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-in-Bundle Encapsulation
draft-irtf-dtnrg-bundle-encapsulation-06

Status of this Memo

Copyright Notice

Abstract

This document defines an encapsulation-specific application agent capability and a bundle payload format for use with the Bundle Protocol [refs.DTNBP] within the context of a Delay-Tolerant Network architecture [refs.DTNarch]. It defines the capability and format for placing one or more bundles inside of the payload field of an encapsulating bundle's Bundle Payload Block.

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 an encapsulation-specific application agent capability and a bundle payload format for use with the Bundle Protocol [refs.DTNBP] within the context of a Delay-Tolerant Network architecture [refs.DTNarch]. It defines the capability and format for placing one or more bundles inside of the payload field of an encapsulating bundle's Bundle Payload Block.

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

- -generating and transmitting bundles that have Bundle Payload Blocks whose payload fields are formatted to contain one or more bundles (encapsulation), and

- -receiving and processing bundles that have Bundle Payload Blocks whose payload fields are formatted to contain one or more bundles (de-encapsulation)

as defined in this document.

Bundle protocol agents claiming to support this document MUST be capable of diverting and inserting bundles as described in [refs.DTNdivert], and must provide a raw bundle interface to the encapsulating/de-encapsulating application agents across which bundles may be passed.

2. Motivating Use Cases

The bundle-in-bundle encapsulation mechanism is expected to be of general utility in DTN. So far, three use cases have been identified for this capability:

Content-based dissemination of data from a DTN node's storage cache - A caching bundle node may have a bundle store that contains bundles with content-based metadata extension blocks [refs.Metadata]. If this caching bundle node receives a request for bundles having data of a certain content type from a particular requesting node, the caching bundle node will want to retrieve the bundle(s) matching that content type from its bundle store and send those bundle(s) to the requesting node. If the endpoint of which the requesting node is a member is different from the destination endpoint of the stored bundle(s), in order to send the bundle(s), the caching node either has to replace the destination EID of the stored bundle(s) with the EID of the requesting node or place the stored bundle(s) into one or more encapsulating bundles that have as their destination EID the EID of the requesting node. If the stored bundles are protected with a Payload Integrity Block (PIB), the caching node has no choice but to send these bundles using encapsulation, because if the destination EIDs of the stored bundles are changed, the security results in their PIB(s) will not compute to the correct value at the security destination and they will fail their integrity check.

Efficient custodial retransmission of a multicast bundle - As defined in [refs.DTNcustMC], if a custodian of a multicast bundle receives a failed custody signal from a specific downstream node, the custodian will want to get that multicast bundle to that downstream node so that it can be forwarded on from that downstream node. However, the custodian may not want to simply retransmit the multicast bundle because this may cause the bundle to be resent to many other nodes on the multicast distribution path that have already received the bundle, in addition to the node that the custodian wants the bundle to reach. As an alternative, the custodian may want to place the multicast bundle inside of an encapsulating unicast bundle and send the encapsulating bundle to the downstream node that generated the failed custody signal so that the downstream node can de-encapsulate the multicast bundle and forward it on from there.

Tunneling - Bundle-in-bundle encapsulation is a method of tunneling bundles across some portion of a network. As mentioned in [refs.DTNBPsec], in the security case, one or more bundles can be placed inside of the payload of another bundle and then the payload of the encapsulating bundle can be encrypted. The

encapsulating bundle is then sent from the encapsulating security gateway to the de-encapsulating security gateway which, in effect, form the ends of a security tunnel. This security tunnel protects the entire contents of the encapsulated bundle(s) from being disclosed, so that even the confidentiality of each bundle's source EID and destination EID are maintained on the portion of the network that is spanned by the tunnel.

3. Payload Field Format for Bundle-in-Bundle Encapsulation

Bundle-in-bundle encapsulation is accomplished by placing the bundle(s) to be encapsulated into the payload field of an encapsulating bundle's Bundle Payload Block. The elements that must be placed in the payload field of the Bundle Payload Block and their format are as follows:

- Number of Encapsulated Bundles field (SDNV) - indicates the number N of bundles that are present in the Encapsulated Bundle List field (defined below), where N MUST be greater than 0.

