---------->	B-3		
TR1	RIPng Response <----------	StepB-3		
		B-4	RIPng Request (PRFX_i with i=4,5,8) <----------	TR3
		StepB-4	RIPng Response ---------->	TR3
TR2	RIPng Response ---------->	B-5		
TR2	RIPng Response ---------->	C-1		
TR2	RIPng Response ---------->	C1-2		
TR2	RIPng Response ---------->	C2-2		
TR2	RIPng Response ---------->	C3-2		
TR2	RIPng Response ---------->	C4-2		
TR2	RIPng Response ---------->	C5-2		
TR2	RIPng Response ---------->	C6-2		
TR1	RIPng Request (PRFX_i with i=4,5,8) ---------->	C-3		
TR1	RIPng Response <----------	StepC-3		
		C-4	RIPng Request (PRFX_i with i=4,5,8) <----------	TR3



Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT by sending a **RIPng Response** packet from **TR2** in which all Metrics are set to infinity(16).

5.4 RTEs Request

5.4.1 Entire Table Request (*)

Purpose: Check the correct handling of the Entiere table Request

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.
- This test is dependant from the algorithm used (ie Poison Reverse, Split Horizon, and none). According to **[RFC2080]** the preferred method of operation is Poison Reverse although the implementations Should provide a per-interface control allowing No Horizoning, Split Horizoning and Poison Reverse

Discussion:

A Request is used to ask for a response containing all or part of a router's routing table. Normally, Requests are sent as multicasts, from the RIPng port, by routers which have just come up and are seeking to fill in their routing tables as quickly as possible. However, there may be situations (e.g., router monitoring) where the routing table of only a single router is needed. In this case, the Request should be sent directly to that router from a UDP port other than the RIPng port. If such a Request is received, the router responds directly to the requestor's address and port with a globally valid source address since the requestor may not reside on the directly attached network.

The Request is processed entry by entry. If there are no entries, no response is given. There is one special case. If there is exactly one entry in the request, and it has a destination prefix of zero, a prefix length of zero, and a metric of infinity (i.e., 16), then this is a request to send the entire routing table. In that case, a call is made to the output process to send the routing table to the requesting address/port. Except for this special case, processing is quite simple. Examine the list of RTEs in the Request one by one.

For each entry, look up the destination in the router's routing database and, if there is a route, put that route's metric in the metric field of the RTE. If there is no explicit route to the specified destination, put infinity in the metric field. Once all the entries have been filled in, change the command from Request to Response and send the datagram back to the requestor.

Note that there is a difference in metric handling for specific and whole-table requests. If the request is for a complete routing table, normal output processing is done, including Split Horizon (see section 2.6 on Split Horizon). If the request is for specific entries, they are looked up in the routing table and the information is returned as is; no Split Horizon processing is done. The reason for this distinction is the expectation that these requests are likely to be used for different purposes. When a router first comes up, it multicasts a Request on every connected network asking for a complete routing table. It is assumed that these complete routing tables are to be used to update the requestor's routing table. For this reason, Split Horizon must be done. It is further assumed that a Request for specific networks is made only by diagnostic software, and is not used for routing. In this case, the requester would want to know the exact contents of the routing table and would not want any information hidden or modified.

Packet:

- *RTEs sent by TR1 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_1	1	0x1
PRFX_15	2	0x15
PRFX_16	3	0x16

Procedure:

- **Do The following operation for each Algorithm Selection:**
  - **No Horizoning**
  - **Split Horizon**
  - **Poison Reverse**

1. **TR1** sends a **RIPng Response** to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. `521`).
   The source address is the Link-local address of TR5 and the destination address is the RIPng Multicast group `FF02::9`.
   The Hop limit has to be set to `255`.
2. **TR2** sends a **RIPng Request** to get the whole routing table of the RUT.
   The UDP ports source and destination are the RIPng port (ie. `521`).
   The Hop limit is set to **`254`.**
   There is exactly one entry in the request, and it has a destination prefix of zero, a prefix length of zero, and a metric of infinity (i.e., 16).
   Addresses are the following:
   (a) Source: **link-local address of TR2**; Destination: **link-local of the RUT on Link1.**
   (b) Source: **link-local address of TR2**; Destination: **global address of the RUT on Link1.**
   (c) Source: **global address of TR2**; Destination: **link-local of the RUT on Link1.**

(d) Source: **global address of TR2**; Destination: **global address of the RUT on Link1.**

(e) Source: **global address of TR2**; Destination: **global address of the RUT on Link2.**

(f) Source: **link-local address of TR2**, Destination: **FF02::2 (Multicast Address All routers on the link).**

(g) Source: **global address of TR2**; Destination: **FF02::9 (Multicast Address All Rinpg routers on the link).**

3. **TR3** sends a **RIPng Request** to get the whole routing table of the RUT.
The UDP ports source and destination are the RIPng port (ie. `521`).
The Hop limit is set to **`254`**.
There is exactly one entry in the request, and it has a destination prefix of zero, a prefix length of zero, and a metric of infinity (i.e., 16).
Addresses are the following:

   (a) Source: **link-local address of TR3**; Destination: **link-local of the RUT on Link2.**

   (b) Source: **link-local address of TR3**; Destination: **global address of the RUT on Link2.**

   (c) Source: **global address of TR3**; Destination: **link-local of the RUT on Link2.**

   (d) Source: **global address of TR3**; Destination: **global address of the RUT on Link2.**

   (e) Source: **global address of TR3**; Destination: **global address of the RUT on Link1.**

   (f) Source: **link-local address of TR3**, Destination: **FF02::2 (Multicast Address All routers on the link).**

   (g) Source: **global address of TR3**; Destination: **FF02::9 (Multicast Address All Rinpg routers on the link).**

4. **TR2** sends a **RIPng Request** to get the whole routing table of the RUT.
The UDP ports source is set to **`777`** and destination is the RIPng port (ie. `521`).
The Hop limit is set to **`254`**.
There is exactly one entry in the request, and it has a destination prefix of zero, a prefix length of zero, and a metric of infinity (i.e., 16).
Addresses are the following:

   - Source: **link-local address of TR2**; Destination: **link-local of the RUT on Link1.**
   - Source: **link-local address of TR2**; Destination: **global address of the RUT on Link1.**
   - Source: **global address of TR2**; Destination: **link-local of the RUT on Link1.**
   - Source: **global address of TR2**; Destination: **global address of the RUT on Link1.**
   - Source: **global address of TR2**; Destination: **global address of the RUT on Link2.**
   - Source: **link-local address of TR2**, Destination: **FF02::2 (Multicast Address All routers on the link).**
   - Source: **global address of TR2**; Destination: **FF02::9 (Multicast Address All Rinpg routers on the link).**

5. **TR3** sends a **RIPng Request** to get the whole routing table of the RUT.
The UDP ports source is set to **`777`** and destination is the RIPng port (ie. `521`).
The Hop limit is set to **`254`**.
There is exactly one entry in the request, and it has a destination prefix of zero, a prefix length of zero, and a metric of infinity (i.e., 16).
Addresses are the following:

   (a) Source: **link-local address of TR3**; Destination: **link-local of the RUT on Link2.**

   (b) Source: **link-local address of TR3**; Destination: **global address of the RUT on Link2.**

   (c) Source: **global address of TR3**; Destination: **link-local of the RUT on Link2.**

   (d) Source: **global address of TR3**; Destination: **global address of the RUT on Link2.**

   (e) Source: **global address of TR3**; Destination: **global address of the RUT on Link1.**

   (f) Source: **link-local address of TR3**, Destination: **FF02::2 (Multicast Address All routers on the link).**

   (g) Source: **global address of TR3**; Destination: **FF02::9 (Multicast Address All Rinpg routers on the link).**

Observable Results:

- **Step 1:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.
- **Step 2:** A Response has to be generated by the RUT on link1.
  The Response source address is the RUT Link-local address of Link 1, the destination address is the source address of the RIPng Request done. The source port is the RIPng port source, the destination port is `777`.
  The Response should contain the following fields in any order:

  i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_1	2	0x1
PRFX_11	1	0
PRFX_15	3	0x15
PRFX_16	4	0x16

  ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_11	1	0

  iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_1	16	0
PRFX_11	1	0
PRFX_15	16	0
PRFX_16	16	0

- **Step 3:** A Response has to be generated by the RUT on link2.
  The Response source address is the RUT Link-local address of Link 2, the destination address is TR3 global address, the source port is the RIPng port source and the destination port is `777`.
  The Response should contain the following fields in any order with No Horizoning, Split Horizon and Poison Reverse:

IPv6 Prefix	Metric	Route Tag
PRFX_1	2	0x1
PRFX_10	1	0
PRFX_15	3	0x15
PRFX_16	4	0x16

- **Step 4:** A Response has to be generated by the RUT on link1.

  The Response source address is the RUT Global address of Link 1, the destination address is the source address of the RIPng Request done. The source and destination port are the RIPng port.
  The Response should contain the following fields in any order:

  i. With No Horizoning

IPv6 Prefix	Metric	Route Tag
PRFX_1	2	0x1
PRFX_11	1	0
PRFX_15	3	0x15
PRFX_16	4	0x16

ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_11	1	0

iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_1	16	0
PRFX_11	1	0
PRFX_15	16	0
PRFX_16	16	0

- **Step 5:** A Response has to be generated by the RUT on link2.
  The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source and destination port are the RIPng port. The Response should contain the following fields in any order with No Horizoning, Split Horizon and Poison Reverse:

IPv6 Prefix	Metric	Route Tag
PRFX_1	2	0x1
PRFX_10	1	0
PRFX_15	3	0x15
PRFX_16	4	0x16

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
	RIPng Response			
TR1	---------->	1		
	RIPng Request (ALL)			
TR2	---------->	2		
	RIPng Response			
TR2	<----------	Step2		
			RIPng Request (ALL)	
		3	<----------	TR3
			RIPng Response	
		Step3	---------->	TR3
	RIPng Request (ALL)			
TR2	---------->	4		
	RIPng Response			
TR2	<----------	Step4		
			RIPng Request (ALL)	
		5	<----------	TR3
			RIPng Response	
		Step5	---------->	TR3

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT: the Tester sends a **RIPng Response** packet from **TR1** in which all Metrics are set to infinity (16).

