COINRG
Internet-Draft
Intended status: Informational
Expires: 28 April 2022

I. Kunze
K. Wehrle
RWTH Aachen University
D. Trossen
Huawei

25 October 2021

Transport Protocol Issues of In-Network Computing Systems

draft-kunze-coinrg-transport-issues-05

Abstract

Today's transport protocols offer a variety of functionality based on the notion that the network is to be treated as an unreliable communication medium. Some, like TCP, establish a reliable connection on top of the unreliable network while others, like UDP, simply transmit datagrams without a connection and without guarantees into the network. These fundamental differences in functionality have a significant impact on how COIN approaches can be designed and implemented. Furthermore, traditional transport protocols are not designed for the multi-party communication principles that underlie many COIN approaches. This document raises several questions regarding the use of transport protocols in connection with COIN.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

A fundamental consideration for the Internet's design is that functions can be implemented correctly and completely only with the knowledge of the applications, as formulated in [E2E]. This choice is reflected in the end-to-end (E2E) principle [RFC1958],[RFC2775] in that end-hosts perform most, if not all, relevant computations. The network only performs transparent, reasonable operations such as delivering the packets without modifying them with transport protocols designed to facilitate the direct communication between those end-hosts.

[E2E], however, does consider that "sometimes an incomplete version of the function provided by the communication system may be useful as a performance enhancement". We link this consideration to the field of computing in the network (COIN), which encourages explicit computations in the network, introducing an intertwined complexity as the computations on the end-hosts depend on the functionality deployed in the network.

Such thinking, to some extent, challenges traditional ``end-to-end'' transport protocols as they are not designed to address in-network computation entities or to include more than two devices into a communication, even for inherent functionalities provided by the transport protocol. Some of the resulting problems when considering in-network computation in the context of an overall E2E problem are already presented in [I-D.draft-kutscher-coinrg-dir-02].

This draft focusses on the potential opportunities and research questions for the design of transport protocols that may assume the availability of in-network computing capabilities, including the possible collaboration with other IRTF and IETF groups, such as PAN RG, the IETF transport area in general, or the LOOPS BOF, for finding suitable solutions. In particular, the draft first describes how different aspects of transport protocols might be affected by in-network computing functionality before it is analyzed how existing transport protocols map to the identified questions and challenges.

2. Terminology

COIN element: Device on which COIN functionality can be deployed

3. Technology Areas

3.1. Addressing

The traditional addressing concept of the Internet is that end-hosts directly address each other with all computational intelligence residing at the network edges. With COIN, computations move into the network and need to be integrated into the established infrastructure. In systems where the whole network is under the control of the network operator this integration can be implemented by explicitly adjusting the communication schemes based on the COIN functionality. Considering larger scales, this approach of manually adjusting traffic patterns and applications to correctly incorporate changes made by the network is not feasible.

What is needed are ways to specify which kind of functionality should be applied to the transmitted data on the path inside the network and maybe even where or by whom the execution should take place. From an identification perspective, addressing may not only need to specify the set of functionality that is being desired but also enable to provide affinity to a member of the set of computational nodes that provide said functionality.

For instance, orchestration functionality may be implemented using an indirection mechanism which routes a packet along a pre-defined or dynamically chosen path which then realizes the desired functionality. One possibility is to directly route on service or functionality identifiers instead of sending individual packets between locator-addressed network elements [I-D.draft-irtf-coinrg-use-cases]. While this aligns the routing more clearly with the communication between computational elements, selecting the 'right' computational endpoint (out of possibly several ones) becomes critical to the proper functioning of the overall service.

3.1.1. Research questions and challenges

1. How should end-hosts address the COIN functionality?

2. How can the treatment of the transmitted data, i.e., which COIN functionality to execute, be represented in the addressing of the request?

3. How can end-hosts direct computational requests to different computational endpoints of the same service in different network locations, i.e., decide where the COIN functionality is executed?

4. How to decide (and encode the decision) which computational endpoint to choose (from possibly several ones existing in the network)?

5. How can devices which do not implement COIN functionality be integrated into the systems without breaking the COIN or legacy functionality?

