An Approach for Transferring an End-to-End Transport Service into a Functional Building Block Structure

Thomas Dreibholz, Martin Becke, Erwin P. Rathgeb
University of Duisburg-Essen, Institute for Experimental Mathematics
Ellernstrasse 29, 45326 Essen, Germany
{dreibh,martin.becke,rathgeb}@iem.uni-due.de

Christian Henke
Technical University Berlin
Straße des 17. Juni 135, 10623 Berlin, Germany
c.henke@tu-berlin.de

Abstract—**Current network stacks based on the classic OSI layered reference model are restricted and inflexible, which makes the addition and deployment of new features difficult. Operating systems offer an interface for the Transport Layer functionalities to applications. The usage of this interface is very widespread, making it useful to maintain this interface. However, the layered structure should be replaced.**

In this paper, we propose our approach for breaking up the functionalities of the OSI Transport Layer into the concept so called functional building blocks. Each functional building block provides a particular service, which – in interaction with the other functional building blocks – is able to provide a service similar to the classic OSI Transport Layer. The concept allows for easy removal, replacement or addition of existing and new functional building blocks to adapt the service to state-of-the-art and future requirements, particularly including multi-path transport and QoS.

Keywords: Transport, Multi-Path, Functional Building Blocks, Cross-Layer Optimization, Future Internet

I. INTRODUCTION

The transport of application data over a communications network is a rather complex task. Communication systems have to care for various features like error detection, reliable transport, congestion control, QoS and connection maintenance. A long time ago, the ISO has published their well-known Open Systems Interconnection (OSI) reference model [1] which splits these features into seven independent, hierarchical layers. These layers are instantiated by independent protocols. However, the independence of these layers is increasingly becoming a problem – and the introduction of middle layers (like MPLS at layer 2.5, IPSEC at layer 3.5, TLS at layer 4.5) and approaches like cross-layer optimization increasingly blur the borders between the layers. The concept of *Functional Building Blocks* [2]–[5] gets rid of the hierarchical layers and defines independent functionalities as blocks instead. Basic functional building blocks are so fine-granular that they only conduct one specific functionality which cannot be subdivided. The basic building blocks can be composed to composite building blocks which conduct functionalities of any granularity, up to the functionality of a whole network stack. These blocks can be loosely coupled – which provides reusability and information exchange between functionalities on formerly different layers. Based on the loose coupling, new functional blocks can be easily removed, changed or added. The system adapts the data transmission by composing functional blocks based on the application specific requirements and the network state. In this paper, we propose our approach for breaking up the functionalities of the OSI Transport Layer into a functional building blocks structure. Particularly, our approach also takes care of new transport requirements like multi-path transfer [6] and QoS.

II. MULTI-HOMING AND MULTI-PATH TRANSPORT

The term *Multi-Homing* denotes the property of a node to manage multiple addresses, e.g. when possessing multiple network interfaces connected to different Internet Service Providers (ISP) or simply when using an IPv4 and IPv6 address on the same interface. Multi-homing is a common approach to improve service resilience by having multiple network paths to a peer (denoted by different peer addresses) as protection against link failures. While multi-homing only denotes the possibility of having multiple paths, of which e.g. one path may be selected for the user data transport while the other paths remain for backup only, the term *Multi-Path Transport* denotes the *simultaneous* usage of all working paths for user data transfer.

The functionality of multi-homing may be applied on different OSI layers: IETF standards such as SHIM6 [7] and HIP [8] support multi-homed access to a network on the Network Layer of the OSI reference model. They build up as a shim to separate locator and identifier and support access to a network from more than one interface. The benefit of a Network Layer approach to provide transparent multi-homing functionality to the Transport Layer is also the drawback of this approach: for Best Effort congestion control in the Transport Layer, it is not sufficient to just handle each network path separately (see [9] for details). That is, multi-path transport is not possible without modification to central Transport Layer responsibilities.

