Transport Protocol Issues of In-Network Computing Systems
draft-kunze-coinrg-transport-issues-00

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 discusses selected characteristics of transport protocols which have to be adapted to support COIN functionality.

Table of Contents

1. Introduction

A fundamental design choice of the Internet is that the network should be kept as simple as possible while complexity in the form of processing should ideally be located at the edges of the network, i.e., on end-hosts. This choice is reflected in the widely known end-to-end principle which states that end-hosts directly address each other and perform all relevant computations while the only purpose of the network is to deliver the packets without modifying them. Transport protocols are consequently designed to facilitate the direct communication between end-hosts.

In practice, the end-to-end principle is often violated by intransparent middleboxes which alter transmitted packets, e.g., by dropping or changing header fields. Contrary to that, COIN encourages explicit computations in the network which introduces an intertwined complexity as the concrete computations on the end-hosts critically depend on the functionality that is deployed in the network. On another note, it challenges traditional end-to-end transport protocols as they lack means for addressing in-network computation entities and as they are generally not designed to include more than two devices into a communication. Some of these problems are already presented in [I-D.draft-kutscher-coinrg-dir-00]. This draft intends to discuss potential problems for traditional transport protocols in more detail to raise questions that the COIN community needs to solve before a widespread application of COIN

functionality is sensible. Collaboration with other IRTF and IETF groups, such as PANRG, the IETF transport area in general, or the LOOPS BOF, can help in finding suitable solutions.

2. Addressing

The traditional addressing concept of the Internet are end-hosts directly addressing each other, with all computational intelligence residing at the network edges. As other COIN drafts, such as [I-D.draft-montpetit-coin-xr-03], [I-D.draft-he-coin-managed-networks-01] and [I-D.draft-kunze-coin-industrial-use-cases-00], and extensive publications, such as [DANG], [RUETH] and [SAPIO], in the last years have shown, performing some computations within the network offers the prospect of improved application performance. In systems like data centers, which do not rely on the Internet and where the whole network is under the control of the network operator, integrating such functionality can be done by explicitly adjusting the communication schemes within these fully controlled systems based on the functionality that is executed in the network.

With a widespread application of COIN and a consequent increase in computational capacities in the network, however, it might also become viable to deploy functionality in the 'wild' Internet so that everyday applications might also benefit from these computations. At this point, it is no longer feasible to manually adjust the traffic patterns or the applications to correctly incorporate changes made by the network.

It might thus make sense to make it possible to specify which kind of functionality should be applied to the transmitted data on the path inside the network, maybe even where or by whom exactly the execution should take place. Such functionality could for example be implemented using an indirection mechanisms which routes a packet along a pre-defined or dynamically chosen path which then realizes the desired functionality. Related concepts which might be of use are Segment and Source Routing as well as (Service/Network) Function Chaining/Composition.

The main challenges/questions at this point are:

1. How should end-hosts address the functionality within the network?

2. How exactly can the end-hosts influence where or by whom the functionality is executed?

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

Assuming that there is a suitable addressing scheme which allows to define which kinds of functionality should be applied to the transmitted data, a next question that arises is how the transmitted data is to be treated by the devices implementing the functionality.

3. Flow granularity

Core networking hardware pipelines such as backbone switches serving several tens of GBit/s are built to process incoming packets on a per-packet basis, keeping little to no state between them. While this is appropriate for the general task of forwarding packets, it might not be sufficient for performing computations as related 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. The common scheme is that 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 should be treated individually. This, above all, perfectly meets the possibilities that are already offered by all networking equipment.

2. Every packet should be 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 and might hence also be relevant for the computations.

3. Every packet should be treated as part of a byte stream. Here, all previous packets and potentially even all following 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 of a sequence of packets.

The flow granularity consequently has a significant impact on how computations can be performed and where. Apart from how the COIN

elements should treat the transmitted data, another important aspect is how it can be ensured that the end-hosts know who has altered the data and how.

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. While this can be beneficial if implemented correctly, it also 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 thus potentially cause harm to the end-hosts. What is consequently needed is a mechanism 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 2) and subsequently authenticated.

The challenges at this point are thus the following:

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 is it verified that indeed the proclaimed COIN elements have performed the changes and not some impostor?

5. Security

Today, most, if not all, COIN concepts base on the fact that the data is transmitted in plain text as this makes working on the data easy. This is in contrast to a general development which sees more and more security features added to communication protocols, nicely highlighted by QUIC where the all payload data and almost all header content (except for the spin bit) is already encrypted inside the transport layer. This, in turn, makes COIN concepts infeasible in settings where QUIC connections are used as the COIN elements do not have access to any packet content. As waiving security features is generally not acceptable, the widespread success of COIN might very well also depend on how well security mechanisms like encryption can be integrated into COIN frameworks.