3.1.2. Related concepts and efforts

* Segment and Source Routing (see [SPRING-WG])

Source Routing allows a sender to (partially) define the route of a packet through the network. This mechanism can be leveraged to steer the traffic along COIN nodes and thus trigger desired COIN functionality. The SPRING WG is scoped to define procedures around Segment Routing [SR], a modern variant of Source Routing for IPv6 and MPLS.

* (Service/Network) Function Chaining/Composition (see [SFC-WG])

Service Function Chaining (SFC) describes a process to first define an ordered list of service functions (e.g., firewalls) and then steer traffic through these functions [SFC-PS]. The SFC WG is tasked with defining suitable orchestration techniques for SFC. The existing SFC architecture [SFC-Arch] and the Network Service Header [SFC-NSH] already provide fundamental mechanisms. Interpreting COIN functionality as service functions could make SFC applicable to COIN at Layer 2 and Layer 3, but also at name-based, e.g., HTTP level [RFC8677]

* Internet services over ICN (see [ICNIP])

Work in the ICN RG [ICNRG] has generally studied the addressing of information rather than endpoints, opening up the possibility for providing information from different sources, including in-network elements, such as for caching purposes. The work in [ICNIP] utilizes the ICN capabilities to address services directly as a named entity, including IP endpoints, in order to support concepts like virtualization of service endpoints and provisioning within edge and in-network locations. The solution in [ICNIP] proposes the use of a Layer 2 path-based forwarding with service identifiers used to address the specific service endpoint.

* Flexible Addressing (see [I-D.draft-jia-intarea-scenarios-problems-addressing])

Although the work in the INT Area of the IETF does not postulate a specific solution, it outlines a number of communication scenarios and challenges, some of which aligns with those outlined above. The companion draft [I-D.draft-jia-intarea-internet-addressing-gap-analysis] provides a

deeper gap analysis of existing solutions (the above mentioned solutions are presented here, too), identifying a number of issues that arise from the specific point solutions realized by those solutions. The authors argue for both flexibility and extensibility of addressing; key aspects that any solution addressing the research questions outlined above would benefit from.

* Semantic Routing (see [I-D.draft-king-irtf-semantic-routing-survey])

The survey at [I-D.draft-king-irtf-semantic-routing-survey] provides an overview of efforts on addressing and routing that incorporate a semantic beyond the one defined by IPv6, covering both existing IETF solutions as well as ongoing research, defined as 'semantic routing' in the draft. The companion draft at [I-D.draft-king-irtf-challenges-in-routing] outlines a number of challenges that exist for such extension of the addressing semantic, some of which align with the issues identified in this document. More importantly, the draft discusses the possible deployment of semantic routing solutions, e.g., as an overlay or limited to a single domain (following the Limited Domain concept of [RFC8799]). Some of the challenges identified in [I-D.draft-king-irtf-challenges-in-routing] apply to a COIN environment, while not being limited to it. For instance, the intended scope of any enhanced addressing (e.g., identifying COIN elements on-path in a scenario) or the description of path characteristics that COIN traffic would need to adhere to.

3.2. Flow granularity

Core networking hardware pipelines such as backbone switches are built to process incoming packets on a per-packet basis, keeping little to no state between them. This is appropriate for the general task of forwarding packets, but might not be sufficient for COIN as information that is needed for the computations can be spread across several packets. In a TCP stream, for example, data is dynamically distributed across different segments which means that the data needed for application-level computations might also be split up. In contrast to that the content of UDP datagrams is defined by the application itself which is why the datagrams could either be self-contained or information can be cleverly distributed onto different datagrams. Summarizing, different transport protocols induce different meanings to the packets that they send out which needs to be accounted for in COIN elements as they have to know how the received data is to be interpreted. There are at least three options for this.

1. Every packet is treated individually. This maps to the capabilities of existing networking equipment.

2. Every packet is treated as part of a message. In this setting, the packet alone does not have enough information for the computations. Instead, it is important to know the content of the surrounding packets which together form the overall message.

3. Every packet is treated as part of a byte stream. Here, all previous packets and potentially even all subsequent packets need to be taken into consideration for the computations as the current packet could, e.g., be the first of a group of packets, a packet in the middle, or the final packet.

