Internet Research Task Force (IRTF)
Request for Comments: 6258
Category: Experimental
ISSN: 2070-1721

S. Symington
The MITRE Corporation

May 2011

Delay-Tolerant Networking Metadata Extension Block

Abstract

This document defines an extension block that may be used with the Delay-Tolerant Networking (DTN) Bundle Protocol. This Metadata Extension Block is designed to carry additional information that DTN nodes can use to make processing decisions regarding bundles, such as deciding whether to store a bundle or determining to which nodes to forward a bundle. The metadata that is carried in a metadata block must be formatted according to the metadata type that is identified in the block's metadata type field. One specific metadata type, for carrying URIs as metadata, is defined in this document. Other metadata types may be defined in separate documents. This document is a product of the Delay Tolerant Networking Research Group and has been reviewed by that group. No objections to its publication as an RFC were raised.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for examination, experimental implementation, and evaluation.

This document defines an Experimental Protocol for the Internet community. This document is a product of the Internet Research Task Force (IRTF). The IRTF publishes the results of Internet-related research and development activities. These results might not be suitable for deployment. This RFC represents the consensus of the Delay-Tolerant Networking Research Group of the Internet Research Task Force (IRTF). Documents approved for publication by the IRSG are not a candidate for any level of Internet Standard; see Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc6258.

Table of Contents

1. Introduction

This document defines an extension block that may be used with the Bundle Protocol [RFC5050] within the context of a Delay-Tolerant Networking architecture [RFC4838]. The DTN Bundle Protocol [RFC5050] defines the bundle as its protocol data unit. This document defines a bundle block called a "metadata block". This block is designed to carry additional information that DTN nodes can use to make processing decisions regarding bundles.

The metadata block has been deliberately defined to be flexible enough that it would be capable of supporting a variety of metadata types and formats. Indeed, the only restriction imposed on the metadata to be used is that its type and format be predefined and registered (if public) so that it can be parsed and processed by DTN nodes that support metadata of that type. Section 4 defines a

specific metadata type and discusses the use of other metadata types that may be defined elsewhere. As mentioned, it is the intention that the metadata that is carried in this block be application-related information. For example, the metadata might be information that is associated with the payload of a bundle. Additional extension blocks could be (and have been) defined for carrying additional network-related information.

While the bundle payload may be processed opaquely by DTN nodes, metadata is intended to serve as a mechanism for providing DTN nodes with access to additional information that they can use to process the bundle. Examples of such additional information include keywords found in the payload; payload provenance information; GPS coordinates (if the payload is a map, for instance); message IDs; and artist, album, and track name (if the payload is a song). Even though the metadata is additional information related to the application, its purpose is to be used by DTN nodes to make decisions regarding how to process bundles within the network, such as whether or not a bundle should be stored or to which nodes a bundle should be forwarded. Metadata that is about bundle payload, for example, might serve as a content-based index of bundles that are stored in a DTN cache. So, in response to a request for bundles related to a certain subject or related to specific GPS coordinates, for example, the metadata of stored bundles could be searched, and all bundles with metadata matching the search criteria could be retrieved and returned to the requestor.

This document defines the general format of and the processing required to support the metadata block. The actual metadata to be inserted into a metadata block MUST be formatted according to the metadata type that is identified in the block's metadata type field. One specific metadata type, for carrying Uniform Resource Identifiers (URIs) [RFC3986] as metadata, is defined in this document. Other metadata types may be defined in separate documents, along with the steps required to process records of that type, if necessary. If such other metadata types are defined, they should be registered to ensure global uniqueness (see the IANA Considerations section).

The capabilities described in this document are OPTIONAL for deployment with the Bundle Protocol. As defined in this document, Bundle Protocol implementations claiming to support the metadata block MUST be capable of:

- generating a metadata block and inserting it into a bundle,

- receiving bundles containing a metadata block and making the information contained in this metadata block's metadata field available for use, e.g., in bundle storage or forwarding decisions, and

- deleting a metadata block from a received bundle before forwarding the bundle.

Bundle Protocol implementations claiming to support a specific metadata type must both support the metadata block as defined above and be capable of parsing and processing the metadata itself according to the definition of the metadata type in which the metadata is expressed. This metadata type may be the URI metadata type (see the URI metadata type section), or it may be another metadata type defined in a separate document.

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 RFC 2119 [RFC2119].

