Roll Working Group   O. Bergmann
Internet-Draft   C. Bormann
Intended status: Standards Track   S. Gerdes
Expires: May 3, 2018   Universitaet Bremen TZI
                                                        H. Chen
                                             Huawei Technologies
                                                 October 30, 2017

Constrained-Cast: Source-Routed Multicast for RPL
draft-ietf-roll-ccast-01

Abstract

   This specification defines a protocol for forwarding multicast
   traffic in a constrained node network employing the RPL routing
   protocol in non-storing mode.

Table of Contents

1. Introduction

   As defined in [RFC6550], RPL Multicast assumes that the RPL network
   operates in Storing Mode. Multicast DAOs are used to indicate
   subscription to multicast address to a parent; these DAOs percolate
   up and create bread-crumbs. This specification, although part of RFC
   6550, appears to be incomplete and untested. More importantly,
   Storing Mode is not in use in constrained node networks outside
   research operating environments.

   The present specification addresses multicast forwarding for RPL
   networks in the much more common Non-Storing Mode. Non-Storing is
   based on the root node adding source-routing information to downward
   packets. Evidently, to make this work, RPL multicast needs to
   source-route multicast packets. A source route here is a list of
   identifiers to instruct forwarders to relay the respective IP
   datagram.

   As every forwarder in an IP-based constrained node network has at
   least one network interface, it is straight-forward to use the
   address of an outgoing interface as identifiers in this source-route.
   (Typically, this is a globally unique public address of the node's
   only network adapter.)

The source-route subsets the whole set of potential forwarders available in the RPL DODAG to those that need to forward in order to reach known multicast listeners.

Including an actual list of outgoing interfaces is rarely applicable, as this is likely to be a large list of 16-byte IPv6 addresses. Even with [RFC6554] style compression, the size of the list becomes prohibitively quickly.

1.1. Terminology

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

In this specification, the term "byte" is used in its now customary sense as a synonym for "octet".

All multi-byte integers in this protocol are interpreted in network byte order.

2. The BIER Approach

Bit-Indexed Explicit Replication [I-D.ietf-bier-architecture] lists all egress routers in a bitmap included in each multicast packet. This requires creating a mostly contiguous numbering of all egress routers; more importantly, BIER requires the presence of a network map in each forwarders to be able to interpret the bitmap and map it to a set of local outgoing interfaces.

3. The Constrained-Cast Approach

Constrained-Cast employs Bloom Filters [BLOOM] as a compact representation of a match or non-match for elements in a large set: Each element to be included is hashed with multiple hash functions; the result is used to index a bitmap and set the corresponding bit. To check for the presence of an element, the same hash functions are applied to obtain bit positions; if all corresponding bits are set, this is used to indicate a match. (Multiple hash functions are most easily obtained by adding a varying seed value to a single hash algorithm.)

By including a bloom filter in each packet that matches all outgoing interfaces that need to forward the packet, each forwarder can efficiently decide whether (and on which interfaces) to forward the packet.

4. False Positives

   Bloom filters are probabilistic. A false positive might be
   indicating a match where the bits are set by aliasing of the hash
   values. In case of Constrained-Cast, this causes spurious
   transmission and wastes some energy and radio bandwidth. However,
   there is no semantic damage (hosts still filter out unneeded
   multicasts). The total waste in energy and spectrum can be
   visualized as the false-positive-rate multiplied by the density of
   the RPL network. A network can easily live with a significant
   percentage of false positives. By changing the set of hash functions
   (i.e., seed) over time, the root can avoid a single node with a false
   positive to become an unnecessary hotspot for that multicast group.

5. Protocol

   The protocol uses DAO-like "MLAO" messages to announce membership to
   the root as specified in Section 5.1.

   For downward messages, the root adds a new routing header that
   includes a hash function identifier and a seed value; another one of
   its fields gives the number of hash functions (k) to ask for k
   instances of application of the hash function, with increasing seed.
   The format of the new routing header is specified in Section 5.2.