Other approaches apply multi-homing on the Transport Layer: SCTP [10] is a general-purpose, unicast, connection-oriented, transport protocol, which provides the reliable transport of user messages. CMT-SCTP [6] is a multi-path extension for SCTP. MPTCP (Multipath TCP, see [11]) is an experimental multi-path transport approach for TCP. In contrast to CMT-SCTP, MPTCP extends the originally single-path protocol TCP with multi-homing and multi-path transport capabilities. This requires a significant number of changes to the original protocol functionalities. But the current, layered architecture based makes these changes difficult.

III. A MULTI-PATH-CAPABLE CONNECTION SERVICE

In the following, we present our approach for transferring the tasks of the OSI Transport Layer into a functional building blocks structure. This requires to define some terminology first.

A. Definitions

A *Channel* denotes a point-to-point communications association between two endpoints, identified by their addresses. It may be an

arbitrary virtual network path [12] or e.g. an Ethernet or WLAN connection. The channel allows for bi-directional communications between the two endpoints; it *may* (but is not required to) provide additional features like QoS (e.g. resource reservations).

A *Sub-Flow* denotes the abstraction of a data flow on a channel. A *Flow* denotes the set of all sub-flows between the same two endpoints, belonging to a connection. The sub-flows of a flow may use different channels.

A *Connection* denotes a point-to-point association between two endpoints which uses a flow to transport application data. Since applications like web site access usually transfer multiple independent data streams (e.g. the web page itself, images, multimedia objects), it is useful to multiplex one or more *Streams* over the same flow instead of using separate connections. This so called *Multi-Streaming* feature is e.g. provided by SCTP [10].

B. Functional Building Blocks

Figure 1 presents our functional building block approach for a connection service based on an end-to-end service. A multi-path connection is logically structured in composite functional blocks which we call containers. The connection functionality is offered by a Connection Control Container. The Flow Control container maintains the connection's flow and is associated to N Sub-Flow container instances. The connection consists of M streams, which are offered by the Stream Control Container and multiplexed over the connection's flow. Note, that the functional building blocks – unlike OSI layers – are intended to interact with each other when necessary, even when they are not placed in the same container. For readability reasons, these relationships have been neglected in the figure. Each container provides a pre-defined interface for the inter-block communication. The focus of this paper is to introduce the required building blocks themselves. Their composition is out of scope.

From the perspective of an application, the service provided by the Connection Control Container is similar to the OSI Transport Layer interface. That is, existing applications using this well-known and very widespread interface may be adapted to our new approach with small effort. Using the functional building blocks, which we will describe in the following, the features of current state-of-the art Transport Layer protocols (UDP, TCP, SCTP [10], DCCP [13]) are covered. However, our concept allows for easy removal, change or addition of new blocks to adapt the service to future requirements.

The "Channel Management" block is responsible for the transmission of data over an underlying channel. The channel itself, which may e.g. be a virtual network path [12] or a physical path, takes care of routing the data.

The "Error Detection and Correction" block takes care of detecting sub-flow bit errors (e.g. by using a checksum) and possibly correcting them (e.g. by using Forward Error Correction mechanisms). Furthermore, it interacts with the QoS and Acknowledgement blocks to trigger the retransmission of damaged data while not assuming loss due to congestion. Such a functionality is e.g. realized by the "Packet Drop Reporting" [14] extension of SCTP. Finally, this block also takes care of probing the usability of the associated channel (e.g. by keep-alive messages as realized by SCTP [10]).

The "Sub-Flow QoS Control" block is responsible for the sub-flow's QoS. That is, it ensures that reservations of resources are processed, takes care of resource usage metering and monitoring the QoS of the data transport and handles congestion control. An important class of QoS is Best Effort, i.e. the amount of data output

has to be adapted to current network conditions. Interaction with the "Error Detection and Correction" and "Acknowledgement" blocks as well as possible support of the underlying channel (e.g. Explicit Congestion Notification [15]) e.g. allows for TCP-like congestion control [9] using a sliding window, rate-based approaches or even mechanisms like ABR (Available Bit-Rate) provided by ATM networks [16]. Note, that each sub-flow may use its own kind of QoS control, which allows e.g. for using different congestion control variants (similar to DCCP [13]) on different paths.

