Custody Transfer Concepts

Scott Burleigh, JHU APL

7 November 2023

What is custody transfer for?

- The fundamental “custody transfer” concept in BP is that the sender of a bundle remains responsible for retransmitting it, in the event of data loss or corruption, until that node has been informed by the receiver of the bundle that the bundle has been successfully and completely received. At that point, the sender’s retransmission responsibility is terminated.
- That is, BP adopts an exception to the end-to-end principle: to improve performance, the forwarding nodes along the path (not just the source) are responsible for retransmission.
- The Custody Transfer mechanism built into BPv6 was designed to implement this concept.

What's wrong with using Custody Transfer to provide reliability?

- CT provides no way for the receiving node to signal data loss. The only negative acknowledgment is "I got it all, but I don't want it; here's why."
- Since there's no way for data loss (complete or partial) to be signaled, partial retransmission is likewise impossible. If you send a 2 GB bundle and 1400 bytes of it are lost in transmission, you retransmit 2 GB.
- And the only way for the sender to realize that it must retransmit a lost bundle is for a countdown timer to expire prior to reception of a custody acceptance signal.

But timeout-driven retransmission is okay, right?

- Sure, but in the general case there is no suitable algorithm for computing the appropriate timer interval for transmission to a neighboring node: accurate computation of this interval is always a function of the underlying network infrastructure.
  - In the Internet, TCP knows how to do this; an LTP-like algorithm would perform poorly.
  - Over space links, LTP knows how to do this; a TCP-like algorithm would perform poorly.
  - Other algorithms would be fine for unusual environments such as the ISS stack, but nowhere else.
  - No algorithm or set of algorithms built into BP will function efficiently in all network environments.

But that's moot, because…

- Nodes are never required to accept custody.
- The custody acceptance signal might come from a node that is 3 hops away along the end-to-end path, or 6 hops away, or 13.
- The custodian doesn't even know that the bundle is going to reach that node, so obviously it can't have any idea when that will happen.
- In the general case, whatever interval you select will be wrong:
  - The timer will expire too early, wasting bandwidth in retransmission of a bundle that was actually received.
  - Or it will expire too late, wasting storage and retarding delivery.
- And, again, it will always be too much, because there's no partial retransmission.

Still worse

- Bundles can be fragmented. Every fragment of a bundle is a bundle, so custody transfer can be applied to fragments.
- A bundle for which there is a custodian may be fragmented into multiple bundles by a forwarder that doesn't take custody of the original bundle.
- Now the custodian EID cited in every fragmentary bundle is a node that did not create that bundle. Whenever a node takes custody of one of those fragments, the resulting custody signal is for a fragmentary bundle for which the receiving custodian never took custody. No custody signal for the original bundle is ever received, so custody of that bundle is never released, so it is retransmitted endlessly.

And it’s hopeless for multicast

- When we multicast a bundle, at every fork in the multicast tree we receive 1 copy of the bundle and transmit N copies.
- In order for custody of one of those N copies to be transferred, the node that receives that copy must send a custody acceptance bundle, containing the bundle ID, back to the current custodian for that bundle.
- But nothing in the bundle ID tells the custodian which one of the N forwarded copies was successfully transferred.
- So the signal can’t be used to release any retransmission buffer or turn off any retransmission timer.

Why not identify the received bundle by source EID of the custody signal?

- You may or may not be able to associate that source endpoint ID with one of the child nodes to which the bundle was sent (there's no protocol for this).
- But that doesn't matter anyway, because the responding node doesn't have to be any of the nodes to which the custodian transmitted that copy.
  - Nodes are not required to either accept or refuse custody.
  - So the new custodian might 2 or 3 or 19 hops down the tree from the current custodian, a node that the current custodian has never heard of and knows nothing about.

So what happens when a custodial retransmission timer expires?

- The custodian has no idea which child node(s) it must retransmit the bundle to, because it doesn't know which child nodes received the bundle.
- The number of custody signals received might be much larger than the number of child nodes that the custodian sent the bundle to, because any subset of those child nodes might have sent multiple copies of the bundle to any number of other nodes without ever having taken custody of the bundle.
- In the general case, there is no way to make any sense out of the custody signaling that results from multicast transmission.

What's the alternative?

- A node that has a bundle destined for a multicast endpoint sends a copy of the bundle to each child node that has expressed interest (for itself and/or on behalf of its own child nodes) in bundles destined for that endpoint.
- The bundles are sent to each such child node using whatever convergence-layer protocol is appropriate (per ECOS, routing, whatever) for transmission to that node.
- This is reliable multicast wherever, and only where, a reliable convergence-layer protocol (TCP, LTP, BSSP) is selected.
- In general, convergence-layer reliability is the efficient way to implement the custody transfer concept.

But wait…

- When convergence-layer transmission simply fails, the bundle may need to be "forwarded" back upstream to some node that can compute an alternate route to the destination.
- So, again, we pay the cost of forwarding the entire bundle.
- Alternatively, one or more upstream forwarders might retain copies of bundles in case such re-forwarding is necessary. Then the node at which CL transmission failed could just send a small "please try again" message (like a negative CT signal) back to such a custodian – not return the entire bundle.
- The CCSDS DTN Working Group is currently discussing a mechanism for **status report** aggregation that could do this.