6MAN
Internet-Draft
Intended status: Standards Track
Expires: April 26, 2011

J. Hui
Arch Rock Corporation
JP. Vasseur
Cisco Systems, Inc
D. Culler
UC Berkeley
V. Manral
IP Infusion

October 23, 2010

An IPv6 Routing Header for Source Routes with RPL
draft-ietf-6man-rpl-routing-header-01

Abstract

In Low power and Lossy Networks (LLNs), memory constraints on routers may limit them to maintaining at most a few routes. In some configurations, it is necessary to use these memory constrained routers to deliver datagrams to nodes within the LLN. The Routing for Low Power and Lossy Networks (RPL) protocol can be used in some deployments to store most, if not all, routes on one (e.g. the Directed Acyclic Graph (DAG) root) or few routers and forward the IPv6 datagram using a source routing technique to avoid large routing tables on memory constrained routers. This document specifies a new IPv6 Routing header type for delivering datagrams within a RPL domain.

Table of Contents

1. Introduction

Routing for Low Power and Lossy Networks (RPL) is a distance vector IPv6 routing protocol designed for Low Power and Lossy networks (LLN) [I-D.ietf-roll-rpl]. Such networks are typically constrained in resources (limited communication data rate, processing power, energy capacity, memory). In particular, some LLN configurations may utilize LLN routers where memory constraints limit nodes to maintaining only a small number of default routes and no other destinations. However, it may be necessary to utilize such memory-constrained routers to forward datagrams and maintain reachability to destinations within the LLN.

To utilize paths that include memory-constrained routers, RPL relies on source routing. In one deployment model of RPL, necessary mechanisms are used to collect routing information at more capable routers and form paths from those routers to arbitrary destinations within the RPL domain. However, a source routing mechanism supported by IPv6 is needed to deliver datagrams.

This document specifies the Type 4 Routing header (RH4) (to be confirmed by IANA) for use strictly within a RPL domain.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Overview

The basic format of RH4 draws from that of the Type 0 Routing header (RH0) [RFC2460]. However, RH4 introduces mechanisms to compact the source route entries when all entries share the same prefix with the IPv6 Destination Address of the encapsulating header, a typical scenario in LLNs using source routing. The compaction mechanism reduces consumption of scarce resources such as bandwidth.

RH4 also differs from RH0 in the processing rules to alleviate security concerns that lead to the deprecation of RH0 [RFC5095]. First, routers processing RH4 MUST implement a strict source route policy where each and every IPv6 hop is specified within the datagram itself. Second, a RH4 header MUST only be used within a RPL domain. RPL Border Routers, responsible for connecting RPL domains and IP domains that use other routing protocols, MUST NOT allow datagrams already carrying a RH4 header to enter or exit the RPL domain. Third, to avoid some attacks that lead to the deprecation of RH0, routers along the way MUST verify that loops do not exist with in the source route.

To deliver a datagram, a router MAY specify a source route to reach the destination using a RH4. There are two cases that determine how to include an RH4 with a datagram.

1. If the RH4 specifies the complete path from source to destination, the RH4 should be included directly within the datagram itself.

2. If the RH4 only specifies a subset of the path from source to destination, IPv6-in-IPv6 tunneling MUST be used as specified in [RFC2473]. The router MUST prepend a new IPv6 header and RH4 to the original datagram. Use of tunneling ensures that the datagram is delivered unmodified and that ICMP errors return to the source of the RH4 rather than the source of the original datagram.

In a RPL network, Case 1 occurs when both source and destinations are within a RPL domain and a single RH4 header is used to specify the entire path from source to destination, as shown in the following figure:

```
                    +--------------------+
                    |                    |
                    |  (S) -------> (D)  |
                    |                    |
                    +--------------------+
```



```
                                RPL Domain
```

In the above scenario, datagrams traveling from source, S, to destination, D, have the following packet structure:

```
                  +------+------+------+--------//-+
                  | IPv6 | IPv6 | IPv6 | Packet    |
                  | Src  | Dst  | RH4  | Payload   |
                  +------+------+------+--------//-+
```

S's address is carried in the IPv6 Source Address field. D's address is carried in the last entry of RH4 for all but the last hop, when D's address is carried in the IPv6 Destination Address field.

