A Minimal Set of Transport Services for TAPS Systems

draft-ietf-taps-minset-00

Michael Welzl, Stein-Gjessing

neat

TAPS @ IETF 100
14. 11. 2017

This update in a nutshell

1. Also: early configuration to guide protocol choice, avoid "cornering" ourselves (e.g.: first pick UDP, then get a request for reliable data transfer)

2. IETF 99 request #1: consider fall-back to UDP

3. IETF 99 request #2: make it clear that this is not an API proposal
   – Text states this very clearly now (we think)

Early configuration: Example decision tree

- Will you need some form of reliability?
  No: all protocols can be used.
  Is any of the following useful to the application?
  - Specify checksum coverage used by the sender
  - Specify min. checksum coverage required by receiver
    Yes: UDP-Lite is preferred; No: UDP is preferred.

Yes: SCTP or TCP can be used.

Example decision tree /2

- Is any of the following useful to the application?
  - Hand over a message to reliably transfer (possibly multiple times) before connection establishment
  - Suggest timeout to the peer
  - Notification of Excessive Retransmissions (early warning below abortion threshold)
  - Notification of ICMP error message arrival

Yes: TCP is preferred.

No: SCTP and TCP are equally preferable.

Updated abstract interface description

- **CREATE** (flow-group-id, reliability, checksum_coverage, config_msg_prio, earlymsg_timeout_notifications)
- **CONFIGURE_TIMEOUT** (flow-group-id [timeout] [peer_timeout] [retrans_notify])
- **CONFIGURE_URGENCY** (flow-group-id [scheduler] [capacity_profile] [low_watermark])
- **CONFIGURE_PRIORITY** (flow-id priority)
- **CONFIGURE_CHECKSUM** (flow-id [send [send_length]] [receive [receive_length]])
- **CONNECT** (flow-id dst_addr), **LISTEN** (flow-id)
- **CLOSE** (flow-id), **ABORT** (flow-id)
- **SEND_FRAME** (flow-id frame [reliability] [ordered] [bundle] [delack] [fragment] [idempotent])
- **RECEIVE_FRAME** (flow-id buffer)

RED = No UDP fall-back

minset abstract API, cont'd

- **NOTIFICATIONS**
  - Excessive Retransmissions
  - ICMP Arrival (parameter: ICMP message); ECN Arrival
  - Timeout (parameter: s seconds)
  - Close; Abort
  - Drain
  - Path Change (parameter: path identifier)
  - Send Failure

- **QUERY_PROPERTIES**
  - maximum frame size that may be sent without fragmentation; maximum transport frame size that can be sent; maximum transport frame size that can be received; maximum amount of data that can possibly be sent before or during connection establishment

Conclusion

- Reminder: what you saw was a more efficient/condensed way of writing the transport features, not a proposed API
  - and the draft now explicitly says this

- What next?
  - Fall-back to TLS? Or HTTPS?
  - What are the usage scenarios we're envisioning?