DTN Research Group
Internet-Draft
Intended status: Experimental
Expires: February 13, 2010

S. Symington
R. Durst
K. Scott
The MITRE Corporation
August 12, 2009

Delay-Tolerant Networking Bundle Diversion
draft-irtf-dtnrg-bundle-divert-01

Status of this Memo

Copyright Notice

Abstract

This document defines two extensions to the capabilities of a Bundle Protocol Agent (BPA) (as defined in [refs.DTNBP]) that is processing bundles within the context of a Delay-Tolerant Network architecture [refs.DTNarch]. It defines an operation called "diversion", which is the act of a bundle protocol agent moving an entire bundle from some point in bundle processing in the BPA to a DTN application agent. This diversion of a bundle from the BPA to an application agent is distinct from delivery of the bundle at that application agent. This document defines a second operation, called "injection", which is the inverse of diversion. Injection is the act of an application agent moving an entire bundle from the application agent into some point in bundle processing in the BPA. This injection of a bundle from an application agent to the BPA is distinct from bundle transmission.

Table of Contents

1. Introduction

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 [refs.RFC2119].

This document defines two extensions to the capabilities of a Bundle Protocol Agent (BPA) (as defined in [refs.DTNBP]) that is processing bundles within the context of a Delay-Tolerant Network architecture [refs.DTNarch]. It defines an operation called "diversion", which is the act of a bundle protocol agent moving an entire bundle from some point in bundle processing in the BPA to a DTN application agent. This diversion of a bundle from the BPA to an application agent is distinct from delivery of the bundle at that application agent. This document defines a second operation, called "injection", which is the inverse of diversion. Injection is the act of an application agent moving an entire bundle from the application agent into some point in bundle processing in the BPA. This injection of a bundle from an application agent to the BPA is distinct from bundle transmission.

The capabilities described in this document are OPTIONAL for deployment with the Bundle Protocol. Bundle Protocol implementations claiming to support bundle diversion MUST be capable of both:

-moving entire bundles from bundle processing in the BPA to an application agent in an operation that is distinct from delivery of those bundles at that application agent (diversion), and

-moving entire bundles from an application agent to bundle processing in the BPA in an operation that is distinct from bundle transmission (injection)

as defined in this document, and of

-providing a raw bundle interface between the BPA and all application agents to which bundles will be diverted or from which bundles will be injected.

2. Motivating Use Case

Bundle diversion is expected to be of general utility in DTN in instances in which all or part of a bundle needs to be transcoded while the bundle is being forwarded through the network. A bundle that is received at a BPA in one coding format can be diverted to an application agent that has as its purpose the conversion of the bundle to an alternate coding format. After transcoding, an application agent can inject a transcoded bundle back into BPA processing to be forwarded or delivered in its new format.

Bundle-in-bundle encapsulation [refs.DTNencaps] is a specific example of a use case that has been identified for bundle diversion. If a bundle is forwarded to a bundle protocol agent that is intended to serve as the entrance of an encapsulation tunnel, that BPA would divert the bundle to an encapsulation-specific application agent. This encapsulation-specific application agent would encapsulate the diverted bundle into another, encapsulating bundle and request transmission of this encapsulating bundle. The source endpoint of the encapsulating bundle would have the encapsulation-specific application agent as its member and the destination endpoint of the encapsulating bundle would have as a member the application agent that will de-encapsulate the bundle. The encapsulating bundle would therefore be transmitted from the encapsulating application agent for delivery to the de-encapsulating application agent. At the bundle node where the de-encapsulating application agent is located, the de-encapsulating application agent would extract the encapsulated bundle from the encapsulating bundle and inject that original bundle into the bundle protocol agent for processing, including forwarding and/or delivery. In order to perform such tunneling, bundle diversion is required to move the bundle to be encapsulated from BPA processing to the encapsulating application at the start of the tunnel, and bundle injection is required to move the de-encapsulated bundle from the de-encapsulating application back into BPA bundle processing at the end of the tunnel.

3. Bundle Diversion Operation

The "divert" operation occurs when some (here unspecified) mechanism, e.g. a routing table lookup, causes an entire bundle to be moved from bundle processing in the BPA to a (here unspecified, but arbitrary) DTN application agent. Diversion of a bundle from the BPA to an application agent is distinct from delivery of the bundle at that application agent. Some of the ways in which diversion differs from delivery are as follows: the application agent to which the bundle is diverted is not a member of the bundle's destination endpoint; an entire bundle is diverted, not just an application data unit and associated metadata; a bundle may be diverted at an arbitrary point in bundle processing, so that it may include extension blocks or a fragmented payload; and no bundle delivery status or custodial delivery reports are transmitted upon diversion as they might be on delivery.

Bundles may be diverted at any point after bundle transmission or reception and before bundle dispatching relative to the bundle processing procedures defined in [refs.DTNBP] and to the procedures required to process any extension blocks (e.g. security blocks, Previous Hop Insertion Block, etc.) in the bundle. The point in bundle processing at which diversion will occur should be documented as part of the specification of the AA to which the diversion will occur.

When diverting a bundle, the BPA may pass not only the bundle, but additional information, if needed, that is meaningful to the AA to which the bundle is being diverted.