5.4.2 RTEs Request

Purpose: Check the correct handling of solicited Request

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

A Request is used to ask for a response containing all or part of a router's routing table. Normally, Requests are sent as multicasts, from the RIPng port, by routers which have just come up and are seeking to fill in their routing tables as quickly as possible. However, there may be situations (e.g., router monitoring) where the routing table of only a single router is needed. In this case, the Request should be sent directly to that router from a UDP port other than the RIPng port. If such a Request is received, the router responds directly to the requestor's address and port with a globally valid source address since the requestor may not reside on the directly attached network.

The Request is processed entry by entry. If there are no entries, no response is given. There is one special case. If there is exactly one entry in the request, and it has a destination prefix of zero, a prefix length of zero, and a metric of infinity (i.e., 16), then this is a request to send the entire routing table. In that case, a call is made to the output process to send the routing table to the requesting address/port. Except for this special case, processing is quite simple. Examine the list of RTEs in the Request one by one.

For each entry, look up the destination in the router's routing database and, if there is a route, put that route's metric in the metric field of the RTE. If there is no explicit route to the specified destination, put infinity in the metric field. Once all the entries have been filled in, change the command from Request to Response and send the datagram back to the requestor.

Note that there is a difference in metric handling for specific and whole-table requests. If the request is for a complete routing table, normal output processing is done, including Split Horizon (see section 2.6 on Split Horizon). If the request is for specific entries, they are looked up in the routing table and the information is returned as is; no Split Horizon processing is done. The reason for this distinction is the expectation that these requests are likely to be used for different purposes. When a router first comes up, it multicasts a Request on every connected network asking for a complete routing table. It is assumed that these complete routing tables are to be used to update the requestor's routing table. For this reason, Split Horizon must be done. It is further assumed that a Request for specific networks is made only by diagnostic software, and is not used for routing. In this case, the requester would want to know the exact contents of the routing table and would not want any information hidden or modified.

Packets:

- *RTEs sent by TR4 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_4	1	0x4
PRFX_5	1	0x4
PRFX_8	2	0x4
PRFX_9	2	0x4

Procedure:

1. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255.

2. **TR1** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR1 and the destination address is the global address of the RUT on Link1. Moreover the request is the following:

IPv6 Prefix	Route Tag	Metric
PRFX_4	0x4	0
PRFX_5	0	2
PRFX_8	0x4	3
PRFX_9	0x5	4
PRFX_10	0x1	0
PRFX_11	0	5
PRFX_15	0x1	0
PRFX_16	0	16
PRFX_255	0xFF	255

3. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
The UDP ports source is set to 777 and destination is the RIPng port (ie. 521). The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. Moreover the request is the following:

IPv6 Prefix	Route Tag	Metric
PRFX_4	0x4	0
PRFX_5	0	2
PRFX_8	0x4	3
PRFX_9	0x5	4
PRFX_10	0x1	0
PRFX_11	0	5
PRFX_15	0x1	0
PRFX_16	0	16
PRFX_255	0xFF	255

Observable Results:

- **Step 1:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.
- **Step 2:** A Response has to be generated by the RUT on link1. The Response source address is the RUT Global address of Link 1, the destination address is TR1 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	2	0x4
PRFX_5	2	0x4
PRFX_8	3	0x4
PRFX_9	3	0x4
PRFX_10	1	0
PRFX_11	1	0
PRFX_15	16	0
PRFX_16	16	0
PRFX_255	16	0

The response SHOULD not care about the use of Split Horizon and Poison Reverse Algorithm.

- **Step 3:** A Response has to be generated by the RUT on link2. The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address. The source port is the RIPng port source, the destination port is 777:

IPv6 Prefix	Metric	Route Tag
PRFX_4	2	0x4
PRFX_5	2	0x4
PRFX_8	3	0x4
PRFX_9	3	0x4
PRFX_10	1	0
PRFX_11	1	0
PRFX_15	16	0
PRFX_16	16	0
PRFX_255	16	0

The response SHOULD not care about the use of Split Horizon and Poison Reverse Algorithm

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response <----------	TR4
TR1	RIPng Request (PRFX_i with i=4,5,8) ---------->	2		
TR1	RIPng Response <----------	Step2		
		3	RIPng Request (PRFX_i with i=4,5,8) <----------	TR3
		Step3	RIPng Response ---------->	TR3

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT: the Tester sends a **RIPng Response** packet from **TR4** in which all Metrics are set to infinity (16).

5.4.3 Empty Request

Purpose: Check the correct handling of the Empty Request

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

A Request is used to ask for a response containing all or part of a router's routing table. Normally, Requests are sent as multicasts, from the RIPng port, by routers which have just come up and are seeking to fill in their routing tables as quickly as possible. However, there may be situations (e.g., router monitoring) where the routing table of only a single router is needed. In this case, the Request should be sent directly to that router from a UDP port other than the RIPng port. If such a Request is received, the router responds directly to the requestor's address and port with a globally valid source address since the requestor may not reside on the directly attached network.

The Request is processed entry by entry. If there are no entries, no response is given. There is one special case. If there is exactly one entry in the request, and it has a destination prefix of zero, a prefix length of zero, and a metric of infinity (i.e., 16), then this is a request to send the entire routing table. In that case, a call is made to the output process to send the routing table to the requesting address/port. Except for this special case, processing is quite simple. Examine the list of RTEs in the Request one by one.

For each entry, look up the destination in the router's routing database and, if there is a route, put that route's metric in the metric field of the RTE. If there is no explicit route to the specified destination, put infinity in the metric field. Once all the entries have been filled in, change the command from Request to Response and send the datagram back to the requestor.

Note that there is a difference in metric handling for specific and whole-table requests. If the request is for a complete routing table, normal output processing is done, including Split Horizon (see section 2.6 on Split Horizon). If the request is for specific entries, they are looked up in the routing table and the information is returned as is; no Split Horizon processing is done. The reason for this distinction is the expectation that these requests are likely to be used for different purposes. When a router first comes up, it multicasts a Request on every connected network asking for a complete routing table. It is assumed that these complete routing tables are to be used to update the requestor's routing table. For this reason, Split Horizon must be done. It is further assumed that a Request for specific networks is made only by diagnostic software, and is not used for routing. In this case, the requester would want to know the exact contents of the routing table and would not want any information hidden or modified.

Packet:

- *RTEs sent by TR1 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_1	1	0
PRFX_15	2	0
PRFX_16	3	0

Procedure:

1. **TR1** sends a **RIPng Response** to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. `521`).
   The source address is the Link-local address of TR1 and the destination address is the RIPng Multicast group `FF02::9`.
   The Hop limit has to be set to `255`.
2. **TR2** sends an **Empty RIPng Request** to the RUT (ie without any route).
   The UDP ports source and destination are the RIPng port (ie. `521`).
   The Hop limit is set to **`254`**.
   Addresses are the following:
   (a) Source: **link-local address of TR2**; Destination: **link-local of the RUT on Link1.**
   (b) Source: **link-local address of TR2**; Destination: **global address of the RUT on Link1.**
   (c) Source: **global address of TR2**; Destination: **link-local of the RUT on Link1.**
   (d) Source: **global address of TR2**; Destination: **global address of the RUT on Link1.**
   (e) Source: **global address of TR2**; Destination: **global address of the RUT on Link2.**
   (f) Source: **link-local address of TR2**, Destination: **FF02::2 (Multicast Address All routers on the link).**
   (g) Source: **global address of TR2**; Destination: **FF02::9 (Multicast Address All Rinpg routers on the link).**
3. **TR3** sends an **Empty RIPng Request** to the RUT (ie without any route).
   The UDP ports source and destination are the RIPng port (ie. `521`).
   The Hop limit is set to **`254`**.
   Addresses are the following:
   (a) Source: **link-local address of TR3**; Destination: **link-local of the RUT on Link2.**
   (b) Source: **link-local address of TR3**; Destination: **global address of the RUT on Link2.**
   (c) Source: **global address of TR3**; Destination: **link-local of the RUT on Link2.**
   (d) Source: **global address of TR3**; Destination: **global address of the RUT on Link2.**
   (e) Source: **global address of TR3**; Destination: **global address of the RUT on Link1.**
   (f) Source: **link-local address of TR3**, Destination: **FF02::2 (Multicast Address All routers on the link).**
   (g) Source: **global address of TR3**; Destination: **FF02::9 (Multicast Address All Rinpg routers on the link).**
4. **TR2** sends an **Empty RIPng Request** to the RUT (ie without route).
   The UDP ports source is set to **`777`** and destination is the RIPng port (ie. `521`).
   The Hop limit is set to **`254`**.
   Addresses are the following:
   (a) Source: **link-local address of TR2**; Destination: **link-local of the RUT on Link1.**
   (b) Source: **link-local address of TR2**; Destination: **global address of the RUT on Link1.**
   (c) Source: **global address of TR2**; Destination: **link-local of the RUT on Link1.**
   (d) Source: **global address of TR2**; Destination: **global address of the RUT on Link1.**
   (e) Source: **global address of TR2**; Destination: **global address of the RUT on Link2.**
   (f) Source: **link-local address of TR2**, Destination: **FF02::2 (Multicast Address All routers on the link).**
   (g) Source: **global address of TR2**; Destination: **FF02::9 (Multicast Address All Rinpg routers on the link).**

5. **TR3** sends a **Empty RIPng Request** to the RUT (ie without route).
   The UDP ports source is set to **777** and destination is the RIPng port (ie. 521).
   The Hop limit is set to **254 .**
   Addresses are the following:

   (a) Source: **link-local address of TR3**; Destination: **link-local of the RUT on Link2.**

   (b) Source: **link-local address of TR3**; Destination: **global address of the RUT on Link2.**

   (c) Source: **global address of TR3**; Destination: **link-local of the RUT on Link2.**

   (d) Source: **global address of TR3**; Destination: **global address of the RUT on Link2.**

   (e) Source: **global address of TR3**; Destination: **global address of the RUT on Link1.**

   (f) Source: **link-local address of TR3**, Destination: **FF02::2 (Multicast Address All routers on the link).**

   (g) Source: **global address of TR3**; Destination: **FF02::9 (Multicast Address All Rinpg routers on the link).**

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
TR1	RIPng Response ---------->	1		
TR2	RIPng Request (Empty) ---------->	2		
TR2	RIPng Response <----------	Step2		
		3	RIPng Request (Empty) <----------	TR3
		Step3	RIPng Response ---------->	TR3
TR2	RIPng Request (Empty) ---------->	4		
TR2	RIPng Response <----------	Step4		
		5	RIPng Request (Empty) <----------	TR3
		Step5	RIPng Response ---------->	TR3

Observable Results:

- **Step 1:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent.
- **Step 2:** Because the Request was empty, no Response has to be generated by the RUT on link1.
- **Step 3:** Because the Request was empty, no Response has to be generated by the RUT on link2.
- **Step 4:** Because the Request was empty, no Response has to be generated by the RUT on link1.
- **Step 5:** Because the Request was empty, no Response has to be generated by the RUT on link2.

