Network Working Group   S. Burleigh
Internet-Draft   D. Horres
Intended status: Standards Track   JPL, Calif. Inst. Of Technology
Expires: September 3, 2018   K. Viswanathan
                                                        M. Benson
                                                       F. Templin
                                         Boeing Research & Technology
                                                     March 2, 2018

           Architecture for Delay-Tolerant Key Administration
                    draft-burleigh-dtnwg-dtka-01.txt

   Abstract

      Delay-Tolerant Key Administration (DTKA) is a system of public-key
      management protocols intended for use in Delay Tolerant Networking
      (DTN). This document outlines a DTKA proposal for space-based
      communications, which are characterized by long communication delays
      and planned communication contacts.

   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 https://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 September 3, 2018.

   Copyright Notice

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.

Table of Contents

1. Introduction

   Delay-Tolerant Key Administration (DTKA) is a system of public-key
   management protocols intended for use in Delay Tolerant Networking
   (DTN) [RFC4838]. This document outlines a DTKA proposal for space-
   based communications, which are characterized by long communication
   delays and planned communication contacts. The proposal satisfies
   the requirements for DTN Security Key Management
   [I-D.templin-dtnskmreq].

1.1. Motivation and Design Strategy

   In general, on-demand interactive communications, like client-server
   interactions, are not feasible in DTN's network model. Terrestrial
   public-key management protocols require on-demand interactions with
   remote computing nodes to distribute and validate public-keys. For
   example, terrestrial public-key management protocols require on-
   demand interactions with a remote trusted authority (Certificate
   Revocation List (CRL)) to determine if a given public-key certificate
   has been revoked or not. Therefore, such terrestrial public-key
   management protocols cannot be used in DTN.

   Periodic and planned communications are an inherent property of
   space-based communication systems. Thus, the core principle of DTKA
   is to exploit this property of space-based communication systems in
   order to avoid the need for on-demand interactive communications for
   key management. Therefore, the design strategy for DTKA is to pro-
   actively distribute authenticated public-keys to all nodes in a given
   DTN instance in advance to ensure that keys will be available when
   needed even if there may be significant delays or disruptions. This
   design strategy is to be contrasted with protocols for terrestrial
   Public-Key Infrastructures, in which authenticated public-keys are
   exchanged interactively, just-in-time and on demand.

1.2. Scope

   DTKA was originally designed for space-based DTN environments, but it
   could potentially be used in terrestrial DTN environments as well.

1.3. About This Document

   This document describes the high-level architecture of DTKA and lists
   the architectural entities, their interactions, and system
   assumptions.

1.4. Related Documents

   The following documents provide the necessary context for the high-
   level design described in this document.

      RFC 4838 [RFC4838] describes the architecture for DTN and is
      titled, "Delay-Tolerant Networking Architecture." That document
      provides a high-level overview of DTN architecture and the
      decisions that underpin the DTN architecture.

      RFC 5050 [RFC5050] describes the protocol and message formats for
      DTN and is titled, "Bundle Protocol Specification." That document
      provides the format of the network protocol message for DTN,

called a Bundle, along with descriptions of processes for
generating, sending, forwarding, and receiving Bundles. It also
specifies an encoding format called SDNV (Self-Delimiting Numeric
Values) for use in DTN. Each bundle comprises a primary block, a
payload block, and zero or more additional extension blocks. A
node may receive and process a bundle even when the bundle
contains one or more extension blocks that the node is not
equipped to process.

RFC 6257 [RFC6257] is titled, "Bundle Security Protocol
Specification." It specifies the message formats and processing
rules for providing three types of security services to bundles,
namely: confidentiality, integrity, and authentication. It does
not specify mechanisms for key management. Rather, it assumes
that cryptographic keys are somehow in place and then specifies
how the keys shall be used to provide the security services.
Additionally, it attempts to standardize a default cipher suite
for DTN.

The revised Internet Draft [I-D.ietf-dtn-bpsec] for DTN
communication security is titled, "Bundle Security Protocol
Specification (bpsec)." When compared with RFC 6257, it is silent
on concepts such as Security Regions, at-most-once-delivery
option, and cipher suite specification. It deletes the Bundle
Authentication Block and generalized the Payload Integrity and
Payload Confidentiality Blocks to Block Integrity Block and Block
Confidentiality Block. It provides more detailed specification
for bundle canonicalization and rules for processing bundles
received from other nodes. Like RFC 6257, the draft does not
describe any key management mechanisms for DTN but assumes that a
suitable key management mechanism shall be in place.

5050bis [I-D.ietf-dtn-bpbis] is an Internet Draft on standards
track that intends to update RFC 5050. It introduces a new
concept called "node ID" as distinguished from the existing
concept of "endpoint ID": a single DTN endpoint may contain one or
more nodes. It also migrates some primary block fields into
extension blocks, making the primary block immutable. In the
Security Considerations section, 5050bis explicitly describes end-
to-end security using Block-Integrity-Block (BIB) and Block-
Confidentiality-Block (BCB). It does not specify link-by-link
security considerations to be part of the bundle protocol level
using the Bundle-Authenticity-Block (BAB), which was described in
RFC 6257. The convergence layers may provide link-by-link
authentication instead of bundle protocol agent.

The Internet Draft for specifying requirements for DTN Key
Management [I-D.templin-dtnskmreq] is titled, "DTN Security Key

Management - Requirements and Design." It sketches nine
requirements and four design criteria for DTN Key Management
system. The last two requirements are the need to support
revocation in a delay tolerant manner. It also specifies the
requirements for avoiding single points of failure and
opportunities for the presence of multiple key management
authorities.

2. Terminology

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 [RFC2119]. Lower
case uses of these words are not to be interpreted as carrying
RFC2119 significance.

