A proposed framework for Naming, Addressing and Forwarding

Rick Taylor

IETF DTN Working Group
27th May 2021 Interim meeting

Introduction

I have a concern that is difficult to reason clearly around the complex topics of naming, addressing, forwarding and routing in DTNs without a semi-formal framework to use as a reference point.

In this presentation, I will propose such a framework, with reference to the guiding principles outline in Bundle Protocol version 7 (BPv7), and attempt to create a common conceptual model of the inner workings of an individual Bundle Processing Agent (BPA).

I will proceed from the BPv7 definitions, and introduce a basic model; and then incrementally propose increased capability, at the price of more complexity.

I am not attempting to standardise the implementation of a BPA.

Disclaimer

- This is my personal view, and should not be read as any kind of steer from the chairs, or any insight into activities by my employer.
- These ideas are based on discussion with many others, and I present my distillation of these discussions.
- I may not have fully understood some of the ramifications of my proposals.
- I may be stating the blindingly obvious, but I haven't found anywhere else this is recorded.
- My vision may be incompatible with your vision, but that's okay.
- I make ugly slides, sorry.

Endpoint IDs

Bundle Protocol version 7 defines Endpoint IDs as a tuple: `[schema, content]`

Where:

- `schema` is the numeric identifier of the schema in use.
- `content` is a CBOR encoded schema-specific value.

- Applications executing somewhere in the DTN are the receivers of bundles, and receivers can have multiple Endpoint IDs.
- Endpoint IDs may refer to zero or more receivers, i.e. they can be used for multicast (> 1) or as a 'black hole' (0)?
  - I shall define this value as the Endpoint ID's *multiplicity*.
- Node IDs are defined as a specialisation of Endpoint IDs that are explicitly unicast (*multiplicity* = 1), and name individual Bundle Processing Agent (BPA) applications.
- BPv7 defines 2 schemas, the *ipn* and *dtn* schema.
  - Neither defines how to determine the *multiplicity* an Endpoint ID.
  - The *ipn* scheme concatenates the name of the consuming application and the Node ID of the BPA processing bundles on behalf of the application, introducing the concept of a *Service Number*.
- By defining two naming schemes, BPv7 suggests that there is no requirement for a universal naming scheme, and each DTN deployment may choose a scheme according to its needs.
  - This implies no central naming allocation authority, as is the case with IP addresses and DNS names.

Late Binding

BPv7 introduces the concept of *Late Binding*:

- The destination of a bundle is not completely resolved at the sender, but should be re-evaluated at each step along the path to the destination.
  - This removes the need for a bundle source to have a complete connected graph to the destination, a key part of the DTN architecture.

Late Binding implies that there must be a *Name Resolution* operation that is performed at each node along the path.

BPv7 does not define this *Name Resolution* operation, but does hint to some of its expected behaviour:

"Late binding of overlay network endpoint identifiers to underlying constituent network addresses"

This implies that BPv7 expects the *Name Resolution* operation to function between the logical layers defined in the document, as well as at each bundle processing agent.

- This implies Convergence Layers need to have a concept of a Name with a different scope than the bundle layer.

I understand this as: BPv7 requires that the *Name Resolution* operation must be executed, possibly repeatedly, for each bundle in flight, at each bundle processing agent until delivery completes.

Simple Name Resolution

The Simple Name Resolution operation performs the following steps:

