TAPS
Internet-Draft
Intended status: Informational
Expires: September 19, 2016

S. Gjessing
M. Welzl
University of Oslo

March 18, 2016

A Minimal Set of Transport Services for TAPS Systems
draft-gjessing-taps-minset-01

Abstract

This draft will eventually recommend a minimal set of IETF Transport Services offered by end systems supporting TAPS, and give guidance on choosing among the available mechanisms and protocols. It categorizes the set of transport services given in the TAPS document draft-ietf-taps-transports-usage-00, assuming that the eventual minimal set of transport services will be based on a similar form of categorization.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

An application has an intended usage and demands for transport services, and the task of any system that implements TAPS is to offer these services to its applications, i.e. the applications running on top of TAPS, without binding an application to a particular transport protocol.

The present draft is based on [TAPS1] and [TAPS2] and follows the same terminology (also listed below). The purpose of these two drafts is, according to the TAPS charter, to "Define a set of Transport Services, identifying the services provided by existing IETF protocols and congestion control mechanisms." This is item 1 in the list of working group tasks. Also according to the TAPS charter, the working group will then "Specify the subset of those Transport Services, as identified in item 1, that end systems supporting TAPS will provide, and give guidance on choosing among available mechanisms and protocols. Note that not all the capabilities of IETF Transport protocols need to be exposed as Transport Services." Hence it is necessary to minimize the number of services that are offered. We begin this by grouping the transport features.

Following [TAPS2], we divide the transport service features into two main groups as follows:

1. Connection related transport service features
   - Establishment
   - Availability

- Maintenance
- Termination

2. Data Transfer Related Transport Service Features
   - Sending Data
   - Receiving Data
   - Errors

Because QoS is out of scope of TAPS, this document assumes a "best effort" service model [RFC5290], [RFC7305]. Applications using a TAPS system can therefore not make any assumptions about e.g. the time it will take to send a message. There are however certain requirements that are strictly kept by transport protocols today, and these must also be kept by a TAPS system. Some of these requirements relate to features that we call "Functional".

Functional features provide functionality that cannot be used without the application knowing about them, or else they violate assumptions that might cause the application to break. For example, unordered message delivery is a functional feature: it cannot be used without the application knowing about it because the application's assumption could be that messages arrive in-order, and in this case unordered delivery could cause the application to break. Change DSCP and data bundling (Nagle in TCP) are optimizing features: if a TAPS system autonomously decides to enable or disable them, an application will not break, but a TAPS system may be able to communicate more efficiently if the application is in control of this optimizing feature. Change DSCP and data bundling are examples of features that require application-specific knowledge (about delay/bandwidth requirements and the length of future data blocks that are to be transmitted, respectively). Some features, however, do not always require application-specific knowledge, and could therefore sometimes be used by a TAPS system without exposing them to the application. We call these features potentially automatable.

To summarize, features offered to applications are divided into two groups as follows:

o Potentially automatable
  It may sometimes be possible to use this feature without support by the application.
o Application-specific
  It is not possible to use this feature without support by the application.

The Application-specific features are further divided into two groups:

- o Functional
  This feature is application-specific, and using it without explicitly involving the application could lead to incorrect operation.
- o Optimizing
  This feature is application-specific, and can allow an application to improve its performance.

In the following, some features are additionally marked as DELETED. These features are IETF Transport protocol features that are not exposed to the TAPS user because they include functionality that is automatable. A few features are marked as "ADDED". These provide non-automatable functionality of DELETED features.

2. Terminology (as defined by draft-ietf-taps-transports-10)

The following terms are used throughout this document, and in subsequent documents produced by TAPS that describe the composition and decomposition of transport services.

Transport Service Feature: a specific end-to-end feature that the transport layer provides to an application. Examples include confidentiality, reliable delivery, ordered delivery, message-versus-stream orientation, etc.

Transport Service: a set of Transport Features, without an association to any given framing protocol, which provides a complete service to an application.

Transport Protocol: an implementation that provides one or more different transport services using a specific framing and header format on the wire.

Transport Service Instance: an arrangement of transport protocols with a selected set of features and configuration parameters that implements a single transport service, e.g., a protocol stack (RTP over UDP).

Application: an entity that uses the transport layer for end-to-end delivery data across the network (this may also be an upper layer protocol or tunnel encapsulation).

3. The superset of transport service features

This section is based on the classification of the transport service features in pass 3 of [TAPS2]. As noted earlier, whether the usage of potentially automatable features can be automatized in a TAPS system depends on how much network-specific information an application wants to manipulate (e.g., to directly expose to its user). Therefore, in the following, "application-specific knowledge" refers to knowledge that only applications have, as opposed to all knowledge that applications may want to have.

3.1. CONNECTION Related Transport Service Features

ESTABLISHMENT:

- Connect
  Protocols: TCP, SCTP
  Functional because the notion of a connection is often reflected in applications as an expectation to be able to communicate after a "Connect" succeeded, with a communication sequence relating to this feature that is defined by the application protocol.
  ADDED.

- Specify IP Options
  Protocols: TCP
  Potentially automatable because IP Options relate to knowledge about the network, not the application.
  DELETED.