3. High Level Architecture

```
+---------+                   +----------+2.ListOfAuthenticated +------+
|Key      |1.(NodeID, Key)| Key        |(Node ID, Key)          | Key  |
|Owner    +--------------> Authority +---------------------> User |
|(Node ID)|                   |          |                        |      |
+---^-----+                   +----------+                        +---^--+
    |                         3.Secure Communications                 |
    +---------------------------------------------------------------+
```
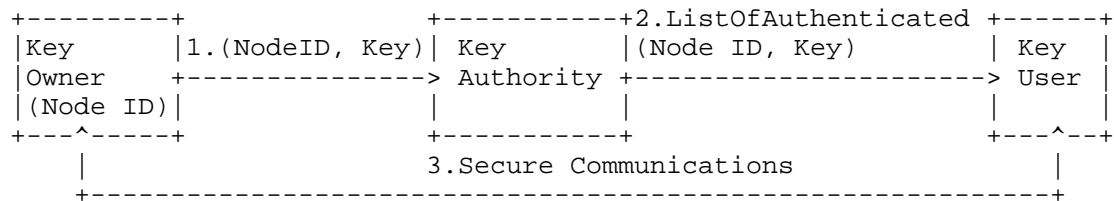
          Figure 1: Abstract Data-Flow-Diagram for DTKA

The DTKA system includes Key Owners, Key Agents (which, in aggregate,
constitute the Key Authority), and Key Users. For the sake of
simplicity and to promote conceptual clarity, Figure 1 shows a single
Key Agent. In order to avoid a single point-of-trust, DTKA provides
mechanisms to distribute the Key Authority function among one or more
DTKA Key Agents using an erasure-coding technique. This trust-
distributing mechanism is discussed later in this document.

Each Key Owner has a unique DTN Node ID and chooses its own public-
private key pair. In order to associate a public-key (Key) with its
Node ID, a Key Owner sends an assertion of the form: (Node ID, Key)
to the Key Authority. Key Owners need to authenticate their
respective keys in one of two ways:

1. in the case of out-of-band bootstrapping, Key Authority shall
    rely on the physical security of the out-of-band channel to
    validate the integrity of the received message and the Key Owner
    needs to sign the association (Node ID, Key) using the private
    key corresponding to the Key. Association realized using such an

interaction will be called Out-of-band-authentication (OOBAuth);
or,

2. in the case of in-band authentication, the Key Owner or a trusted
    third-party needs to sign the association (Node ID, Key) using
    the private key corresponding to the previously authenticated and
    currently effective public-key for that NodeID. If the Key Owner
    signs the association, there will be roll-over association. If a
    trusted third-party signs the association, the association will
    have the type endorse so as to indicate an endorsement.

Each Key User periodically receives a list of authenticated public-
keys from the Key Authority and uses the authenticated public-keys as
needed.

3.1. Application Domains

DTN can be used in various theatres such as space, airspace, on earth
and at sea. There can be more than one installation of DTN in each
of these theatres administered by different administrative entities,
which may represent countries, companies and institutions. A
particular installation of DTN with a single aggregate key authority
is called an Application Domain.

3.2. System Entities

The architectural elements of DTKA, which shall henceforth be called
DTKA Entities, are listed below.

DTKA Key Agent (DTKA-KA)
    DTKA-KA is part of the root of trust for authenticated
    distribution of public-keys for a given application domain. All
    DTKA Entities must have physically authenticated public-keys of
    all DTKA Key Agents (DTKA-KAs), which together constitute the DTKA
    Key Authority for a given application domain.

DTKA Key Owner[Node ID] (DTKA-KO[Node ID])
    DTKA-KO[Node ID] is a computing node that has possession of the
    private key corresponding to the public-key authenticated for a
    given Node Identity (Node ID) by the DTKA-KAs for the Key Owner's
    application domain.

DTKA Key User (DTKA-KU)
    DTKA-KU is a computing node that receives authenticated public-
    keys from DTKA-KAs and distributes the same within a single
    computing machine through a suitable Interprocess Communication
    mechanism, which is outside the scope of this document.

   DTKA Key Manager (DTKA-KM) and DTKA Key Manager Client (DTKA-KMC)
      DTKA-KM is a DTKA Key User that receives authenticated public-keys
      from DTKA-KAs and distributes the same over a communication
      network to DTKA-KMCs, which are not DTKA Entities. DTKA-KMC can
      be a DTN node that can receive key distributions from DTKA KMs.
      The communication and security protocols for the interactions
      between DTKA-KMs and DTKA-KMCs are outside the scope of this
      document.

3.3. System Interconnections

```
     ++                                                          ++
      ++ +----------------------------------------------------------+ ++
      ++ Sub-second One-Way-Light-Time (OWLT) and Rarely Disrupted Link ++
      ++ +---------^-----------------------------------^--------+ ++
      ++          |                                   |          ++
          +-------v-------+                   +-------v-------+
          | DTKA Entity   |                   | DTKA Entity   |
          |               |                   |               |
          | +----------+  |  ................ |  +----------+ |
          | | DTKA Key |  |  ................ |  | DTKA Key | |
          | | Agent    |  |                   |  | Agent    | |
          | +----------+  |                   |  +----------+ |
          | +----------+  |                   |  +----------+ |
          | |   TSM    |  |                   |  |   TSM    | |
          | +----------+  |                   |  +----------+ |
          +-------^-------+                   +-------^-------+
       ++         |                                   |        ++
       ++ +-------v-----------------------------------v--------+ ++
       ++      Communication Link with Delay and Disruptions      ++
        ++ +---------^-----------------------------------^--------+ ++
         ++          |                                   |          ++
          +-------v-------+                   +-------v-------+
          | DTKA Entity   |                   | DTKA Entity   |
          |               |                   |               |
          | +----------+  |                   |  +----------+ |
          | | DTKA Key |  |                   |  | DTKA Key | |
          | | Owner    |  |                   |  | User     | |
          | +----------+  |                   |  +----------+ |
          | +----------+  |                   |  +----------+ |
          | |Autonomous|  |                   |  |Autonomous| |
          | |Clock     |  |                   |  |Clock     | |
          | +----------+  |                   |  +----------+ |
          +--------------+                   +--------------+
```
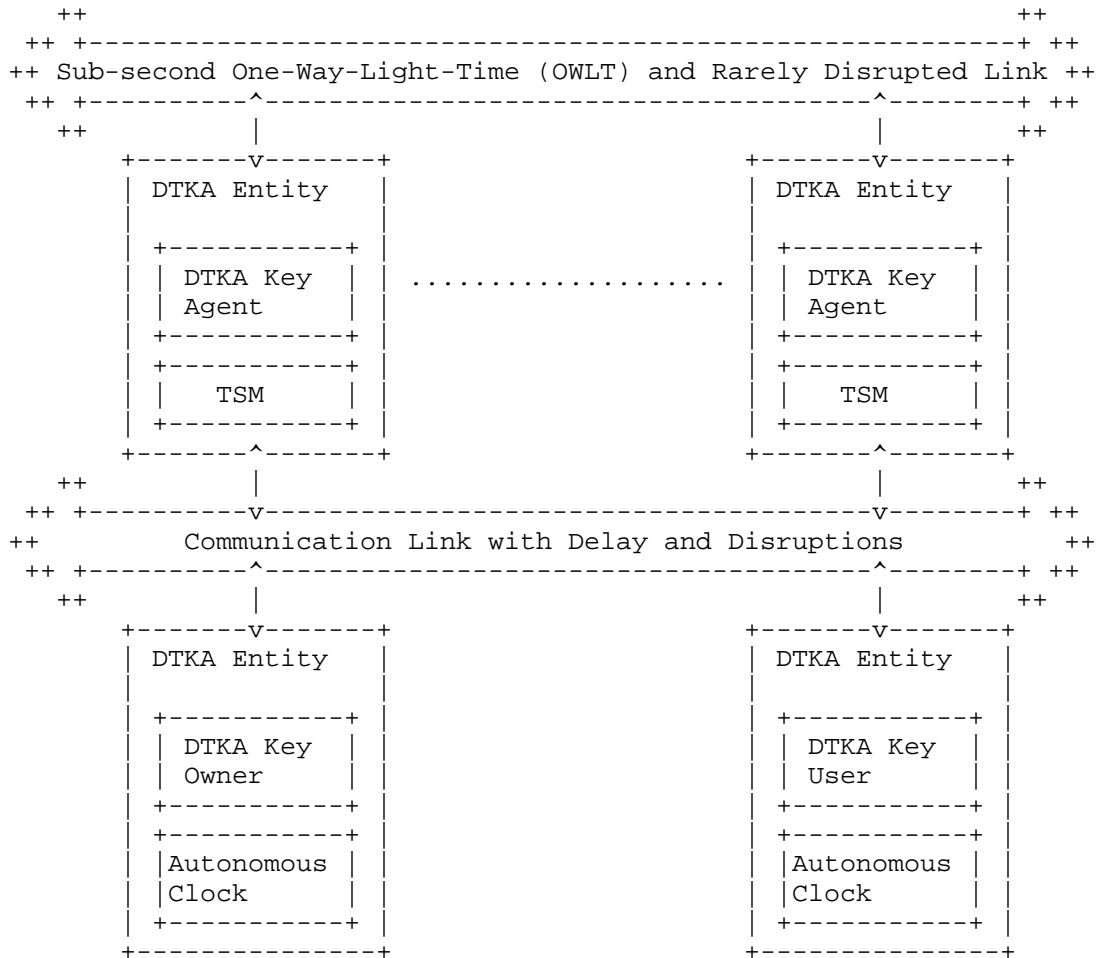
                Figure 2: DTKA System Interconnections

