Network Working Group
Internet-Draft
Intended status: Informational
Expires: June 22, 2015

G. Fairhurst, Ed.
University of Aberdeen
B. Trammell, Ed.
M. Kuehlewind, Ed.
ETH Zurich

December 19, 2014

Services provided by IETF transport protocols and congestion control mechanisms
draft-ietf-taps-transports-01

Abstract

This document describes services provided by existing IETF protocols and congestion control mechanisms. It is designed to help application and network stack programmers and to inform the work of the IETF TAPS Working Group.

1. Introduction

Most Internet applications make use of the Transport Services provided by TCP (a reliable, in-order stream protocol) or UDP (an unreliable datagram protocol). We use the term "Transport Service" to mean the end-to-end service provided to an application by the transport layer. That service can only be provided correctly if information about the intended usage is supplied from the application. The application may determine this information at design time, compile time, or run time, and may include guidance on whether a feature is required, a preference by the application, or something in between. Examples of features of Transport Services are reliable delivery, ordered delivery, content privacy to in-path devices, integrity protection, and minimal latency.

The IETF has defined a wide variety of transport protocols beyond TCP and UDP, including TCP, SCTP, DCCP, MP-TCP, and UDP-Lite. Transport services may be provided directly by these transport protocols, or layered on top of them using protocols such as WebSockets (which runs over TCP) or RTP (over TCP or UDP). Services built on top of UDP or UDP-Lite typically also need to specify additional mechanisms, including a congestion control mechanism (such as a windowed congestion control, TFRC or LEDBAT congestion control mechanism). This extends the set of available Transport Services beyond those provided to applications by TCP and UDP.

Transport protocols can also be differentiated by the features of the services they provide: for instance, SCTP offers a message-based service that does not suffer head-of-line blocking when used with multiple stream, because it can accept blocks of data out of order, UDP-Lite provides partial integrity protection, and LEDBAT can provide low-priority "scavenger" communication.

2. Terminology

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

[Editor Note: The terminology below was presented at the TAPS WG meeting in Honolulu. While the factoring of the terminology seems uncontroversial, there may be some entities which still require names (e.g. information about the interface between the transport and lower layers which could lead to the availablity or unavailibility of

certain transport protocol features). Comments are welcome via the TAPS mailing list.]

Transport Service Feature: a specific end-to-end feature that a transport service provides to its clients. Examples include confidentiality, reliable delivery, ordered delivery, message-versus-stream orientation, etc.

Transport Service: a set of transport service 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 Protocol Component: an implementation of a transport service feature within a protocol.

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 encpasulation).

3. Existing Transport Protocols

This section provides a list of known IETF transport protocol and transport protocol frameworks.

[Editor Note: Contributions to the sections in the list below are welcome]

3.1. Transport Control Protocol (TCP)

TCP is an IETF standards track transport protocol. [RFC0793] introduces TCP as follows: "The Transmission Control Protocol (TCP) is intended for use as a highly reliable host-to-host protocol between hosts in packet-switched computer communication networks, and in interconnected systems of such networks." Since its introduction, TCP has become the default connection-oriented, stream-based transport protocol in the Internet. It is widely implemented by endpoints and widely used by common applications.

3.1.1. Protocol Description

TCP is a connection-oriented protocol, providing a three way handshake to allow a client and server to set up a connection, and mechanisms for orderly completion and immediate teardown of a connection. TCP is defined by a family of RFCs [RFC4614].

TCP provides multiplexing to multiple sockets on each host using port numbers. An active TCP session is identified by its four-tuple of local and remote IP addresses and local port and remote port numbers.

TCP partitions a continuous stream of bytes into segments, sized to fit in IP packets, constrained by the maximum size of lower layer frame. PathMTU discovery is supported. Each byte in the stream is identified by a sequence number. The sequence number is used to order segments on receipt, to identify segments in acknowledgments, and to detect unacknowledged segments for retransmission. This is the basis of TCP's reliable, ordered delivery of data in a stream. TCP Selective Acknowledgment [RFC2018] extends this mechanism by making it possible to identify missing segments more precisely, reducing spurious retransmission.