   Typical sizes of the bloom filter bitmap that the root inserts into
   the packet can be 64, 128, or 256 bit, which may lead to acceptable
   false positive rates if the total number of forwarders in the 10s and
   100s. (To do: write more about the math here. Note that this number
   tallies forwarding routers, not end hosts.)

   A potential forwarder that receives a multicast packet adorned with a
   constrained-cast routing header first checks that the packet is
   marked with a RPL rank smaller than its own (loop prevention). If
   yes, it then forwards the packet to all outgoing interfaces that
   match the bloom filter in the packet.

5.1. Multicast Listener Advertisement Object (MLAO)

   The header format of the MLAO is depicted in Figure 1. The basic
   structure of the MLAO message is similar to the RPL Destination
   Advertisement Object (DAO). In particular, it starts with RPL ICMP
   base header with a type value of 155 and the code {IANA TBD1} (MLAO),
   followed by the Checksum, RPLInstanceID, parameters and flags as in a
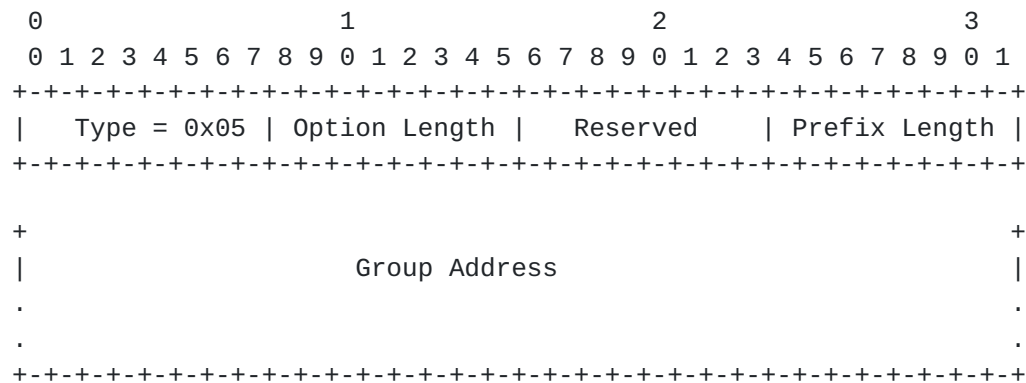   DAO.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Type = 0x05 | Option Length |   Reserved    | Prefix Length |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
+                                                               +
|                        Group Address                          |
.                                                               .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: RPL Target Option for MLAO

The group address field indicates the group that the sender of the
MLAO is interested in. This field usually contains a 128 bit IPv6
multicast group address. Shorter group identifiers could be used
together with a protocol for explicit creation of groups. The MLAO
message must have at least one RPL target option to specify the
address of the listener that has generated the MLAO. The message is
directed to the global unicast address of the DODAG root and travels
upwards the routing tree.

Note: It has been suggested to use the RPL Transit Option (Type
   0x06) instead as it is used in Non-Storing mode to inform the
   DODAG root of path attributes. Specifically, this option can be
   used to limit the subscription by providing a proper Path
   Lifetime.