Along those options above, the question arises how shorter-term 'messages' (or transactions) of the computation should be handled compared to the often longer-term management of the network resources needed to transmit the packets across one or more such messages. For instance, error control may be best applied to the individual messages between computational endpoints, while congestion control may be applied across several messages at the level of the relation between the network elements hosting the computational endpoints. In this view, the notion of a 'flow' may separate message or transaction handling from the resource management aspect, where a flow may be divided into sub-flows (said messages or transactions) with error control being applied to those sub-flows but resource management being applied to the overall flow. Such choice of flow granularity would consequently have a significant impact on how and where computations can be performed as well as ensuring that end-hosts know who has altered the data and how.

3.2.1. Research questions and challenges

1. Which flow granularities are sensible for which scenarios and upper layer protocols?

2. How do the different flow granularities map to error and congestion control?

3. How is flow granularity used for creating affinity in, e.g., routing choices?

4. How may flow granularity information be used in COIN elements, e.g., to support routing and transport protocol realizations?

3.2.2. Related concepts and efforts

As mentioned above, flow granularities are defined in transport protocols through their semantic for the unit of transfer, which can be a 'datagram' or a 'flow'. Upper layer protocols, such as HTTP, map their application data into this semantic, resulting, for instance, in a flow of HTTP requests. Note that the flow identified by the 5-tuple for the transport connection usually also carries the reverse direction of communication, e.g., in the form of HTTP responses. The introduction of 'TCP re-use' in HTTP/1.1 introduced the capability of sending many HTTP request/response interactions in a single TCP flow. The notion of flow granularity is being used in [DYNCAST] to link the relation of one or more application level interactions to a specific service instance in deployment scenarios where more than one service instance may serve requests for a given service; [DYNCAST] refers to the problem of 'instance affinity', i.e., the need to send one or more such interactions to the same instance before being able to choose another instance (e.g., based on computing or network metrics, as suggested in [DYNCAST]). At this point of the work, the potential realization of such 'instance affinity' and the relation to transport (as well as application) protocols has not been discussed yet.

Within the concept of Service Function Chaining (SFC) [SFC-Arch], a chain of services is formed and expressed through the next service header (NSH) [SFC-NSH], which provides entries into a next hop table maintained at each Service Function Forwarder (SFF) [SFC-Arch]. Packet classification takes place at the entry point of the chain, therefore providing a notion of flow granularity where the chain is treated as the 'unit of transfer'. Chaining can take place at Layer 2 or Layer 3, but also at a name-based layer (such as HTTP), as proposed in [RFC8677].

3.3. Collective Communication

COIN scenarios may exhibit a collective communication semantic, i.e., a communication between one and more computational endpoints, as is for example illustrated by use cases in [I-D.draft-sarathchandra-coin-appcentres-04]. With this, unicast and multicast transmissions become almost equal forms of communication, as also observed in work on information-centric networking (ICN) [ICNRG].

Yet, these many-point relations may be ephemeral down to the granularity of individual service requests between computational endpoints which questions the viability of stateful routing and transport approaches used for long-lived multicast scenarios such as liveTV transmissions.

This is particularly pertinent for the transport layer where reliability and flow control among a quickly changing set of receivers is a challenging problem. The ability to divide receiver groups with the support of in-network COIN elements may provide solutions that will cater to the possible dynamics of collective communication among computational endpoints.

3.3.1. Research questions and challenges

1. How to handle ephemeral transport relations at the request level across more than one endpoint?

2. How to separate longer-term resource management from shorter-term transaction handling for, e.g., error and flow control?

3. What role could COIN elements play in improving on solutions for questions 1 and 2?

3.3.2. Related concepts and efforts

As stated above, work in the [ICNRG] has long considered multicast and unicast delivery as two communication models, realized by the same communication method that utilizes the interest-data model of ICN. The work in [ICNIP] utilizes a different approach by relying on path-based forwarding of packets identified through service-level identifiers (such as URLs but also IP addresses), where return path multicast is achieved through binary operations over the path information of incoming service requests. The utilized transport network technology is that of 5GLAN or SDN, where the latter uses an OpenFlow-compatible approach to path-based forwarding with constant state requirements for the in-network forwarders. A similar approach is used in [BIER-MC] albeit at the level of a BIER overlay network. [ICNIP] also discusses, albeit briefly only, the separation of longer-lived resource management from shorter-lived transaction handling to increase efficiency of the ephemeral return path communication at the transport level.