Receiver flow control is provided by a sliding window: limiting the amount of unacknowledged data that can be outstanding at a given time. The window scale option [RFC7323] allows a receiver to use windows greater than 64KB.

All TCP senders provide Congestion Control: This uses a separate window, where each time congestion is detected, this congestion window is reduced. A receiver detects congestion using one of three mechanisms: A retransmission timer, loss (interpreted as a congestion signal), and Explicit Congestion Notification (ECN) [RFC3168] to provide early signaling (see [I-D.ietf-aqm-ecn-benefits])

A TCP protocol instance can be extended [RFC4614] and tuned. Some features are sender-side only, requiring no negotiation with the receiver; some are receiver-side only, some are explicitly negotiated during connection setup.

By default, TCP segment partitioning uses Nagle's algorithm [RFC0896] to buffer data at the sender into large segments, potentially incurring sender-side buffering delay; this algorithm can be disabled by the sender to transmit more immediately, e.g. to enable smoother interactive sessions.

A TCP service is unicast.

3.1.2. Interface description

A TCP API is defined in [REF], but there is currently no API specified in the RFC series.

In API implementations derived from the BSD Sockets API, TCP sockets are created using the "SOCK_STREAM" socket type.

The features used by a protocol instance may be set and tuned via this API.

(more on the API goes here)

3.1.3. Transport Protocol Components

The transport protocol components provided by TCP are:

- unicast

- connection-oriented setup with feature negotiation

- port multiplexing

- reliable delivery

- ordered delivery

- segmented, stream-oriented delivery in a single stream

- congestion control

(discussion of how to map this to features and TAPS: what does the higher layer need to decide? what can the transport layer decide based on global settings? what must the transport layer decide based on network characteristics?)

3.2. Multipath TCP (MP-TCP)

[Editor Note: a few sentences describing Multipath TCP [RFC6824] go here. Note that this adds transport-layer multihoming to the components TCP provides]

3.3. Stream Control Transmission Protocol (SCTP)

SCTP [RFC4960] is an IETF standards track transport protocol that provides a bidirectional s set of logical unicast meessage streams over a connection-oriented protocol. The protocol and API use messages, rather than a byte-stream. Each stream of messages is

independently managed, therefore retransmission does not hold back data sent using other logical streams.

The SCTP Partial Reliability Extension (SCTP-PR) is defined in [RFC3758].

[EDITOR'S NOTE: Michael Tuexen and Karen Nielsen signed up as contributors for these sections.]

3.3.1. Protocol Description

An SCTP service is unicast.

3.3.2. Interface Description

The SCTP API is described in the specifications published in the RFC series.

3.3.3. Transport Protocol Components

The transport protocol components provided by SCTP are:

- unicast
- connection-oriented setup with feature negotiation
- port multiplexing
- reliable or partially reliable delivery
- ordered delivery within a stream
- support for multiple prioritised streams
- message-oriented delivery
- congestion control

[EDITOR'S NOTE: Please update list.]

3.4. User Datagram Protocol (UDP)

The User Datagram Protocol (UDP) [RFC0768] [RFC2460] is an IETF standards track transport protocol. It provides a uni-directional minimal message-passing transport that has no inherent congestion control mechanisms or other transport functions. IETF guidance on the use of UDP is provided in [RFC5405]. UDP is widely implemented by endpoints and widely used by common applications.

[EDITOR'S NOTE: Kevin Fall signed up as a contributor for this section.]

3.4.1. Protocol Description

UDP is a connection-less datagram protocol, with no connection setup or feature negotiation. The protocol and API use messages, rather than a byte-stream. Each stream of messages is independently managed, therefore retransmission does not hold back data sent using other logical streams.

It provides multiplexing to multiple sockets on each host using port numbers. An active UDP session is identified by its four-tuple of local and remote IP addresses and local port and remote port numbers.

UDP fragments packets into IP packets, constrained by the maximum size of lower layer frame.

Mechanisms for receiver flow control, congestion control, PathMTU discovery, support for ECN, etc need to be provided by upper layer protocols [RFC5405].