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT: the Tester sends a **RIPng Response** packet from **TR1** in which all Metrics are set to infinity (16).

5.5 Regular Routing Update / Triggered Update & Algorithm selection (*)

This part will test Unsollicited Answers (ie: Regular Routing Updates and Triggered Update). This part will also check the algorithm impact on these Unsollicited Answers.

5.5.1 Unsollicited Answers Contains

5.5.1.1 RTE Creation (*)

Purpose: Check that a Route Creation is reported in the next Regular Routing Update and Triggered Update on each Link

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.
- This test is dependant from the algorithm used (ie Poison Reverse, Split Horizon, and none). According to **[RFC2080]** the preferred method of operation is Poison Reverse although the implementations Should provide a per-interface control allowing No Horizoning, Split Horizoning and Poison Reverse

Discussion:

Every 30 seconds, the RIPng process is awakened to send an unsolicited Response message, containing the complete routing table, to every neighboring router.

If routes have changed, the RIPng Process SHOULD sent a Triggered Update. Triggered updates do not need to include the entire routing table. In principle, only those routes which have changed need to be included. Therefore messages generated as part of a triggered update must include at least those routes that have their route change flag set. They may include additional routes, at the discretion of the implementor; however, sending complete routing updates is strongly discouraged. When a triggered update is processed, messages should be generated for every directly-connected network. Split Horizon processing is done when generating triggered updates as well as normal updates. If, after Split Horizon processing for a given network, a changed route will appear unchanged on that network (e.g., it appears with an infinite metric), the route need not be sent. If no routes need be sent on that network, the update may be omitted. Once all of the triggered updates have been generated, the route change flags should be cleared.

Packets:

- *RTEs sent by TR4 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_4	1	0x4
PRFX_5	1	0x4
PRFX_8	2	0x4

- *RTEs sent by TR1 in Ripng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_6Bone	5	0x1
PRFX_1	1	0x1

Procedure:

- **Do The following operation for each Algorithm Selection:**
  - **No Horizoning**
  - **Split Horizon**
  - **Poison Reverse**

1. Wait For the First Regular Routing Update on Link2
2. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. `521`).
   The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group `FF02::9`.
   The Hop limit has to be set to `255`.
3. Wait For the second Regular Routing Update on Link2 (between 15 and 45 seconds from the previous one). A triggered Update SHOULD also be sent prior to the Regular routing Update.
4. **TR1** sends a **RIPng Response** to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. `521`).
   The source address is the Link-local address of TR1and the destination address is the RIPng Multicast group `FF02::9`.
   The Hop limit has to be set to `255`.
5. Wait For the next Regular Routing Update on Link2 (between 15 and 45 seconds from the previous one). A triggered Update SHOULD also be sent prior to the Regular routing Update.

Observable Results:

- **Step 1:** If a Regular Routing Update is not received 45 seconds after the previous one, the test verdict will be FAIL. The Regular Routing Update on Link2 should contain:

  i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0

  ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0

  iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0

- **Step 3:**
  - A triggered Update SHOULD be sent prior on Link2 according to the algorithm used. It should contain:

    i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_4	2	0x4
PRFX_5	2	0x4
PRFX_8	3	0x4

ii. **With Split Horizon**

No Triggered Update SHOULD be sent on Link2

iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_4	16	0
PRFX_5	16	0
PRFX_8	16	0

- If a Regular Routing Update is not received 45 seconds after the previous one, the test verdict will be FAIL. The Regular Routing Update on Link2 should contain:

I. With No Horizoning

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0
PRFX_4	2	0x4
PRFX_5	2	0x4
PRFX_8	3	0x4

II. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0

III. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0
PRFX_4	16	0
PRFX_5	16	0
PRFX_8	16	0

- **Step 5:**
  - A triggered Update SHOULD be sent prior on Link2 according to the algorithm used. It should contain:

i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_6Bone	6	0x1
PRFX_1	2	0x1

ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_6Bone	6	0x1
PRFX_1	2	0x1

iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_6Bone	6	0x1
PRFX_1	2	0x1

- If a Regular Routing Update is not received 45 seconds after the previous one, the test verdict will be FAIL. The Regular Routing Update on Link2 should contain:

I. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0
PRFX_4	1	0x4
PRFX_5	1	0x4
PRFX_8	2	0x4
PRFX_6Bone	6	0x1
PRFX_1	2	0x1

II. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_6Bone	6	0x1
PRFX_1	2	0x1

III. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0
PRFX_4	16	0
PRFX_5	16	0
PRFX_8	16	0
PRFX_6Bone	6	0x1
PRFX_1	2	0x1

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response (regular Update) ---------->	
		2	RIPng Response <----------	TR4
		3	RIPng Response (Trigger Update) ----------> RIPng Response (regular Update) ---------->	
TR1	RIPng Response ---------->	5		
		6	RIPng Response (Trigger Update) ----------> RIPng Response (regular Update) ---------->	

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT: the Tester sends a **RIPng Response** packet from **TR4** and **TR1** in which all Metrics are set to infinity (16).

5.5.1.2 Route Attributes Update in RTE (*)

Purpose: Check that a Route Tag and a metric Update is reported in the next Regular Routing Update and Triggered Update on each Link

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.
- This test is dependant from the algorithm used (ie Poison Reverse, Split Horizon, and none). According to **[RFC2080]** the preferred method of operation is Poison Reverse although the implementations Should provide a per-interface control allowing No Horizoning, Split Horizoning and Poison Reverse

Discussion:

Every 30 seconds, the RIPng process is awakened to send an unsolicited Response message, containing the complete routing table, to every neighboring router.

Procedure:

- **Do The following operation for each Algorithm Selection:**
  - **No Horizoning**
  - **Split Horizon**
  - **Poison Reverse**

1. Wait For the First Regular Routing Update on Link2
2. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. `521`).
   The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group `FF02::9`.
   The Hop limit has to be set to `255`. RTEs sent by TR4 in Ripng Response are the following *:*

IPv6 Prefix	Metric	Route Tag
PRFX_4	2	0x41
PRFX_5	2	0x41
PRFX_8	2	0x41

3. Wait For the second Regular Routing Update on Link2 (between 15 and 45 seconds from the First one). A triggered Update SHOULD also be sent prior to the Regular routing Update.
4. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. `521`).
   The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group `FF02::9`.
   The Hop limit has to be set to `255`. RTEs sent by TR4 in Ripng Response are the following *:*

IPv6 Prefix	Metric	Route Tag
PRFX_4	1	0x42
PRFX_5	2	0x42
PRFX_8	3	0x42

5. Wait For the Third Regular Routing Update on Link2. A triggered Update SHOULD also be sent prior to the Regular routing Update.

6. **TR4** sends a **RIPng Response** to clear the routing table of the RUT in which all Metrics are set to infinity (16).

7. **WAIT 120s until the Garbage-Collection timer has expired.**

8. Wait For the next Regular Routing Update on Link2. A triggered Update MAY also be sent prior to the Regular routing Update.

9. **TR1** sends a **RIPng Response** to give its RIPng routing table to the RUT.
The UDP ports source and destination are the RIPng port (ie. `521`).
The source address is the Link-local address of TR1and the destination address is the RIPng Multicast group `FF02::9`.
The Hop limit has to be set to `255`. RTEs sent by TR1 in Ripng Response are the following *:*

IPv6 Prefix	Metric	Route Tag
PRFX_1	2	0x11
PRFX_6Bone	2	0x11
PRFX_6to4	2	0x11

10. Wait For the next Regular Routing Update on Link2 (between 15 and 45 seconds from the previous one). A triggered Update SHOULD also be sent prior to the Regular routing Update.

11. **TR1** sends a **RIPng Response** to give its RIPng routing table to the RUT.
The UDP ports source and destination are the RIPng port (ie. `521`).
The source address is the Link-local address of TR1and the destination address is the RIPng Multicast group `FF02::9`.
The Hop limit has to be set to `255`. RTEs sent by TR1 in Ripng Response are the following *:*

IPv6 Prefix	Metric	Route Tag
PRFX_1	1	0x12
PRFX_6Bone	2	0x12
PRFX_6to4	3	0x12

12. Wait For the next Regular Routing Update on Link2 (between 15 and 45 seconds from the previous one). A triggered Update SHOULD also be sent prior to the Regular routing Update.