- Request multiple streams
  Protocols: SCTP
  Potentially automatable because using multi-streaming does not require application-specific knowledge.
  DELETED.

- Obtain multiple sockets
  Protocols: SCTP
  Potentially automatable because the usage of multiple paths to communicate to the same end host relates to knowledge about the network, not the application.
  DELETED.

AVAILABILITY:

- Listen
  Protocols: All
  Functional because the notion of accepting connection requests is often reflected in application as an expectation to be able to communicate after a "Listen" succeeded, with a communication

sequence relating to this feature that is defined by the
application protocol.
ADDED.

- Listen, 1 specified local interface
  Protocols: TCP, SCTP
  Potentially automatable because decisions about local interfaces relate to knowledge about the network and the Operating System, not the application.
  DELETED.

- Listen, N specified local interfaces
  Protocols: SCTP
  Potentially automatable because decisions about local interfaces relate to knowledge about the network and the Operating System, not the application.
  DELETED.

- Listen, all local interfaces (unspecified)
  Protocols: TCP, SCTP
  Potentially automatable because decisions about local interfaces relate to knowledge about the network and the Operating System, not the application.
  DELETED.

- Obtain requested number of streams
  Protocols: SCTP
  Potentially automatable because using multi-streaming does not require application-specific knowledge.

MAINTENANCE:

- Change timeout for aborting connection (using retransmit limit or time value)
  Protocols: TCP, SCTP
  Functional because this is closely related to potentially assumed reliable data delivery.

- Control advertising timeout for aborting connection to remote endpoint
  Protocols: TCP
  Functional because this is closely related to potentially assumed reliable data delivery.

- Disable Nagle algorithm
  Protocols: TCP, SCTP
  Optimizing because this decision depends on knowledge about the size of future data blocks and the delay between them.

- Request an immediate heartbeat, returning success/failure
  Protocols: SCTP
  Potentially automatable because this informs about network-specific knowledge.

- Set protocol parameters
  Protocols: SCTP
  SCTP parameters: RTO.Initial; RTO.Min; RTO.Max; Max.Burst; RTO.Alpha; RTO.Beta; Valid.Cookie.Life; Association.Max.Retrans; Path.Max.Retrans; Max.Init.Retransmits; HB.interval; HB.Max.Burst
  Potentially automatable because these parameters relate to knowledge about the network, not the application.

- Notification of Excessive Retransmissions (early warning below abortion threshold)
  Protocols: TCP
  Optimizing because it is an early warning to the application, informing it of an impending functional event.

- Notification of ICMP error message arrival
  Protocols: TCP
  Optimizing because these messages can inform about success or failure of functional features (e.g., host unreachable relates to "Connect")

- o Status (query or notification)
  Protocols: SCTP
  SCTP parameters: association connection state; socket list; socket reachability states; current receiver window size; current congestion window sizes; number of unacknowledged DATA chunks; number of DATA chunks pending receipt; primary path; most recent SRTT on primary path; RTO on primary path; SRTT and RTO on other destination addresses; socket becoming active / inactive
  Potentially automatable because these parameters relate to knowledge about the network, not the application.

- o Set primary path
  Protocols: SCTP
  Potentially automatable because it requires using multiple sockets, but obtaining multiple sockets in the CONNECTION.ESTABLISHMENT category is potentially automatable.

- o Change DSCP
  Protocols: TCP
  Optimizing because choosing a suitable DSCP value requires application-specific knowledge.

TERMINATION:

- o Close after reliably delivering all remaining data, causing an event informing the application on the other side
  Protocols: TCP, SCTP
  Functional because the notion of a connection is often reflected in applications as an expectation to have all outstanding data delivered and no longer be able to communicate after a "Close" succeeded, with a communication sequence relating to this feature that is defined by the application protocol.

- o Abort without delivering remaining data, causing an event informing the application on the other side
  Protocols: TCP, SCTP
  Functional because the notion of a connection is often reflected in applications as an expectation to potentially not have all outstanding data delivered and no longer be able to communicate

after an "Abort" succeeded, with a communication sequence relating to this feature that is defined by the application protocol.

- Timeout event when data could not be delivered for too long
  Protocols: TCP, SCTP
  Functional because this notifies that potentially assumed reliable data delivery is no longer provided.

3.2. DATA Transfer Related Transport Service Features

3.2.1. Sending Data

- Reliably transfer data
  Protocols: TCP, SCTP
  Functional because this is closely tied to properties of the data that an application sends or expects to receive.

- Notifying the receiver to promptly hand over data to application
  Protocols: TCP
  Optimizing because this is meant to control sleep times of the application’s receiving process.

- Message identification
  Protocols: SCTP
  Functional because this is closely tied to properties of the data that an application sends or expects to receive.

- Choice of stream
  Protocols: SCTP
  Potentially automatable because it requires using multiple streams, but requesting multiple streams in the CONNECTION.ESTABLISHMENT category is potentially automatable.

- Choice of path (destination address)
  Protocols: SCTP

Potentially automatable because it requires using multiple sockets, but obtaining multiple sockets in the CONNECTION.ESTABLISHMENT category is potentially automatable.

