JOHNS HOPKINS
APPLIED PHYSICS LABORATORY

Bundle Protocol Version 7 Wireshark Dissectors

IETF 112 DTN WG

Brian Sipos
JHU APL

Background

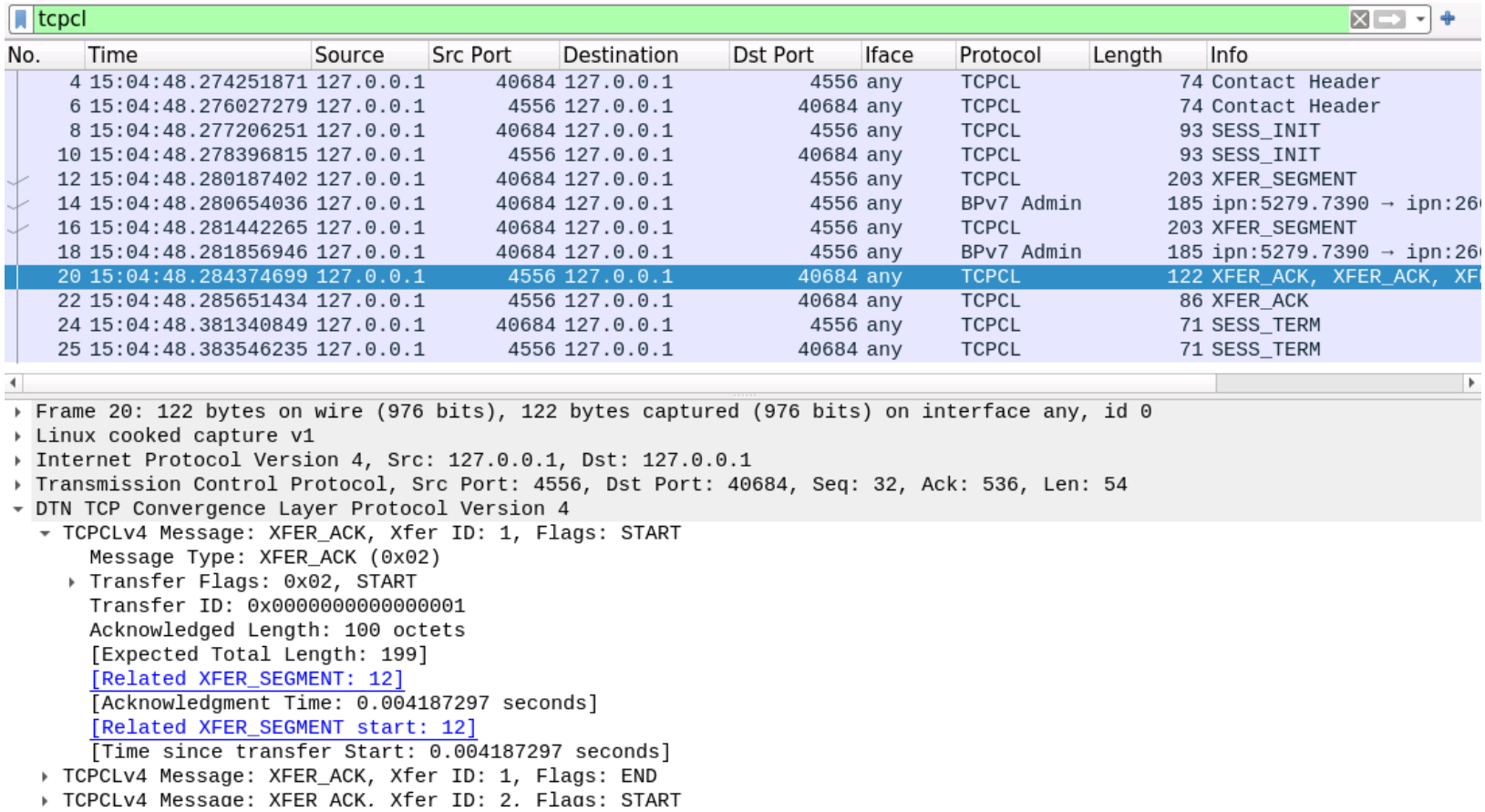
- Since early 2019 there has been a development version of TCPCLv4, BPv7, and (since late 2019) BPSec.
- These were used as both proof-of-concept and steering for TCPCL and as a framework for higher-level protocols:
  - ACME Node ID Validation
  - BPSec COSE context
- Recently these were integrated into wireshark main-branch dissectors for TCPCL and “bundle”.
  - TCPCL uses explicit contact header version negotiation, v3 was pre-existing.
  - The “bundle” dissector (used by TCPCL, UDPCL, LTP) uses version introspection, v4-6 were pre-existing.