3.4. Authentication

The realisation of COIN legitimizes and actively promotes that data transmitted from one host to another can be altered on the way inside the network. This opens the door for foul play as all intermediate network elements - no matter if they are malicious or misbehaving by accident, COIN elements, or 'traditional' middleboxes - could simply start altering parts of the original data and potentially cause harm to the end-hosts. What is needed are mechanisms with which the receiving host can verify (a) how and (b) by whom the data has been altered on the way. In fact, these might very well be two distinct

mechanisms as one (a) only focusses on the changes that are made to the data while (b) requires a scheme with which COIN elements can be uniquely identified (could very well relate to Section 3.1) and subsequently authenticated.

3.4.1. Research questions and challenges

1. How are changes to the data within the network communicated to the end-hosts?

2. How are the COIN elements that are responsible for the changes communicated to the end-hosts?

3. How are changes made by the COIN elements authenticated?

3.4.2. Related concepts and efforts

* Proof of Transit [SFC-PoT]

The Proof of Transit concept of the SFC WG allows for proving that packets have passed a defined path. Using this concept, it could at least be possible to make sure that a packet has indeed passed the desired COIN elements. However, it does not provide means to validate which changes were made by the known nodes.

3.5. Security

Many early COIN concepts require an unencrypted transmission of data. At the same time, there is a general tendency towards more and more security features in communication protocols. QUIC, e.g., encrypts all payload data and almost all header content already inside the transport layer. This makes current COIN concepts infeasible in settings where QUIC connections are used as the COIN elements do not have access to any packet content. Using COIN thus also depends on how well security mechanisms like encryption can be integrated into COIN frameworks.

3.5.1. Research questions and challenges

To be added.

3.5.2. Related concepts and efforts

To be added.

3.6. Transport Features

Depending on application needs, different transport protocols provide different features. These features shape the behavior of the protocol and have to be taken into account when developing COIN functionality. In this section, we focus on the impact of reliability as well as flow and congestion control to create awareness for the multifaceted interaction between the transport protocols and COIN elements.

3.6.1. Reliability

Applications require a reliable transport whenever it is important that all data is transmitted successfully. TCP[TCP] provides such a reliable communication as it first sets up a dedicated connection and then ensures the successful reception of all data. In contrast, UDP[UDP] is a connectionless protocol without guarantees and COIN elements working on UDP transmissions must be robust to lost information. This is not the case for applications on top of TCP, but the retransmissions and the TCP state, which TCP uses to achieve the reliability, make packet processing for COIN more complex due to at least three reasons.

The concept of retransmissions bases on the end-to-end principle as retransmissions are performed by the sender if it has determined that the receiver did not receive the corresponding original message. Both participants can then act knowing that parts of the overall data are still missing. For simple COIN elements, which are not aware of the involved TCP states and which do not track sequence numbers, it is difficult to identify (a) that a packet in the sequence is missing and (b) that a packet is a retransmission. One question is whether COIN elements should incorporate an understanding for retransmissions on the basis of existing transport mechanisms or if a COIN-capable transport should include dedicated signals for the COIN elements.

Apart from challenges in identifying retransmissions, there is also the fact that they are sent out of order with the original packet sequence. Depending on the chosen flow granularity (see Section 3.2), COIN elements might have to hold contextual information for a prolonged time once they identify an impeding retransmission. Moreover, they might have to postpone or cancel computations if data is missing and instead schedule later computations. The main question arising from this is: to what extent should COIN elements be capable of incorporating retransmissions into their computation schemes and how much additional storage capabilities are required for this?

When incorporating COIN elements into the retransmission mechanisms, it is also an interesting question whether it should be possible to request or perform retransmissions from COIN elements. Considering a setting with COIN elements that are capable of detecting missing packets and retransmission requests, it might improve the overall performance if the COIN element directly requests or performs the retransmission instead of forwarding the packet/request through the complete sequence of elements. This is especially interesting in the context of collective communication where reliability mechanisms could make use of the multi-source nature of the communication and leverage the presence of many COIN elements in the network, for instance by using network coding techniques, which in turn may benefit from COIN elements participating in the reliability mechanism. In all cases, the aforementioned storage capabilities are relevant so that the COIN elements can store enough information. The general question, i.e., which nodes in the sequence should do the retransmission, has already been worked on in the context of multicast transport protocols.