The "Flow QoS Control" is responsible for the QoS of the whole flow, i.e. the set of all sub-flows. While the "Sub-Flow QoS Control" can only take care of its associated sub-flow itself, the "Flow QoS Control" takes care of the QoS interaction among the sub-flows. This interaction is important when considering the fairness of a multi-path data transport, which is e.g. required by CMT/RP-SCTP (CMT-SCTP with Resource Pooling, see [9]) and MPTCP [11].

The reliability of the data transport is controlled by the "Acknowledgement" block. It realizes an acknowledgement mechanism to detect packet losses and may trigger retransmissions in interaction with the "Buffer Mgt" block. In particular, fine-granular retransmission policies (e.g. limiting the number of retransmission trials by number or time) – analogously to the "Partial Reliability" [17] extension of SCTP – may be realized. Furthermore, it may trigger congestion handling in interaction with "Flow QoS Control" and corresponding "Sub-Flow QoS Control".

Handling of transmission, retransmission and reception buffers is the responsibility of the "Buffer Mgt" block. Efficient buffer management is crucial for the deployment of the data transport, since a highly utilized system probably has to maintain a large number of simultaneous connections.

The identification of endpoints and the mapping to channels is in the responsibility of the "Path Mgt" block. In particular, it also takes care of mobility, i.e. adding or removing sub-flows (e.g. similar to the "Dynamic Address Reconfiguration" [18] extension of SCTP). Connection establishment and teardown is similar to adding the first or removing the last sub-flow. Interaction with the "Security" block is necessary to ensure the authenticity of the peer endpoint and to avoid connection hijacking attacks.

Integrity, authenticity and confidentiality of the data transport are in the responsibility of the "Security" block. It can e.g. realize appropriate encryption and signature mechanisms for user as well as control data (e.g. similar to D-TLS [19], Secure-SCTP [20] or the "Chunk Authentication" extension of SCTP, see [21]).

The "Scheduler" block multiplexes/demultiplexes user data fragments of the different streams to/from the sub-flows. In particular, it can interact with the "Flow QoS Control" and the "Sub-Flow QoS Control" blocks in order to ensure certain QoS properties for the data (e.g. low delay or low jitter). It also takes care of flow control (e.g. ensuring that the receiving endpoint will not be overloaded), in interaction with the "Buffer Mgt" block.

The segmentation and de-segmentation of large user messages is performed by the "Segmentation" block, in interaction with the "Scheduler" and "Flow QoS Control" (which have information of an appropriate MTU size [22]).