Observable Results:

- **Step 1:**The Regular Routing Update on Link2 should contain:

  i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0

  ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0

  iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0

- **Step 3:**

  - A triggered Update SHOULD be sent prior on Link2 according to the algorithm used. It should contain:

i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_4	3	0x41
PRFX_5	3	0x41
PRFX_8	3	0x41

ii. **With Split Horizon**

No Triggered Update SHOULD be sent on Link2

iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_4	16	0
PRFX_5	16	0
PRFX_8	16	0

- If a Regular Routing Update is not received 45 seconds after the previous one, the test verdict will be FAIL. The Regular Routing Update on Link2 should contain:

I. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0
PRFX_4	3	0x41
PRFX_5	3	0x41
PRFX_8	3	0x41

II. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0

III. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0
PRFX_4	16	0
PRFX_5	16	0
PRFX_8	16	0

- **Step 5:**
  - A triggered Update SHOULD be sent prior on Link2 according to the algorithm used. It should contain:

i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_4	2	0x42
PRFX_5	3	0x42
PRFX_8	4	0x42

ii. **With Split Horizon**

No Triggered Update SHOULD be sent on Link2

iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_4	16	0
PRFX_5	16	0
PRFX_8	16	0

- If a Regular Routing Update is not received 45 seconds after the previous one, the test verdict will be FAIL. The Regular Routing Update on Link2 should contain:

I. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0
PRFX_4	2	0x42
PRFX_5	3	0x42
PRFX_8	4	0x42

ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0

iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0
PRFX_4	16	0
PRFX_5	16	0
PRFX_8	16	0

- **Step 7:** Route for PRFX_4, PRFX_5 and PRFX_8 MUST now be removed of the RUT.
- **Step 8:**
  - A triggered Update MAY be sent prior on Link2 according to the algorithm used. It should contain:

i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_4	16	0
PRFX_5	16	0
PRFX_8	16	0

ii. **With Split Horizon**

No Triggered Update SHOULD be sent on Link2

iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_4	16	0

PRFX_5	16	0
PRFX_8	16	0

- The Regular Routing Update on Link2 should contain:

i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0

ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0

iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0

- **Step 10:**
  - A triggered Update SHOULD be sent prior on Link2. It should contain:

IPv6 Prefix	Metric	Route Tag
PRFX_1	3	0x11
PRFX_6Bone	3	0x11
PRFX_6to4	3	0x11

  - If a Regular Routing Update is not received 45 seconds after the previous one, the test verdict will be FAIL. The Regular Routing Update on Link2 should contain:

i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0
PRFX_1	3	0x11
PRFX_6Bone	3	0x11
PRFX_6to4	3	0x11

ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_11	1	0
PRFX_1	3	0x11
PRFX_6Bone	3	0x11
PRFX_6to4	3	0x11

iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	16	0
PRFX_11	1	0
PRFX_1	3	0x11
PRFX_6Bone	3	0x11
PRFX_6to4	3	0x11

- **Step 12:**
  - A triggered Update SHOULD be sent prior on Link2. It should contain:

IPv6 Prefix	Metric	Route Tag
PRFX_1	2	0x12
PRFX_6Bone	3	0x12
PRFX_6to4	4	0x12

  - If a Regular Routing Update is not received 45 seconds after the previous one, the test verdict will be FAIL. The Regular Routing Update on Link2 should contain:

    i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0
PRFX_1	2	0x12
PRFX_6Bone	3	0x12
PRFX_6to4	4	0x12

    ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_11	1	0
PRFX_1	2	0x12
PRFX_6Bone	3	0x12
PRFX_6to4	4	0x12

    iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	16	0
PRFX_11	1	0
PRFX_1	2	0x12
PRFX_6to4	3	0x12
PRFX_6Bone	4	0x12

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response (regular Update) ---------->	
		2	RIPng Response <----------	TR4
		3	RIPng Response (Trigger Update) ----------> RIPng Response (regular Update) ---------->	
		4	RIPng Response <----------	TR4
		5	RIPng Response (Trigger Update) ----------> RIPng Response (regular Update) ---------->	
		6	RIPng Response (Metric 16) <----------	TR4

7：Wait 120 seconds. The Garbage-collection timer will have expired for PRFX_4, PRFX_5, PRFX_8.

		8	RIPng Response (regular Update) ---------->	
TR1	RIPng Response ---------->	9		
		10	RIPng Response (Trigger Update) ----------> RIPng Response (regular Update) ---------->	
TR1	RIPng Response ---------->	11		
		12	RIPng Response (Trigger Update) ----------> RIPng Response (regular Update) ---------->	

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT: the Tester sends a RIPng Response packet from TR1 and TR4 in which all Metrics are set to infinity (16).

5.5.1.3 Route Update from the same Link (*)

Purpose: Check that an Update with a better route is reported in the next Regular Routing Update and Triggered Update on each Link

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.
- This test is dependant from the algorithm used (ie Poison Reverse, Split Horizon, and none). According to **[RFC2080]** the preferred method of operation is Poison Reverse although the implementations Should provide a per-interface control allowing No Horizoning, Split Horizoning and Poison Reverse

Discussion:

Every 30 seconds, the RIPng process is awakened to send an unsolicited Response message, containing the complete routing table, to every neighboring router.

Procedure:

- **Do The following operation for each Algorithm Selection:**
  - **No Horizoning**
  - **Split Horizon**
  - **Poison Reverse**

1. Wait For the First Regular Routing Update on Link2
2. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. `521`).
   The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group `FF02::9`.
   The Hop limit has to be set to `255`. RTEs sent by TR4 in Ripng Response are the following *:*

IPv6 Prefix	Metric	Route Tag
PRFX_3	3	0x41
PRFX_4	2	0x41
PRFX_5	2	0x41
PRFX_7	2	0x41
PRFX_8	2	0x41

3. Wait For the second Regular Routing Update on Link2 (between 15 and 45 seconds from the First one). A triggered Update SHOULD also be sent prior to the Regular routing Update.
4. **TR3** sends a RIPng Response to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. `521`).
   The source address is the Link-local address of TR2 and the destination address is the RIPng Multicast group `FF02::9`.
   The Hop limit has to be set to `255`. RTEs sent by TR2 in Ripng Response are the following *:*

IPv6 Prefix	Metric	Route Tag
PRFX_3	2	0x3

PRFX_4	2	0x3
PRFX_5	3	0x3
PRFX_7	1	0x3
PRFX_8	1	0x3

5. Wait For the Third Regular Routing Update on Link2. A triggered Update SHOULD also be sent prior to the Regular routing Update.

6. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT.
The UDP ports source and destination are the RIPng port (ie. `521`).
The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group `FF02::9`.
The Hop limit has to be set to `255`. RTEs sent by TR4 in Ripng Response are the following *:*

IPv6 Prefix	Metric	Route Tag
PRFX_3	1	0x42
PRFX_4	2	0x42
PRFX_5	3	0x42
PRFX_8	2	0x42

7. Wait For the next Regular Routing Update on Link2 (between 15 and 45 seconds from the previous one). A triggered Update SHOULD also be sent prior to the Regular routing Update.

Observable Results:

- **Step 1:** If a Regular Routing Update is not received 45 seconds after the previous one, the test verdict will be FAIL. The Regular Routing Update on Link2 should contain:

  i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0

  ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0

  iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0

- **Step 3:**
  - A triggered Update SHOULD be sent prior on Link2 according to the algorithm used. It should contain:

    i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_3	4	0x41
PRFX_4	3	0x41
PRFX_5	3	0x41
PRFX_7	3	0x41
PRFX_8	3	0x41

ii. **With Split Horizon**

No Triggered Update SHOULD be sent on Link2

iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_3	4	0x41
PRFX_4	3	0x41
PRFX_5	3	0x41
PRFX_7	3	0x41
PRFX_8	3	0x41

- If a Regular Routing Update is not received 45 seconds after the previous one, the test verdict will be FAIL. The Regular Routing Update on Link2 should contain:

i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0
PRFX_3	4	0x41
PRFX_4	3	0x41
PRFX_5	3	0x41
PRFX_7	3	0x41
PRFX_8	3	0x41

ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0

iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0
PRFX_3	16	0
PRFX_4	16	0
PRFX_5	16	0
PRFX_7	16	0
PRFX_8	16	0

- **Step 5:**
  - A triggered Update SHOULD be sent prior on Link2 according to the algorithm used. It should contain:

i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_3	2	0x42
PRFX_4	3	0x42
PRFX_5	4	0x42

PRFX_8	3	0x42

ii. **With Split Horizon**

No Triggered Update SHOULD be sent on Link2

iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_3	2	0x42
PRFX_4	3	0x42
PRFX_5	4	0x42
PRFX_8	3	0x42

- If a Regular Routing Update is not received 45 seconds after the previous one, the test verdict will be FAIL. The Regular Routing Update on Link2 should contain:

i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0
PRFX_3	3	0x3
PRFX_4	3	0x41
PRFX_5	3	0x41
PRFX_7	2	0x3
PRFX_8	2	0x3

ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0

iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0
PRFX_3	16	0
PRFX_4	16	0
PRFX_5	16	0
PRFX_7	16	0
PRFX_8	16	0

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response (regular Update) ---------->	
		2	RIPng Response <----------	TR4
		3	RIPng Response (Trigger Update) ----------> RIPng Response (regular Update) ---------->	
		4	RIPng Response <----------	TR3
		5	RIPng Response (Trigger Update) ----------> RIPng Response (regular Update) ---------->	

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT: the Tester sends a RIPng Response packet from TR4 and TR1 in which all Metrics are set to infinity (16).

5.5.1.4 Route Update from a different Link (*)

Purpose: Check that an Update with a better route is reported in the next Regular Routing Update and Triggered Update on each Link

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.
- This test is dependant from the algorithm used (ie Poison Reverse, Split Horizon, and none). According to **[RFC2080]** the preferred method of operation is Poison Reverse although the implementations Should provide a per-interface control allowing No Horizoning, Split Horizoning and Poison Reverse

Discussion:

Every 30 seconds, the RIPng process is awakened to send an unsolicited Response message, containing the complete routing table, to every neighboring router.

