

COIN
Internet-Draft
Intended status: Informational
Expires: January 3, 2020

J. He
A. Li, Ed.
Huawei
M. Montpetit
Triangle Video
July 2, 2019

In-Network Computing for Managed Networks: Use Cases and Research
Challenges
draft-he-coin-managed-networks-01

Abstract

This draft wants to review the existing research and the open issues that relate to the addition of data plane programmability in managed networks. While some of the research hypotheses that are at the center of in-network-computing have been investigated since the time of active networking, recent developments in software defined networking, virtualization programmable switches and new network programming languages like P4 have generated a new enthusiasm in the research community and a flourish of new projects in systems and applications alike. This is what this draft is addressing.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 3, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

(<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
1.1. Requirements Language	3
2. In Network Computing and Data Centers	3
3. State of the Art in DC Programmability	5
3.1. In-Network Computing	5
3.2. In-Network Caching	7
3.3. In Network Consensus	7
4. Industrial Networks	9
5. Research Topics	9
5.1. Data Plane Issues in Managed Networks	10
5.2. Interaction with Transport Protocols	10
5.3. Interaction with Security Mechanisms	10
5.4. Privacy aspects	10
6. Conclusion	11
7. References	11
7.1. Normative References	11
7.2. Informative References	11
Authors' Addresses	13

1. Introduction

It is now a given in the computing and networking world that traditional approaches to cloud and client-server architectures lead to complexity and scalability issues. New solutions are necessary to address the growth of next generation managed network operation (in data centers and edge devices alike) including automation, self-management, orchestration across components and federation across network nodes to enable emerging services and applications.

Mobility, social network and big data and AI/ML as well as emerging content application in the XR (virtual, augmented and mixed reality) as well as emerging industrial networking applications require more scalable, available and reliable solution not only in real time, anywhere and over a wide variety of end devices. While these solutions involve edge resources for computing, rendering and distribution, this paper focuses on the data center what are the current research approaches to create more flexible solutions. We must define what we understand by data centers. In this draft, we

are not going to limit them to single location cloud resources but add multiple locations as well as interwork with edge resources to enable the network programmability that is central to next generation DCs in term of supported services and dynamic resilience. This leads to innovative research opportunities, including but not limited to:

- Software defined networking (SDN) in distributed environments.
- Security and trust models.
- Data plane programmability for consensus and key-value operations.
- High Level abstractions as in network computing should focus on primitives, which can be widely re-used in a class of applications and workloads, and identify those high level abstractions to promote deployment.
- Machine Learning (ML) and Artificial Intelligence (AI) to detect faults and failures and allow rapid responses as well as implement network control and analytics.
- New services for mixed reality (XR) deployment with in-network optimization and advanced data structures and rendering for interactivity, security and resiliency.
- New applications in industrial networking.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. In Network Computing and Data Centers

As DC hardware components becoming interchangeable, the advent of software-defined technologies suggests that a change is underway. In the next-generation data center, an increasing percentage of critical business and management functions will be activated in the software layer rather than the underlying hardware. This will allow organizations to move away from the current manual configurations to handle more dynamic, rules-based configurations. Hence, virtualization and cloud computing have redefined the datacenter (DC) boundaries beyond the traditional hardware-centric view [SAPIO]. Servers, storage, monitoring and connectivity are becoming one. The network is more and more the computer.

Hence, there is now a number of distributed networking and computing systems which are the basis of big-data and AI-related applications in DCs in particular. They include Distributed file system (e.g. the Hadoop Distributed File System or HDFS [HADOOP]), distributed memory database (e.g. MemCached [MEM]), distributed computing system (e.g. mapReduce from Hadoop, [HADOOP], Tensorflow [TENSOR], and Spark GraphX [SPARK]), as well as distributed trust systems on the blockchain, such as hyperledgers and smart contracts.

In parallel the emergence of the P4 language [P4] and programmable switches facilitates innovation and triggers new research. For example, the latest programmable switches make the concept of the totally programmable and dynamically reconfigurable network closer to reality. And, as distributed systems are increasingly based on memory instead of hard disks, distributed system-based application performance is increasingly constrained by network resources not computing.