The "Ordered Delivery" block ensures that the message order of a stream is preserved, if this is required by the application (e.g. by using sequence numbers similar to SCTP [10] or DCCP [13]).
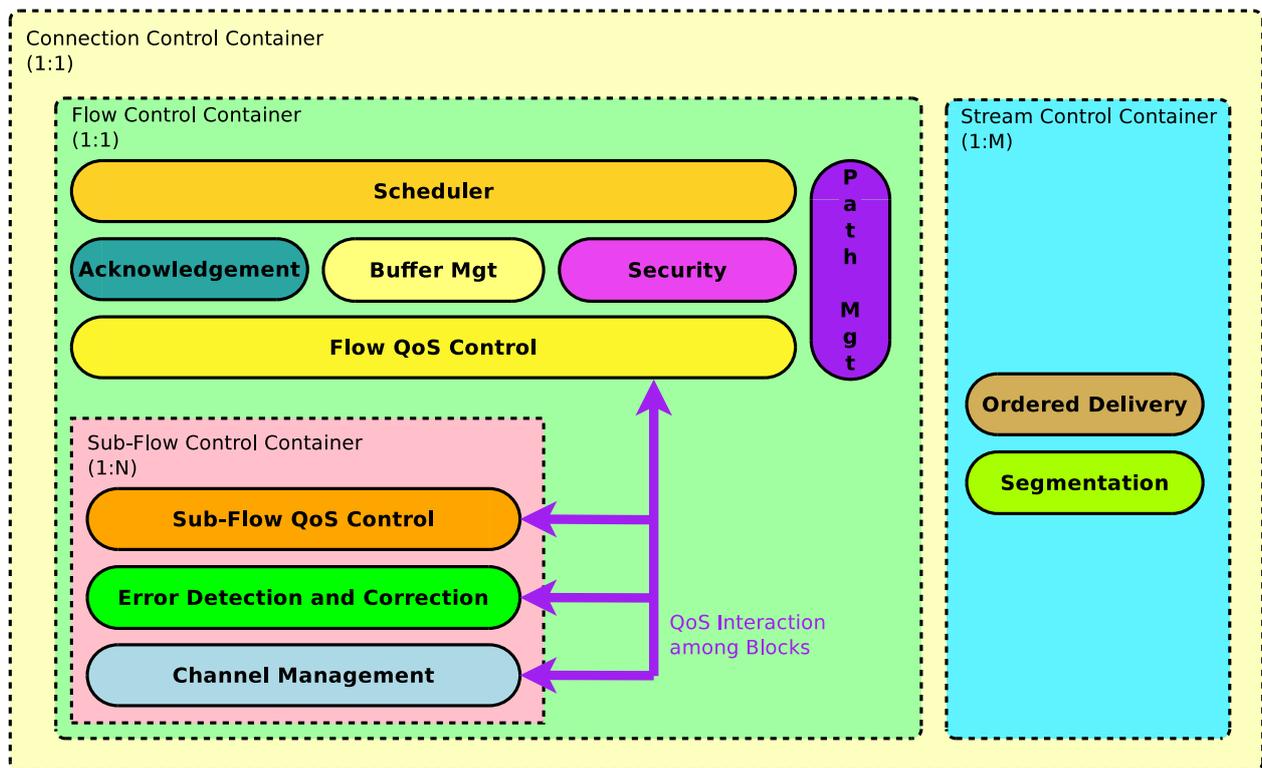
Figure 1.   Functional Building Block Approach for a Connection Service

IV. Conclusions

The concept of functional building blocks is a new approach to overcome the limitations of the 7-layer OSI reference model. It splits the rather complicated task of the user data transport over a network into functionalities, which can interact with each other and be modified, removed or extended easily. Using this concept, new application requirements can be realized with small effort in comparison to a traditionally layered approach.

In this paper, we have proposed our approach of transferring the tasks of the OSI Transport Layer into a functional building blocks structure. Our approach covers the features of the current state-of-the-art Transport Layer protocols UDP, TCP, SCTP and DCCP, including important transport features like multi-streamed, multi-homed and multi-path transport as well as QoS.

As part of future work, we are going to integrate our concept into a Future Internet experimental facility for a proof of concept.

References

[1] International Telecommunication Union, "Open Systems Interconnection - Base Reference Model," ITU-T, Recommendation X.200, Aug. 1994.

[2] R. Braden, T. Faber, and M. Handley, "From protocol stack to protocol heap: role-based architecture," *SIGCOMM Comput. Commun. Rev.*, vol. 33, no. 1, pp. 17–22, 2003.

[3] *Basic Abstractions for an Autonomic Network Architecture*, 2007.

[4] R. Dutta, G. N. Rouskas, I. Baldine, A. Bragg, and D. Stevenson, "The silo architecture for services integration, control, and optimization for the future internet," in *IEEE ICC*, 2007, pp. 24–27.

