

Delay-Tolerant Networking Research Group

E. Birrane

Internet-Draft Johns Hopkins University Applied Physics Laboratory

Intended status: Experimental

October 01, 2013

Expires: April 04, 2014

Contact Graph Routing Extension Block
draft-irtf-dtnrg-cgreb-00

Abstract

This draft describes an extension block used to convey path information in networks using Contact Graph Routing (CGR). The CGR Extension Block (CGR-EB) captures the set of contacts comprising the message path calculated by a CGR agent prior to forwarding a message.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 04, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Internet-Draft

CGR-EB

October 2013

Table of Contents

1.	Introduction	2
1.1.	Goals	2
1.1.1.	Hybrid Source-Path Routing	3
1.1.2.	Graph Synchronization	3
1.1.3.	Congestion Modeling	4
1.2.	Scope	4
1.3.	Requirements Language	4
2.	CGR Extension Block Format	4
3.	CGR Block Processing	7
4.	Processing Procedures	8
4.1.	Block Validation	8
4.2.	Local Graph Update	9
5.	Policy Considerations	10
5.1.	Validation Threshold	11
5.2.	Validation Lookahead	11
5.3.	Block Replacement	11
5.4.	Block Trimming	12
5.5.	Local Contact Replacement	12
5.6.	ECC Modification	12
6.	IANA Considerations	12
7.	Security Considerations	12
8.	References	12
8.1.	Normative References	13
8.2.	Informative References	13
	Author's Address	13

[1.](#) Introduction

This document describes an extension to the Delay-Tolerant Networking (DTN) Bundle Protocol (BP) [[RFC5050](#)] that marks bundles with routing information from an instance of the Contact Graph Routing [[CGR](#)] algorithm. This information establishes a both a nominal path calculated for this bundle through the DTN as well as a snapshot of the graph information available to the route-originating node at the time the bundle routing information was generated [[Acta](#)].

[1.1.](#) Goals

Bundle agents may use the information in this block in one of three ways: (1) to implement a source-path routing system in DTNs whose dynamicism would otherwise confuse opportunistic forwarding

decisions, (2) to synchronize pieces of contact graph information across portions of a network, and (3) to infer congestion metrics across heavily used paths in a DTN.

[1.1.1.](#) Hybrid Source-Path Routing

The standard CGR routing algorithm is a hop-by-hop forwarding algorithm which produces a set of plausible next-hops for a bundle in a DTN given a source and destination node. By only persisting the next hop, CGR must be re-run on a bundle at every hop in the network. This is advantageous behavior in networks whose topology changes faster than new contact graphs can be synchronized in the network.

However, this is also expensive behavior as the CGR calculations are computationally intensive [PAPER REF], as they involve calculating multiple paths through the network to derive plausible next hops. Further, the opportunistic nature of CGR as a per-hop forwarding algorithm place restrictions on the types of cost functions that may be used to route bundles. Namely, since there is no mechanism for carrying path information with a bundle as it moves through the DTN, the cost function must provide some over-arching, out-of-band information to keep the distributed algorithm from falling into routing loops.

Alternatively, source path routing refers to the practice of encoding the nominal message delivery path with the bundle. Since the path computation is atomic and not spread across multiple nodes in the network, any cost function may be selected by the network for message routing. The path, once discovered, will be followed even if the network topology changes at a later time, eliminating the danger of falling into a routing loop.

Source-path routing, however, fails to capture the case when the original, nominal path is invalidated by the dynamics of the network, either because a link in the DTN has been lost, or is congested with other traffic. When this occurs, a hybrid mechanism is preferred: the nominal path is used whenever the nominal path is feasible. If, and only if, the nominal path is unfeasible is a new path generated. This new path replaces the old path and the bundle continues as before.

[1.1.2.](#) Graph Synchronization

The hybrid source-path approach requires that a downstream node not only be able to understand the nominal path, but to evaluate it in the context of its local contact graph. This requires that the path information not only identify the links comprising the path, but also critical characteristics and assumptions that held when the path was created. This information is required to determine whether the links still exist at the evaluating node and, if they do, that they still have the necessary characteristics (data rate, available capacity, end time, etc...).