However, there are some challenges when introducing in-network computing and caching:

- Limited memory size: for example, the SRAM size [TOFINO] can be as small as tens of MBs.
- Limited instruction sets: the operations are mainly simple arithmetic, data (packet) manipulation and hash operation. Some switches can provide limited floating-point operation. This enables network performance tools like forward error correction but limits more advanced applications such as machine learning for congestion control for example.
- Limited speed/CPU processing capabilities: only a few operations can be performed on each packet to ensure line speed (tens of nano-seconds on fast hardware). Looping could allow a processed packet to re-enter the ingress queue, but with a cost of increasing latency and reducing forwarding capabilities.
- Performance: for devices located on the links of the distributed network; it is to be evaluated how on-path processing can reduce the FCT (Flow Completion Time) in data center network hence reduce the network traffic/congestion and increase the throughput.

The next sections of this draft review how some of these questions are currently being addressed in the research community.

3. State of the Art in DC Programmability

Recent research has shown that in-network computing can greatly improve the DC network performance of three typical scenarios: aggregate on path- computing, key-value (K-V) cache, and strong consistency. Some of these research results are summarized below.

3.1. In-Network Computing

The goals on on-path computing in DC is 1. to reduce delay and/or increase throughput for improved performance by allowing advanced packet processing and 2. to help reduce network traffic and alleviate congestion by implementing better traffic and congestion management [REXFORD] [SOULE] [SAPIO].

However, in terms of research and implementation, there are still open issues that need to be addressed in order to fulfill these promises beyond what was mentioned in the previous section. In particular, the end-to-end principle which has driven most of the networking paradigms of the last 20 years is challenged when in-network computing devices are inserted on the ingress-egress path. This is still an open discussion topic.

The type of computing that can be performed to improve the DC performance is another of the open topics. Computing should improve performance but not at the expense of existing application degradation. Computing should also enable new applications to be developed. At the time of this writing those include data intensive applications in workload mode with partition and aggregation functionality.

Data-intensive applications include big data analysis (e.g. data reduction, deduplication and machine learning), graph processing, and stream processing. They support scalability by distributing data and computing to many worker servers. Each worker performs computing on a part of the data, and there is a communication phase to update the shared state or complete the final calculation. This process can be executed iteratively. It is obvious that communication cost and availability of bottleneck resources will be one of the main challenges for such applications to perform well as a large amount of data need to be transmitted frequently in many-to-many mode. But already, there are several distributed frameworks with user-defined aggregation functions, such as mapReduce from Hadoop [HADOOP], Pregel from Google [PREGEL], and DryadLinq from Microsoft [DRYAD]. These functions enable application developers to reduce the network load used for messaging by aggregating all single messages together and consequently reduce the task execution time. Currently, these aggregation functions are used only at the worker level. If they are

used at the network level, a higher traffic reduction ratio can be reached.

The aggregation functions needed by the data intensive applications, have some features that make it suitable to be at least executed in a programmable. They usually reduce the total amount of data by arithmetic (add) or logical function (minima/maxima detection) that can be parallelized. Performing these functions in the DC at the ingress of the network can be beneficial to reduce the total network traffic and lead to reduced congestion. The challenge is of course not to lose important data in the process especially when applied to different parts of the input data without considering the order and affect the accuracy of the final result.

In-network computing can also improve the performance of multipath routing by aggregating path capacity to individual flows and providing dynamic path selection, improving scalability and multitenancy.

Other data intensive applications that can be improved in terms of network load by in-network computing include: machine learning, graph analysis, data analytics and map reduce. For all of those, aggregation functions in the computing hardware provides a reduction of potential network congestion; in addition, because of the reduced load, the overall application performance is improved. The traffic reduction was shown to range from 48% up to 93% [SAPIO].

Machine learning is a very active research area for in-network computing because of the large datasets it both requires and generates. For example, in TensorFlow [TENSOR], parameters updates are small deltas that only change a subset of the overall tensor and can be aggregated by a vector addition operation. The overlap of the tensor updates, i.e. the portion of tensor elements that are updated by multi workers at the same time, is representative of the possible data reduction achievable when the updates are aggregated inside the network.

In graph analysis, three algorithms with various characteristics have been considered in [SAPIO]: PageRank, Single Source Shortest Path (SSSP) and Weakly Connected Components (WCC) with a commutative and associative aggregation function. Experiment shows that the potential traffic reduction ratio in the three applications is significant.

Finally, in map-reduce, experiments in the same paper show that after aggregation computing, the number of packets received by the reducer decreases by 88%~90% compared UDP, by 40% compared to

using TCP. There is thus great promise for mapReduce-like to take advantage of computing and storage optimization.

