TAPS Working Group   P. Tiesel
Internet-Draft   T. Enghardt
Intended status: Informational   TU Berlin
Expires: May 7, 2019   November 03, 2018

Communication Units Granularity Considerations for Multi-Path Aware Transport Selection

draft-tiesel-taps-communitgrany-03

Abstract

This document provides an approach how to reason about the composition of multi-path aware transport stacks. It discusses how to compose the functionality needed by stacking existing internet protocols and the fundamental mechanisms that are used in multi-path systems and the consequences of applying them to different granularities of communication units, e.g, on a message or stream granularity. This document is targeted as guidance for automation of destination selection, path selection, and transport protocol selection.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Today's Internet architecture faces a communication endpoint with a set of choices, including choosing a transport protocol and picking an IP protocol version. In many cases, e.g., when fetching data from a CDN, an endpoint has also the choice of which endpoint instance, [I-D.brunstrom-taps-impl] calls these instances "Derived Endpoint", to contact as DNS can return multiple alternative addresses.

If endpoints want to take advantage of multiple available paths, there is another bunch of, partially interdependent, choices:

- Which path(s) between the endpoints could be used?

- Which path(s) between the endpoints should be used?

- Should the paths be used in an active/active way or only as active/fallback?

- Which protocols or sets of protocols should be used?

- Which role will other on-path elements, e.g. middle-boxes, take in servicing this flow?

Implementing an heuristic or strategy for choosing from this overwhelming set of transport options by each application puts a huge burden on the application developer. Thus, the decisions regarding all transport options mentioned so far should be supported and, if requested by the application, automated within a the transport layer. In order to build such automatization, we need to be able to compare the product of all transport options (destinations, paths, transport protocols and protocol options) available to choose the most appropriate.

As the protocols to be used are not known a priori and can differ depending on other transport options, this reasoning has to be independent of a specific protocol or implementation and allow to compare them even if they operate on different communication unit granularities.

1.1. Communication Units vs. Layering

When reasoning about network systems, layering traditionally has been the main guidance on where functionality is placed. Looking at modern systems, the classical concept of layers and their mapping to protocols becomes blurry. Protocols can operate on different granularities of communication units, i.e., the semantic units such as messages that the protocols distinguish. These communication units often do not match the PDUs used by the protocols, e.g., TCP segments do not necessarily align with messages at the application layer.

In this document, we do not want to take a protocol-centric perspective, but we focus on mechanisms a multi-access system is composed of and the communication units they operate on. This has several advantages:

- We can much easier abstract from the protocols used and look at the composition itself.

- By disseminate on which kind of communication unit these mechanisms can operate, we can reason about the overall design space.

- If seeing the same mechanism multiple times within the same system composition, we can reason about possibly conflicting optimizations.

Overall, this perspective allows us to compare mechanism like distributing requests of an application among different paths, MPTCP and using bandwidth aggregation proxies (as discussed within the IETF in the BANANA working group) despite their different nature and layer of implementation.

2. Abstract Hierarchy of Communication Units

These communication units definitions are primarily used for reasoning about automatic stack composition. Therefore, depending on the protocol stack instance, a communication unit can span multiple protocol instances.

Some of these hierarchy levels correspond to objects in [I-D.ietf-taps-minset], but in case of Association and Association Set, we have to split categories as they may indeed be separate on the transport. Note the naming confusion concerning the term "flow" deriving from different perspective.

We also annotate the corresponding terminology used in [I-D.ietf-taps-arch] if it differs from the one used in this document.

2.1. Message

An Message is a piece of data that has a meaning for the application. It is the smallest communication unit that we consider.

[I-D.ietf-taps-minset] correspondent: Message

Examples:

- A HTTP-Request/Response-Header/Body for HTTP/2

- An XML message in XMPP

2.2. Stream

A Stream is an ordered sequence of related Messages that should be treated the same by the transport system.

[I-D.ietf-taps-minset] correspondent: Flow

Examples:

- A Stream in QUIC or SCTP

- A TCP connection used as transport for XMPP

2.3. Association / Connection Group

An Association multiplexes a set of Messages or Streams within the same Flow with common source and destination. Therefore these communication units become indistinguishable for the network. Association and flow describe the same concept, the former from the perspective of the application, the latter from the perspective of the network.

[I-D.ietf-taps-minset] correspondent: Flow-Group

[I-D.ietf-taps-arch] correspondent: Connection Group

Examples:

- A TCP connection carrying HTTP/2 frames

- A set of IP packets that carry TCP or UDP segments and share the same 5-tuple of src-address, dst-address, protocol, src-port, dest-port.

2.4. Association Set / Flow-Group

An Association Set or Flow Set is a set of Associations or Flows that belong together from an application point of view.

[I-D.ietf-taps-minset] correspondent: Flow-Group

Examples:

- Two flows, one carrying RTP payloads and one used for RTCP control messages.

3. Mechanisms Used in Multi-Path Systems

Transport protocols on the Internet provide a large variety of functionality. While the functionality of simple protocols like UDP is easy to describe (multiplexing streams of messages), describing the functionality of complex protocols such as QUIC, MPTCP or SCTP is manyfold as these protocols provide a set of commonly used functionality. Also, the same functionality can be provided at many places throughout the whole stack. In the following, we explore the set of functionality that can be provided by transport protocols.