In a RPL network, Case 2 occurs for all datagrams that have either source or destination outside the RPL domain, as shown in the following diagram:

```
                    +-----------------+
                    |                 |
                    |  (S) -------> (BR1) -------> (D)
                    |                 |
                    +-----------------+
                         RPL Domain

                    +-----------------+
                    |                 |
                    |  (D) <------- (BR1) <------- (S)
                    |                 |
                    +-----------------+
                         RPL Domain
```

In the above scenario, datagrams traveling within the RPL domain have the following packet structure:

```
             +------+------+------+------+------+--------//-+
             | IPv6 | IPv6 | IPv6 | IPv6 | IPv6 | Packet     |
             | Src  | Dst  | RH4  | Src  | Dst  | Payload    |
             +------+------+------+------+------+--------//-+
                                  <--- Original Packet --->
              <---           Tunneled Packet           --->
```


Note that the outer header (including the RH4) is added and removed by the RPL Border Router.

Case 2 also occurs whenever a RPL router needs to insert a source route when forwarding datagram. One such use case with RPL is to have all RPL traffic flow through a Border Router and have the Border Router use source routes to deliver datagrams to their final destination. The Border Router in this case would encapsulate the received datagram unmodified using IPv6-in-IPv6 and include a RH4 in the outer IPv6 header.

```
                          +-----------------+
                          |                 |
                          |  (S) -------\   |
```

```
                         |              \  |
                         |              (BR1)
                         |              /  |
                         |  (D) <------/   |
                         |                 |
                         +-----------------+
                              RPL Domain
```

In the above scenario, datagrams travel from S to D through BR1. Between S and BR1, the datagrams are routed using the DAG built by RPL and do not contain a RH4. BR1 encapsulates received datagrams unmodified using IPv6-in-IPv6 and the RH4 is included in the outer IPv6 header.

To help avoid IP-layer fragmentation, the RH4 header has a maximum size of RH4_MAX_SIZE octets and links within a RPL domain SHOULD have a MTU of at least 1280 + 40 (outer IP header) + RH4_MAX_SIZE (+ additional extension headers or options needed within RPL domain) octets.

3. Format of the RPL Routing Header

The Type 4 Routing header has the following format:

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
 |  Next Header   |  Hdr Ext Len  | Routing Type=4| Segments Left |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 | CmprI | CmprE |  Pad  |               Reserved                |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                                                               |
 .                                                               .
 .                        Addresses[1..n]                        .
 .                                                               .
 |                                                               |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Next Header — 8-bit selector. Identifies the type of header immediately following the Routing header. Uses the same values as the IPv4 Protocol field [RFC3232].

Hdr Ext Len — 8-bit unsigned integer. Length of the Routing header in 8-octet units, not including the first 8 octets. Hdr Ext Len MUST NOT exceed RH4_MAX_SIZE / 8. Note that when Addresses[1..n] are compressed (i.e. value of CmprI or CmprE is not 0), Hdr Ext Len does not equal twice the number of Addresses.

Routing Type — 8-bit selector. Set to 4 (to be confirmed by IANA).

Segments Left — 8-bit unsigned integer. Number of route segments remaining, i.e., number of explicitly listed intermediate nodes still to be visited before reaching the final destination. Value MUST be between 0 and Segments, inclusive.

CmprI — 4-bit unsigned integer. Number of prefix octets from each segment, except than the last segment, that are elided. For example, a Type 4 Routing header carrying full IPv6 addresses in Addresses[1..n-1] sets CmprI to 0.

CmprE — 4-bit unsigned integer. Number of prefix octets

from the segment that are elided. For example, a Type 4 Routing header carrying a full IPv6 address in Addresses[n] sets CmprE to 0.

Pad 4-bit unsigned integer. Number of octets that are used to for padding after Address[n] and the end of the Type 4 Routing header.

Address[1..n] Vector of addresses, numbered 1 to n. Each vector element in [1..n-1] has size (16 - CmprI) and element [n] has size (16-CmprE).

The Type 4 Routing header shares the same basic format as the Type 0 Routing header [RFC2460]. When carrying full IPv6 addresses, the CmprI, CmprE, and Pad fields are set to 0 and the only difference between the Type 4 and Type 0 encodings is the value of the Routing Type field.