For IPv4 the UDP checksum is optional, but recommended for use in the general Internet [RFC5405]. [RFC2460] requires the use of this checksum for IPv6, but [RFC6935] permits this to be relaxed for specific types of application. The checksum support considerations for omitting the checksum are defined in [RFC6936].

A UDP service may support IPv4 broadcast, multicast, anycast and unicast.

3.4.2. Interface Description

There is no current API specified in the RFC Series, but guidance on use of common APIs is provided in [RFC5405].

3.4.3. Transport Protocol Components

The transport protocol components provided by UDP are:

- unicast

- IPv4 broadcast, multicast and anycast

- non-reliable, non-ordered delivery

- message-oriented delivery

o optional checksum protection.

3.5. Lightweight User Datagram Protocol (UDP-Lite)

The Lightweight User Datagram Protocol (UDP-Lite) [RFC3828] is an IETF standards track transport protocol. UDP-Lite provides a bidirectional set of logical unicast or multicast message streams over a datagram protocol. IETF guidance on the use of UDP-Lite is provided in [RFC5405].

[EDITOR'S NOTE: Gorry Fairhurst signed up as a contributor for this section.]

3.5.1. Protocol Description

UDP-Lite is a connection-less datagram protocol, with no connection setup or feature negotiation. The protocol and API use messages, rather than a byte-stream. Each stream of messages is independently managed, therefore retransmission does not hold back data sent using other logical streams.

It provides multiplexing to multiple sockets on each host using port numbers. An active UDP-Lite session is identified by its four-tuple of local and remote IP addresses and local port and remote port numbers.

UDP-Lite fragments packets into IP packets, constrained by the maximum size of lower layer frame.

UDP-Lite changes the semantics of the UDP "payload length" field to that of a "checksum coverage length" field. Otherwise, UDP-Lite is semantically identical to UDP. Applications using UDP-Lite therefore can not make assumptions regarding the correctness of the data received in the insensitive part of the UDP-Lite payload.

As for UDP, mechanisms for receiver flow control, congestion control, PathMTU discovery, support for ECN, etc need to be provided by upper layer protocols [RFC5405].

Examples of use include a class of applications that can derive benefit from having partially-damaged payloads delivered, rather than discarded. One use is to support are tolerate payload corruption and over paths that include error-prone links, another application is when header integrity checks are required but payload integrity is provided by some other mechanism (e.g. [RFC6936].

A UDP-Lite service may support IPv4 broadcast, multicast, anycast and unicast.

3.5.2. Interface Description

There is no current API specified in the RFC Series, but guidance on use of common APIs is provided in [RFC5405].

The interface of UDP-Lite differs from that of UDP by the addition of a single (socket) option that communicates a checksum coverage length value: at the sender, this specifies the intended checksum coverage, with the remaining unprotected part of the payload called the "error-insensitive part". The checksum coverage may also be made visible to the application via the UDP-Lite MIB module [RFC5097].

3.5.3. Transport Protocol Components

The transport protocol components provided by UDP-Lite are:

- unicast

- IPv4 broadcast, multicast and anycast

- non-reliable, non-ordered delivery

- message-oriented delivery

- partial integrity protection

3.6. Datagram Congestion Control Protocol (DCCP)

Datagram Congestion Control Protocol (DCCP) [RFC4340] is an IETF standards track bidirectional transport protocol that provides unicast connections of congestion-controlled unreliable messages. DCCP is suitable for applications that transfer fairly large amounts of data and that can benefit from control over the trade off between timeliness and reliability [RFC4336].

[EDITOR'S NOTE: Gorry Fairhurst signed up as a contributor for this section.]

3.6.1. Protocol Description

DCCP is a connection-oriented datagram protocol, providing a three way handshake to allow a client and server to set up a connection, and mechanisms for orderly completion and immediate teardown of a connection. The protocol is defined by a family of RFCs.

It provides multiplexing to multiple sockets on each host using port numbers. An active DCCP session is identified by its four-tuple of local and remote IP addresses and local port and remote port numbers.

At connection setup, DCCP also exchanges the the service code [RFC5595] mechanism to allow transport instantiations to indicate the service treatment that is expected from the network.