Procedure:

- **Do The following operation for each Algorithm Selection:**
  - **No Horizoning**
  - **Split Horizon**
  - **Poison Reverse**

1. Wait For the First Regular Routing Update on Link2
2. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. `521`).
   The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group `FF02::9`.
   The Hop limit has to be set to `255`. RTEs sent by TR4 in Ripng Response are the following *:*

IPv6 Prefix	Metric	Route Tag
PRFX_2	3	0x41
PRFX_4	2	0x41
PRFX_5	2	0x41
PRFX_7	2	0x41
PRFX_8	2	0x41

3. Wait For the second Regular Routing Update on Link2 (between 15 and 45 seconds from the First one). A triggered Update SHOULD also be sent prior to the Regular routing Update.
4. **TR2** sends a RIPng Response to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. `521`).
   The source address is the Link-local address of TR2 and the destination address is the RIPng Multicast group `FF02::9`.
   The Hop limit has to be set to `255`. RTEs sent by TR2 in Ripng Response are the following *:*

IPv6 Prefix	Metric	Route Tag
PRFX_2	2	0x2

PRFX_4	2	0x2
PRFX_5	3	0x2
PRFX_7	1	0x2
PRFX_8	1	0x2

5. Wait For the Third Regular Routing Update on Link2. A triggered Update SHOULD also be sent prior to the Regular routing Update.

6. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT.
The UDP ports source and destination are the RIPng port (ie. `521`).
The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group `FF02::9`.
The Hop limit has to be set to `255`. RTEs sent by TR4 in Ripng Response are the following *:*

IPv6 Prefix	Metric	Route Tag
PRFX_2	1	0x42
PRFX_4	2	0x42
PRFX_5	3	0x42
PRFX_8	2	0x42

7. Wait For the next Regular Routing Update on Link2 (between 15 and 45 seconds from the previous one). A triggered Update SHOULD also be sent prior to the Regular routing Update.

Observable Results:

- **Step 1:** If a Regular Routing Update is not received 45 seconds after the previous one, the test verdict will be FAIL. The Regular Routing Update on Link2 should contain:

  i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0

  ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0

  iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0

- **Step 3:**

  - A triggered Update SHOULD be sent prior on Link2 according to the algorithm used. It should contain:

    i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_2	4	0x41
PRFX_4	3	0x41
PRFX_5	3	0x41
PRFX_7	3	0x41
PRFX_8	3	0x41

ii. **With Split Horizon**

No Triggered Update SHOULD be sent on Link2

iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_2	4	0x41
PRFX_4	3	0x41
PRFX_5	3	0x41
PRFX_7	3	0x41
PRFX_8	3	0x41

- If a Regular Routing Update is not received 45 seconds after the previous one, the test verdict will be FAIL. The Regular Routing Update on Link2 should contain:

i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0
PRFX_2	4	0x41
PRFX_4	3	0x41
PRFX_5	3	0x41
PRFX_7	3	0x41
PRFX_8	3	0x41

ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0

iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0
PRFX_2	16	0
PRFX_4	16	0
PRFX_5	16	0
PRFX_7	16	0
PRFX_8	16	0

- **Step 5:**
  - A triggered Update SHOULD be sent prior on Link2. It should contain:

IPv6 Prefix	Metric	Route Tag
PRFX_2	3	0x2
PRFX_7	2	0x2
PRFX_8	2	0x2

- If a Regular Routing Update is not received 45 seconds after the previous one, the test verdict will be FAIL. The Regular Routing Update on Link2 should contain:

i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0
PRFX_2	3	0x2
PRFX_4	3	0x41
PRFX_5	3	0x41
PRFX_7	2	0x2
PRFX_8	2	0x2

ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_2	3	0x2
PRFX_7	2	0x2
PRFX_8	2	0x2

iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0
PRFX_2	3	0x2
PRFX_4	16	0
PRFX_5	16	0
PRFX_7	2	0x2
PRFX_8	2	0x2

- **Step 7:**

- A triggered Update SHOULD be sent prior on Link2 according to the algorithm used. It should contain:

i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_2	2	0x42
PRFX_4	3	0x42
PRFX_5	3	0x42

ii. **With Split Horizon**

No Triggered Update SHOULD be sent on Link2

iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_2	16	0
PRFX_4	16	0
PRFX_5	16	0

- If a Regular Routing Update is not received 45 seconds after the previous one, the test verdict will be FAIL. The Regular Routing Update on Link2 should contain:

i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0
PRFX_2	2	0x42
PRFX_4	3	0x42
PRFX_5	3	0x42
PRFX_7	2	0x2
PRFX_8	2	0x2

ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_7	2	0x2
PRFX_8	2	0x2

iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0
PRFX_2	16	0
PRFX_4	16	0
PRFX_5	16	0
PRFX_7	2	0x2
PRFX_8	2	0x2

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response (regular Update) ---------->	
		2	RIPng Response <----------	TR4
		3	RIPng Response (Trigger Update) ----------> RIPng Response (regular Update) ---------->	
TR2	RIPng Response ---------->	4		
		5	RIPng Response (Trigger Update) ----------> RIPng Response (regular Update) ---------->	
		6	RIPng Response <----------	TR4
		7	RIPng Response (Trigger Update) ----------> RIPng Response (regular Update) ---------->	

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT: the Tester sends a RIPng Response packet from TR4 and TR1 in which all Metrics are set to infinity (16).

5.5.1.5 RTE Deletion (*)

Purpose: Check that Deleted Route by setting the Metric to 16 are correctly reported in Regular Routing Update and Triggered Update on each Link

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.
- This test is dependant from the algorithm used (ie Poison Reverse, Split Horizon, and none). According to **[RFC2080]** the preferred method of operation is Poison Reverse although the implementations Should provide a per-interface control allowing No Horizoning, Split Horizoning and Poison Reverse

Discussion:

Every 30 seconds, the RIPng process is awakened to send an unsolicited Response message, containing the complete routing table, to every neighboring router.

Procedure:

- **Do The following operation for each Algorithm Selection:**
  - **No Horizoning**
  - **Split Horizon**
  - **Poison Reverse**

2. Wait For the First Regular Routing Update on Link2
3. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. `521`).
   The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group `FF02::9`.
   The Hop limit has to be set to `255`. RTEs sent by TR4 in Ripng Response are the following *:*

IPv6 Prefix	Metric	Route Tag
PRFX_4	1	0

4. Wait For the second Regular Routing Update on Link2 (between 15 and 45 seconds from the First one). A triggered Update SHOULD also be sent prior to the Regular routing Update.
5. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. `521`).
   The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group `FF02::9`.
   The Hop limit has to be set to `255`. RTEs sent by TR4 in Ripng Response are the following *:*

IPv6 Prefix	Metric	Route Tag
PRFX_4	16	0

6. Wait For the Third Regular Routing Update on Link2. A triggered Update SHOULD also be sent prior to the Regular routing Update.
7. **WAIT 120s until the Garbage-Collection timer has expired.**
8. Wait For the next Regular Routing Update on Link2.

9. **TR1** sends a **RIPng Response** to give its RIPng routing table to the RUT.
The UDP ports source and destination are the RIPng port (ie. `521`).
The source address is the Link-local address of TR1and the destination address is the RIPng Multicast group `FF02::9`.
The Hop limit has to be set to `255`. RTEs sent by TR1 in Ripng Response are the following *:*

IPv6 Prefix	Metric	Route Tag
PRFX_1	1	0

10. Wait For the next Regular Routing Update on Link2 (between 15 and 45 seconds from the previous one). A triggered Update SHOULD also be sent prior to the Regular routing Update.

11. **TR1** sends a **RIPng Response** to give its RIPng routing table to the RUT.
The UDP ports source and destination are the RIPng port (ie. `521`).
The source address is the Link-local address of TR1and the destination address is the RIPng Multicast group `FF02::9`.
The Hop limit has to be set to `255`. RTEs sent by TR1 in Ripng Response are the following *:*

IPv6 Prefix	Metric	Route Tag
PRFX_1	16	0

12. Wait For the next Regular Routing Update on Link2 (between 15 and 45 seconds from the previous one). A triggered Update SHOULD also be sent prior to the Regular routing Update.

13. **WAIT 120s until the Garbage-Collection timer has expired.**

14. Wait For the next Regular Routing Update on Link2.

Observable Results:

- **Step 1:**The Regular Routing Update on Link2 should contain:

  i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0

  ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0

  iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0

- **Step 3:**

  - A triggered Update SHOULD be sent prior on Link2 according to the algorithm used. It should contain:

    i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_4	2	0

    ii. **With Split Horizon**

    No Triggered Update SHOULD be sent on Link2

    iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_4	16	0

- If a Regular Routing Update is not received 45 seconds after the previous one, the test verdict will be FAIL. The Regular Routing Update on Link2 should contain:

  i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0
PRFX_4	2	0

  ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0

  iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0
PRFX_4	16	0

- **Step 5:**
  - A triggered Update SHOULD be sent prior on Link2 according to the algorithm used. It should contain:

IPv6 Prefix	Metric	Route Tag
PRFX_4	16	0

  - If a Regular Routing Update is not received 45 seconds after the previous one, the test verdict will be FAIL. The Regular Routing Update on Link2 should contain:

    i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0
PRFX_4	16	0

    ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0

    iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0
PRFX_4	16	0

- **Step 6:** Route for PRFX_4 MUST now be removed of the RUT.
- **Step 7:** The Regular Routing Update on Link2 should contain:

  i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0

  ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0

  iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0

- **Step 9:**
  - A triggered Update SHOULD be sent prior on Link2. It should contain:

IPv6 Prefix	Metric	Route Tag
PRFX_1	2	0

  - If a Regular Routing Update is not received 45 seconds after the previous one, the test verdict will be FAIL. The Regular Routing Update on Link2 should contain:

  i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0
PRFX_1	2	0

  ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_1	2	0

  iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0
PRFX_1	2	0

- **Step 11:**
  - A triggered Update SHOULD be sent prior on Link2. It should contain:

IPv6 Prefix	Metric	Route Tag
PRFX_1	16	0

- If a Regular Routing Update is not received 45 seconds after the previous one, the test verdict will be FAIL. The Regular Routing Update on Link2 should contain:

  i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0
PRFX_1	16	0

  ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_1	16	0

  iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0
PRFX_1	16	0

- **Step 13:** The Regular Routing Update on Link2 should contain:

  i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0

  ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0

  iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response (regular Update) ---------->	
		2	RIPng Response <----------	TR4
		3	RIPng Response (Trigger Update) ----------> RIPng Response (regular Update) ---------->	
		4	RIPng Response <----------	TR4
		5	RIPng Response (Trigger Update) ----------> RIPng Response (regular Update) ---------->	

6：Wait 120 seconds. The Garbage-collection timer will have expired for PRFX_4.