This information is determined from the contact graph of the node that populated the extension block. Downstream nodes that inspect the extension block, beyond validation, may choose to use the information in the extension block to update their own, local contact graph if the link information in the extension block represents a more recent set of information about the state of the network. This provides the potential for a path-based synchronization mechanism.

[1.1.3.](#) Congestion Modeling

Nominal paths represent the anticipated path of a bundle through the DTN. Since this path is only abandoned when it loses feasibility (as opposed to more simply losing optimality) the path is expected to be honored in all cases where the network remains relatively stable. This provides every node in the network with a set of anticipated hops, and associated times, for each bundle it sees. Together this information provides a lower bound estimate on future traffic going over downstream links.

Each node may use this path information to inform the Estimated Capacity Consumption (ECC) of links in its own local contact graph. This significantly provides a congestion model that is both automatic and predicted, eliminating the need to perform congestion management as an out-of-band management function.

[1.2.](#) Scope

[1.3.](#) 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 [RFC 2119](#) [[RFC2119](#)].

2. CGR Extension Block Format

The CGR block conforms to sections [4.5.2](#) and [4.6](#) of [[RFC5050](#)], contrained as follows:

- o Block type code is 0xED.
- o The following block processing control flag MUST be set to 1:
 - * Bit 0 - block must be replicated in every fragment.

The setting of other block processing control flags, where not mandated by the Bundle Protocol specification, is an implementation matter.

The extension block content is illustrated in Figure 1.

CGR Extension Block Format

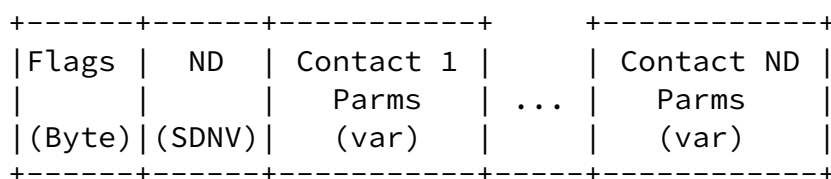
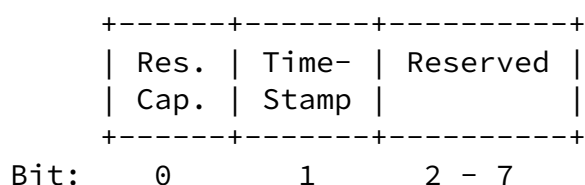


Figure 1

Flags

The CGR block flags identify the amount of contact parameter information available for each of the following contact parameters. The format of the byte is as follows.



Res. Cap. Whether the residual capacity associated with each contact is provided (1) or not (0).

Timestamp Whether the last update timestamp associated with each contact is provided (1) or not (0).

ND

The network distance of the encoded path. This specifies the number of contacts remaining in the block.

Contact The information about the nth contact. Contacts are listed in order from close-to-source to close-to-destination.

Contact parameters contain the information necessary to determine if the contact exists in the same manner at the local node and, if not, to update the contact in the local contact graph. The parameters are illustrated in Figure 2.

Contact Parameters

+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
Start	End	*Res.	Data	Rx	*Def.
Time	Time	Cap.	Rate	Node	Time
(SDNV)	(SDNV)	(SDNV)	(SDNV)	(EID)	(SDNV)
+-----+	+-----+	+-----+	+-----+	+-----+	+-----+

Figure 2

Start Time

The UTC time of when the contact will be available, encoded in an SDNV.

End Time

The UTC time of when the contact will stop being available, encoded in an SDNV.

(Optional) Residual Capacity

The estimated capacity of the link, without accounting for the bundle traveling over the link. This is measured in bytes. The current bundle is not factored into this value, as the contact may update the local contact graph as part of graph synchronization but still not use the contact if an alternate path is calculated as part of the hybrid path verification algorithm.