1. Remove a bundle from the Ingress Bundle Set, according to some de-queueing policy.

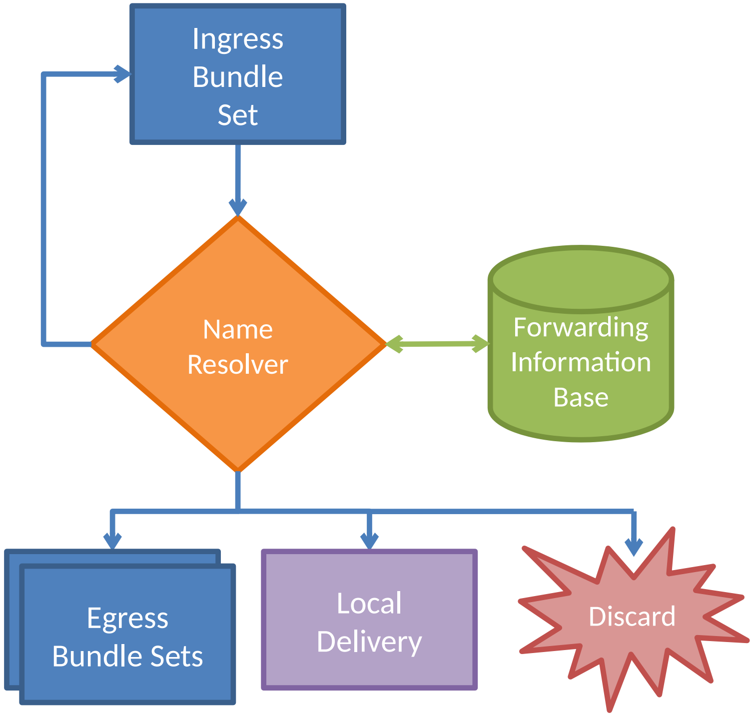
2. Match the rules in the Forwarding Information Base (FIB) with the blocks in the bundle to determine the action to take.

3. Perform one of the following actions:
   a. "Deliver": Pass the bundle to a local application.
   b. "Forward": Add the bundle to a per-CLA Egress Bundle Set, ready to be forwarded.
   c. "Wait": Add the bundle to the Ingress Bundle Set, to be processed again later.
   d. "Drop": Discard the bundle.

All the information required to determine the action to be taken exists in the blocks of the bundle being processed.

Note: The FIB is just a conceptual set of rules that describes the actions to be taken. How the content of the FIB is populated and maintained is not important for now, and the responsibility of 'routing' protocols.

This forwarding algorithm is complete enough to function within a single consistently named DTN, with a-priori known connectivity, such as provided by contact-graph routing.



Annotating Bundles

- In order to support Convergence Layer Adaptors with point-to-multipoint behaviour, each CL must define its own naming scheme.
  - E.g. TCP-CL needs a mapping from Endpoint IDs to one or more IP addresses of peers.

- Therefore there must be a function to translate a name from one scope (e.g. Endpoint ID) to another (e.g. TCP-CL IP address).

- This can be done with another action for the *Name Resolution* operation:
  - "Annotate": Attach some local metadata to the bundle that records a name mapping from one scope to another, and then return the bundle to the Ingress Bundle Set to be processed again.

- During processing, any attached local metadata may be used to enhance forwarding in the following ways:
  - Pass the bundle to a specific CLA for forwarding, mapping Endpoint ID to CL-scoped address.
  - Pass the bundle to a specific process, e.g. mapping *ipn* Service Number to internal process identifier.

Annotating Name Resolution

The Annotating Name Resolution operation performs the following steps:

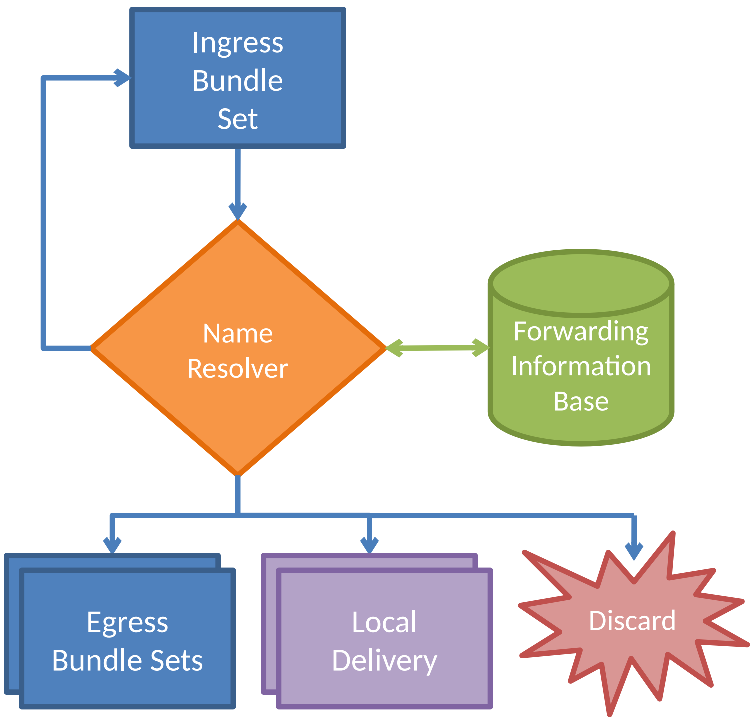
1. Remove a bundle from the Ingress Bundle Set, according to some de-queueing policy.