- Bundle Offsets List (optional) - contains the list of N-1 offsets (each of which is an SDNV) of the beginning of each bundle that is located in the Encapsulated Bundle List field after the first bundle in that field. (The offset of the first bundle in the Encapsulated Bundle List is understood to be 0.) If only 1 bundle is present in the Encapsulated Bundle List (i.e., if the value in the Number of Encapsulated Bundles field is 1), the Bundle Offsets List field MUST NOT be present.

- Encapsulated Bundle List - contains the list of N bundles that are encapsulated.

The format of the payload field of the Bundle Payload Block of an encapsulating bundle is as follows:

Payload Field Format of the Bundle Payload Block
of an Encapsulating Bundle:

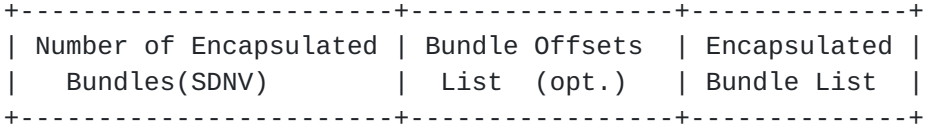

Figure 1

4. Encapsulation Requirements

4.1. Diversion of Bundles for Encapsulation

Bundles that are diverted to a bundle-in-bundle encapsulation application agent (AA) as described in [refs.DTNdivert] MUST be diverted at the following point in bundle processing, 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, Metadata Extension Block, etc.) in the bundle:

-after the bundle has been processed for reception (if it was received from another node), including the processing of all of its extension blocks (if any), e.g. after verifying the security result in the Bundle Authentication Block (BAB) [refs.DTNBPsec] (if present) and deleting the BAB, after verifying the security result in the Payload Security Block (PSB) if the node is the security destination, after using and deleting the Previous Hop Insertion Block (if present), etc., and

-after the bundle has been prepared for forwarding, having been given all necessary extension blocks, e.g. a new BAB (if appropriate), a new Previous Hop Insertion Block (if appropriate), etc., but

-before the bundle is actually forwarded or delivered.

Note that, as described in [refs.DTNdivert], if, as part of the procedures for preparing the bundle for forwarding, the bundle is given a BAB, the EID of the node that is creating this BAB MUST be identified within the bundle, either through an EID reference to the BAB security-source or using another mechanism, such as the Previous Hop Insertion Block [refs.PrevHopExt] because it will not be possible for the bundle protocol agent that is responsible for validating the security result in the BAB to determine the EID of this BAB-creating node through implementation-specific means (such as from the convergence layer).

4.2. Encapsulation Processing Steps

Upon receipt of one or more such diverted bundles, the bundle-in-bundle encapsulation AA MUST place the bundle or bundles that have been diverted to it into the Encapsulated Bundle List field of an application data unit that is formatted as described above and request transmission of a bundle containing this application data unit as the payload field of its Bundle Payload Block. If more than one bundle has been placed in the Encapsulated Bundle List field, the

value of the Number of Encapsulated Bundles field MUST have as its value the number of bundles that have been placed in the Encapsulated Bundle List and the value of the Bundle Offsets List field MUST be the list of offsets of each of the encapsulated bundles, after the first encapsulated bundle, in the payload field. If exactly one bundle has been placed in the Encapsulated Bundle List field, the value of the Number of Encapsulated Bundles field MUST have as its value the number 1 and the Bundle Offsets List field MUST NOT be present.

The parameters of the bundle transmission request for this encapsulating bundle must result in the following:

If any of the diverted bundles that were placed in the Encapsulated Bundle List has its "Custody transfer is requested" Bundle Processing Control Flag [refs.DTNBP] set, the encapsulating bundle must have its "Custody transfer is requested" Bundle Processing Control Flag set.

The value of the destination EID in the new, encapsulating bundle MUST identify an endpoint of which the node(s) that will be de-encapsulating the bundle is/are a member.

The value of the source EID of the encapsulating bundle MUST be set to an endpoint ID of which the bundle-in-bundle encapsulation AA is a member.