Depending on the extent of realization of the presented retransmission features, COIN elements might almost have to implement some of TCP's state to fulfil their tasks. Considering that different COIN elements have different computational and storage capacities, it is very likely that not every form of transport integration into COIN can be supported by every available COIN platform. The choice of devices included into the communication will hence certainly affect the types of transport protocols that can be operated on the COIN networks.

Another aspect to consider is the 'unit' that needs to be reliably transferred. In stream-based transport protocols, such as TCP, packets represent the smallest unit of transfer. However, different choices in the flow granularity and a possible move to larger-than-a-packet messages or transactions, as suggested in Section 3.2, might make other approaches to reliability viable that operate on the basis of such messages.

3.6.1.1. Research questions and challenges

1. What is the unit of reliable transfer?

2. How to utilize more than one computational endpoint in the reliability mechanism?

3. Should COIN elements be aware of retransmissions?

4. How can COIN elements identify missing packets or retransmissions?

5. Should COIN elements be explicitly notified about retransmissions?

6. To what extent should COIN elements be capable of incorporating retransmissions into their computation schemes?

7. How much storage capabilities are required for incorporating retransmissions?

8. How can COIN elements incorporate missing packets into their computations?

9. How to deal with state changes in COIN elements caused by data lost later in the communication chain and then retransmitted?

10. Should COIN elements be capable of requesting retransmissions/ answering retransmission requests?

11. Which devices should perform retransmissions?

12. Do COIN elements have to keep transport state?

13. How much transport state do COIN elements have to keep?

3.6.1.2. Related concepts and efforts

* Transmission Control Protocol [TCP]

TCP provides reliable, ordered, and error-checked delivery of a byte stream. As such, TCP does not allow for payload changes. This means that COIN elements could only make changes to lower header information.

* Stream Control Transmission Protocol [SCTP]

SCTP provides ensures a reliable exchange of messages. In contrast to TCP, it decouples reliability from in-order delivery and thus allows for sending messages without ordering. Additionally, it has also been extended to provide partial reliability, i.e., controlling the desired reliability on a per-message basis [SCTP-PR].

* Constrained Application Protocol [CoAP]

CoAP is a specialized protocol targeting nodes that are constrained, e.g., in terms of compute power or available bandwidth. It is message-based and distinguishes between confirmable and non-confirmable messages, i.e., similar to SCTP, allows for controlling the reliability on a per-message basis.

* User Datagram Protocol [UDP]

UDP is a message-based protocol that does not provide any guarantees regarding reliability to the application layer.

3.6.2. Flow/Congestion Control

TCP incorporates mechanisms to avoid overloading the receiving host (flow control) and the network (congestion control) and determines its sending rate as the minimum value of what both mechanisms determine as feasible for the system. This approach is based on the notion that computing and forwarding hosts are separated and is challenged by the inclusion of COIN elements, i.e., computing nodes in the network.

Flow control bases on explicit end-host information as the participating end-hosts notify each other about how much data they are capable of processing and consequently do not transmit more data as the other host can handle. This only changes if one of the end-hosts updates its flow control information.

Congestion control, on the other hand, interprets volatile feedback from the network to guess a sending rate that is possible given the current network conditions. Most congestion control algorithms hereby follow a cyclical procedure where the sending end-hosts constantly increase their sending rate until they detect network congestion. They then decrease their sending rate once and start to increase it again.

In this traditional two-fold approach, loss, delay, or any other congestion signal (depending on the congestion control algorithm) induced by COIN elements (only in case that they are the bottleneck) is interpreted as network congestion and thus accounted for in the congestion control mechanism. This means that the sending end-host may repeatedly overload the computational capabilities of the COIN elements when probing for the current network conditions instead of respecting general device capabilities as is done by flow control.

In the context of COIN, the granularity of flows may see a division into sub-flows or messages to better represent the used computational semantic as discussed in Section 3.2. This raises the question whether flow and congestion control should be applied to longer term flows (of many sub-flows or messages) or directly to sub-flows. Eventually, this could possibly lead to a separation of error control (for sub-flows) and flow control (for longer-term flows). A subsequent challenge is then how to reconcile the possible volatile nature of sub-flow relations (between computational endpoints) with the longer-term relationship between network endpoints that will see

a flow of messages between them. This is particularly pertinent in collective communication scenarios, where many forward unicast sub-flows may lead to a single multicast sub-flow response albeit only for that one response message. Reconciling the various unicast resource regimes into a single (ephemeral) multicast one poses a significant challenge.