Figure 2 depicts the system level interconnections that are assumed
for the design of DTKA. An application domain can have one or more
DTKA-KAs, all of which must be interconnected using a sub-second One-
Way-Light-Time (OWLT) and rarely disrupted link. Such communication
link can be realized using terrestrial Internet or specialized point-
to-point space communication techniques. This link shall be used by
the DTKA-KAs to synchronize between themselves. The DTKA-KAs shall
run a reliable Time Synchronization Mechanism (TSM), like the Network
Time Protocol (NTP) service. TSM shall ensure that time is
synchronized between the DTKA-KAs that realize the DTKA Key Authority
for a given application domain.

A potentially delayed and frequently disrupted communication link is
assumed to interconnect DTKA-KAs, DTKA-KOs and DTKA-KUs. This
delayed-and-disrupted communication link is used by the DTKA-KAs to
multicast authenticated public-key associations to DTKA-KUs. The
DTKA-KUs are assumed to have access to autonomous clocks. Autonomous
clocks keep time without external correction signals and with an
allowed drift in the order of a few seconds. But, delay-tolerant
mechanisms for clock agreement such as issuance of UTC offsets in
network management messages may be present.

3.4. Architectural Assumption on Communication

In the subsequent sections, it shall be seen that DTKA-KAs shall
dispatch updates to the list of authenticated public-keys in the
system using erasure coding techniques. It is evident that at least
a sub-set of such communications updates must reach each DTKA-KU.
Therefore, the DTN upon which the DTKA operates must satisfy the
following communication assumption before DTKA can function along
expected lines: all addressed receivers MUST receive sufficient
number of bundles from the DTKA-KAs before the earliest effective
time among the effective times of all public-key associations in the
payloads of the bundles. Note that the underlying DTN will not be
aware of the effective times of the public-key associations in the
payloads of the bundles.

The above assumption can be restated using DTKA protocol
terminologies, which shall be seen in the subsequent sections, as
follows: All addressed receivers MUST receive enough of the code
blocks for a given bulletin to enable reassembly of that bulletin
before the earliest effective-time among all associations in the
bulletin.

3.5. System Security Configuration

   The current public-keys of all designated DTKA-KAs for a given
   application domain must be securely configured into every DTKA-KA and
   DTKA-KU that needs to participate in that application domain; this is
   a pre-condition for initializing those DTKA-Entities. This process
   will ensure that the DTKA Agents are established as the root of trust
   for that application domain.

4. Detailed Design

4.1. Message Formats

   Every DTKA-KA in an application domain will receive requests for
   associating public-keys with Node IDs from the respective DTKA-KOs.
   After authenticating the requests and any pending revocations (as
   described in Section 4.4 below), every DTKA-KA reaches consensus with
   all other DTKA-KAs, which constitute the Key Authority in its
   application domain, on some subset of the authenticated requests and
   revocations. The protocols and algorithms for DTKA-KA consensus is
   an implementation aspect and out-of-scope of this document. After
   each successful consensus, each DTKA-KA must increment its local
   value called Bulletin Serial Number (BSN) and agree on the Trust
   Model Number (TMN) for the bulletin. Thereafter, each DTKA-KA must
   independently multicast to all participating DTKA Entities the subset
   of authenticated list of address-and-key associations on which
   consensus was reached along with the new BSN value. The message
   format for this multicast, which is called a Bulletin, supports
   message authentication and redundancy. The goal of message
   authentication is to prevent DTKA Entities' acceptance of malicious
   multicast messages issued by hostile nodes. The goal of message
   redundancy is to ensure that a minimal set of collaborating DTKA-KAs
   in the application domain will be able to successfully send out-of-
   band-authentication (OOBAuth) or revocations for address-and-key
   associations to all DTKA Entities -- the DTKA Entities need not know
   which DTKA-KAs are not collaborating.