A common network configuration for a RPL domain is that all nodes within a LLN share a common prefix. Type 4 Routing header introduces the CmprI, CmprE, and Pad fields to allow compaction of the Address[1..n] vector when all entries share the same prefix as the IPv6 Destination Address field of the encapsulating datagram. The CmprI and CmprE field indicates the number of prefix octets that are shared with the IPv6 Destination Address of the encapsulating header. The shared prefix octets are not carried within the Routing header and each entry in Address[1..n-1] has size (16 - CmprI) octets and Address[n] has size (16 - CmprE) octets. When CmprI or CmprE is non-zero, there may exist unused octets between the last entry, Address[n], and the end of the Routing header. The Pad field indicates the number of unused octets that are used for padding. Note that when CmprI and CmprE are both 0, Pad MUST be null and carry a value of 0.

The Type 4 Routing header MUST NOT specify a path that visits a node more than once. When generating a Type 4 Routing header, the source may not know the mapping between IPv6 addresses and nodes. Minimally, the source MUST ensure that IPv6 Addresses do not appear more than once and the IPv6 Source and Destination addresses of the encapsulating datagram do not appear in the Type 4 Routing header.

Multicast addresses MUST NOT appear in a Type 4 Routing header, or in the IPv6 Destination Address field of a datagram carrying a Type 4 Routing header.

4. RPL Router Behavior

4.1. Generating Type 4 Routing Headers

To deliver an IPv6 datagram to its destination, a router may need to generate a new Type 4 Routing header and specify a strict source route. Routers MUST use IPv6-in-IPv6 tunneling, as specified in [RFC2473] to include a new Type 4 Routing header in datagrams that are sourced by other nodes. This ensures that the delivered datagram remains unmodified and that ICMPv6 errors generated by a Type 4 Routing header are sent back to the router that generated the routing header.

Performing IP-in-IP encapsulation may grow the datagram to a size larger than the IPv6 min MTU of 1280 octets. To help avoid IP-layer fragmentation caused by IP-in-IP encapsulation, links within a RPL domain SHOULD be configured with a MTU of at least 1280 + 40 (outer IP header) + RH4_MAX_SIZE (+ additional extension headers or options needed within RPL domain) octets.

4.2. Processing Type 4 Routing Headers

As specified in [RFC2460], a routing header is not examined or processed until it reaches the node identified in the Destination Address field of the IPv6 header. In that node, dispatching on the Next Header field of the immediately preceding header causes the Routing header module to be invoked.

The function of Type 4 Routing header is intended to be very similar to IPv4's Strict Source and Record Route option [RFC0791]. When processing the Type 4 Routing header, a router MUST drop the packet if the next entry is not on-link and SHOULD send an ICMP Destination Unreachable (ICMPv6 Type 1) message with ICMPv6 Code set to 7 (to be confirmed by IANA) to the packet's Source Address. An ICMPv6 Code of 7 indicates that the next Address entry is not on-link and the router cannot satisfy the strict source route. When generating ICMPv6 error messages, the rules in Section 2.4 of [RFC4443] must be observed.

To detect loops in the Type 4 Routing headers, a router MUST determine if the Type 4 Routing header includes more than one address assigned any interface on that router. If such addresses appear more than once, the router MUST drop the packet and SHOULD send an ICMP Parameter Problem, Code 0, to the Source Address.

The following describes the algorithm performed when processing a Type 4 Routing header:

```
    if Segments Left = 0 {
       proceed to process the next header in the packet, whose type is
       identified by the Next Header field in the Routing header
    }
    else {
       compute n, the number of addresses in the Routing header, by
       n = ((Hdr Ext Len * 8) - Pad) / (16 - Comp)

       if Segments Left is greater than n {
          send an ICMP Parameter Problem, Code 0, message to the Source
          Address, pointing to the Segments Left field, and discard the
          packet
       }
       else {
          decrement Segments Left by 1;
          compute i, the index of the next address to be visited in
          the address vector, by subtracting Segments Left from n

          if Address[i] or the IPv6 Destination Address is multicast {
             discard the packet
          }
          else if 2 entries in Address[1..n] are assigned to local
                  interface and are separated by an address not assigned
                  to local interface {
             discard the packet
          }
          else if i < n and Address[i] is not on-link {
             send an ICMP Destination Unreachable, Code 7, message to
             the Source Address and discard the packet
          }
          else {
             swap the IPv6 Destination Address and Address[i]

             if the IPv6 Hop Limit is less than or equal to 1 {
                send an ICMP Time Exceeded -- Hop Limit Exceeded in
                Transit message to the Source Address and discard the
                packet
             }
             else {
```

```
                decrement the Hop Limit by 1

                resubmit the packet to the IPv6 module for transmission
                to the new destination
             }
          }
       }
    }
```


5. RPL Border Router Behavior

RPL Border Routers (referred to as LBRs in [I-D.ietf-roll-terminology]) are responsible for ensuring that a Type 4 Routing header is only used within the RPL domain it was created.