Together, the four aspects presented in Section 2 to Section 5 form a set of fundamental properties that should be taken into account for a basic transport-compatible realization of COIN. What is not yet considered is the fact that different transport protocols typically differ in functionality and thus have a significantly different behavior. In the following, we briefly discuss select additional transport features to create awareness for the multifaceted interaction between the transport protocols and COIN elements.

6. Advanced Transport Features

There is a variety of transport features which are only supported in some concrete transport implementations. Still, they have a significant impact on the behavior and performance of the protocols. One aspect is that some protocols offer reliability while others do not. This is for example visible when comparing UDP, a connectionless protocol without guarantees, to TCP which first sets up a dedicated connection and then ensures the successful reception of all data. When facing UDP transmissions, COIN elements potentially have to cope with lost information while with TCP it is fairly save to say that the packets will reach them at some point.

This, however, also makes it more difficult for COIN elements as TCP retransmissions, which are issued once a packet has been detected as lost, are sent drastically out of order with the original packet sequence.

Thinking one step further, retransmissions are sent from the original sender of the packet. In a communication setting with COIN elements in between, resending the packet through the complete sequence of elements might not be necessary if, e.g., a packet is lost at the last stage of a sequence of COIN elements. Here, it might be enough if this last element resends the packet, although this in turn means that there have to be storage capabilities on the devices enabling them to store a certain number of packets and have them ready for retransmission. The general question, i.e., which of the nodes in the sequence should actually do the retransmission, has already been worked on in the context of multicast transport protocols.

Now focussing on the aspect of storage capabilities, it can be said that different COIN devices have different computational and storage capacities which can become a challenge if they have to hold some packets (potentially for retransmission) or if they are supposed to embed into TCP for which they might be forced to hold some form of TCP's state. Consequently, 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 is flow and congestion control to avoid overloading the receiving end-host and the network; it is included in TCP, but not in UDP, but has an impact on how the end-hosts can send their data.

All in all, there is a wide range of non-essential transport features which nonetheless offer improved performance in certain settings and for certain application combinations. However, as presented, it is likely that not all of the features/types of transport protocols can be supported on every COIN element. A potential approach might be to define different classes of COIN-ready transport protocols which can then be deployed depending on the concretely available networking/ hardware elements.

7. Security Considerations

TBD

8. IANA Considerations

N/A

9. Conclusion

The advent of COIN comes with many new use cases and promises improved solutions for various problems. It is, however, not directly compatible with the end-to-end nature of transport protocols. To enable a transport-based communication, it is thus important to answer key questions regarding COIN and transport protocols, some of which are raised in this document.

10. Informative References

[DANG] Dang, HT., "NetPaxos: Consensus at Network Speed", DOI: 10.1145/2774993.2774999, in SOSR, 2015.

[I-D.draft-he-coin-managed-networks-01]
He, J., Li, A., and M. Montpetit, "In-Network Computing for Managed Networks: Use Cases and Research Challenges", draft-he-coin-managed-networks-01 (work in progress), July 2019.

[I-D.draft-kunze-coin-industrial-use-cases-00]
Kunze, I., Rueth, J., and K. Wehrle, "Industrial Use Cases for In-Network Computing", draft-kunze-coin-industrial-use-cases-00 (work in progress), July 2019.

[I-D.draft-kutscher-coinrg-dir-00]
Kutscher, D., Karkkainen, T., and J. Ott, "Directions for Computing in the Network", draft-kutscher-coinrg-dir-00 (work in progress), July 2019.

[I-D.draft-montpetit-coin-xr-03]
Montpetit, M., "In Network Computing Enablers for Extended Reality", draft-montpetit-coin-xr-03 (work in progress), July 2019.

[RUETH] Rueth, J., "Towards In-Network Industrial Feedback Control", DOI: 10.1145/3229591.3229592, in ACM SIGCOMM NetCompute, August 2018.

[SAPIO] Sapio, A., "Scaling Distributed Machine Learning with In-Network Aggregation", 2019, <https://arxiv.org/abs/1903.06701>.

Authors' Addresses

Ike Kunze
RWTH Aachen University
Ahornstr. 55
Aachen D-50274
Germany

Phone: +49-241-80-21422
Email: kunze@comsys.rwth-aachen.de

Klaus Wehrle
RWTH Aachen University
Ahornstr. 55
Aachen D-50274
Germany

Phone: +49-241-80-21401
Email: wehrle@comsys.rwth-aachen.de