Data Rate

The data rate associated with this contact, measured in bps and encoded in an SDNV.

RX Node

The EID of the DTN node receiving transmission over this contact. The transmitting node is inferred based on the position of the contact parms in the block: the transmitting node for contact N is the receiving node for contact (N-1).

(Optional) Definition Time

The UTC time when the contact was authoritatively defined by the node populating this extension block. This value is used to determine whether the information associated with a particular contact in the extension block is more or less recent than the information on the same contact in the node local contact graph.

[3.](#) CGR Block Processing

The steps taken by a bundle agent when handling the CGR Extension Block, is illustrated in Figure 3.

Extension Block Lifecycle

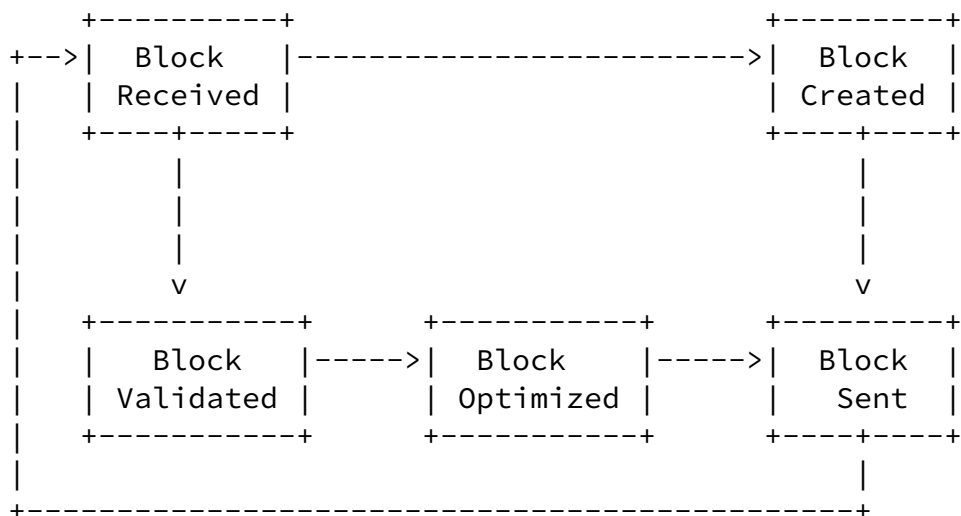


Figure 3

Block Created

A Bundle Protocol Agent will create a CGR-EB and insert it into a bundle in one of two circumstances: based on policy when a bundle is sources at the agent, or based on policy when a received bundle fails to validate an existing CGR-EB. When creating a new block, the path is always specified as from the current node onward to the bundle destination.

Block Received

Upon receiving a bundle with a CGR-EB, a BPA must validate the contents of the block to ensure that the contacts encoded within remain feasible. Before the block may move to the "validated" state, it must survive the block validation procedure. If the block fails to validate, then it must be discarded and a new block created from the local node onward. The validation procedure may validate all contacts along the specified path, or smaller number of contacts looking into the future.

Block Validated

When the block has been validated, it indicates that at least the next N hops have been validated, based on the look-ahead

the block that represent the portion of the path prior to the current BPA are assumed correct since they always represent the path taken to the current BPA. Based on the validated contacts, the local graph may be updated using the Local Graph Update procedure.

Block Optimized

Once the path in the block has been validated, a local optimization may be made to determine whether there is a faster way to get to the next hop. Since this optimization only considers ways to get to the next hop in the path, this does not risk falling into a routing loop.

Block Sent

The local BPA may add the CGR-EB to any outgoing bundle in accordance with associated processing flags. There is no requirement for positioning of the block within the bundle. The only constraint on the bundle is that only a single CGR-EB may exist in a bundle at any time.

[4.](#) Processing Procedures

This section identifies the pseudo-code associated with the core processing procedures.

[4.1.](#) Block Validation

The validate block procedure examines a subset of contacts in the block, looks up their associated information from the local contact graph, and matches them within some tolerance. There is no need to check contacts that were traversed prior to reaching the current BPA. The core variables and functions that comprise this procedure are listed as follows.