   As mentioned previously, bulletin is a collection of association
   blocks (or Key Information Message [KIM] data structure) such that
   each association block represents a single association of a Node ID
   with a public-key as depicted in Figure 3. Each block issues either
   an out-of-band-authentication (OOBAuth) or endorse or revoke or roll-
   over instruction to the receiving DTKA Entities, which use the key
   information message to execute the instruction locally. The
   semantics for each of the instruction shall be described in
   subsequent sections. The block labelled "Bulletin Hash" contains the
   cryptographic hash computed over all association blocks (key
   information messages), the Bulletin Serial Number (BSN) and the Trust

Model Number (TMN) in that bulletin. The BSN is a unique and
sequential bulletin identifier. The TMN is a unique identifier to
indicate the trust model configuration that is to be used to validate
this bulletin. The trust model configuration can be seen as a list
of DTKA-KAs (Key Agents), who are trusted to authenticate this
bulletin to all DTKA Users in the system. The trust model
configuration is also used to indicate the t-out-of-n threshold
configuration that shall be described in the next paragraph. The
precise syntax for the trust model configuration is a DTKA-KA
implementation aspect and, is therefore, out-of-scope of this
document.

```
+--------+--------+---+---+------------------------------------+ +---+
|Bulletin|Bulletin|TMN|BSN|Key information message (KIM):       | |   |
|        |hash    |   |   |{([Node ID, Effective Time, Public Key],|..|KIM|
|        |        |   |   |   OOBAuth/endorse/revoke/roll_over)}| |   |
+--------+--------+---+---+------------------------------------+ +---+
```

Figure 3: Bulletin

After forming a bulletin, a (Q+k)-erasure code algorithm is used to
create an erasure code for the bulletin. Thus, receipt of any Q
distinct code blocks will be sufficient to decode the bulletin. To
ensure that the incapacity or compromise -- or veto (disagreement on
bulletin content) -- of any single DTKA-KA will not result in
malfunction of the key authority mechanism, each DTKA-KA is assigned
primary responsibility for transmission of some limited subset of the
bulletin's code blocks and backup responsibility for some other
limited subset. The assigned code block subsets for the various
DTKA-KAs are selected in such a way that every code block is to be
transmitted by two different DTKA-KAs. The combination of these two
transmission redundancy mechanisms (parity code blocks and duplicate
transmissions), together with reliable bundle transmission at the
convergence layer under bundle multicast, minimizes the likelihood of
any client node being unable to reconstruct the bulletin from the
code blocks it receives.

During system initialization, the code-block assignments for each
DTKA-KA need to be configured into every DTKA Entity. The code-block
assignment for the example considered in this section is shown below
in the table, in which an x-mark depicts the assignment of a code
block to a DTKA-KA. It can be seen in the table that, in this
example, code-blocks from at least five (t=5) DTKA-KAs must be
received before the bulletin blocks can be decoded. Also, when all
DTKA-KAs multicast their pre-defined code blocks, n * m (8*3 = 24)
code blocks are sent to all DTKA Entities. To further defend against
a compromised DTKA-KA node introducing error into the key
distribution system:

o All nodes are informed of the code block subsets for which all
   DTKA-KA nodes are responsible. Any received code block that was
   transmitted by a DTKA-KA node which was not responsible for
   transmission of that code block is discarded by the receiving
   node.

o Each code block issued by the each KA is signed under that KA's
   private key. The bulletin hash in the code block uniquely
   identifies the bulletin that will be reconstructed using this code
   block. Every transmitted code block is accompanied by the
   bulletin hash. All - and only - code blocks tagged with the
   unique bulletin hash are reassembled into the bulletin identified
   by that hash.

o If the hash of a bulletin reassembled from a set of received code
   blocks is not verified then, for each the DTKA-KA node that
   transmitted one or more of the constituent code blocks, all code
   blocks transmitted by that node are excluded from the reassembled
   bulletin. Upon success, the node whose transmitted code blocks
   had been excluded from the reassembled bulletin may be presumed to
   be compromised.

Code Block Numbers (0 to (Q + k -1))	0	1	2	3	4	5	6	7
KA 1	x	x	x					
KA 2		x	x	x				
KA 3			x	x	x			
KA 4				x	x	x		
KA 5					x	x	x	
KA 6						x	x	x
KA 7	x						x	x
KA 8	x	x						x

Table 1: Example Trust-Table: Code Blocks Assignments for Key Agents

The message format for transmitting the assigned code-blocks by each
DTKA-KA is shown in Figure 4. Note that each such message is the
payload of a Bundle and that the authenticity of that payload is
nominally protected by a Block Integrity Block containing a digital
signature computed in the private key of the issuing Key Agent; the
message itself contains no self-authentication material. Reading the
figure left to right, we have: (a) a field indicating the type of
this message, namely Bulletin code block; (b) the bulletin hash as
defined in Figure 3; (c) the trust model number that provides trust-
table configuration as depicted in Table 1; (d) the code-block

numbers (column numbers in the trust-table) for which code-blocks are available in this code block message; and, (e) the specified code-blocks from the DTKA-KA. The identity of the DTKA-KA (KAx) that generated the code blocks must be available as the source node ID of the DTN bundle that carried this code block message. KAx is used to validate the signature in the bundle's Block Integrity Block before the message is delivered to DTKA by the underlying DTN protocol layer.

```
+----------+----------+---+----------+------------+
| Bulletin | Bulletin |TMN| Code Block| Code Blocks|
| Codeblock| Hash     |   | Numbers  |            |
+---------+----------+---+----------+------------+
```

Figure 4: Message Format for Code Blocks

4.2. Non-receipt of a Bulletin

When a DTKA Entity receives sufficient number of bulletin blocks from the DTKA Key Agents, it can reconstruct the corresponding bulletin with its unique Bundle Serial Number (BSN) in the format depicted in Figure 3. By maintaining a historical list of successfully reconstructed BSN values and analysing for gaps in the BSN historical list, a DTKA Entity can detect non-receipt of past bulletins. Upon such a detection, the DTKA Entity must send a request to all the DTKA Key Agents in the format specified in Figure 5 in order to request retransmission of the past bulletins for a given BSN value. When such request is received by a DTKA Key Agent, the DTKA Key Agent must retransmit its code blocks corresponding to the requested BSN only to the requesting DTKA Entity in the format shown in Figure 4. The security for this communication from the DTKA Key Agents must be similar to the security for the bulletin broadcast communication. Upon receiving sufficient number of bulletin blocks for the requested bulletin, the requesting DTKA Entity may reconstruct the bulletin and verify that the bulletin with the requested BSN has indeed been received. Thereupon, the DTKA Entity must update its BSN historical list with the received BSN value.

```
+----------+----------+--------------+-------+
| Bulletin | Request  | Requesting   |List   |
| Request  | Timestamp| Node (Node ID)|of BSNs|
+---------+----------+--------------+-------+
```