Current Behavior

- The dissectors use wireshark “dissector tables” to delegate handling:
  - TCPCLv4 session and transfer extension types
  - BPv7 block types (with all types from BPbis included)
  - BPv7 Administrative Record types (types from BPbis included)
  - BPv7 payload (based on IPN service number and DTN demux name)
  - BPSec per-context parameter types and result types (with default context included)
- The TCPCL dissector includes de-fragmentation and sequence analysis (time-for-full-transfer, time-to-ACK-segment, etc.).
  - Packet icons indicate segment ACK associations.
  - During integration this was applied to TCPCLv3 as well.
- The TCPCL dissector lost the ability to opportunistically dissect messages when contact headers are not captured.
- The BPv7 dissector includes de-fragmentation and sequence analysis (duplicate identity detection, status report cross-reference, etc.)
  - Packet icons indicate status subjects and bundle retransmits.
  - The block dissector is also BPSec-aware, indicating which blocks are BIB or BCB targets.

TCPCLv4 Example Capture

Example shows init/termination, segmented transfer, and message packing.



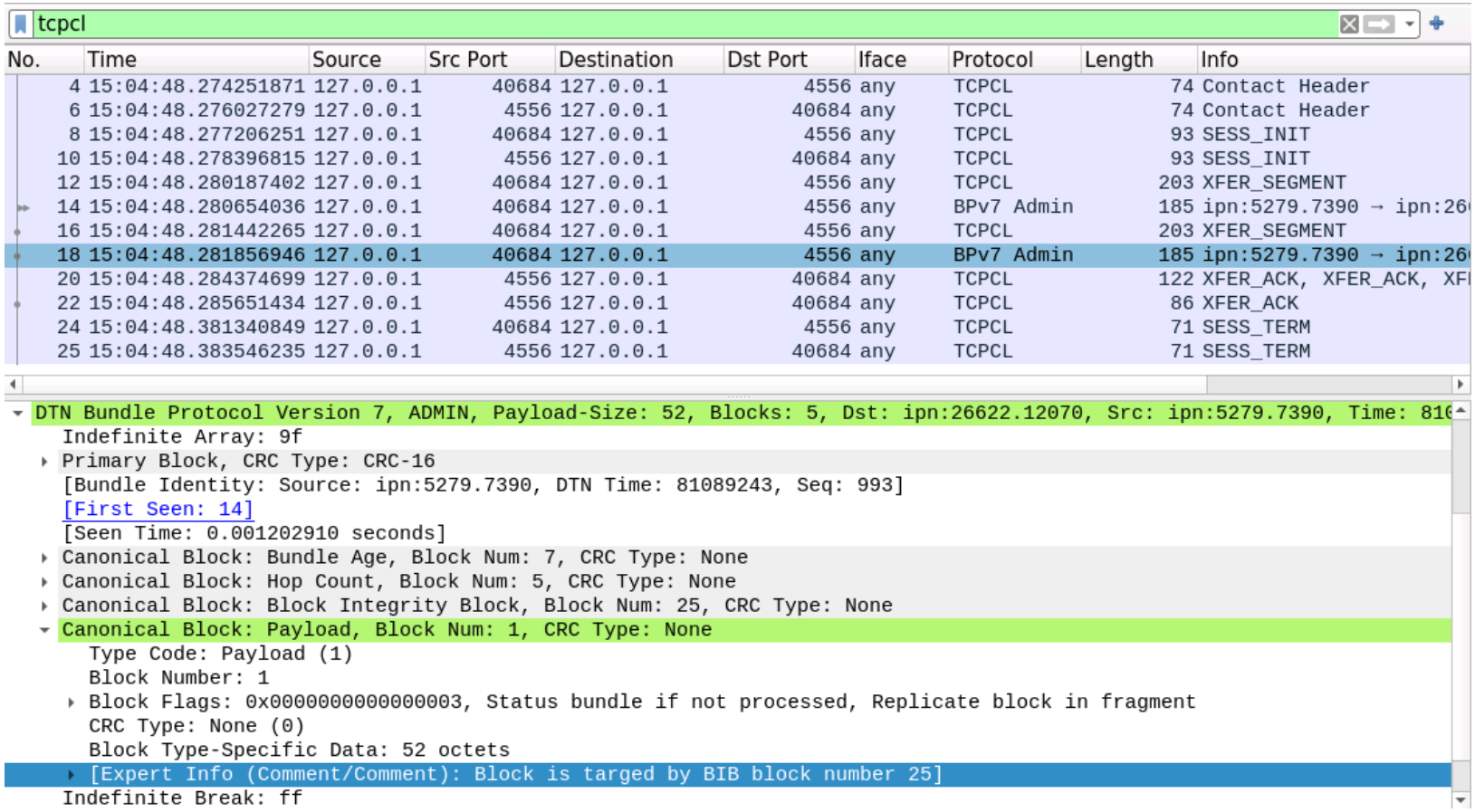
tcpcl

No.	Time	Source	Src Port	Destination	Dst Port	Iface	Protocol	Length	Info
4	15:04:48.274251871	127.0.0.1	40684	127.0.0.1	4556	any	TCPCL	74	Contact Header
6	15:04:48.276027279	127.0.0.1	4556	127.0.0.1	40684	any	TCPCL	74	Contact Header
8	15:04:48.277206251	127.0.0.1	40684	127.0.0.1	4556	any	TCPCL	93	SESS_INIT
10	15:04:48.278396815	127.0.0.1	4556	127.0.0.1	40684	any	TCPCL	93	SESS_INIT
12	15:04:48.280187402	127.0.0.1	40684	127.0.0.1	4556	any	TCPCL	203	XFER_SEGMENT
14	15:04:48.280654036	127.0.0.1	40684	127.0.0.1	4556	any	BPv7 Admin	185	ipn:5279.7390 → ipn:26
16	15:04:48.281442265	127.0.0.1	40684	127.0.0.1	4556	any	TCPCL	203	XFER_SEGMENT
18	15:04:48.281856946	127.0.0.1	40684	127.0.0.1	4556	any	BPv7 Admin	185	ipn:5279.7390 → ipn:26
20	15:04:48.284374699	127.0.0.1	4556	127.0.0.1	40684	any	TCPCL	122	XFER_ACK, XFER_ACK, XF