LOOKAHEAD

The policy associated with a BPA may choose to evaluate only the next hop in the path, the entire path, or some number of hops in-between.

C[i]

The variable C[i] represents the ith set of contact parameters encoded in the block.

FIND_CONTACT

The BPA must specify a look-up function that accepts a set of contact parameters from a block and must produce the contact in the local contact graph that matches these parameters, within some specified tolerance.

The pseudo-code listing is as follows.

```
PROC VALIDATE_BLOCK()
```

```
  MAX = MIN[ND,LOOKAHEAD]
```

```
  MIN = Index of contact terminating in local node
```

```
  FOR I = MIN..MAX
```

```
    IF I > 0 THEN
```

```
      CUR_TX = C[I-1].rxNode
```

```
    ELSE
```

```
      CUR_TX = Previous hop for this bundle.
```

```
    END IF
```

```
    LC = FIND_CONTACT(C[I].start, C[I].end, CUR_TX, C[I].rxNode)
```

```
    IF LC = NIL
```

```
      RETURN FALSE
```

```
    END IF
```

```
    IF LC does not have capacity for the bundle
```

```
      RETURN FALSE
```

```
    END IF
```

```
  END FOR
```

```
  RETURN TRUE
```

```
END PROC
```

[4.2.](#) Local Graph Update

The local graph update procedure examines those validated contacts in the extension block and compares them to the contents of the local contact graph. Two levels of update are performed by this function: (1) newer contacts parameters from the block update contacts in the graph and (2) the estimated capacity associated with transmitting the bundle is propagated through the local contact graph.

The key functions and variables used by this procedure are as

follows:

NC The number of contacts from the extension block, starting with the first in the block, that have been validated. Note that all contacts in the block representing hops prior to the current node are considered valid.

B The bundle being sent.

MAX The number of contacts in the extension block.

COPY_PARAMS This helper function copies contact parameters from the extension block into the local contact.

UPDATE_ECC This helper function reduces the contact's estimated capacity consumption value by the anticipated capacity necessary to transmit the bundle. This function is optional to implement and may simply return an unupdated ECC value.

The pseudo-code listing is as follows.

```
PROC UPDATE_LOCAL_GRAPH(NC, B)

  FOR I = 0..MAX

    IF I > 0 THEN
      CUR_TX = C[I-1].rxNode
    ELSE
      CUR_TX = Previous hop for this bundle.
    END IF

    LC = FIND_CONTACT(C[I].start, C[I].end, CUR_TX, C[I].rxNode)

    IF I <= NC THEN
      IF LC.Timestamp < C[I].Timestamp THEN
        COPY_PARAMS(LC, C[I])
      END IF
    END IF

    UPDATE_ECC(LC, B)
  END FOR
```

END PROC

[5.](#) Policy Considerations

This section identifies policy decisions available to BPAs processing CGR blocks and provides non-normative guidance on the impact of these decisions.

Birrane

Expires April 04, 2014

[Page 10]

Internet-Draft

CGR-EB

October 2013

[5.1.](#) Validation Threshold

When attempting to match contact parameters in the CGR block with a contact in the local contact graph there may be minor differences between the parameter values. Differences in transmit and receive node should not be tolerated, but minor differences in start and end times and residual capacities may not be significant enough to claim that a contact match was not made. The definition of matching thresholds, and how they are applied when matching contacts is an implementation matter left to a particular implementing bundle agent. However, some thresholds should be set for start times, end times, and residual capacity.

[5.2.](#) Validation Lookahead

The block validation procedure has the option of validating every future contact encoded in the block path or looking at some subset of future contacts.

Validating the entire encoded path has the benefit of learning, very early, whether or not a problem exists with routing the bundle as originally planned at the bundle source. This is desirable in relatively stable network topologies where even far-down-stream path problems can be worked through by the local node's CGR algorithm. For example, stable networks that evaluate a path based on congestion forecasting may prefer validating the entire remaining bundle path at each hop in the network.