Figure 5: Message Format for Requesting Retransmission of Bulletin

4.3. Node Registration

   In order to register a new DTKA-KO in the system, DTKA requires the
   DTKA-KO with a Node ID (DTKA-KO[Node ID]) to generate a public-
   private key pair and preserve the secrecy of its private key. The
   DTKA-KO[Node ID] needs to generate an association message of the form
   (Node ID, effective-time, public-key), where effective-time specifies
   the start time after which the public-key is valid. That is, each
   bundle sent by this node is to be authenticated using the node's most
   recently effective public key whose effective time is less than the
   bundle's creation time. The DTKA-KO[Node ID] must send the
   association message, along with a signature on the message using its
   private key, to the DTKA-KA as depicted in Figure 6. Since DTKA-KA
   would not have seen the association of the public-key to that key
   owner previously, it cannot trust that the message indeed originated
   from DTKA-KO[Node ID]. Therefore, for registration purposes, this
   initial message from the DTKA-KO[Node ID] to the DTKA-KA MUST be
   protected by transmitting it over an independently (e.g., physically)
   authenticated channel. The independently authenticated channel can
   be realized by physically securing the access to the DTKA-KA server,
   using a physical communication medium, such as a USB dongle, and
   manually verifying the authenticity of the communication from the
   DTKA-KO. The manual verification is a one-time process for a given
   Key Owner. When an application domain has more than one DTKA-KA
   (KAx), the message from DTKA-KO[Node ID] must be sent to each DTKA-KA
   (KAx) in a similarly secure manner.

   Although the messages to DTKA-KA (KAx) are independently
   authenticated, the DTKA-KO[Node ID] must sign the association message
   using its private key. The signature is not intended to
   cryptographically authenticate the message but only to prove to the
   DTKA-KA that the DTKA-KO[Node ID] is indeed in possession of the
   private key. This self-signed message by the DTKA-KO is useful to
   ensure that the physical courier, which is used to realize the
   physically authenticated channel, has not tampered the message sent
   by the DTKA-KO to the DTKA-KA. Additionally, the self-signed message
   is useful to audit the operations of the DTKA-KA.

```
        +--------------------+                +---------------+
        |   DTKA-KO[Node ID]  |                | DTKA-KA (KAx) |
        +--------+-----------+                +-------+-------+
              **|*****************************************|**
               * |                                       | *
               * |{[Node ID, Effective Time, Public Key, s]   | *
               * | such that s = Sign(Private Key, [Node ID,  | *
               * |         Effective Time, Public Key,...])}   | *
               * +--------------------------------------->  *
               * |                                       | *
               * | Physically authenticated channel(USB,...) | *
              **|*****************************************|**
                |                                       |
                |                                       |
                |                                   +--+
                | TRUE = Verify(Public Key, s, [Node ID, | |
                |        Effective Time, Public Key,...]) | |
                |                                   +-->
                |                                       |
                +                                       +
```
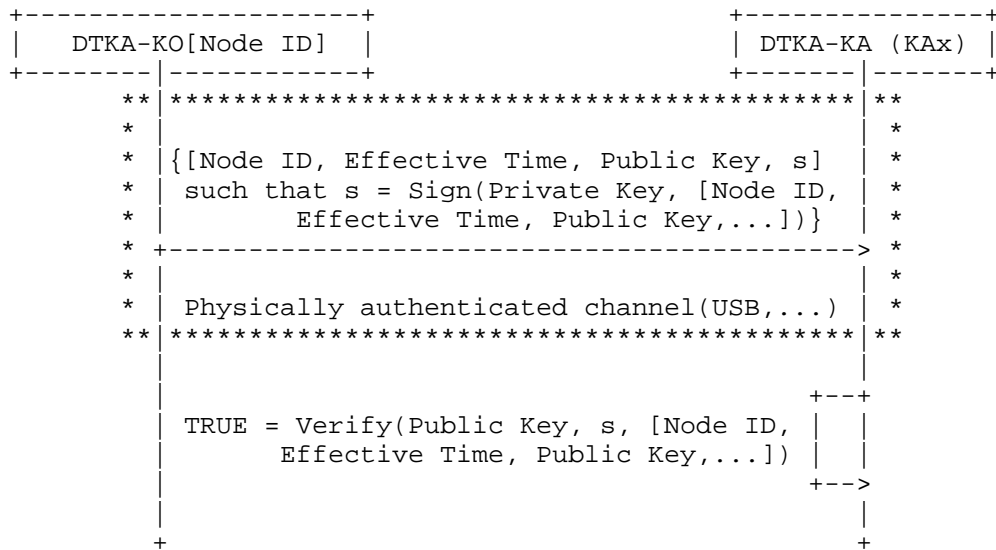
Figure 6: Interaction Diagram 1: Node Registration

Each DTKA-KA will insert the received association message into its
next bulletin (refer to Figure 3), for multicast as an out-of-band-
authentication (OOBAuth) association: when registration is received
through a physically authenticated channel. The bulletin will be
multicast to all DTKA Entities using the protocol described in
Section 4.7.

As an alternative to the use of a physically authenticated channel,
the registration association message may be sent by a trusted third-
party node whose authenticated public key is already registered and
known to all DTKA-KAs, so that the message may be authenticated by
verifying the digital signature (formed using the trusted third-party
node's current private key) in the BIB of the bundle containing the
message. Each DTKA-KA will insert such association requests in its
next bulletin for multicast as an endorsed association by tagging the
corresponding Key Information message in the bulletin as "endorse"
(refer to Figure 3). The bulletin will be multicast to all DTKA
Entities using the protocol described in Section 4.7.