The protocol segments data into messages, sized to fit in IP packets, constrained by the maximum size of lower layer frame. Each message is identified by a sequence number. The sequence number is used to identify segments in acknowledgments, to detect unacknowledged segments, to measure RTT, etc. The protocol may support ordered or unordered delivery of data, and does not itself provide retransmission.

Receiver flow control is supported: limiting the amount of unacknowledged data that can be outstanding at a given time.

A DCCP protocol instance can be extended [RFC4340] and tuned. Some features are sender-side only, requiring no negotiation with the receiver; some are receiver-side only, some are explicitly negotiated during connection setup.

DCCP supports negotiation of the congestion control profile, examples of specified profiles include [RFC4341] [RFC4342] [RFC5662]. All IETF-defined methods provide Congestion Control.

Examples of suitable applications include interactive applications, streaming media or on-line games [RFC4336].

A DCCP service is unicast.

3.6.2. Interface Description

There is no current API specified in the RFC Series.

3.6.3. Transport Protocol Components

The transport protocol components provided by DCCP are:

- unicast

- connection-oriented setup

- feature negotiation

- non-reliable, ordered delivery

- message-oriented delivery

- partial integrity protection

3.7. Realtime Transport Protocol (RTP)

RTP provides an end-to-end network transport service, suitable for applications transmitting real-time data, such as audio, video or data, over multicast or unicast network services, including TCP, UDP, UDP-Lite, DCCP.

[EDITOR'S NOTE: Varun Singh signed up as contributor for this section.]

3.8. Transport Layer Security (TLS) and Datagram TLS (DTLS) as a

pseudotransport

(A few words on TLS [RFC5246] and DTLS [RFC6347] here, and how they get used by other protocols to meet security goals as an add-on interlayer above transport.)

3.8.1. Protocol Description

3.8.2. Interface Description

3.8.3. Transport Protocol Components

3.9. Hypertext Transport Protocol (HTTP) as a pseudotransport

[RFC3205]

3.9.1. Protocol Description

3.9.2. Interface Description

3.9.3. Transport Protocol Components

3.10. WebSockets

[RFC6455]

3.10.1. Protocol Description

3.10.2. Interface Description

3.10.3. Transport Protocol Components

4. Transport Service Features

(drawn from the candidate features provided by protocol components in the previous section - please discussion on list)

4.1. Complete Protocol Feature Matrix

(a comprehensive matrix table goes here; Volunteer: Dave Thaler)

5. IANA Considerations

This document has no considerations for IANA.

6. Security Considerations

This document surveys existing transport protocols and protocols providing transport-like services. Confidentiality, integrity, and authenticity are among the features provided by those services. This document does not specify any new components or mechanisms for providing these features. Each RFC listed in this document discusses the security considerations of the specification it contains.

7. Contributors

Non-editor contributors of text will be listed here, as in the authors section.

8. Acknowledgments

This work is partially supported by the European Commission under grant agreement FP7-ICT-318627 mPlane; support does not imply endorsement.

9. References

9.1. Normative References

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

9.2. Informative References

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980.

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, September 1981.

[RFC0896] Nagle, J., "Congestion control in IP/TCP internetworks", RFC 896, January 1984.

[RFC1122] Braden, R., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, October 1989.

[RFC2018] Mathis, M., Mahdavi, J., Floyd, S., and A. Romanow, "TCP Selective Acknowledgment Options", RFC 2018, October 1996.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, September 2001.

[RFC3205] Moore, K., "On the use of HTTP as a Substrate", BCP 56, RFC 3205, February 2002.

[RFC3390] Allman, M., Floyd, S., and C. Partridge, "Increasing TCP's Initial Window", RFC 3390, October 2002.

[RFC3758] Stewart, R., Ramalho, M., Xie, Q., Tuexen, M., and P. Conrad, "Stream Control Transmission Protocol (SCTP) Partial Reliability Extension", RFC 3758, May 2004.

[RFC3828] Larzon, L-A., Degermark, M., Pink, S., Jonsson, L-E., and G. Fairhurst, "The Lightweight User Datagram Protocol (UDP-Lite)", RFC 3828, July 2004.