		7	RIPng Response (regular Update) ---------->	
TR1	RIPng Response ---------->	8		
		9	RIPng Response (Trigger Update) ----------> RIPng Response (regular Update) ---------->	
TR1	RIPng Response ---------->	10		
		11	RIPng Response (Trigger Update) ----------> RIPng Response (regular Update) ---------->	

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT: the Tester sends a RIPng Response packet from TR1 and TR4 in which all Metrics are set to infinity (16).

5.5.2 Unsollicited Answers Processing

5.5.2.1 Regular Routing Update Timer (*)

Purpose: Check the correct handling of the Regular Routing Update. Ie. Check that Regular routing Update are handled by a timer with a random value between 15 and 45s.

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.
- This test is dependant from the algorithm used (ie Poison Reverse, Split Horizon, and none). According to **[RFC2080]** the preferred method of operation is Poison Reverse although the implementations Should provide a per-interface control allowing No Horizoning, Split Horizoning and Poison Reverse

Discussion:

Every 30 seconds, the RIPng process is awakened to send an unsolicited Response message, containing the complete routing table, to every neighboring router. When there are many routers on a single network, there is a tendency for them to synchronize with each other such that they all issue updates at the same time. This can happen whenever the 30 second timer is affected by the processing load on the system. It is undesirable for the update messages to become synchronized, since it can lead to unnecessary collisions on broadcast networks.for more details). Therefore, implementations are required to take one of two precautions:

- The 30-second updates are triggered by a clock whose rate is not affected by system load or the time required to service the previous update timer.

- The 30-second timer is offset by a small random time (+/- 0 to 15 seconds) each time it is set. The offset is derived from: 0.5 * the update period (i.e. 30).

Procedure:

- **Do The following operation for each Algorithm Selection:**
  - **No Horizoning**
  - **Split Horizon**
  - **Poison Reverse**

1. Wait For the First Regular Routing Update on Link1
2. Wait For the second Regular Routing Update on Link1. (between 15 and 45 seconds). Let's call **T12**, the delay between the first and the second Regular Routing Update.
3. Wait For the third Regular Routing Update on Link1. (between 15 and 45 seconds more). Let's call **T23**, the delay between the first and the second Regular Routing Update.
4. Wait For the fourth Regular Routing Update on Link1. (between 15 and 45 seconds more). Let's call **T34**, the delay between the first and the second Regular Routing Update.
5. Wait For the First Regular Routing Update on Link2
6. Wait For the second Regular Routing Update on Link2. (between 15 and 45 seconds). Let's call **T'12**, the delay between the first and the second Regular Routing Update on Link2.
7. Wait For the third Regular Routing Update on Link2. (between 15 and 45 seconds more). Let's call **T'23**, the delay between the first and the second Regular Routing Update on Link2.
8. Wait For the fourth Regular Routing Update on Link2. (between 15 and 45 seconds more). Let's call **T'34**, the delay between the first and the second Regular Routing Update on Link2.

Observable Results:

- **Step 4:** If a Regular Routing Update is not received 45 seconds after the previous one on Link 1, the test verdict will be FAIL. Each Regular Routing Update should contain:

  i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0

  ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_11	1	0

  iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	16	0
PRFX_11	1	0

  Moreover to avoid the synchronization problem the different delays **T12**, **T23**, **T34** MUST be all different.

- **Step 8:** If a Regular Routing Update is not received 45 seconds after the previous one on Link2, the test verdict will be FAIL. Each Regular Routing Update should contain:

  i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0

  ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0

  iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0

  Moreover to avoid the synchronization problem the different delays **T'12**, **T'23**, **T'34** MUST be all different.

Test Sequence:

Time between Responses	Link1	RUT	Link2	Tester (TRi)
0	RIPng Response (regular Update) <----------	1		
T12	RIPng Response (regular Update) <----------	2		
T23	RIPng Response (regular Update) <----------	3		
T34	RIPng Response (regular Update) <----------	4		
		5	RIPng Response (regular Update) ---------->	0
		6	RIPng Response (regular Update) ---------->	T'12
		7	RIPng Response (regular Update) ---------->	T'23
		8	RIPng Response (regular Update) ---------->	T'34

Postamble:

Because the Routing Table of the RUT is clear nothing should be done in this postamble.

5.5.2.2 Garbage-collection Time (*)

Purpose: Check that route are correctly announced during the garbage-collection time.

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.
- This test is dependant from the algorithm used (ie Poison Reverse, Split Horizon, and none). According to **[RFC2080]** the preferred method of operation is Poison Reverse although the implementations Should provide a per-interface control allowing No Horizoning, Split Horizoning and Poison Reverse

Discussion:

There are two timers associated with each route, a "timeout" and a "garbage-collection time." Upon expiration of the timeout, the route is no longer valid; however, it is retained in the routing table for a short time so that neighbors can be notified that the route has been dropped. Upon expiration of the garbage-collection timer, the route is finally removed from the routing table.

The timeout is initialized when a route is established, and any time an update message is received for the route. If 180 seconds elapse from the last time the timeout was initialized, the route is considered to have expired, and the deletion process described below begins for that route.

Deletions can occur for one of two reasons: the timeout expires, or the metric is set to 16 because of an update received from the current router (see section 2.4.2 for a discussion of processing updates from other routers). In either case, the following events happen:

- The garbage-collection timer is set for 120 seconds.

- The metric for the route is set to 16 (infinity). This causes the route to be removed from service.

- The route change flag is to indicate that this entry has been changed

- The output process is signalled to trigger a response.

Until the garbage-collection timer expires, the route is included in all updates sent by this router. When the garbage-collection timer expires, the route is deleted from the routing table.

Should a new route to this network be established while the garbage-collection timer is running, the new route will replace the one that is about to be deleted. In this case the garbage-collection timer must be cleared

Procedure:

- **Do The following operation for each Algorithm Selection:**
  - **No Horizoning**
  - **Split Horizon**
  - **Poison Reverse**

1. Wait For the First Regular Routing Update on Link2
2. **TR2** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521). The source address is the Link-local address of TR2 and the destination address is the RIPng Multicast group FF02::9. Moreover the Hop limit has to be set to 255. RTEs sent by TR2 in Ripng Response are the following :

IPv6 Prefix	Metric	Route Tag
PRFX_2	1	0

3. Wait 180 seconds: The timeout should have expired juste after a Routing Update. Wait for the next Triggered Update on Link2 (in maximum 5 seconds) since the next Routing Update is in 14 to 44s.
4. Check all the regular Routing Update on Link2 during 110 seconds.
5. Wait 10 seconds. The Garbage-collection timer will have now expired.
6. Wait For the next Regular Routing Update on Link2

Observable Results:

- **Step 1:**The Regular Routing Update on Link2 should contain:

  i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0

  ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0

  iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0

- **Step 3:**The Triggered Update on Link2 should at least contain whatever the algorithm used:

IPv6 Prefix	Metric	Route Tag
PRFX_2	16	0

  It MAY also contains some other entries.

- **Step 4:** All the Regular Routing Update on Link2 should contain:

  i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0
PRFX_2	16	0

  ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_2	16	0

  iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0
PRFX_2	16	0

- **Step 6:**The Regular Routing Update on Link2 should contain:

i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0

ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0

iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response (regular Update) ---------->	
TR2	RIPng Response ---------->	2		
		3	RIPng Response (regular Update) ---------->	

3：Wait 180 seconds. The Timeout wil have expired for TR4

		4	RIPng Response (regular Update) ----------> Check all the regular Routing Update on Link2 during 110 seconds	

5：Wait 10 seconds. The Garbage-collection timer wil have expired for TR4

		6	RIPng Response (regular Update) ---------->	

Postamble:

Because the Routing Table of the RUT is clear nothing should be done in this postamble. Nevertheless it could be required to clear all remaining routes in the RUT by sending a RIPng Response packet from TR2 in which all Metrics are set to infinity(16).

5.5.2.3 Trigger Update Limitation (*)

Purpose: Check the correct handling of the Regular Routing Update. Ie. Check that Regular routing Update are handled by a timer with a random value between 15 and 45s.

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.
- This test is dependant from the algorithm used (ie Poison Reverse, Split Horizon, and none). According to **[RFC2080]** the preferred method of operation is Poison Reverse although the implementations Should provide a per-interface control allowing No Horizoning, Split Horizoning and Poison Reverse

Discussion:

Every 30 seconds, the RIPng process is awakened to send an unsolicited Response message, containing the complete routing table, to every neighboring router. When there are many routers on a single network, there is a tendency for them to synchronize with each other such that they all issue updates at the same time. This can happen whenever the 30 second timer is affected by the processing load on the system. It is undesirable for the update messages to become synchronized, since it can lead to unnecessary collisions on broadcast networks.for more details). Therefore, implementations are required to take one of two precautions:

- The 30-second updates are triggered by a clock whose rate is not affected by system load or the time required to service the previous update timer.

- The 30-second timer is offset by a small random time (+/- 0 to 15 seconds) each time it is set. The offset is derived from: 0.5 * the update period (i.e. 30).

Triggered updates require special handling for two reasons. First, experience shows that triggered updates can cause excessive loads on networks with limited capacity or networks with many routers on them.

Therefore, the protocol requires that implementors include provisions to limit the frequency of triggered updates. After a triggered update is sent, a timer should be set for a random interval between 1 and 5 seconds. If other changes that would trigger updates occur before the timer expires, a single update is triggered when the timer expires. The timer is then reset to another random value between 1 and 5 seconds. Triggered updates may be suppressed if a regular update is due by the time the triggered update would be sent.