Consequently, the question arises whether COIN elements should be able to participate in end-to-end flow control.

3.6.2.1. Research questions and challenges

1. Should COIN elements be covered by congestion control?

2. Should COIN elements be able to participate in end-to-end flow control?

3. How could a resource constraint scheme similar to flow control be realized for COIN elements?

4. How to reconcile message-level flexibility in transport relations between computational endpoints with longer-term resource stability between network elements participating in the computational scenario?

3.6.2.2. Related concepts and efforts

* Transmission Control Protocol [TCP]

TCP implements flow and congestion control. The traffic is controlled using TCP's receiver and congestion windows.

* Separation of Data Path and Data Flow

[I-D.draft-asai-tsvwg-transport-review-02] proposes to explicitly divide transport protocols into two parts: a data path and a data flow layer. Essentially, the data path layer is responsible for handling path-related tasks, such as congestion control, and as such spans multiple flows. The data flow layer on top then handles flow-related tasks such as retransmissions and flow control. As indicated by the early stage of the document, the concrete structure is still up for debate. Yet, explicitly dividing congestion and flow control could give the opportunity to devise more sophisticated approaches to incorporate COIN elements.

4. Summary of related research and standardization efforts

Issue	Efforts
Addressing	Segment and Source Routing: - [SPRING-WG] - Segment Routing [SR] (Service/Network) Function Chaining/Composition: - [SFC-WG] - SFC Problem Statement [SFC-PS] - SFC Architecture [SFC-Arch] - SFC Network Service Header [SFC-NSH] - Internet Services over IP [ICNIP]
Flow Granularity	Service Function Chaining [SFC-Arch], [SFC-NSH] Use cases and problem statement for dynamic anycast [DYNCAST]
Collective Communication	Information-centric networking [ICNRG] Internet Services over IP [ICNIP] HTTP multicast over BIER [BIER-MC]
Authentication	SFC Proof of Transit [SFC-PoT]
Reliability	Transmission Control Protocol [TCP] Stream Control Transmission Protocol [SCTP] Constrained Application Protocol [CoAP] User Datagram Protocol [UDP]
Flow/Congestion Control	[I-D.draft-asai-tsvwg-transport-review-02]

Figure 1: Related research and standardization efforts.

5. Gap Analysis

This section provides a gap analysis within the identified technology areas with respect to existing IETF solutions and ongoing efforts in transport protocols that were summarized in the previous section.

The goal of such analysis is to identify issues with those existing solutions through and within COIN environments. From the viewpoint of structuring the gap analysis, approaches such as those taken in [I-D.draft-jia-intarea-internet-addressing-gap-analysis] may be used as examples as well as direct (if suitable) input into the gap analysis performed here.

5.1. Addressing

TBD

5.2. Flow granularity

TBD

5.3. Collective Communication

TBD

5.4. Authentication

TBD

5.5. Security

TBD

5.6. Transport Features

5.6.1. Reliability

TBD

5.6.2. Flow/Congestion Control

TBD

6. Summary of Issues Identified

This section will summarize the main issues across the investigated transport technology areas of the previous section.

7. Security Considerations

COIN changes the traditional paradigm of a simple network and the corresponding end-to-end principle as it encourages computations in/by the network. Approaches designed to protect transmitted data, such as Transport Layer Security (TLS), which is even embedded into newer transport protocols like QUIC, rely on the end-to-end principle and are thus conceptually not compatible with COIN without a consistent view on how in-network compute elements would fit into the traditional end-to-end model.

Additionally, COIN elements often do not support required cryptographic functionality.

Thus, there may be a need for new security concepts specific to COIN environment that may have to be developed to allow for a secure use within COIN environments.

8. IANA Considerations

N/A

9. Conclusion

The advent of COIN may bring many new use cases, as documented in [I-D.draft-irtf-coinrg-use-cases], with promises of improved solutions for various problems. The concept of in-network computing capabilities, however, is not directly compatible with the end-to-end nature of transport protocols, thereby posing a number of key questions regarding COIN and transport protocols.