- Message lifetime
  Protocols: SCTP
  Optimizing because only applications know about the time criticality of their communication.

- Choice between unordered (potentially faster) or ordered delivery
  Protocols: SCTP
  Functional because this is closely tied to properties of the data that an application sends or expects to receive.

- Request not to bundle messages
  Protocols: SCTP
  Optimizing because this decision depends on knowledge about the size of future data blocks and the delay between them.

- Specifying a "payload protocol-id" (handed over as such by the receiver)
  Protocols: SCTP
  Functional because it allows application data with every message, for the sake of identification of data, which by itself is application-specific.

3.2.2. Receiving Data

- Receive data
  Protocols: TCP, SCTP
  Functional because a TAPS system must be able to send and receive data.

- Choice of stream to receive from
  Protocols: SCTP

Potentially automatable because it requires using multiple streams, but requesting multiple streams in the CONNECTION.ESTABLISHMENT category is potentially automatable.

- Message identification
  Protocols: SCTP
  Functional because this is closely tied to properties of the data that an application sends or expects to receive.

- Information about partial message arrival
  Protocols: SCTP
  Functional because this is closely tied to properties of the data that an application sends or expects to receive.

3.2.3. Errors

- Notification of send failures
  Protocols: All
  Functional because this notifies that potentially assumed reliable data delivery is no longer provided.
  ADDED.

- Notification of unsent messages
  Protocols: SCTP
  Automatable because the distinction between unsent and unacknowledged is network-specific.
  DELETED.

- Notification of unacknowledged messages
  Protocols: SCTP
  Automatable because the distinction between unsent and unacknowledged is network-specific.
  DELETED.

4. Conclusion

The eventual recommendations are:

- A TAPS system should exhibit all functional features that are offered by the transport protocols that it uses because these features could otherwise not be utilized by the TAPS system. It can still be possible to implement a TAPS system that does not offer all functional features, e.g. for the sake of uniform application operation across a broader set of protocols, but then the corresponding functionality of transport protocols is not exploited.
- A TAPS system should exhibit all application-specific optimizing features. If an application-specific optimizing feature is only available in a subset of the transport protocols used by the TAPS system, it should be acceptable for the TAPS system to ignore its usage when the transport protocol that is currently used does not provide it because of the performance-optimizing nature of the feature and the initially mentioned assumption of "best effort" operation.
- By hiding potentially automatable features from the application, a TAPS system can gain opportunities to automatize network-related functionality. This can facilitate using the TAPS system for the application programmer and it allows for optimizations that may not be possible for an application. For instance, a kernel-level TAPS system that hides SCTP multi-streaming from applications could theoretically map application-level connections from multiple applications onto the same SCTP association. Similarly, system-wide configurations regarding the usage of multiple interfaces could be exploited if the choice of the interface is not given to the application. However, if an application wants to directly expose such choices to its user, not offering this functionality can become a disadvantage of a TAPS system. This is a trade-off that must be considered in TAPS system design.

Given that the intention of TAPS is to break the design-time binding between applications and transport protocols, the decision on which features a TAPS system provides should also depend on the protocols that support them. Features that are provided by only one particular transport protocol have the potential to tie applications to that protocol. They should either not be offered, or replaced by fall-back functionality that allows for semantically correct operation (for example, ordered data delivery is correct but potentially slower for an application that requests unordered data delivery. "Potentially slower" is not a hindrance to correct operation within the "best effort" service model).

5. Acknowledgements

This work has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 644334 (NEAT). The views expressed are solely those of the author(s).

6. IANA Considerations

XX RFC ED - PLEASE REMOVE THIS SECTION XXX

This memo includes no request to IANA.

7. Security Considerations

Security will be considered in future versions of this document.

8. Informative References

[RFC5290] Floyd, S. and M. Allman, "Comments on the Usefulness of Simple Best-Effort Traffic", RFC 5290, DOI 10.17487/RFC5290, July 2008, <http://www.rfc-editor.org/info/rfc5290>.

[RFC7305] Lear, E., Ed., "Report from the IAB Workshop on Internet Technology Adoption and Transition (ITAT)", RFC 7305, DOI 10.17487/RFC7305, July 2014, <http://www.rfc-editor.org/info/rfc7305>.

[TAPS1] Fairhurst, G., Trammell, B., and M. Kuehlewind, "Services provided by IETF transport protocols and congestion control mechanisms", Internet-draft draft-ietf-taps-transports-10, March 2016.

[TAPS2] Welzl, M., Tuexen, M., and N. Khademi, "An Approach to Identify Services Provided by IETF Transport Protocols and Congestion Control Mechanisms", Internet-draft draft-ietf-taps-transports-usage-00, June 2015.

Authors’ Addresses

Stein Gjessing
University of Oslo
PO Box 1080 Blindern
Oslo N-0316
Norway

Phone: +47 22 85 24 44
Email: steing@ifi.uio.no

Michael Welzl
University of Oslo
PO Box 1080 Blindern
Oslo N-0316
Norway

Phone: +47 22 85 24 20
Email: michawe@ifi.uio.no