[RFC4336] Floyd, S., Handley, M., and E. Kohler, "Problem Statement for the Datagram Congestion Control Protocol (DCCP)", RFC 4336, March 2006.

[RFC4340] Kohler, E., Handley, M., and S. Floyd, "Datagram Congestion Control Protocol (DCCP)", RFC 4340, March 2006.

[RFC4341] Floyd, S. and E. Kohler, "Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 2: TCP-like Congestion Control", RFC 4341, March 2006.

[RFC4342] Floyd, S., Kohler, E., and J. Padhye, "Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 3: TCP-Friendly Rate Control (TFRC)", RFC 4342, March 2006.

[RFC4614] Duke, M., Braden, R., Eddy, W., and E. Blanton, "A Roadmap for Transmission Control Protocol (TCP) Specification Documents", RFC 4614, September 2006.

[RFC4960] Stewart, R., "Stream Control Transmission Protocol", RFC 4960, September 2007.

[RFC5097] Renker, G. and G. Fairhurst, "MIB for the UDP-Lite protocol", RFC 5097, January 2008.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[RFC5348] Floyd, S., Handley, M., Padhye, J., and J. Widmer, "TCP Friendly Rate Control (TFRC): Protocol Specification", RFC 5348, September 2008.

[RFC5405] Eggert, L. and G. Fairhurst, "Unicast UDP Usage Guidelines for Application Designers", BCP 145, RFC 5405, November 2008.

[RFC5595] Fairhurst, G., "The Datagram Congestion Control Protocol (DCCP) Service Codes", RFC 5595, September 2009.

[RFC5662] Shepler, S., Eisler, M., and D. Noveck, "Network File System (NFS) Version 4 Minor Version 1 External Data Representation Standard (XDR) Description", RFC 5662, January 2010.

[RFC5925] Touch, J., Mankin, A., and R. Bonica, "The TCP Authentication Option", RFC 5925, June 2010.

[RFC5681] Allman, M., Paxson, V., and E. Blanton, "TCP Congestion Control", RFC 5681, September 2009.

[RFC6093] Gont, F. and A. Yourtchenko, "On the Implementation of the TCP Urgent Mechanism", RFC 6093, January 2011.

[RFC6298] Paxson, V., Allman, M., Chu, J., and M. Sargent, "Computing TCP's Retransmission Timer", RFC 6298, June 2011.

[RFC6935] Eubanks, M., Chimento, P., and M. Westerlund, "IPv6 and UDP Checksums for Tunneled Packets", RFC 6935, April 2013.

[RFC6936] Fairhurst, G. and M. Westerlund, "Applicability Statement for the Use of IPv6 UDP Datagrams with Zero Checksums", RFC 6936, April 2013.

[RFC6455] Fette, I. and A. Melnikov, "The WebSocket Protocol", RFC 6455, December 2011.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.

[RFC6691] Borman, D., "TCP Options and Maximum Segment Size (MSS)", RFC 6691, July 2012.

[RFC6824] Ford, A., Raiciu, C., Handley, M., and O. Bonaventure, "TCP Extensions for Multipath Operation with Multiple Addresses", RFC 6824, January 2013.

[RFC7323] Borman, D., Braden, B., Jacobson, V., and R. Scheffenegger, "TCP Extensions for High Performance", RFC 7323, September 2014.

[I-D.ietf-aqm-ecn-benefits]
Welzl, M. and G. Fairhurst, "The Benefits and Pitfalls of using Explicit Congestion Notification (ECN)", draft-ietf-aqm-ecn-benefits-00 (work in progress), October 2014.

Authors' Addresses

Godred Fairhurst (editor)
University of Aberdeen
School of Engineering, Fraser Noble Building
Aberdeen AB24 3UE

Email: gorry@erg.abdn.ac.uk

Brian Trammell (editor)
ETH Zurich
Gloriastrasse 35
8092 Zurich
Switzerland

Email: ietf@trammell.ch

Mirja Kuehlewind (editor)
ETH Zurich
Gloriastrasse 35
8092 Zurich
Switzerland

Email: mirja.kuehlewind@tik.ee.ethz.ch