Second, triggered updates do not need to include the entire routing table. In principle, only those routes which have changed need to be included. Therefore messages generated as part of a triggered update must include at least those routes that have their route change flag set. They may include additional routes, at the discretion of the implementor; however, sending complete routing updates is strongly discouraged. When a triggered update is processed, messages should be generated for every directly-connected network. Split Horizon processing is done when generating triggered updates as well as normal updates. If, after Split Horizon processing for a given network, a changed route will appear unchanged on that network (e.g., it appears with an infinite metric), the route need not be sent. If no routes need be sent on that network, the update may be omitted. Once all of the triggered updates have been generated, the route change flags should be cleared.

Procedure:

- **Do The following operation for each Algorithm Selection:**
    - **No Horizoning**
    - **Split Horizon**
    - **Poison Reverse**

1. Wait For the First Regular Routing Update on Link2

2. **TR2** sends a **RIPng Response** to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. `521`).
   The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group `FF02::9`.
   The Hop limit has to be set to `255`. RTEs sent by TR4 in Ripng Response are the following *:*

IPv6 Prefix	Metric	Route Tag
PRFX_2	4	0

3. Wait For the next Triggered Update (between 1s to 5s) on Link2.

4. **TR2** quickly sends 3 **RIPng Responses** to give its RIPng routing table to the RUT.
   The UDP ports source and destination are the RIPng port (ie. `521`).
   The source address is the Link-local address of TR4 and the destination address is the RIPng Multicast group `FF02::9`.
   The Hop limit has to be set to `255`. RTEs sent by TR4 in the 3 Ripng Responses will be the following *:*

a.

IPv6 Prefix	Metric	Route Tag
PRFX_2	2	0x21
PRFX_4	1	0x21

b.

IPv6 Prefix	Metric	Route Tag
PRFX_2	1	0x22
PRFX_4	16	0x22
PRFX_8	1	0x22

c.

IPv6 Prefix	Metric	Route Tag
PRFX_2	3	0x23

5. Wait For the next Triggered Update (between 1s to 5s) on Link2.

Observable Results:

- **Step 1:**The Regular Routing Update on Link2 should contain:

  i. **With No Horizoning**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	1	0

  ii. **With Split Horizon**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0

  iii. **With Poison Reverse**

IPv6 Prefix	Metric	Route Tag
PRFX_10	1	0
PRFX_11	16	0

- **Step 3:**The Triggered Update on Link2 should at least contain whatever the algorithm used:

IPv6 Prefix	Metric	Route Tag
PRFX_2	4	0

It MAY also contains some other entries.

- **Step 5:** Because all three Update have been sent very quicly (less than 1 seconds)**,** only the final Triggered Update has to be sent on Link2. It should at least contain whatever the algorithm used:

IPv6 Prefix	Metric	Route Tag
PRFX_2	3	0x23
PRFX_4	16	0
PRFX_8	1	0x22

It MAY also contains some other entries.

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
		1	RIPng Response (regular Update) ---------->	
TR2	RIPng Response ---------->	2		
		3	RIPng Response (Triggered Update) ---------->	
TR2	RIPng Response ---------->	4a		
	RIPng Response ---------->	4b		
	RIPng Response ---------->	4c		
		5	RIPng Response (Triggered Update) ---------->	

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT: the Tester sends a RIPng Response packet from TR2 in which all Metrics are set to infinity (16).

5.6 Routing Process

5.6.1.1 Route Selection

Purpose: The goal of this test is to check that the correct route with the longest prefix is chosen by the RUT even if another route with a lower metric is available

References:

- **[RFC2080]**

Resource Requirement:

- None

Test Requirement:

- PRFX_10 has to be announced by the RUT on Link1.
- PRFX_11 has to be announced by the RUT on Link2.

Discussion:

The route with the longest prefix must be chosen to forward packets even if another route with a lower metric is available.

Packet:

- *RTEs sent by TR1 in RIPng Response are the following :*

IPv6 Prefix	Metric	Route Tag
2001:10::f1f1/0	1	0x1

- *RTEs sent by TR3 in RIPng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_3/52	2	0x3
PRFX_4/64	2	0x3

- *RTEs sent by TR4 in RIPng Response are the following :*

IPv6 Prefix	Metric	Route Tag
PRFX_3/64	3	0x4
PRFX_4/52	3	0x4

Procedure:

1. **TR1** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521).
   The source address is the Link-local address of TR1 and the destination address is the RIPng Multicast group FF02::9.
   The Hop limit has to be set to 255. This response contains the route 2001:10::f1f1/0 which is the global address of the RUT on Link1 with a prefix length of 0. It must be considered as the default route since the prefix length is 0.

2. **TR3** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521).
   The source address is the Link-local address of TR3 and the destination address is the RIPng Multicast group FF02::9.
   The Hop limit has to be set to 255.
3. **TR4** sends a **RIPng Response** to give its RIPng routing table to the RUT. The UDP ports source and destination are the RIPng port (ie. 521).
   The source address is the Link-local address of TR3 and the destination address is the RIPng Multicast group FF02::9.
   The Hop limit has to be set to 255.
4. **TR2** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521).
   The source address is the global address of TR2 and the destination address is the global address of the RUT on Link1.
   The Hop limit has to be set to 255.
   The request is done for PRFX_3/52,PRF_3/64, PRFX_4/52, PRFX_4/64 and the Default Route (0:0:0:0:0:0:0:0 with a prefix length of 0).
5. **TR3** sends a **RIPng Request** to get some information of the routing table of the RUT.
   The UDP ports source is set to 777 and destination is the RIPng port (ie. 521).
   The source address is the global address of TR3 and the destination address is the global address of the RUT on Link2. The request is done for PRFX_3/52,PRF_3/64, PRFX_4/52, PRFX_4/64 and the Default Route (0:0:0:0:0:0:0:0 with a prefix length of 0).
6. **TR2** sends an **Echo Request** through the RUT to:
   a. PRFX_3::1
   b. PRFX_4::1
   c. PRFX_15::1

   The source Address is TR2 Global Address.
7. **TR3** sends an **Echo Request** through the RUT to:
   a. PRFX_3::1
   b. PRFX_4::1
   c. PRFX_15::1

   The source Address is TR3 Global Address.

Observable Results:

- **Step 1,2, 3:** Regular routing updates are generated every 30s. The corresponding response contains the whole routing table. Because routes are created, an update has to be triggered. Nevertheless, triggered update may be suppressed if a regular update is due by the time the triggered update would be sent. Route Entries with metric 15 or 16 have not to be taken into account by the RUT.
- **Step 4:** A Response has to be generated by the RUT on link1.
  The Response source address is the RUT Global address of Link 1, the destination address is TR2 global address, the source port is the RIPng port source, the destination port is 777.
  The Response should contain the following fields:

IPv6 Prefix	Metric	Route Tag
PRFX_3/52	3	0x3
PRFX_3/64	4	0x4
PRFX_4/52	4	0x4
PRFX_4/64	3	0x3
0:0:0:0:0:0:0:0/0	2	0x1

- **Step 5:** A Response has to be generated by the RUT on link2.
  The Response source address is the RUT Global address of Link 2, the destination address is TR3 global address, the source port is the RIPng port source and the destination port is 777.
  The Response should contain the following fields:

IPv6 Prefix	Metric	Route Tag
PRFX_3/52	3	0x3
PRFX_3/64	4	0x4
PRFX_4/52	4	0x4
PRFX_4/64	3	0x3
0:0:0:0:0:0:0:0/0	2	0x1

- **Step 6:** The **Echo Request** for PRFX_3::1 MUST be forwarded to **TR4** and the **Echo Request** for PRFX_4::1 MUST be forwarded to **TR3**. The chosen route MUST have the longest prefix even if an alternative route exist with a lower prefix. The **Echo Request** for PRFX_15::1 MUST be forwarded to **TR1.**
- **Step 7:** The **Echo Request** for PRFX_3::1 MUST be forwarded to **TR4** and the **Echo Request** for PRFX_4::1 MUST be forwarded to **TR3**. The chosen route MUST have the longest prefix even if an alternative route exist with a lower prefix. The **Echo Request** for PRFX_15::1 MUST be forwarded to **TR1.**

Test Sequence:

Tester (TRi)	Link1	RUT	Link2	Tester (TRi)
TR1	RIPng Response ---------->	1		
TR2		2	RIPng Response <----------	TR3
TR2		3	RIPng Response <----------	TR4
TR2	RIPng Request ---------->	4		
TR2	RIPng Response <----------	Step4		
		5	RIPng Request <----------	TR3
		Step5	RIPng Response ---------->	TR3
TR2	Echo Request (PRFX_3::1) ---------->	6(a)		
		Step6(a)	Echo Request (PRFX_3::1) ---------->	TR4
TR2	Echo Request (PRFX_4::1) ---------->	6(b)		
		Step6(b)	Echo Request (PRFX_4::1) ---------->	TR3
TR2	Echo Request (PRFX_15::1) ---------->	6(c)		
TR1	Echo Request (PRFX_15::1) <----------	Step6(c)		
		7(a)	Echo Request (PRFX_3::1) <----------	TR3
		Step7(a)	Echo Request (PRFX_3::1) ---------->	TR4
		7(b)	Echo Request (PRFX_4::1) <----------	TR3
		Step7(b)	Echo Request (PRFX_4::1) ---------->	TR3
		7(c)	Echo Request (PRFX_15::1) <----------	TR3
TR1	Echo Request (PRFX_15::1) <----------	Step7(c)		

Postamble:

Because the Routing Table of the RUT has been modified, it is needed to clear all routes in the RUT: the Tester sends a **Response Packet** from **TR1**, **TR3** and **TR4** in which all Metrics are set to infinity (16).

5.7 Future Extensions

- router sollicitation check (NDP)
- Link down correctly propagated
- ICMP Redirect and routing
- Several global or LL @ => Answer/Update should use the same @ everytime
- Test d'unsollicited answer may be enhanced by using some different way to update or delete a route such as described in previous section Update/Delete
- Test of a Request in which the PREFIX Length is a sub PREFIX of a network announced by the RUT.