3.1. Destination Selection

Destination Selection refers to selecting one of multiple different destinations. This mechanism is applicable to any kind of communication unit and can occur on all layers.

Typical cases for destination selection include:

- Choosing one address of a multi-homed server for an upcoming communication.
- Choosing a server among a list of servers retuned by DNS, e.g for servers that host the same content as part of a CDN.
- Choosing a backend server within a load balancer.

In practice, destination address selection is often tied to name resolution. As name resolution relies on both local decisions on the endpoint as well as decisions within the DNS infrastructure, this mechanism spreads across different administrative domains which each independently contribute to the overall selection result.

3.2. Path Selection

Path Selection refers to choosing which of the available paths to use. and can occur on the network layer and any layer below.

- Within an end-host, path selection is usually realized by choosing the source IP address and thus choosing one of the local network interfaces for the communication to the remote endpoint.
- Within a path layer traffic system like an MPTCP-Proxy or a BANANA-Box, path selection is usually realized by choosing the outer source and destination address.
- In case of an ECMP router, path selection is usually done based on a 3- or 5-tupel and just determines the interface to the next hop.
- Within MPTCP, each TCP segment has to be assigned to one or more subflows for transmission to the receiver.

While path selection involves a choice of access network it does not need knowledge of or changes to the routing choices within the core network.

When doing path selection on small communication units like TCP segments, it is not uncommon to split path selection into two subproblems: _Candidate Path Selection_ determines feasible and

preferred choices, e.g., in case of MPTCP by establishing subflows. Afterwards, _Per-Chunk Path Selection_ selects among these alternatives for each chunk. Thus, the first can be more expensive while the latter should be easy to execute.

TODO: Discuss difference between Multiple Provisioning Domains [RFC7556] or multiple access networks within the same provisioning domain - especially when it comes to integrating 3GPP mechanisms like [RFC5555] or [RFC7864].

3.3. Chunking

Chunking refers to splitting an message, a stream or a set of associations into one or more parts. Typically, chunking splits only large messages or streams into multiple ones while keeping smaller entities untouched. Associations or Flows are typically not split, but sets of Associations or Flows might be partitioned. Once split into chunks, each chunk can be transferred individually over different transfer options.

Chunking can and does occur at different layers within a system:

- A Web site consists of multiple objects or files. Thus, the files can be seen as the natural chunks of a Web site.

- TCP takes as input a byte stream and chunks it into segments. TCP chunking (segmentation) occurs at arbitrary byte ranges, thus it will most likely not align with boundaries of Messages that were multiplexed within an application layer Association on top of a TCP connection.

In practice, chunking is often constrained in order to maintain certain properties that are desirable for the overall system. Examples such restrictions include the following:

- Segmentation in TCP restrict the chunk size, i.e. TCP segment size, to the IP MTU or IP Path MTU to avoid fragmentation at the IP layer.

- Equal cost multipath routing does not distribute packets, but Flows to avoid reordering.

3.4. Scheduling

Scheduling refers to distributing chunks or sets of chunks across multiple pre-chosen path. Thus, depending on the objectives, it can make sense to see scheduling as is nothing else than per-chunk path selection as defined above. In other cases, e.g. when trying to

balance traffic, it makes sense to look at scheduling as a concept itself that uses chunking and per-chunk path selection as sub-mechanisms.

Examples of scheduling strategies include:

- Schedule all chunks on one path as long as this path is available, otherwise fall pack to another.
- Distribute chunks based on path capacity.

4. Cost of Transport Option Selection

Transport option selection mechanisms are often intertwined. Which mechanism is used by which layer or which network component depends on the transfer objectives as well as the state of the network, e.g., availability, path throughput, path RTT, server load.

The cost and complexity of transport option selection depends on the network state used and the number of transfer options. If the transfer option selection only uses local state e.g., link availability, and the mechanism is predetermined and/or uses simple mechanisms, e.g., a simple hash function, the cost can even be negligible. An example where transfer option selection is cheap is ECMP within a router. In other cases, the cost can be non-trivial, e.g. when the selection involves queries to remote entities or even active network performance measurements. Such examples include DNS or DHT lookups, as used by some file sharing protocols, or network measurements like RTT and bandwidth estimations used by many video streaming applications. Indeed, costs may be prohibitive, e.g when requiring multiple DNS lookups for every 1 second chunk of a 20 minute video.

5. Involvement of On-Path Elements

It may become necessary to take path layer components (middle-boxes) into account that interfere with the transport layer.

While the classical "End-To-End Arguments in System Design" [End-To-End] advocates for a dumb network and placing functionality as close to the edge and up in the stack as possible, there are always tussles of moving functionality up or down the stack. This document does not argue against pushing some multi-path functionality down the stack, but advocates to maintain the control of the overall system composition at the end host. Functionality provided by a path can indeed be a reason to choose this path for a given communication unit.

Some flow off-loading mechanisms that come in gestalt of of logical interfaces, e.g., [RFC7847]. These interfaces treat some association sets differently, which can be considered on-path functionality.