Validating only the immediate next hop, or some subset of hops, will only ensure that the bundle can achieve incremental progress along its path. This is desirable behavior in networks where local nodes have more current information relating to their immediate N-hop

neighbors, where N is the coded lookahead value. In this configuration problems with the bundle path are dealt with when they present an immediate routing problem under the assumption that the nodes closest to the path problem will have the most up-to-date information on how to best route around the problem.

[5.3.](#) Block Replacement

When a CGR block fails to validate, the local BPA must make the decision to either abandon path encoding altogether or generate a new block. It is recommended that the block be replaced in networks whose routing cost functions are otherwise prone to routing loops. However, networks that have periods of high topological change may wish to abandon path routing until such time as the network stabilizes. The detection of topological change and related stabilization are implementation matters of the network.

Birrane

Expires April 04, 2014

[Page 11]

Internet-Draft

CGR-EB

October 2013

[5.4.](#) Block Trimming

The validated CGR block may be sent to the next hop intact, or trimmed. The value of sending the CGR block intact to the next hop is that the previously-used contacts may be used to inform the local graph update procedure for downstream nodes. However, if the local contact graph update procedure is not to be used in the network, or only used for to-be-traversed contacts, then the block size may be reduced by trimming old contacts.

There is no change to the CGR block format when choosing to trim contacts, as long as the remaining network diameter variable is updated to reflect the new contact value.

[5.5.](#) Local Contact Replacement

Local BPAs must decide whether to use the information in the CGR block to update contacts in the local contact graph. This requires the ability to timestamp contacts in the block and in the local contact graph. If the timestamp and residual capacity fields of the contacts are not included in the block there is no additional information outside of contact definition to update in the local graph.

[5.6.](#) ECC Modification

The decision to update the local contact graph with the ECC associated with the path of the bundle should reflect the confidence that the bundle will actually follow the prescribed path. Much like the lookahead decision when validating paths, network characteristics must be considered when applying this update. Very stable topologies may wish to update ECCs in the local contact graph for the entire path. More dynamic topologies may wish to only update the ECC for the next N hops.

6. IANA Considerations

This memo includes no request to IANA.

7. Security Considerations

Transport security is handled by the transport layer, for example the Bundle Security Protocol [[RFC6257](#)] when using the Bundle Protocol [[RFC5050](#)].

8. References

Birrane

Expires April 04, 2014

[Page 12]

Internet-Draft

CGR-EB

October 2013

8.1. Normative References

- [CGR] Burleigh, S., "Contact Graph Routing ", [draft-burleigh-dtnrg-cgr-01](#), July 2010.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.
- [RFC5050] Scott, K. and S. Burleigh, "Bundle Protocol Specification", [RFC 5050](#), November 2007.
- [RFC6257] Symington, S., Farrell, S., Weiss, H., and P. Lovell, "Bundle Security Protocol Specification", [RFC 6257](#), May 2011.

8.2. Informative References

- [Acta] Birrane, E., Burleigh, S., and N. Kasch, "Analysis of the

contact graph routing algorithm: Bounding interplanetary paths ", Acta Astronautica, Volume 75, Pages 108-119, ISSN 0094-5765, June-July 2012.

[IJAHUC] Birrane, E., "Building Routing Overlays in Disrupted Networks: Inferring Contacts in Challenged Sensor Internetworks ", International Journal of Ad Hoc and Ubiquitous Computing (IJAHUC) Special Issue on Algorithms and Protocols for Opportunistic and Delay Tolerant Networks, Volume 11, No. 2/3, Pages 139-156, 2012.

[WD] Birrane, E., "Improving Graph-Based Overlay Routing in Delay Tolerant Networks. ", IFIP Wireless Days (WD 2011), October 2011.

Author's Address

Edward J. Birrane
Johns Hopkins University Applied Physics Laboratory
11100 Johns Hopkins Road
Laurel , Maryland 20723
USA

Phone: +1 410 778 7423
Email: Edward.Birrane@jhuapl.edu