3.2. In-Network Caching

Key-value stores are ubiquitous and one of their major challenges are to process their associated data-skewed workload in a dynamic fashion. As in any caches, popular items receive more queries, and the set of popular items can change rapidly, with the occurrence of well-liked posts, limited-time offers, and trending events. The skew generated by the dynamic nature of the K-V can lead to severe load imbalance and significant performance deterioration. The server is either overused in an area or underused in another, the throughput can decrease rapidly, and the response time latency degrades significantly. When the storage server uses per core sharding/partitioning to process high concurrency, this degradation will be further amplified. The problem of unbalanced load is especially acute for high performance in memory K-V store.

The selective replication copying of popular items is often used to keep performance high. However, in addition to more hardware resource consumption, selective replication requires a complex mechanism to implement data mobility, data consistency and query routing. As a result, system design becomes complex and overhead is increased.

This is where in-network caching can help. Recent research experiments show that K-V cache throughput can be improved by 3~10 times by introducing in net cache. Analytical results in [FAN] show that a small frontend cache can provide load balancing for N back-end nodes by caching only $O(N \log N)$ entries, even under worst-case request patterns. Hence, caching $O(N \log N)$ items is sufficient to balance the load for N storage servers (or CPU cores).

In the NetCache system [JIN], a new rack-scale key-value store design guarantees billions of queries per second (QPS) with bounded latencies even under highly-skewed and rapidly-changing workloads. A programmable switch is used to detect, sort, cache, and obtain a hotspot K-V pair to process load balancing between the switch storage nodes.

3.3. In Network Consensus

Strong consistency and consensus in distributed networks are important. Significant efforts in the in-network computing community have been directed towards it. Coordination is needed to maintain system consistency and it requires a large amount of communication between network nodes and instances, taking away processing

capabilities from other more essential tasks. Performance overhead and extra resources often result in a decrease in consistency. And as a result, potential inconsistencies need to be addressed.

Maintaining consistency requires multiple communications rounds in order to reach agreement, hence the danger of creating messaging bottlenecks in large systems. Even without congestion, failure or lost messages, a decision can only be reached as fast as the network round trip time (RTT) permits. Thus, it is essential to find efficient mechanisms for the agreement protocols. One idea is to use the network devices themselves.

Hence, consensus mechanisms for ensuring consistency are some of the most expensive operations in managing large amounts of data [ZSOLT]. Often, there is a tradeoff that involves reducing the coordination overhead at the price of accepting possible data loss or inconsistencies. As the demand for more efficient data centers increases, it is important to provide better ways of ensuring consistency without affecting performance. In [ZSOLT] consensus (atomic broadcast) is removed from the critical path by moving it to hardware. The Zookeeper atomic broadcast (also in Hadoop) proof of concept is implemented at the network level on an FPGA, using both TCP and an application specific network protocol. This design can be used to push more value into the network, e.g., by extending the functionality of middle boxes or adding inexpensive consensus to in-network processing nodes.

A widely used protocol for consensus is Paxos. Paxos is a fundamental protocol used by fault-tolerant systems, and is widely used by data center applications. In summary, Paxos serializes transaction requests from different clients in the leader, ensuring that each learner (message replicator) in the distributed system is implemented in the same order. Each proposal can be an atomic operation (an inseparable operation set). Paxos does not care about specific content of the proposal. Recently, some research evaluation suggests that moving Paxos logic into the network would yield significant performance benefits for distributed applications [DANG15]. In this scheme network switches can play the role of coordinators (request managers) and acceptors (managed storage nodes). Messages travel fewer hops in the network, therefore reducing the latency for the replicated system to reach consensus since coordinators and acceptors typically act as bottlenecks in Paxos implementations, because they must aggregate or multiplex multiple messages. Experiments suggest that moving consensus logic into network devices could dramatically improve the performance of replicated systems. In [DANG15], NetPaxos achieves a maximum throughput of 57,457 messages/s, while basic Paxos the coordinator being CPU bound, is only able to send 6,369 messages/s. In [DANG16],

a P4 implementation of Paxos is presented as a result of Paxos implementation with programmable data planes.

Based on [JIN] and [DANG16], NetChain[JIN18] builds an on-chip key-value store that is chain replicated with Vertical Paxos for strong consistency and fault-tolerance in the network data plane. The authors have designed NetChain to provide in-network coordination service with sub-RTT latency and high throughput.

Other papers, have shown the use of in-network processing and SDN for Paxos performance improvements using multi-ordered multicast and multi-sequencing [LIJ] [PORTS].

4. Industrial Networks

For industrial networks, in-network processing can enable a new flexibility for automation and control by combining control and communication [RUETH][VESTIN].