4. Bundle Injection Operation

The "inject" operation occurs when some application agent at a bundle node causes an entire bundle to be moved from that AA into the BPA for bundle processing. Injection of a bundle from an application agent to the BPA is distinct from bundle transmission. Some of the ways in which injection differs from transmission are as follows: the injecting application entity is not a member of the source EID in the bundle; an entire bundle is injected by the AA, not just a application data unit; the injected bundle may already have extension blocks, a current custodian, and other values set; an injected bundle may have a fragmentary payload; and a bundle may be injected at an arbitrary point in BPA bundle processing.

Bundles may be injected at any point in BPA processing after bundle transmission or reception and before bundle dispatching relative to the bundle processing procedures defined in [refs.DTNBP] and to the procedures required to process any extension blocks (e.g. security blocks, Previous Hop Insertion Block, etc.) in the bundle. The point in bundle processing at which injection will occur should be documented as part of the specification of the AA that will perform the injection.

When injecting a bundle, the AA may pass not only the bundle, but additional information, if needed, that is meaningful to the BPA to which the bundle is being diverted.

5. Relationship of Diversion to Injection

Normally, when bundles are injected into the same BPA from which they were diverted, the bundle processing step at which a bundle is injected should be the same step at which it was diverted. For example, bundles diverted just before bundle forwarding should be injected just before bundle forwarding; bundles diverted just after bundle reception should be injected just after bundle reception; bundles diverted just before/after processing of a given extension block should be injected just before/after processing of that extension block.

Bundles injected into the same BPA from which they were diverted, MUST re-enter bundle processing so as to avoid the looping that would occur if the same bundle were to be diverted more than once at the same processing step.

When bundles are injected into a different BPA from which they were diverted, it is not possible for the BPA into which the bundles are being injected to determine, through implementation-specific means (such as from the convergence layer), the EID of the forwarding node of these injected bundles. The forwarding node of an injected bundle is the node that diverted the bundle, and the node that diverted the bundle is not necessarily a neighbor (in the DTN overlay) of the node into which the bundle is being injected. Therefore, if the EID of the forwarding node of an injected bundle is needed for use by the BPA into which the bundle is being injected, this EID MUST be present in the diverted bundle, for example, in a Previous Hop Insertion Block [refs.PrevHopExt] or in the EID reference to the security-source of a BAB [refs.DTNBPsec], if the bundle contains a BAB.

6. Security Considerations

When a bundle is diverted, the diverted bundle itself may be protected by one or more security blocks. In particular, it may contain a Bundle Authentication Block (BAB), which is designed to be processed by a next-hop neighboring DTN node. If a bundle with a BAB is diverted by one node and later injected into a non-neighboring node, the bundle protocol agent into which the bundle is injected must be capable of validating the security result in the BAB if its security policy requires such validation. Therefore, diversion and injection of bundles protected by BABs may require that keys that are normally only shared between neighbors in the DTN overlay be distributed further so that they are shared by the diverting and injecting nodes. Furthermore, as explained in the previous section, the EID of the node that creates the BAB and diverts the bundle MUST be identified within the bundle because it will not be possible for the BPA into which the bundle is injected, which is responsible for validating the security result in the BAB, to determine the EID of the BAB-creating node through implementation-specific means (such as from the convergence layer).

7. IANA Considerations

None at this time.

8. References

8.1. Normative References

[refs.RFC2119]
Bradner, S. and J. Reynolds, "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, October 1997.

[refs.DTNBP]
Scott, K. and S. Burleigh, "Bundle Protocol Specification", RFC, 5050, November 2007.

[refs.PrevHopExt]
Symington, S., "Delay-Tolerant Networking Previous Hop Insertion Block", draft-irtf-dtnrg-bundle-previous-hop-block-06.txt, work-in-progress, March 2009.

[refs.DTNencaps]
Symington, S., Durst, R., and K. Scott, "Delay-Tolerant Networking Bundle-in-Bundle Encapsulation", draft-irtf-dtnrg-bundle-encapsulation-06.txt, work-in-progress, August 2009.

[refs.DTNBPsec]
Symington, S., Farrell, S., Weiss, H., and P. Lovell, "Bundle Security Protocol Specification", draft-irtf-dtnrg-bundle-security-08.txt, work-in-progress, March 2009.

8.2. Informative References

[refs.DTNarch]
Cerf, V., Burleigh, S., Hooke, A., Torgerson, L., Durst, R., Scott, K., Fall, K., and H. Weiss, "Delay-Tolerant Network Architecture", RFC 4838, April 2007.

Authors' Addresses

Susan Flynn Symington
The MITRE Corporation
7515 Colshire Drive
McLean, VA 22102
US

Phone: +1 (703) 983-7209
Email: susan@mitre.org
URI: http://mitre.org/

Robert C. Durst
The MITRE Corporation
7515 Colshire Drive
McLean, VA 22102
US

Phone: +1 (703) 983-7535
Email: durst@mitre.org
URI: http://mitre.org/

Keith L. Scott
The MITRE Corporation
7515 Colshire Drive
McLean, VA 22102
US

Phone: +1 (703) 983-6547
Email: kscott@mitre.org
URI: http://mitre.org/