BTPU Updates

DTN Working Group, IETF-123 Bangkok

Rick Taylor, Aalyria

Adoption complete

Both the BTPU and FEC extension drafts have been adopted:

- https://datatracker.ietf.org/doc/draft-ietf-dtn-btpu/
- https://datatracker.ietf.org/doc/draft-ietf-dtn-btpu-fec/

There have been no changes to the FEC document, but there has been one significant change to the BTPU document.

Changes from draft-taylor-dtn-btpu-02

Based on a suggestion from Brian Sipos, the numbering of the segments of a transfer is now reversed:

```
0..N  => N..0
```

- This allows a receiver to have an estimate of the total number of expected segments in advance, so storage can be preallocated.
- The replaces the Transfer End Message with a Transfer Start Message, and that feels more natural.

Question: Does this impact “streaming” in some way? I’m not sure “streaming” is an actual use-case.

Potential future changes

I am considering the following potential improvements in the next revision:

1. Reduce the 24-bit segment length field in the common header to 16-bit and include a ‘flags’ field, to allow optional behaviour, such as:
    a. Sub-TLV between Header and Content to carry optional metadata, such as ‘Total Bundle Length’ or ‘BP version’.
2. Reserve Type 6 and Types 0x80..0x9F to ensure frames containing ‘raw’ BPv6 and BPv7 bundles can be demuxed from frames containing BTPU messages.
    a. Perhaps including text about mixing ‘raw’ frames with BTPU frame, particularly around the use of the Bundle Message.

Derived Work

I am already hearing interest in ‘derived work’, such as:

- Bundle Protocol over Ethernet
  - Erik Kline is sketching out a draft detailing how to use BTPU in Ethernet frames.
- Bundle Protocol over EPP/AOS
  - There are active requirements to run BPv7 over EPP/AOS, and BTPU is seen as a useful way to manage the segmentation.
- BIBE with Fragmentation (BIBF?)
  - If we are segmenting bundles inside BIBE for fragmentation, why not re-use the BTPU segmentation primitives, but in a CBOR encoding? (It would end up looking a lot like the payload formatting from UDP-CLv2).

Any questions?