2. Metadata Block Format

The metadata block uses the Canonical Bundle Block Format as defined in the Bundle Protocol [RFC5050]. That is, it is comprised of the following elements, which are defined as in all bundle protocol blocks except the primary bundle block. (Note that Self-Delimiting Numeric Value (SDNV) encoding is described in the Bundle Protocol.):

- Block-type code (1 byte) - defined as in all bundle protocol blocks except the primary bundle block (as described in the Bundle Protocol). The block-type code for the metadata block is 0x08.

- Block processing control flags (SDNV) - defined as in all bundle protocol blocks except the primary bundle block. SDNV encoding is described in the Bundle Protocol. There are no constraints on the use of the block processing control flags. If a bundle node receives a bundle with a metadata block and it is capable of supporting the metadata block but it is not able to parse and process the metadata itself, either because it does not support the metadata type being used or because the metadata is not well-formed according to the metadata type definition, the bundle node must process the bundle as if it cannot process the metadata block. That is, it must operate according to the settings of the block processing control flags, including the "Delete bundle if block can't be processed" flag and the "Discard block if it can't be processed" flag.

- Block EID-reference count and EID-references (optional) - composite field defined in the Bundle Protocol that is present if and only if the metadata block references EID elements in the primary block's dictionary. Presence of this field is indicated by the setting of the "Block contains an EID-reference field" bit of the block processing control flags. If EIDs are referenced in the metadata block, then their interpretation is defined by the particular metadata type that is being used in this metadata block, as indicated in the metadata type field.

- Block data length (SDNV) - defined as in all bundle protocol blocks except the primary bundle block. SDNV encoding is described in the Bundle Protocol.

- Block-type-specific data fields as follows:

  - Metadata Type field (SDNV) - indicates which metadata type is to be used to interpret both the metadata in the metadata field and the EID-references in the optional Block EID-reference count and EID-references field (if present). One metadata type is defined in this document. Other metadata types may be defined in separate documents.

  - Metadata field - contains the metadata itself, formatted according to the metadata type that has been specified for this block. One metadata type is defined in Section 4.1. Other metadata types may be defined elsewhere, as discussed in Section 4.

The structure of a metadata block is as follows:

Metadata Block Format:

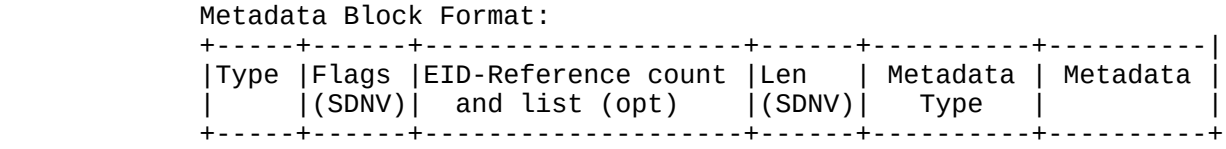

Figure 1

3. Metadata Block Processing

The following are the processing steps that a bundle node may take relative to generation, reception, and processing of metadata blocks.

3.1. Bundle Transmission

When an outbound bundle is created per the parameters of the bundle transmission request, this bundle MAY (as influenced by local policy and the metadata type being used) include one or more metadata blocks (as defined in this specification).

3.2. Bundle Forwarding

A node MAY insert one or more metadata blocks into a bundle before forwarding it; and a node MAY delete one or more metadata blocks from a bundle before forwarding it, as dictated by local policy and the metadata type being used.

3.3. Bundle Reception

If the bundle includes one or more metadata blocks, the metadata information records in these blocks SHALL be made available for use at this node (e.g., in bundle storage or forwarding decisions, or, if the receiving node is the bundle-destination, the metadata information records may be provided to the receiving application).

4. Predefined Metadata Types

As mentioned in the previous section, any number of different metadata types may be defined to indicate the format of both the metadata field and the EID-references in the optional Block EID-reference count and EID-references field (if present) and, if necessary, how metadata of this type should be processed. One metadata type is defined in this document, URI metadata type (0x01). In addition, a range of metadata type values is reserved for private use.

4.1. URI Metadata Type

It is believed that use of URIs will, in many cases, be adequate for encoding metadata, although it is recognized that use of URIs may not be the most efficient method for such encoding. Because of the expected utility of using URI encoding for metadata, the metadata type value of 0x01 is defined to indicate a metadata type of URI. Metadata type values other than 0x01 will be used to indicate alternative metadata types.