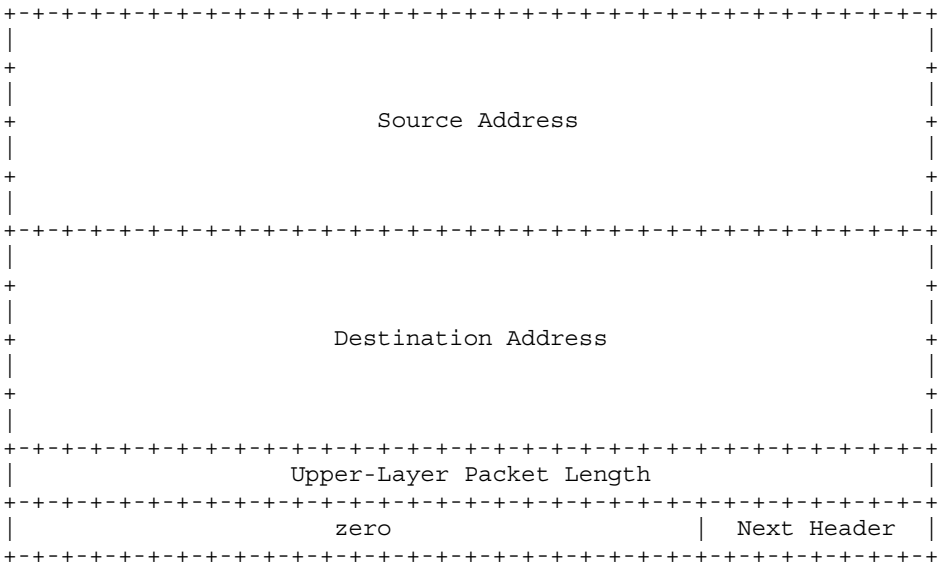
6 Annexes

6.1 IPv6 packets Checksum computation

In this annexe we present the way IPv6 related checksums of this test suite have to be computed..

6.1.1 Pseudo-header

Any transport or other upper-layer protocol that includes the addresses from the IP header in its checksum computation must be modified for use over IPv6, to include the 128-bit IPv6 addresses instead of 32-bit IPv4 addresses. In particular, the following illustration **[RFC2460]** shows the TCP and UDP "pseudo-header" for IPv6:



If the IPv6 packet contains a Routing header, the Destination Address used in the pseudo-header is that of the final destination. At the originating node, that address will be in the last element of the Routing header; at the recipient(s), that address will be in the Destination Address field of the IPv6 header.

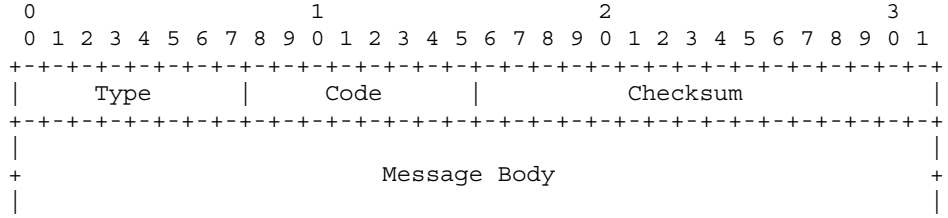
The Next Header value in the pseudo-header identifies the upper-layer protocol (e.g., 6 for TCP, or 17 for UDP). It will differ from the Next Header value in the IPv6 header if there are extension headers between the IPv6 header and the upper-layer header.

The Upper-Layer Packet Length in the pseudo-header is the length of the upper-layer header and data (e.g., TCP header plus TCP data). Some upper-layer protocols carry their own length information (e.g., the Length field in the UDP header); for such protocols, that is the length used in the pseudo-header. Other protocols (such as TCP) do not carry their own length information, in which case the length used in the pseudo-header is the Payload Length from the IPv6 header, minus the length of any extension headers present between the IPv6 header and the upper-layer header.

6.1.2 ICMPv6 Checksum

The ICMPv6 Header Format and the ICMPv6 Header Checksum calculation is defined in **[RFC2463].**

The ICMPv6 Header Format is the following:



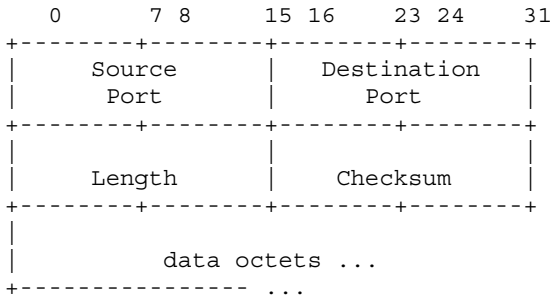
The different fields are explained in **[RFC768].**

The checksum is the 16-bit one's complement of the one's complement sum of the entire ICMPv6 message starting with the ICMPv6 message type field, prepended with the previous "pseudo-header". The Next Header value used in the pseudo-header is 58.

6.1.3 UDP and TCP Checksums

The UDP/TCP Header Formats and checksums calculation are identical in IPv4 and IPv6 but IPv6 uses the previous pseudo-header in its checksum calculation.

The UDP Header Format is the following:

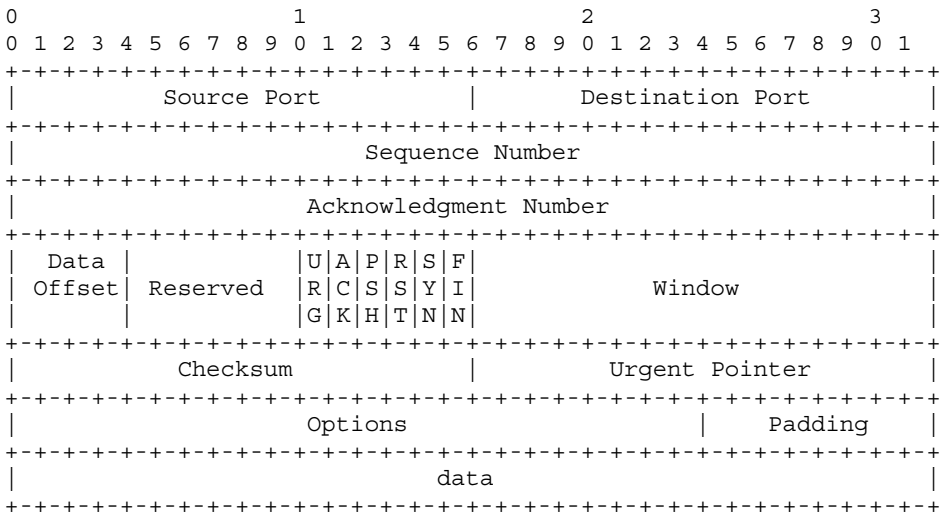


The different fields are explained in **[RFC768].**

Checksum is the 16-bit one's complement of the one's complement sum of a pseudo header of information from the IP header, the UDP header, and the data, padded with zero octets at the end (if necessary) to make a multiple of two octets.

Unlike IPv4, when UDP packets are originated by an IPv6 node, the UDP checksum is not optional. That is, whenever originating a UDP packet, an IPv6 node must compute a UDP checksum over the packet and the pseudo-header.

The TCP Header Format is the following:



The different fields are explained in **[RFC793].**

The checksum field is the 16 bit one's complement of the one's complement sum of all 16 bit words in the header and text. If a segment contains an odd number of header and text octets to be checksummed, the last octet is padded on the right with zeros to form a 16 bit word for checksum purposes. The pad is not transmitted as part of the segment. While computing the checksum, the checksum field itself is replaced with zeros.

7 REFERENCES

[RFC1058]
RFC1058, “Routing Information Protocol”, C.L. Hedrick, Jun-01-1988, RFC1723, HISTORIC.

[RFC1721]
RFC1721, RIP Version 2 Protocol Analysis, G. Malkin, November 1994, INFORMATIONAL.

[RFC1722]
RFC1722, STD0057, “RIP Version 2 Protocol Applicability Statement”,G. Malkin, November 1994, STD.

[RFC1923]
RFC1923, RIPv1 Applicability Statement for Historic Status, J. Halpern, S. Bradner, March 1996, INFORMATIONAL.

[RFC1981]
RFC 1981, “Path MTU Discovery for IP version 6”, J. McCann, S. Deering, J. Mogul, August 1996, DRAFT STANDARD [pub as:PROPOSED STANDARD]

[RFC2080]
RFC2080, “RIPng for IPv6”, G. Malkin, R. Minnear, January 1997, PROPOSED STANDARD.

[RFC2081]
RFC2081, “RIPng Protocol Applicability Statement”, G. Malkin, January 1997, INFORMATIONAL.

[RFC2119]
BCP0014, “Key words for use in RFCs to Indicate Requirement Levels”, S. Bradner, March 1997, BEST CURRENT PRACTICE.

[RFC2453]
RFC2453, STD0056, “RIP Version 2”, G. Malkin, November 1998, STD.

[RFC2460]
RFC 2460, "Internet Protocol, Version 6 (IPv6) Specification", Deering, S., and R. Hinden, December 1998, DRAFT STANDARD.

[RFC2461]
RFC 2461, "Neighbor Discovery for IP Version 6 (IPv6)", Narten, T., Nordmark, E. and W. Simpson, December 1998, DRAFT STANDARD.

[RFC2462]
RFC 2462, "IPv6 Stateless Address Autoconfiguration", Thomson, S., and T. Narten, December 1998, DRAFT STANDARD.

[RFC2463]
RFC 2463, "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", Conta, A., and S. Deering, December 1998, DRAFT STANDARD.

[RFC2464]
RFC 2464, “Transmission of IPv6 Packets over Ethernet Networks”, M. Crawford, December 1998, PROPOSED STANDARD.

[RFC3484]
RFC 3484, “Default Address Selection for Internet Protocol version 6 (IPv6)”, R. Draves, February 2003, PROPOSED STANDARD.

[RFC3513]
RFC 3513, “Internet Protocol Version 6 (IPv6) Addressing Architecture”, R. Hinden, S. Deering, April 2003, PROPOSED STANDARD.