4.4. Key Revocation

Manual decisions trigger the key revocation procedure. Every DTKA-KA
in an application domain is assumed to have a human operator who can
trigger the revocation process. When a key is to be revoked, the
human operator will need to authenticate to the respective DTKA-KA
(KAx) server, identify the public-key and Node ID to be revoked, and

instruct that DTKA-KA (KAx) revocation software to schedule a
revocation message. The revocation software in DTKA-KA (KAx) will
multicast a message as shown in Figure 7. The process for sending
out the code-blocks by all the DTKA-KAs with this revocation
information is described in Section 4.1.

```
 +---------+                                      +---------+
 | DTKA-KA |                                      | DTKA-KA |
 |  (KAx)  |                                      |  (KAy)  |
 +---|-----+                                      +---|-----+
     |                                                |
     |                                                |
     | Multicast{m , s  such that                     |
     | s = Sign(PrivateKey[KAx], m) and               |
     | m = [KAx, Revoke, [Node ID, Effective Time, Pub Key)]} |
     +------------------------------------------------------>
     |                                                |
     |                                             +--+
     |        TRUE = Verify(Public Key[KAx], s, [Node ID, |  |
     |              Effective Time, New Public Key,...])   |  |
     |                                             +-->
     |                                                |
     +                                                +
```
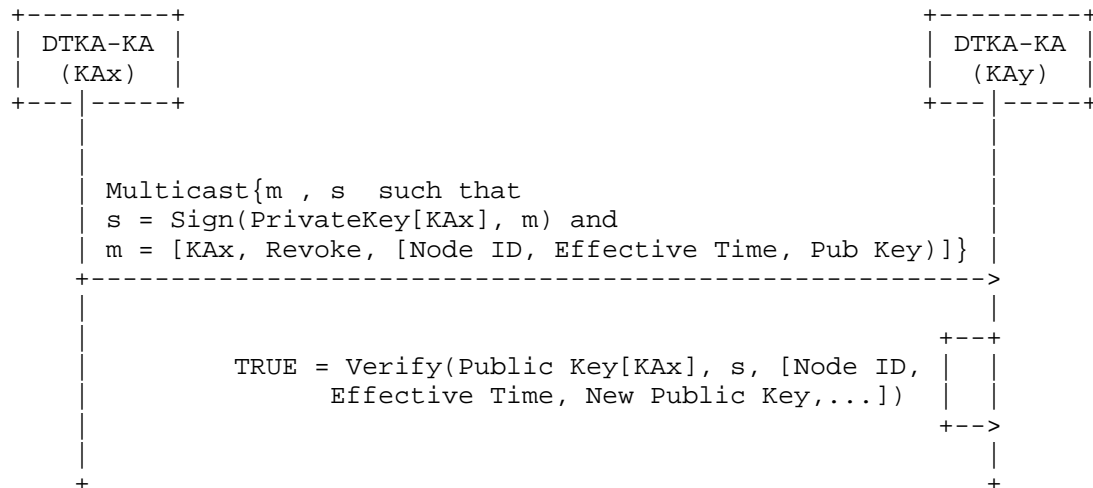
Figure 7: Interaction Diagram 1.1: Key Revocation

4.5. Key Roll-over

   When a DTKA-KO[Node ID] has been registered by the DTKA-KA using the
   protocol described in Figure 6, the DTKA-KO[Node ID] can periodically
   roll-over to a new public-private key pair by following the key roll-
   over protocol described in Figure 8. The protocol for key roll-over
   is similar to the one for key registration except that: (a) the
   protocol can be executed using DTN bundles issued by the KO itself
   without requiring any independently secured out-of-band communication
   channels; and, (b) the old (current) public-key is used to
   authenticate the association of the new public-key with the Node ID
   for that DTKA KO. The DTKA-KO [Node ID] must send this message to
   every key agent in its application domain. Upon accepting the roll-
   over message from the DTKA-KO[Node ID], each key agent will schedule
   the roll-over instruction for identified Node ID and public-key in
   its next bulletin as described in Section 4.1. A DTKA-KO can
   schedule any number of future roll-overs but the number of such roll-
   over schedules may need to be limited to avoid Denial of Service
   attacks by registered nodes -- but this topic is beyond the scope of
   this document.

```
     +---------------------+                +---------------+
     |   DTKA-KO[Node ID]  |                | DTKA-KA (KAx) |
     +--------|------------+                +-------|-------+
              |                                     |
              |{[Node ID, Effective Time, New Public Key, s]  |
              | such that s = Sign(Old Private Key, [Node ID, |
              |      Effective Time, New Public Key,...])}     |
        +--------------------------------------------->
              |                                     |
              |                                     |
              |                                  +--+
              | TRUE = Verify(Old Public Key, s, [Node ID, |  |
              |       Effective Time, New Public Key,...]) |  |
              |                                  +-->
              |                                     |
              +                                     +
```
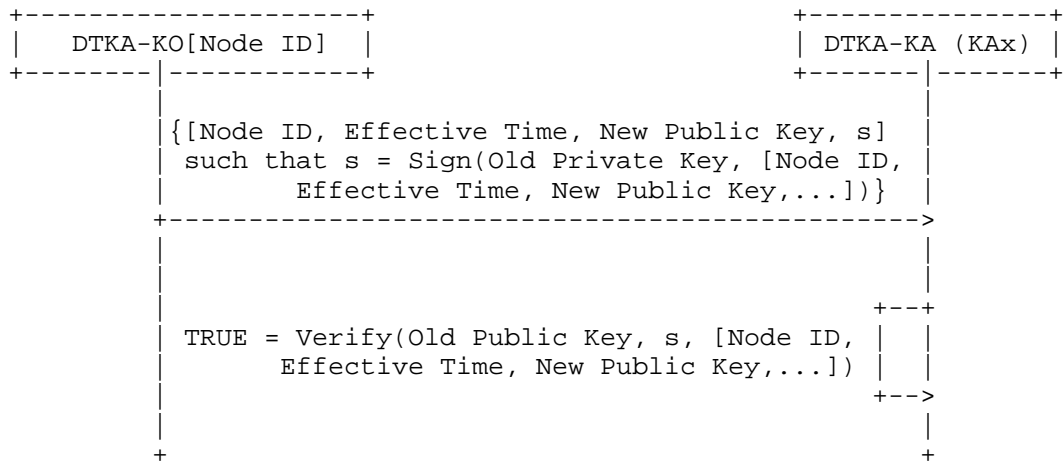
Figure 8: Interaction Diagram 1.2: Key Rollover