Note: this section will be expanded in the next version of the draft.

5. Research Topics

While the previous section introduced the state of the art in data center and industrial in-network computing, there are still some open issues that need to be addressed. In this section, some of these questions are listed as well as the impacts that adding in-network computing will have on existing systems.

Adding computing and caching to the network violates the End-to-End principle central to the Internet. And the interaction with encrypted systems can limit the scope of what in-network can do to individual packet. In addition, even when programmable, every switch is still designed for (line speed) forwarding with the resulting limitations, such as lack of floating-point support for advanced algorithms and buffer size limitation. Especially in the high-performance datacenters for in-network computing to be successful, a balance between functionality, performance and cost must be found.

Hence the research areas in managed networks include but are not limited to:

Protocol design and network architecture: a lot of the current in-network work has targeted network layer optimization; however, transport protocols will influence the performance of any in-network solution. How can in-network optimization interact (or not) with transport layer optimizations?

Can the end-to-end assumptions of existing transport like TCP still be applicable in the in-network compute era? There is heritage in middlebox interactions with existing flows.

Fixed-point calculation for current application vs. of floating-point calculation for more complex operations and services: network switches typically do not support floating-point calculation. Is it necessary to introduce this capability and scale to the demand? For example, AI and ML algorithms currently mainly use floating-point calculation. If the AI algorithm is changed to fixed-point calculation, will the training stop in advance and the training result deteriorate (this needs to be done as joint work with the AI community)?

What are the gains brought by aggregation in distributed and decentralized networks? The built-in buffer of the network device is often limited, and, for example the AI and XR application parameters and caches can reach hundreds of megabytes. There is a trade-off between aggregation of the data on a single network device and its distribution across multiple nodes in terms of performance.

What is the relationship between the depth of packet inspection and not only performance but security and privacy? There is a need to find what application layer cryptography is ready to expose to other layers and even collaborating nodes; this is also related to trust in distributed networks.

Relationship between the speed of creating tables on the data plane and the performance.

5.1. Data Plane Issues in Managed Networks

Note: To be added

5.2. Interaction with Transport Protocols

Note: To be added

5.3. Interaction with Security Mechanisms

Note: To be added

5.4. Privacy aspects

Note: To be added

6. Conclusion

In-network computing as it applies to data centers is a very current and promising research area. Thus, the proposed Research Group creates an opportunity to bring together the community in establishing common goals, identify hurdles and difficulties, provide paths to new research especially in applications and linkage to other new networking research areas at the edge. More information is available in [COIN].

7. References

7.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.

7.2. Informative References

- [CHAN] Chan et al., M., "Network Support for DNN Training", 2018.
- [COIN] He, J., Chen, R., and M. Montpetit, "Computing in the Network, COIN, proposed IRTF group", 2018.
- [DANG15] Dang et al., T., "NetPaxos: Consensus at Network Speed", 2015.
- [DANG16] Dang et al., T., "Paxos Made Switch-y", 2016.
- [DRYAD] Microsoft, "DryadLinq", 2018.
- [FAN] Fan et al., B., "Small Cache, Big Effect: Provable Load Balancing for Randomly Partitioned Cluster Services", 2011.
- [FORSTER] Forster, N., "To be included", 2018.
- [GRAHAM] Graham, R., "Scalable Hierarchical Aggregation Protocol (SHArP): A Hardware Architecture for Efficient Data Reduction.", 2016.
- [HADOOP] Hadoop, "Hadoop Distributed Filesystem", 2016.
- [JIN] Jin et al., X., "NetCache: Balancing Key-Value Stores with Fast In-Network Caching", 2017.

- [JIN18] Jin et al., X., "NetChain: Scale-Free Sub-RTT Coordination", 2018.
- [LIJ] Li et al., J., "NetCache: Balancing Key-Value Stores with Fast In-Network Caching", 2017.
- [LIX] Li et al., X., "Be fast, cheap and in control with SwitchKV", 2016.
- [MEM] memcached.org, "Memcached", 2018.
- [P4] p4.org, "P4 Language", 2018.
- [PORTS] Ports, D., "Designing Distributed Systems Using Approximate Synchrony in Data Center Networks", 2015.
- [PREGEL] github.com/igrigorik/pregel, "Pregel", 2018.
- [REXFORD] Rexford, J., "Sigcomm 2018 Keynote Address", 2018.
- [RUETH] Rueth et al., J., "Towards In-Network Industrial Feedback Control", 2018.
- [SAPIO] Sapio et al., A., "In net computing is a dumb idea whose time has come", 2017.
- [SOULE] Soule, R., "Sigcomm 2018 Netcompute Workshop Keynote Address", 2018.
- [SPARK] Apache, "Spark Graph X", 2018.
- [SUBEDI] Subedi et al., T., "OpenFlow-based in-network Layer-2 adaptive multipath aggregation in data centers", 2015.
- [TENSOR] tensorflow.org, "Tensorflow", 2018.
- [TOFINO] BarefootNetworks, "Tofino", 2018.
- [VESTIN] Vestin et al., J., "FastReact: In-Network Control and Caching for Industrial Control Networks using Programmable Data Planes.", 2018.
- [ZSOLT] Zsolt et al., I., "Consensus in a Box", 2018.