The value of the Creation Timestamp of the encapsulating bundle MUST be set to the time at which the node began constructing the encapsulating bundle, and the value of the Lifetime of the encapsulating bundle MUST be set such that the encapsulating bundle will expire at or later than the expiration time of all of its encapsulated bundle(s). That is, the creation time plus the lifetime of the encapsulating bundle must be greater than or equal to the creation time plus the lifetime of each of the bundles that have been placed in the Encapsulated Bundle List field.

Note that there is currently no Bundle Status Report Status Flag [refs.DTNBP] defined to indicate that the "Reporting node is encapsulating a bundle". Such a status report could possibly be helpful and perhaps should be defined in the future. If such a status report were to be sent to the current custodian (if any) of the encapsulated bundle upon its encapsulation, for example, the status report would indicate to the custodian that custody signals for the encapsulated bundle will not be received until some time after the encapsulated bundle has been de-encapsulated.

5. De-encapsulation Requirements

5.1. Injection of De-encapsulated Bundles

Upon delivery of a bundle at a de-encapsulation AA, the de-encapsulation AA MUST extract the encapsulated bundle(s) from the Encapsulated Bundle List item of the payload field of the encapsulating bundle's Bundle Payload Block and inject these de-encapsulated bundles into the bundle protocol agent at the following point in bundle processing, 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, Metadata Extension Block, etc.) in the bundle:

-after the bundle has been received from another node, but

-before it has been processed for reception, including before the processing of all of its extension blocks (if any), e.g. before reporting bundle reception, before verifying the security result in the Bundle Authentication Block (BAB) (if present) and deleting the BAB, before verifying the security result in the Payload Security Block (PSB) and/or the Payload Confidentiality Block (PCB) if the node is the security destination, before using and deleting the Previous Hop Insertion Block (if present), etc.

5.2. De-encapsulation Processing

As discussed in [refs.DTNdivert], when processing the injected bundles for reception, it will not be possible for the bundle protocol agent to determine the EID of the previous hop node of these de-encapsulated, injected bundles through implementation-specific means (such as from the convergence layer), because the previous hop node was the node that diverted the bundles to the encapsulation AA, which is not necessarily a neighboring node in the DTN overlay. Therefore, if the EID of the previous hop node is needed by the bundle protocol agent, this EID MUST be present in the de-encapsulated 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. If the EID of the previous hop node is needed but is not present in the de-encapsulated bundle, the bundle MUST be discarded and processed no further.

6. Security Considerations

Bundle-in-bundle encapsulation is an application protocol; it does not require an extension block. All encapsulated bundles and associated information needed to perform de-encapsulation are carried in the payload field of the Bundle Payload Block. Therefore, encapsulated bundles are protected, for security purposes, by all three mandatory ciphersuites defined in the Bundle Security Protocol, just as any bundle payload is.

It should be noted that when a bundle is encapsulated, the encapsulated 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 encapsulated by one node and received and de-encapsulated by a non-neighboring node, the bundle protocol agent into which the de-encapsulated bundle is injected for processing must be capable of validating the security result in the BAB if its security policy requires such validation. Therefore, as explained in [refs.DTNdivert], encapsulation of bundles protected by BABs may require that keys that are normally only shared between neighbors be distributed further in the DTN so that they are shared by the encapsulating and de-encapsulating nodes. Furthermore, because it is not possible for the bundle protocol agent into which the de-encapsulated bundle is injected to determine the EID of the diverting node that inserted the BAB into the encapsulated bundle through implementation-specific means (such as from the convergence layer), if the EID of this diverting node is not identified elsewhere in the encapsulated bundle (e.g. in the Previous Hop Insertion Block), this EID must be referenced as the security-source field of the BAB.

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.DTNdivert]
Symington, S., Durst, R., and K. Scott, "Delay-Tolerant Networking Bundle Diversion", draft-irtf-dtnrg-bundle-divert-01.txt, work-in-progress, August 2009.

[refs.DTNcustMC]
Symington, S., Durst, R., and K. Scott, "Delay-Tolerant Networking Custodial Multicast Extensions", draft-irtf-dtnrg-bundle-multicast-custodial-06.txt, work-in-progress, August 2009.

[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.Metadata]
Symington, S., "Delay-Tolerant Networking Metadata Extension Block", draft-irtf-dtnrg-bundle-metadata-block-03.txt, work-in-progress, April 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/