4.6. Key Endorsement

   When a DTKA-KO[Node ID] is not registered and does not have access to
   any out-of-band authentication channel with any DTKA-KA, the DTKA-
   KO[Node ID] will need to have access to an out-of-band authentication
   channel for a trusted third-party (TTP), which is registered with the
   DTKA-KA. Upon receiving the (Node ID, Key, Effective time)
   information from the DTKA-KO[Node ID] over the out-of-band
   authentication channel, the trusted third-party needs to relay that
   information to the DTKA-KA by signing the information under its
   authenticated public key. This interaction is depicted in Figure 9.
   Upon accepting the endorse message from the trusted third-party, each
   key agent will schedule an endorse instruction for identified Node ID
   and public-key in its next bulletin as described in Section 4.1.

```
        +--------------------+              +---------------+
        |Trusted third-party |              | DTKA-KA (KAx) |
        +--------|-----------+              +-------|-------+
                 |                                  |
                 |{[Node ID, Public Key, Effective Time, s]   |
                 | such that s = Sign(TTP Private Key, [Node ID, |
                 |        Public Key, Effective Time,...])}    |
                 +---------------------------------------------->
                 |                                  |
                 |                                  |
                 |                                +--+
                 | TRUE = Verify(TTP Public Key, s, [Node ID, | |
                 |        Effective Time, Public Key,...])    | |
                 |                                +-->
                 |                                  |
                 +                                  +
```
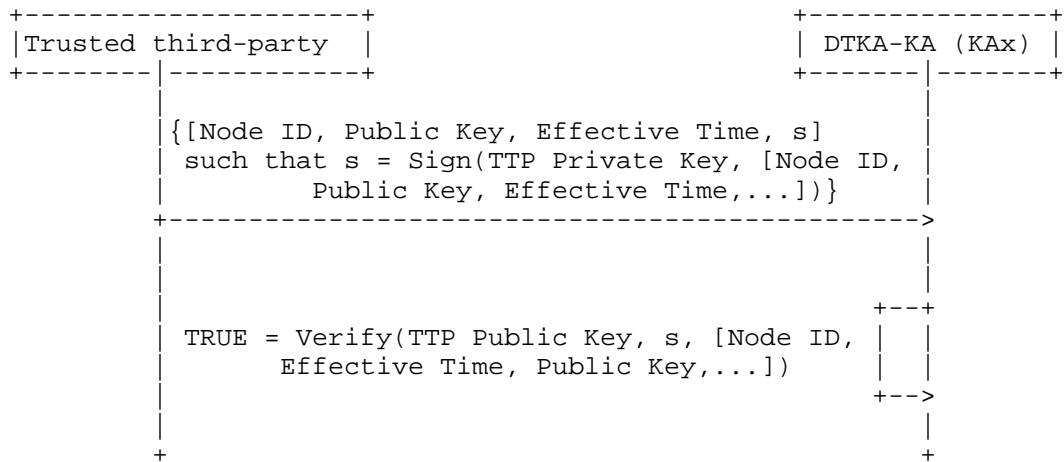
Figure 9: Interaction Diagram 1.3: Key Endorsement

4.7. Key Distribution

   Each DTKA-KA collects multiple out-of-band-authentication (OOBAuth),
   revocation, roll-over and endorse association messages from different
   parties by following the protocols described in Section 4.3,
   Section 4.4, Section 4.5 and Figure 9. Then, each DTKA-KA forms and
   sends multicast communications for the code blocks for its bulletin
   to all DTKA Key Users as explained in Section 4.1. The DTKA-KUs
   verify the authenticity of each code block from all the DTKA-KAs
   before using the code blocks to decode the bulletin, which will
   contains out-of-band key authentication, key revocation, key roll-
   over and endorse instructions. The DTKA-KUs perform these
   instructions in their respective local key database. This
   interaction between the DTKA-KAs and the DTKA-KUs of an application
   domain is shown in Figure 10.

```
        +------------------+                    +--------------+
        |  DTKA-KA (KAx)   |                    |   DTKA-KU    |
        +--------+---------+                    +--------+-----+
                 |                                       |
            +------------------------------------------------->
            | Send(KAx, BulletinHash, CodeBlockNumber        |
            |   CodeBlock of Bulletin such that Bulletin =   |
            |   array[Node ID, Effective Time, PubKey])      |
            |                                                |
            |                                    +---+
            |               Wait for code blocks |   |
            |               from key authorities +--->
            |                                       |
            |                                    +---+
            |               Decode Bulletin using code |   |
            |               blocks from key authorities +--->
            +                                       +
```
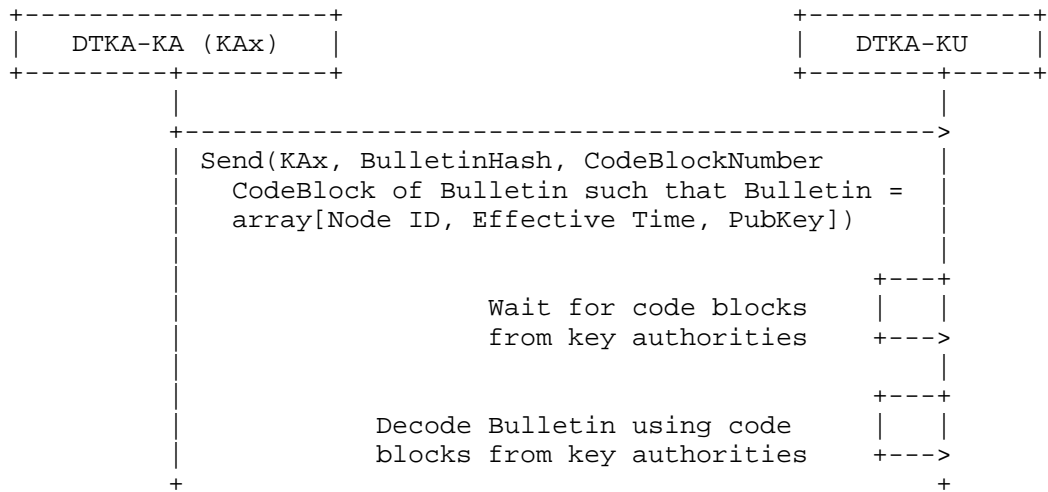
Figure 10: Interaction Diagram 2: Bulk Key Distribution