2. Match the rules in the Forwarding Information Base (FIB) with the blocks in the bundle, **plus any local attached metadata**, to determine the action to take.

3. Perform one of the following actions:
   a. "Deliver": Pass the bundle to a local application.
   b. "Forward": Add the bundle to a per-CLA Egress Bundle Set, ready to be forwarded.
   c. "Wait": Add the bundle to the Ingress Bundle Set, to be processed again later.
   d. "Drop": Discard the bundle.
   e. "Annotate": Attach local metadata to the bundle and add to the Ingress Bundle Set, to be processed again.

This forwarding algorithm is complete enough to function within a single consistently named DTN, with dynamic routing.

- CLA's can learn adjacent neighbour's Endpoint IDs and populate the FIB with the relevant mapping.
- Routing protocols can populate the FIB with next-hop information.



Heterogeneous Names

As mentioned at the beginning, BPv7 suggests that both the choice of naming schema, and the assignment of names with an administrative domain of a DTN network should be a matter of local administrative policy.

- I shall refer to a set of bundle processing agents that share the same naming schema, encoding and assignment policy as a *Naming Domain*.

What if BPAs could add a *Name Resolution Extension Block* to the bundle, with similar metadata, as the bundle transited the network?

- This *Name Resolution Extension Block* could be added to bundles by BPAs acting as gateways as the bundles cross the boundaries between *Naming Domains*.
  - These gateway BPAs must have at least one Endpoint ID in each *Naming Domain*, but need not to be on the topological edge of an administrative domain: well-informed gateways could be placed in the connectivity centre of a *Naming Domain*.

- Bundle processing agents inside a domain could consult any *Name Resolution Extension Block* to assist in it's forwarding decision.
  - This weakens *Late Binding*, as a gateway BPA has effectively performed a binding on behalf of non-gateway BPAs, but it is a compromise, as the alternative is to discard the bundle.
  - However, there is nothing to prevent an intermediate BPA ignoring a *Name Resolution Extension Block* and using the bundle primary block content directly if it results in 'better' forwarding, or if required by administrative policy.

We can generalise this concept to support bundles transiting more than one *Naming Domain*, if the *Name Resolution Extension Block* can be constructed in such a way as its operates as a stack.

- Name resolution information can be pushed and popped from the stack as the bundle follows the path from source to destination.

Redirecting Bundles

Using the same forwarding model as before, we can add additional actions to the *Name Resolution* operation to support this concept:

- “Push”: Push an information item to the top of the *Name Resolution Extension Block* stack of the bundle, and then return the bundle to the Ingress Bundle Set to be processed again.

- “Pop”: Remove the top item of the *Name Resolution Extension Block* stack of the bundle, and then return the bundle to the Ingress Bundle Set to be processed again.

When used by the *Name Resolution* operation, the “Push” action acts as a redirection, adding alternate Node IDs of BPAs that might be able to assist in the delivery of the bundle.

- An example of this is when a BPA receives a bundle with a destination Endpoint ID in a *Naming Domain* it does not know how to parse. The FIB could have a rule for such cases, using the “Push” action to add the Node ID of a nearby gateway BPA that might be able to forward the bundle.