Those key questions, positioned across key technology areas for transport protocols, lead us to look at possible gaps that may be found in existing solutions when it comes to suitably supporting COIN environments and scenarios. The gap analysis performed in this document and the issues identified as a result of this analysis are positioned as possible input into shaping a research agenda for new transport protocols that have in-network computing capabilities in mind and support them securely as well as the best performance possible.

10. Informative References

[BIER-MC] Trossen, D., Rahman, A., Wang, C., and T. T. Eckert, "Applicability of BIER Multicast Overlay for Adaptive Streaming Services", Work in Progress, Internet-Draft, draft-ietf-bier-multicast-http-response-06, 10 July 2021, <https://www.ietf.org/archive/id/draft-ietf-bier-multicast-http-response-06.txt>.

[CoAP] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <https://www.rfc-editor.org/info/rfc7252>.

[DYNCAST] Liu, P., Willis, P., and D. Trossen, "Dynamic-Anycast (Dyncast) Use Cases and Problem Statement", Work in Progress, Internet-Draft , February 2021, <https://datatracker.ietf.org/doc/draft-liu-dyncast-ps-usecases/>.

[E2E] 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, <https://doi.org/10.1145/357401.357402>.

[I-D.draft-asai-tsvwg-transport-review-02]
Asai, H., "Separation of Data Path and Data Flow Sublayers in the Transport Layer", Work in Progress, Internet-Draft, draft-asai-tsvwg-transport-review-02, 15 September 2021, <https://www.ietf.org/archive/id/draft-asai-tsvwg-transport-review-02.txt>.

[I-D.draft-irtf-coinrg-use-cases]
Kunze, I., Wehrle, K., Trossen, D., and M. Montpetit, "Use Cases for In-Network Computing", Work in Progress, Internet-Draft, draft-irtf-coinrg-use-cases-00, 17 February 2021, <https://www.ietf.org/archive/id/draft-irtf-coinrg-use-cases-00.txt>.

[I-D.draft-jia-intarea-internet-addressing-gap-analysis]
Jia, Y., Trossen, D., Iannone, L., Shenoy, N., and P. Mendes, "Gap Analysis in Internet Addressing", Work in Progress, Internet-Draft, draft-jia-intarea-internet-addressing-gap-analysis-00, 12 July 2021, <https://www.ietf.org/archive/id/draft-jia-intarea-internet-addressing-gap-analysis-00.txt>.

[I-D.draft-jia-intarea-scenarios-problems-addressing]
Jia, Y., Trossen, D., Iannone, L., Shenoy, N., Mendes, P., 3rd, D. E. E., and P. Liu, "Challenging Scenarios and Problems in Internet Addressing", Work in Progress, Internet-Draft, draft-jia-intarea-scenarios-problems-addressing-01, 12 July 2021, <https://www.ietf.org/archive/id/draft-jia-intarea-scenarios-problems-addressing-01.txt>.

[I-D.draft-king-irtf-challenges-in-routing]
King, D. and A. Farrel, "Challenges for the Internet Routing Infrastructure Introduced by Changes in Address Semantics", Work in Progress, Internet-Draft, draft-king-irtf-challenges-in-routing-03, 14 June 2021, <https://www.ietf.org/archive/id/draft-king-irtf-challenges-in-routing-03.txt>.

[I-D.draft-king-irtf-semantic-routing-survey]
King, D. and A. Farrel, "A Survey of Semantic Internet Routing Techniques", Work in Progress, Internet-Draft, draft-king-irtf-semantic-routing-survey-02, 28 June 2021, <https://www.ietf.org/archive/id/draft-king-irtf-semantic-routing-survey-02.txt>.

[I-D.draft-kutscher-coinrg-dir-02]
Kutscher, D., Karkkainen, T., and J. Ott, "Directions for Computing in the Network", Work in Progress, Internet-Draft, draft-kutscher-coinrg-dir-02, 31 July 2020, <http://www.ietf.org/internet-drafts/draft-kutscher-coinrg-dir-02.txt>.

[I-D.draft-sarathchandra-coin-appcentres-04]
Trossen, D., Sarathchandra, C., and M. Boniface, "In-Network Computing for App-Centric Micro-Services", Work in Progress, Internet-Draft, draft-sarathchandra-coin-appcentres-04, 26 January 2021, <https://www.ietf.org/internet-drafts/draft-sarathchandra-coin-appcentres-04.txt>.