4.8. Secure Communications

   After receiving out-of-band-authentication (OOBAuth), roll-over or
   endorse information, every DTKA-KU shall have authenticated public-
   keys for different Node IDs in its local database. These
   authenticated public-keys can be used to authenticate messages
   received from the DTKA-KO[Node ID] and to send confidential messages
   to the DTKA-KO[Node ID] after the specified effective-time for each
   Node ID and public-key pair. This interaction is specified in
   Figure 11.

```
        +--------------------+                   +--------------+
        |  DTKA-KO[Node ID]  |                   |   DTKA-KU    |
        +--------+-----------+                   +--------+-----+
                 |Secure communications                   |
                 |[Node ID, Creation Time, Signature, Data] |
                 +----------------------------------------->
                 |                                         |
                 |Secure communications                    |
                 |[Node ID, Creation Time, Encrypted Data] |
                 <-----------------------------------------+
                 |                                         |
                 +                                         +
```
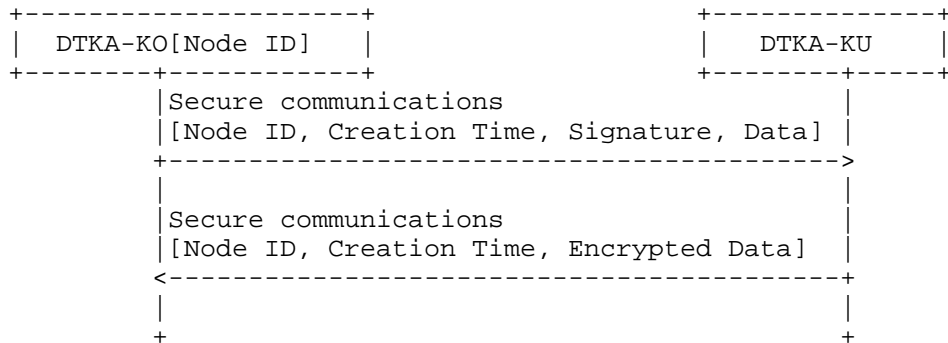
Figure 11: Interaction Diagram 3: Secure communication

4.9. Communication Stack View

   DTKA is designed to be a special DTN application that shall perform
   key management operations using the services of the Bundle Protocol
   and BPSec. DTKA will use BP, which in turn will use BPSec to
   authenticate the messages containing the public-keys that are
   subsequently to be used by BPSec for securing future communications
   as shown in Figure 12.

```
   +-------------+---------+   +------------------------------------+
   |Applications |  DTKA   +--->                                    |
   +-------------+---------+   |          Local Keys Database        |
   |Bundle Protocol (BPSec)<---+(Node ID, Public Key, Effective Time)|
   +---------------------+     |                                    |
   |  Convergence Layer  |     +------------------------------------+
   +---------------------+
   |   Transport Layer   |
   +---------------------+
   |    Network Layer    |
   +-----------------------+
   |   Physical Layer    |
   +---------------------+
```
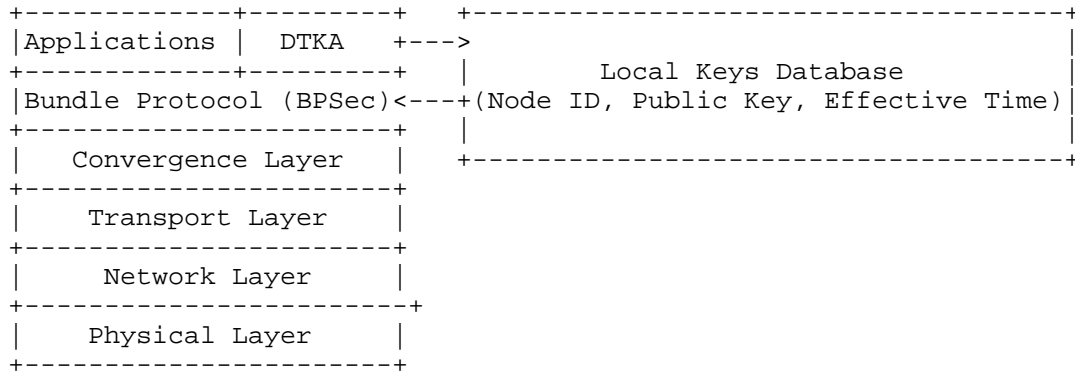
           Figure 12: Block Diagram: Communication Stack View for DTKA

5. IANA Considerations

   This document potentially contains IANA considerations depending on
   the design choices adopted for future work. But, in its present
   form, there are no immediate IANA considerations.

6. Security Considerations

   Security issues and considerations are discussed through out this
   document.

7. References

7.1. Normative References

   [I-D.ietf-dtn-bpbis]
              Burleigh, S., Fall, K., and E. Birrane, "Bundle Protocol
              Version 7", draft-ietf-dtn-bpbis-10 (work in progress),
              November 2017.

   [I-D.ietf-dtn-bpsec]
             Birrane, E. and K. McKeever, "Bundle Protocol Security
             Specification", draft-ietf-dtn-bpsec-06 (work in
             progress), October 2017.

   [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>.

7.2. Informative References

   [I-D.templin-dtnskmreq]
             Templin, F. and S. Burleigh, "DTN Security Key Management
             - Requirements and Design", draft-templin-dtnskmreq-00
             (work in progress), February 2015.

   [RFC4838] Cerf, V., Burleigh, S., Hooke, A., Torgerson, L., Durst,
             R., Scott, K., Fall, K., and H. Weiss, "Delay-Tolerant
             Networking Architecture", RFC 4838, DOI 10.17487/RFC4838,
             April 2007, <https://www.rfc-editor.org/info/rfc4838>.

   [RFC5050] Scott, K. and S. Burleigh, "Bundle Protocol
             Specification", RFC 5050, DOI 10.17487/RFC5050, November
             2007, <https://www.rfc-editor.org/info/rfc5050>.

   [RFC6257] Symington, S., Farrell, S., Weiss, H., and P. Lovell,
             "Bundle Security Protocol Specification", RFC 6257,
             DOI 10.17487/RFC6257, May 2011,
             <https://www.rfc-editor.org/info/rfc6257>.

Authors' Addresses

   Scott Burleigh
   JPL, Calif. Inst. Of Technology
   4800 Oak Grove Dr.
   Pasadena, CA 91109-8099
   USA

   Email: Scott.Burleigh@jpl.nasa.gov

David Horres
JPL, Calif. Inst. Of Technology
4800 Oak Grove Dr.
Pasadena, CA 91109-8099
USA

Email: David.C.Horres@jpl.nasa.gov

Kapali Viswanathan
Boeing Research & Technology
Boeing International Corporation India Private Limited
A Block, 4th Floor, Lake View Building
Bagmane Tech Park, C.V. Raman Nagar
Bangalore, KA 560093
IN

Email: kapaleeswaran.viswanathan@boeing.com

Michael W. Benson
Boeing Research & Technology
The Boeing Company
499 Boeing Boulevard
Huntsville, AL 35824
USA

Email: michael.w.benson@boeing.com

Fred L. Templin
Boeing Research & Technology
P.O. Box 3707
Seattle, WA 98124
USA

Email: fltemplin@acm.org