[5] S. Ganapathy and T. Wolf, "Design of a network service architecture," in *ICCCN*, 2007, pp. 754–759.

[6] J. R. Iyengar, P. D. Amer, and R. Stewart, "Concurrent Multipath Transfer Using SCTP Multihoming Over Independent End-to-End Paths," *IEEE/ACM Transactions on Networking*, vol. 14, no. 5, pp. 951–964, Oct. 2006.

[7] C. de Launois and M. Bagnulo, "The Paths Towards IPv6 Multihoming," *IEEE Communications Surveys and Tutorials*, vol. 8, no. 1-4, pp. 38–51, 2006.

[8] P. Nikander, J. Ylitalo, and J. Wall, "Integrating Security, Mobility, and Multi-homing in a HIP way," in *Proceedings of Network and Distributed Systems Security Symposium*, February 2003.

[9] T. Dreibholz, M. Becke, J. Pulinthanath, and E. P. Rathgeb, "Applying TCP-Friendly Congestion Control to Concurrent Multipath Transfer," in *Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications (AINA)*, Perth/Australia, Apr. 2010.

[10] R. Stewart, "Stream Control Transmission Protocol," IETF, Standards Track RFC 4960, Sept. 2007.

[11] D. Wischik, M. Handley, and C. Raiciu, "Control of Multipath TCP and Optimization of Multipath Routing in the Internet," in *NET-COOP '09: Proceedings of the 3rd Euro-NF Conference on Network Control and Optimization*.   Berlin, Heidelberg: Springer-Verlag, 2009, pp. 204–218.

[12] T. Zinner, K. Tutschku, A. Nakao, and P. Tran-Gia, "Performance Evaluation of Packet Re-ordering on Concurrent Multipath Transmissions for Transport Virtualization," *JCNDS Special Issue on: Network Virtualization – Concepts and Performance Aspects*, May 2010.

[13] E. Kohler, M. Handley, and S. Floyd, "Designing DCCP: Congestion Control without Reliability," *SIGCOMM Computer Communication Review*, vol. 36, no. 4, pp. 27–38, 2006.

[14] R. Stewart, P. Lei, and M. Tüxen, "Stream Control Transmission Protocol (SCTP) Packet Drop Reporting," IETF, Individual Submission, Internet-Draft Version 09, Dec. 2009, draft-stewart-sctp-pktdrprep-09.txt, work in progress.

[15] K. Ramakrishnan, S. Floyd, and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP," IETF, Standards Track RFC 3168, Sept. 2001.

[16] E. P. Rathgeb, *ATM – Infrastruktur für Hochleistungskommunikation*.   Heidelberg/Germany: Springer-Verlag, 1997, ISBN 3-540-60370-0.

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

[18] R. Stewart, Q. Xie, M. Tüxen, S. Maruyama, and M. Kozuka, "Stream Control Transmission Protocol (SCTP) Dynamic Address Reconfiguration," IETF, Standards Track RFC 5061, Sept. 2007.

[19] E. Rescorla and N. Modadugu, "Datagram Transport Layer Security," IETF, Standards Track RFC 4347, Apr. 2006.

[20] C. Hohendorf, E. P. Rathgeb, E. Unurkhaan, and M. Tüxen, "Secure End-to-End Transport Over SCTP," in *Proceedings of the International Conference on Emerging Trends in Information and Communication Security (ETRICS)*, ser. Lecture Notes in Computer Science, vol. 3995.   Springer, 2006, pp. 381–395.

[21] M. Tüxen, R. Stewart, P. Lei, and E. Rescorla, "Authenticated Chunks for the Stream Control Transmission Protocol (SCTP)," IETF, Standards Track RFC 4895, Aug. 2007.

[22] M. Welzl, "PMTU-Options: Path MTU Discovery Using Options," in *60th IETF Meeting (PMTUD WG)*, Aug. 2004.