Redirecting Name Resolution

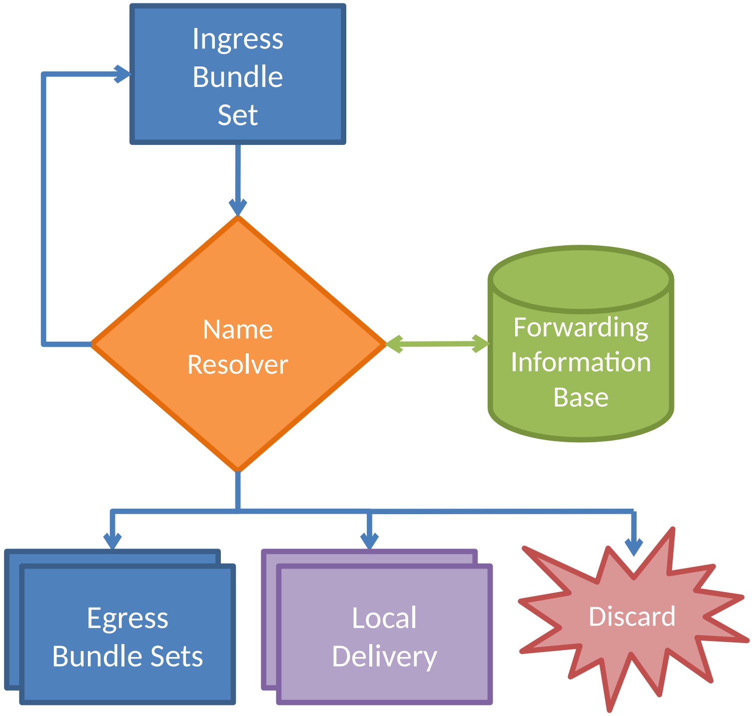
The Redirecting Name Resolution operation performs the following steps:

1. Remove a bundle from the Ingress Bundle Set, according to some de-queueing policy.

2. Match the rules in the Forwarding Information Base (FIB) with the blocks in the bundle, **particularly the *Name Resolution Extension Block***, plus any local attached metadata, to determine the action to take.

3. Perform one of the following actions:
   a. "Deliver": Pass the bundle to a local application.
   b. "Forward": Add the bundle to a per-CLA Egress Bundle Set, ready to be forwarded.
   c. "Wait": Add the bundle to the Ingress Bundle Set, to be processed again later.
   d. "Drop": Discard the bundle.
   e. "Annotate": Attach local metadata to the bundle and add to the Ingress Bundle Set, to be processed again.
   f. "Push": Add metadata to the top of the NREB stack of the bundle and add to the Ingress Bundle Set, to be processed again.
   g. "Pop": Remove metadata from the top of the NREB stack of the bundle and add to the Ingress Bundle Set, to be processed again.

This forwarding algorithm is complete enough to function with a global DTN made of multiple heterogeneous *Naming Domains*, interconnected in a decentralised manner.



Conclusions

Yes, this is label switching.

An alternative formulation would be to use Bundle-in-Bundle Encapsulation, but I believe the ability to inspect down the stack as part of the *Name Resolution* operation is useful and simpler to define with label switching.

- Tunnels have an implied “opaqueness”; like sewer pipes, they aren’t designed to be looked in by users.
- The label stack contains an accessible abbreviated history of the route a bundle followed that might be of interest to intermediate BPAs as well as the bundle destination.

I have not suggested any content or layout of a *Name Resolution Extension Block*, as the block/stack concept may have more use-cases than just name resolution, and perhaps a more generic formulation is appropriate.

- QoS marking seems like another good target for push/pop as bundles transit networks.
- Is it useful to push/pop a set of blocks, rather than a block containing a stack?
- What about BPSec?

Ed Birrane has a similar concept he calls “ticket to ride”, where extension blocks are used to attach labels to assist the transit of bundles through different administrative domains.

- I think we are all circling around the same set of problems.

Questions?