5.2. Routing Header

This specification uses a new Soure Routing 6LowPAN Routing Header
(SRH-6LoRH) type [RFC8138] to convey the Bloom filter that describes
the source route for the IPv6 multicast packet to take within the RPL
routing tree. The 6LoRH Type for this Constrained Cast Routing
Header (CCRH) is set to TBD7. Figure 2 depicts the format of this
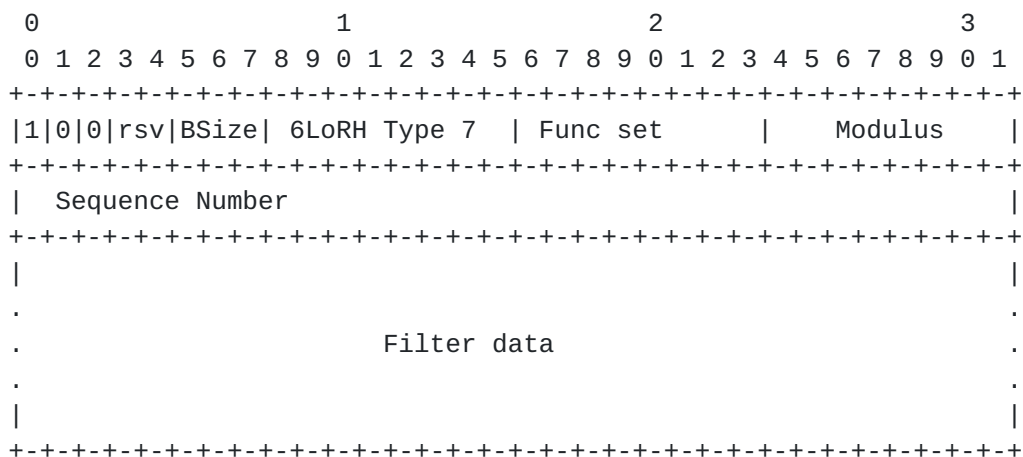new routing header.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|1|0|0|rsv|BSize| 6LoRH Type 7  | Func set      |   Modulus     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Sequence Number                                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
.                                                               .
.                        Filter data                           .
.                                                               .
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: Routing header

rsv: This field is reserved for future use an MUST be set to 0 when
   sending a packet containing a CCRH. A receiver MUST ignore the
   value of the rsv field.

BSize: Specifies the size of the included Bloom filter. Currently,
   only 64 bits, 128 bits, or 256 bits are supported. The BSize
   field is encoded as specified in Table 1.

```
          +-------------+--------------------+
          | BSize value | Filter Size (Bits) |
          +-------------+--------------------+
          | 0           | 64                 |
          | 1           | 128                |
          | 2           | 256                |
          +-------------+--------------------+
```

Table 1: Possible Bloom Filter Lengths

Func set: The set of hash functions used to generate the Filter data
   value.

Note: As the function set contains a combination of several distinct
hash functions, it is currently unclear if 8 bits number space is
large enough.

Modulus: The modulus that is used by the hash functions, minus 64
   (the minimum filter data size that can be used). The DAG root
   chooses the modulus (and thus the filter data size) to achieve its
   objectives for false positive rates (Section 4).

Sequence Number: 32 bits sequence number. The number space is
   unique for a sequence of multicast datagrams for a specific group
   that arrive at the DAG root on their way up. The DAG root
   increments the number for each datagram it sends down the
   respective DODAG.

Filter data: A bit field that indicates which nodes should relay
   this multicast datagram. The length of this field is a multiple
   of 8 bytes (2^(BSize + 3) bytes). The actual length is derived
   from the contents of the field BSize.

6. Implementation

In 2013, Constrained-Cast was implemented in Contiki. It turns out
that forwarders can compute the hash functions once for their
outgoing interfaces and then cache them, simply bit-matching their
outgoing interface hash bits against the bloom filter in the packet
(a match is indicated when all bits in the outgoing interface hash
are set in the bloom filter).

The Root computes the tree for each multicast group, computes the
bloom filter for it, caches these values, and then simply adds the
bloom filter routing header to each downward packet. For adding a
new member, the relevant outgoing interfaces are simply added to the
bloom filter. For removing a leaving member, however, the bloom
filter needs to be recomputed (which can be sped up logarithmically
if desired).

7. Benefits

Constrained-Cast:

o operates in Non-Storing Mode, with the simple addition of a
   membership information service;

o performs all routing decisions at the root.

Further optimizations might include using a similar kind of bloom
filter routing header for unicast forwarding as well (representing,
instead of the outgoing interface list, a list of children that
forwarding parents need to forward to).

8. Security Considerations

TODO

9. IANA Considerations

   The following registrations are done following the procedure
   specified in [RFC6838].