22	15:04:48.285651434	127.0.0.1	4556	127.0.0.1	40684	any	TCPCL	86	XFER_ACK
24	15:04:48.381340849	127.0.0.1	40684	127.0.0.1	4556	any	TCPCL	71	SESS_TERM
25	15:04:48.383546235	127.0.0.1	4556	127.0.0.1	40684	any	TCPCL	71	SESS_TERM

```
▸ Frame 20: 122 bytes on wire (976 bits), 122 bytes captured (976 bits) on interface any, id 0
▸ Linux cooked capture v1
▸ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▸ Transmission Control Protocol, Src Port: 4556, Dst Port: 40684, Seq: 32, Ack: 536, Len: 54
▾ DTN TCP Convergence Layer Protocol Version 4
  ▾ TCPCLv4 Message: XFER_ACK, Xfer ID: 1, Flags: START
      Message Type: XFER_ACK (0x02)
    ▸ Transfer Flags: 0x02, START
      Transfer ID: 0x0000000000000001
      Acknowledged Length: 100 octets
      [Expected Total Length: 199]
      [Related XFER_SEGMENT: 12]
      [Acknowledgment Time: 0.004187297 seconds]
      [Related XFER_SEGMENT start: 12]
      [Time since transfer Start: 0.004187297 seconds]
  ▸ TCPCLv4 Message: XFER_ACK, Xfer ID: 1, Flags: END
  ▸ TCPCLv4 Message: XFER_ACK, Xfer ID: 2, Flags: START
```

BPv7 Example Capture

Example shows Admin status record with BIB covering payload data.



Future and Maintenance

- The BPv7 dissector, without enhancements, is in the 3.6 release branch.
- The TCPCLv4 dissector and some BPv7 related enhancements (fall-through heuristic BTSD dissection) will likely be in 3.8 release.
- The current bundle version introspection logic is not based on any documented standard.
  - There is a similar logic proposed as a part of draft-sipos-dtn-udpcl, and could be extracted as a separate short document.
- TCPCL opportunistic message dissection could be added.
  - This could be done with known-message-type header detection, and some assumptions (i.e. check v4 types first, then v3)
- Now that the dissectors are in the upstream Wireshark project, proposed and future extensions can use this as a basis for experimentation and analysis.
  - The Wireshark project issue tracker is the right place for identifying defects and enhancements.