The Metadata field for metadata of metadata type URI (0x01) consists of an array of bytes formed by concatenating one or more null-terminated URIs. Unless determined by local policy, the specific processing steps that must be performed on bundles with metadata blocks containing metadata of type URI are expected to be indicated

as part of the URI encoding of the metadata. It is envisioned that users might define URI schemes for this purpose. Metadata blocks containing metadata of type URI MUST NOT include a Block EID-reference count and EID-references field. The absence of this field MUST be indicated by a value of 0 for the "Block contains an EID-reference field" flag in the block processing control flags. Support for the URI metadata type is OPTIONAL.

4.2. Private Metadata Types

Metadata type values 192 through 255 are not defined in this specification and are available for private and/or experimental use. Such private metadata types are not required to be registered. All other values of the metadata type are reserved for future use and, when defined, should be registered to ensure global uniqueness (see the IANA Considerations section). Local policy will define how private metadata types are handled.

5. Security Considerations

The DTN Bundle Security Protocol [RFC6257] defines security-related blocks to provide hop-by-hop authentication, end-to-end authentication, end-to-end confidentiality of bundles or parts of bundles, and an extension security block to provide confidentiality and integrity for extension blocks, as well as a set of standard ciphersuites that may be used to calculate security-results carried in these security blocks. All ciphersuites that use the strict canonicalization algorithm [RFC6257] to calculate and verify security-results (e.g., many hop-by-hop authentication ciphersuites) apply to all blocks in the bundle and so would apply to bundles that include an optional metadata block and would include that block in the calculation of their security-result. In particular, bundles including the optional metadata block would be protected in their entirety for the duration of a single hop, from a forwarding node to an adjacent receiving node (but not from source to destination over multiple hops), using the standard BAB-HMAC (Bundle Authentication Block - Hashed Message Authentication Code) ciphersuite defined in the Bundle Security Protocol.

Ciphersuites that use the mutable canonicalization algorithm to calculate and verify security-results (e.g., the mandatory PSH-RSA-SHA256 ciphersuite and most end-to-end authentication ciphersuites) will omit the metadata block from their calculation. Therefore, the fact that metadata in the metadata block may be modified or that metadata blocks themselves may be added to or deleted from a bundle as it transits the network will not interfere with end-to-end security protection when using ciphersuites that use mutable canonicalization.

The metadata block will not be encrypted by the mandatory CH-RSA-AES-PAYLOAD-PSH end-to-end confidentiality ciphersuite, which only allows for payload and PSH encryption.

In order to provide the metadata block with end-to-end confidentiality and authentication independent of any confidentiality or authentication that is provided for the payload or other parts of the bundle, the extension security block may be used to encrypt and authenticate the metadata block. A bundle may contain multiple metadata extension blocks. In some cases, multiple metadata blocks may be carried in the bundle, possibly with each being encrypted separately from each other and from the payload. Such separate encryption of metadata from payload would enable bundle nodes to perform content-based searching and routing on bundle metadata that they are able to decrypt, even if they are not able to decrypt the bundle payload.

Given that metadata can be modified by forwarding nodes, it may be desirable to eventually support the ability to audit changes to the metadata at the individual record level. No such capability has been provided in this specification as currently written.

6. IANA Considerations

6.1. Metadata Type Codes

The metadata block carried in the Metadata Extension Block has a Metadata Type Code field (see Sections 2 and 3). An IANA registry has been set up as follows.

Metadata Type Codes Registry

The registration policy for this registry is:
0-191: Specification Required
192-255: Private and/or Experimental Use. No assignment by IANA.

The Value range is unsigned 8-bit integer.

Value	Description	Reference
0	Reserved	This document
1	URI	This document
2-191	Unassigned	
192-255	Private and/or experimental use	This document

6.2. Block Type Code for the Metadata Block

This specification allocates a codepoint from the Bundle Block Type Codes registry defined in [RFC6255] (see Section 2 of this document):

Additional Entry for the Bundle Block Type Codes Registry:

Value	Description	Reference
8	Metadata Extension Block	This document

7. References

7.1. Normative References

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

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC5050] Scott, K. and S. Burleigh, "Bundle Protocol Specification", RFC 5050, November 2007.

[RFC6255] Blanchet, M., "Delay-Tolerant Networking (DTN) Bundle Protocol IANA Registries", RFC 6255, May 2010.

7.2. Informative References

[RFC4838] Cerf, V., Burleigh, S., Hooke, A., Torgerson, L., Durst, R., Scott, K., Fall, K., and H. Weiss, "Delay-Tolerant Networking Architecture", RFC 4838, April 2007.

[RFC6257] Symington, S., Farrell, S., Weiss, H., and P. Lovell, "Bundle Security Protocol Specification", RFC 6257, May 2011.

Author’s Address

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/