   Note to RFC Editor: Please replace all occurrences of "[RFC-XXXX]"
   with the RFC number of this specification and "TBD1" with the code
   selected for TBD1 below and "TBD7" with the code selected for TBD7.

9.1. ICMPv6 Parameter Registration

   IANA is requested to add the following entry to the Code fields of
   the RPL Control Codes registry:

```
                   +------+------+------------+
                   | Code | Name | Reference  |
                   +------+------+------------+
                   | TBD1 | MLAO | [RFC-XXXX] |
                   +------+------+------------+
```

9.2. Critical 6LowPAN Routing Header Type Registration

   IANA is requested to add the following entries to the Critical
   6LowPAN Routing Header Type Registration registry:

```
               +-------+----------------------+------------+
               | Value | Name                 | Reference  |
               +-------+----------------------+------------+
               |  TBD7 | CCast Routing Header | [RFC-XXXX] |
               +-------+----------------------+------------+
```

10. Acknowledgments

   Thanks to Yasuyuki Tanaka for valuable comments.

   This work has been supported by Siemens Corporate Technology.

11. References

11.1. Normative References

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

   [RFC6550] Winter, T., Ed., Thubert, P., Ed., Brandt, A., Hui, J.,
              Kelsey, R., Levis, P., Pister, K., Struik, R., Vasseur,
              JP., and R. Alexander, "RPL: IPv6 Routing Protocol for
              Low-Power and Lossy Networks", RFC 6550,
              DOI 10.17487/RFC6550, March 2012, <https://www.rfc-
              editor.org/info/rfc6550>.

   [RFC8138] Thubert, P., Ed., Bormann, C., Toutain, L., and R. Cragie,
              "IPv6 over Low-Power Wireless Personal Area Network
              (6LoWPAN) Routing Header", RFC 8138, DOI 10.17487/RFC8138,
              April 2017, <https://www.rfc-editor.org/info/rfc8138>.

   [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC
              2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,
              May 2017, <https://www.rfc-editor.org/info/rfc8174>.

11.2. Informative References

   [BLOOM]   Bloom, B., "Space/time trade-offs in hash coding with
              allowable errors", ISSN 0001-0782, ACM
              Press Communications of the ACM vol 13 no 7 pp 422-426,
              1970, <http://doi.acm.org/10.1145/362686.362692>.

   [I-D.ietf-bier-architecture]
              Wijnands, I., Rosen, E., Dolganow, A., Przygienda, T., and
              S. Aldrin, "Multicast using Bit Index Explicit
              Replication", draft-ietf-bier-architecture-08 (work in
              progress), September 2017.

   [RFC6554] Hui, J., Vasseur, JP., Culler, D., and V. Manral, "An IPv6
              Routing Header for Source Routes with the Routing Protocol
              for Low-Power and Lossy Networks (RPL)", RFC 6554,
              DOI 10.17487/RFC6554, March 2012, <https://www.rfc-
              editor.org/info/rfc6554>.

   [RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type
              Specifications and Registration Procedures", BCP 13,
              RFC 6838, DOI 10.17487/RFC6838, January 2013,
              <https://www.rfc-editor.org/info/rfc6838>.

Authors' Addresses

Olaf Bergmann
Universitaet Bremen TZI
Postfach 330440
Bremen D-28359
Germany

Phone: +49-421-218-63904
Email: bergmann@tzi.org

Carsten Bormann
Universitaet Bremen TZI
Postfach 330440
Bremen D-28359
Germany

Phone: +49-421-218-63921
Email: cabo@tzi.org

Stefanie Gerdes
Universitaet Bremen TZI
Postfach 330440
Bremen D-28359
Germany

Phone: +49-421-218-63906
Email: gerdes@tzi.org

Hao Chen
Huawei Technologies
12, E. Mozhou Rd
Nanjing 211111
China

Phone: +86-25-5662-7052
Email: philips.chenhao@huawei.com