[ICNIP] Trossen, D., Robitzsch, S., Essex, U., AL-Naday, M., and J. Riihijarvi, "Internet Services over ICN in 5G LAN Environments", Work in Progress, Internet-Draft, draft-trossen-icnrg-internet-icn-5glan-04, 1 October 2020, <http://www.ietf.org/internet-drafts/draft-trossen-icnrg-internet-icn-5glan-04.txt>.

[ICNRG] "Information-Centric Networking, IRTF Research Group", <https://datatracker.ietf.org/rg/icnrg/about/>.

[RFC1958] Carpenter, B., Ed., "Architectural Principles of the Internet", RFC 1958, DOI 10.17487/RFC1958, June 1996, <https://www.rfc-editor.org/info/rfc1958>.

[RFC2775] Carpenter, B., "Internet Transparency", RFC 2775, DOI 10.17487/RFC2775, February 2000, <https://www.rfc-editor.org/info/rfc2775>.

[RFC8677] Trossen, D., Purkayastha, D., and A. Rahman, "Name-Based Service Function Forwarder (nSFF) Component within a Service Function Chaining (SFC) Framework", RFC 8677, DOI 10.17487/RFC8677, November 2019, <https://www.rfc-editor.org/info/rfc8677>.

[RFC8799] Carpenter, B. and B. Liu, "Limited Domains and Internet Protocols", RFC 8799, DOI 10.17487/RFC8799, July 2020, <https://www.rfc-editor.org/info/rfc8799>.

[SCTP] Stewart, R., Ed., "Stream Control Transmission Protocol", RFC 4960, DOI 10.17487/RFC4960, September 2007, <https://www.rfc-editor.org/info/rfc4960>.

[SCTP-PR] Stewart, R., Ramalho, M., Xie, Q., Tuexen, M., and P. Conrad, "Stream Control Transmission Protocol (SCTP) Partial Reliability Extension", RFC 3758, DOI 10.17487/RFC3758, May 2004, <https://www.rfc-editor.org/info/rfc3758>.

[SFC-Arch] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <https://www.rfc-editor.org/info/rfc7665>.

[SFC-NSH] Quinn, P., Ed., Elzur, U., Ed., and C. Pignataro, Ed., "Network Service Header (NSH)", RFC 8300, DOI 10.17487/RFC8300, January 2018, <https://www.rfc-editor.org/info/rfc8300>.

[SFC-PoT] Brockners, F., Bhandari, S., Mizrahi, T., Dara, S., and S. Youell, "Proof of Transit", Work in Progress, Internet-Draft, draft-ietf-sfc-proof-of-transit-08, 1 November 2020, <https://www.ietf.org/internet-drafts/draft-ietf-sfc-proof-of-transit-08.txt>.

[SFC-PS] Quinn, P., Ed. and T. Nadeau, Ed., "Problem Statement for Service Function Chaining", RFC 7498, DOI 10.17487/RFC7498, April 2015, <https://www.rfc-editor.org/info/rfc7498>.

[SFC-WG] "Service Function Chaining, IETF Working Group", <https://datatracker.ietf.org/wg/sfc/about/>.

[SPRING-WG] "Source Packet Routing in Networking, IETF Working Group", <https://datatracker.ietf.org/wg/spring/about/>.

[SR] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <https://www.rfc-editor.org/info/rfc8402>.

[TCP] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <https://www.rfc-editor.org/info/rfc793>.

[UDP] Postel, J., "User Datagram Protocol", STD 6, RFC 768, DOI 10.17487/RFC0768, August 1980, <https://www.rfc-editor.org/info/rfc768>.

Authors' Addresses

Ike Kunze
RWTH Aachen University
Ahornstr. 55
D-52074 Aachen
Germany

Email: kunze@comsys.rwth-aachen.de

Klaus Wehrle
RWTH Aachen University
Ahornstr. 55
D-52074 Aachen
Germany

Email: wehrle@comsys.rwth-aachen.de

Dirk Trossen
Huawei Technologies Duesseldorf GmbH
Riesstr. 25C
D-80992 Munich
Germany

Email: Dirk.Trossen@Huawei.com