Working Group Draft for TCPCLv4

Brian Sipos

RKF Engineering Solutions

IETF103

Motivations for Updates to TCPCL

1. During implementation of TCPCLv3, Scott Burleigh found an ambiguity in bundle acknowledgment and refusal.
2. For use in a terrestrial WAN, author has a need for TLS-based authentication and integrity. TCPCLv3 mentions TLS but does not specify its use. IETF strongly in favor of TLS for new general-use protocols.
3. Reduced sequencing variability from TCPCLv3
4. Adding extension capability for TCPCL sessions and transfers.

Goals for TCPCLv4

- Do not change scope or workflow of TCPCL.
  - As much as possible, keep existing requirements and behaviors. The baseline spec was a copy-paste of TCPCLv3.
  - Still using single-phase contact negotiation, re-using existing headers and message type codes.
  - Allow existing implementations to be adapted for TCPCLv4.

Last Draft Edits

- Changes are in draft-ietf-dtn-tcpclv4-10.
- Clarified order of Contact Header exchange in requirements.
  - The active role always transmits first, the passive role only transmits after agreeing on the protocol version.
  - There is no longer ambiguity about what protocol version is agreed upon when exchange finishes.
- Clarified requirements on TLS use.
  - Changes based on feedback from AD Spencer Dawkins.
  - Cited BCP195 directly, rather than RFC7525.
- Clarified default and minimum session timeout behaviors.
  - Restored recommended default from TCPCLv3.
- Added a “reply” marking to SESS_TERM message to avoid trivial feedback loop.
  - Now a termination initiation is distinguishable from its acknowledgement.
- Removed encoding variability in SESS_TERM reason code.
  - An “unknown” code is used where previously there was no encoded value.

Open Issues from Feedback

- Concern about octet-size of extension item encodings.
  - Currently the Extension Item Type is 16-bit and Extension Item Length is 32-bit.
  - This is oversized from minimum expected use.
  - This also avoids any possible issue with large extension items.
  - Is it worth shaving octets to possibly run into size-overflow issues?
  - Author's opinion is that current encoding is reasonable.

Open Issues Continued

- Comment about XFER_INIT (and its Transfer Length) not being strictly necessary.
  - This is true, but XFER_INIT is a convenient place to encode the transfer extension items.
  - The prepended transfer Length is still useful for a receiver to declare resource exhaustion or guard against overly large transfers.
    - This doesn’t guarantee a malevolent sender won’t misrepresent their transfer size, but there are logical guards against indefinite transfers.

Open Issues Continued

- Concern about necessity of SESS_TERM exchange if in-progress transfers can be continued.
  - The point of SESS_TERM now is to avoid truncating and failing a transfer that may be near completion.
  - Both peers in a session can, for any reason and at any time, close the TCP connection itself.
    - This will cause any in-progress transfer to fail immediately.
- Concern about excessive non-requirement text in Section 3 explanations.
  - This text was all driven by earlier confusion about the scope and capability of TCPCL connections, sessions, entities, etc.
  - The author sees value in providing this informative text that in some cases explain non-trivial behavioral side effects.

Way Forward for TCPCLv4

- Working implementation exists and is available for interoperability testing
  - Still needs to be updated for encoding changes in revision 10 of draft.
  - Implemented in `scapy/python` for ease of understanding
  - Handles concurrent sessions
  - Does not implement BP agent behavior, only CL behavior