6. Overview of Mechanisms provided by selected IETF Protocols

Protocol	Congestion Control	Ordering	Reliability	Integrity P.	Confidentiality P.	Authenticity P.	Chunking	Multiplexing
HTTP	r	r	r				bytes	requests
HTTPS	r	r	r	r	r	r	bytes	requests
XMPP	r	r	r	(r)	(r)	(r)		messages
SIP			+	(r)	(r)	(r)		messages
DTLS				+	+	+		services,name
TLS		r	r	+	+	+		services,name
RTP	+(prf)	+(prf)					messages(prf)	messages
SRTP	+(prf)	+(prf)		+	+	r(sig)	messages(prf)	messages
QUIC	+	+	+	+	+	+(tls)	bytes	connection-id,+(tls)
UDP								ports
DCCP	+							ports

TCP	+	+	+				bytes	ports
MPTCP	+	+	+				bytes	ports
SCTP	+	+	+				bytes	ports,streams
IPsec (ESP)				+	+	r(ike)		spi ,next-header
IPsec (AH)				+		r(ike)		spi ,next-header
IP		(+(fr))					(fragments)	address ,next-header
NEMO/IFOM					r	r	assoc.	

Legend:

r: Protocol requires transport service.

+: Protocol provides transport service.

prf: Realized by content specific profiles.

tls: Uses TLSv1.3 as sub-protocol; imports authenticity protection and multiplexing from TLS.

ike: Realized externally by external protocol IKE/IKEv2.

sig: Realized externally by external signaling protocol (e.g., SIP, XMPP, WebRTC).

fr: :Only when fragmentation is used and only to re-assemble IP PUDs

7. Acknowledgements

This work has been supported by Leibniz Prize project funds of DFG - German Research Foundation: Gottfried Wilhelm Leibniz-Preis 2011 (FKZ FE 570/4-1).

8. Informative References

[End-To-End]
Saltzer, J., Reed, D., and D. Clark, "End-to-end arguments in system design", ACM Transactions on Computer Systems Vol. 2, pp. 277-288, DOI 10.1145/357401.357402, November 1984.

[I-D.brunstrom-taps-impl]
Brunstrom, A., Pauly, T., Enghardt, T., Grinnemo, K., Jones, T., Tiesel, P., Perkins, C., and M. Welzl, "Implementing Interfaces to Transport Services", draft-brunstrom-taps-impl-00 (work in progress), March 2018.

[I-D.ietf-taps-arch]
Pauly, T., Trammell, B., Brunstrom, A., Fairhurst, G., Perkins, C., Tiesel, P., and C. Wood, "An Architecture for Transport Services", draft-ietf-taps-arch-02 (work in progress), October 2018.

[I-D.ietf-taps-interface]
Trammell, B., Welzl, M., Enghardt, T., Fairhurst, G., Kuehlewind, M., Perkins, C., Tiesel, P., and C. Wood, "An Abstract Application Layer Interface to Transport Services", draft-ietf-taps-interface-02 (work in progress), October 2018.

[I-D.ietf-taps-minset]
Welzl, M. and S. Gjessing, "A Minimal Set of Transport Services for End Systems", draft-ietf-taps-minset-11 (work in progress), September 2018.

[RFC5555] Soliman, H., Ed., "Mobile IPv6 Support for Dual Stack Hosts and Routers", RFC 5555, DOI 10.17487/RFC5555, June 2009, <https://www.rfc-editor.org/info/rfc5555>.

[RFC7556] Anipko, D., Ed., "Multiple Provisioning Domain Architecture", RFC 7556, DOI 10.17487/RFC7556, June 2015, <https://www.rfc-editor.org/info/rfc7556>.

[RFC7847] Melia, T., Ed. and S. Gundavelli, Ed., "Logical-Interface Support for IP Hosts with Multi-Access Support", RFC 7847, DOI 10.17487/RFC7847, May 2016, <https://www.rfc-editor.org/info/rfc7847>.

[RFC7864] Bernardos, CJ., Ed., "Proxy Mobile IPv6 Extensions to Support Flow Mobility", RFC 7864, DOI 10.17487/RFC7864, May 2016, <https://www.rfc-editor.org/info/rfc7864>.

Appendix A. Changes

A.1. Since -00

- Replaced granularity "Object" with "Message" to align with other TAPS documents.

- Removed empty section on protocol instance selection - this topic will go into a separate document later.

- Minor clarifications.

- Removed definition of normative terms not needed for this document

- Added acknowledgments and updated authors' affiliation (compliance).

A.2. Since -01

- Updated drafts references

- Added Overview of Mechanisms provided by selected IETF Protocols

- Minor clarifications

- Removed superfluous IANA and Security Considerations section

A.3. Since -02

- Prevent expirary (minor formatting fixes)

Authors' Addresses

Philipp S. Tiesel
TU Berlin
Marchstr. 23
Berlin
Germany

Email: philipp@inet.tu-berlin.de

Theresa Enghardt
TU Berlin
Marchstr. 23
Berlin
Germany

Email: theresa@inet.tu-berlin.de