For datagrams entering the RPL domain, RPL Border Routers MUST drop received datagrams that contain a Type 4 Routing header in the IPv6 Extension headers.

For datagrams exiting the RPL domain, RPL Border Routers MUST check for a Type 4 Routing header. If Segments Left is 0, the router MUST remove the RH4 header from the datagram and update the IPv6 Payload Length field accordingly. If Segments Left is non-zero, the router MUST drop the datagram.

6. Security Considerations

6.1. Source Routing Attacks

[RFC5095] deprecates the Type 0 Routing header due to a number of significant attacks that are referenced in that document. Such attacks include network discovery, bypassing filtering devices, denial-of-service, and defeating anycast.

Because this document specifies that Type 4 Routing headers are only for use within a RPL domain, such attacks cannot be mounted from outside the RPL domain. As described in Section 5, RPL Border Routers MUST drop datagrams entering or exiting the RPL domain that contain a Type 4 Routing header in the IPv6 Extension headers.

6.2. ICMPv6 Attacks

The generation of ICMPv6 error messages may be used to attempt denial-of-service attacks by sending error-causing Type 4 Routing headers in back-to-back datagrams. An implementation that correctly follows Section 2.4 of [RFC4443] would be protected by the ICMPv6 rate limiting mechanism.

7. IANA Considerations

This document defines a new IPv6 Routing Type of 4 (to be confirmed).

This document defines a new ICMPv6 Destination Unreachable Code of 7 to indicate that the router does not have the next Address element as a neighbor and could not satisfy the strict source route.

8. Protocol Constants

RH4_MAX_SIZE 136

With a base header size of 8 octets, 136 octets will allow for up to 8 16-octet address entries in the Type 4 Routing header. More entries are possible within 136 octets when compression is used.

9. Acknowledgements

The authors thank Richard Kelsey, Erik Nordmark, Pascal Thubert, and Tim Winter for their comments and suggestions that helped shape this

document.

10. Changes

(This section to be removed by the RFC editor.)

Draft 01:

- Allow Addresses[1..n-1] and Addresses[n] to have a different number of bytes elided.

11. References

11.1. Normative References

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[RFC2473] Conta, A. and S. Deering, "Generic Packet Tunneling in IPv6 Specification", RFC 2473, December 1998.

[RFC4443] Conta, A., Deering, S., and M. Gupta, "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", RFC 4443, March 2006.

[RFC5095] Abley, J., Savola, P., and G. Neville-Neil, "Deprecation of Type 0 Routing Headers in IPv6", RFC 5095, December 2007.

11.2. Informative References

[I-D.ietf-roll-rpl] Winter, T., Thubert, P., Brandt, A., Clausen, T., Hui, J., Kelsey, R., Levis, P., Networks, D., Struik, R., and J. Vasseur, "RPL: IPv6 Routing Protocol for Low power and Lossy Networks", draft-ietf-roll-rpl-13 (work in progress), October 2010.

[I-D.ietf-roll-terminology] Vasseur, J., "Terminology in Low power And Lossy Networks", draft-ietf-roll-terminology-04 (work in progress), September 2010.

[RFC3232] Reynolds, J., "Assigned Numbers: RFC 1700 is Replaced by an On-line Database", RFC 3232, January 2002.

Authors' Addresses

Jonathan W. Hui
Arch Rock Corporation
501 2nd St. Ste. 410
San Francisco, California 94107
USA

Phone: +415 692 0828
Email: jhui@archrock.com

JP Vasseur
Cisco Systems, Inc
11, Rue Camille Desmoulins
Issy Les Moulineaux,   92782
France

Email: jpv@cisco.com

David E. Culler
UC Berkeley
465 Soda Hall
Berkeley, California 94720
USA

Phone: +510 643 7572
Email: culler@cs.berkeley.edu

Vishwas Manral
IP Infusion

Bamankhola, Bansgali
Almora, Uttarakhand 263601
India

Phone: +91-98456-61911
Email: vishwas@ipinfusion.com