Authors' Addresses

Jeffrey He
Huawei

Email: jeffrey.he@huawei.com

Aini Li (editor)
Huawei
Shenzhen, Guangdong
China

Email: liaini@huawei.com

Marie-Jose Montpetit
Triangle Video

Email: marie@mjmontpetit.com

COINRG
Internet-Draft
Intended status: Standards Track
Expires: May 5, 2021

M. McBride
Futurewei
D. Kutscher
Emden University
E. Schooler
Intel
CJ. Bernardos
UC3M
D. Lopez
Telefonica
X. de Foy
InterDigital Communications
Nov 1, 2020

Edge Data Discovery for COIN
draft-mcbride-edge-data-discovery-overview-05

Abstract

This document describes the problem of distributed data discovery in edge computing, and in particular for computing-in-the-network (COIN), which may require both the marshalling of data at the outset of a computation and the persistence of the resultant data after the computation. Although the data might originate at the network edge, as more and more distributed data is created, processed, and stored, it becomes increasingly dispersed throughout the network. There needs to be a standard way to find it. New and existing protocols will need to be developed to support distributed data discovery at the network edge and beyond.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on May 5, 2021.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
1.1. Edge Data	3
1.2. Background	4
1.3. Requirements Language	4
1.4. Terminology	4
2. Edge Data Discovery Problem Scope	5
2.1. A Cloud-Edge Continuum	5
2.2. Types of Edge Data	6
3. Edge Scenarios Requiring Data Discovery	7
4. Edge Data Discovery	7
4.1. Types of Discovery	7
4.2. Early Stage of Discovery	8
4.3. Naming the Data	8
5. Use Cases of Edge Data Discovery	10
5.1. Autonomous Vehicles	10
5.2. Video Surveillance	10
5.3. Elevator Networks	10
5.4. Service Function Chaining	11
5.5. Ubiquitous Witness	12
6. IANA Considerations	13
7. Security Considerations	13
8. Acknowledgement	14
9. Normative References	14
Authors' Addresses	15

1. Introduction

Edge computing is an architectural shift that migrates Cloud functionality (compute, storage, networking, control, data management, etc.) out of the back-end data center to be more proximate to the IoT data being generated and analyzed at the edges

of the network. Edge computing provides local compute, storage and connectivity services, often required for latency- and bandwidth-sensitive applications. Thus, Edge Computing plays a key role in verticals such as Energy, Manufacturing, Automotive, Video Surveillance, Retail, Gaming, Healthcare, Mining, Buildings and Smart Cities.

1.1. Edge Data

Edge computing is motivated at least in part by the sheer volume of data that is being created by endpoint devices (sensors, cameras, lights, vehicles, drones, wearables, etc.) at the very network edge and that flows upstream, in a direction for which the network was not originally designed. In fact, in dense IoT deployments (e.g., many video cameras are streaming high definition video), where multiple data flows collect or converge at edge nodes, data is likely to need transformation (to be transcoded, subsampled, compressed, analyzed, annotated, combined, aggregated, etc.) to fit over the next hop link, or even to fit in memory or storage. Note also that the act of performing computation on the data creates yet another new data stream! Preservation of the original data streams is needed sometimes but not always.

In addition, data may be cached, copied and/or stored at multiple locations in the network on route to its final destination. With an increasing percentage of devices connecting to the Internet being mobile, support for in-the-network caching and replication is critical for continuous data availability, not to mention efficient network and battery usage for endpoint devices.

Additionally, as mobile devices' memory/storage fill up, in an edge context they may have the ability to offload their data to other proximate devices or resources, leaving a bread crumb trail of data in their wakes. Therefore, although data might originate at edge devices, as more and more data is continuously created, processed and stored, it becomes increasingly dispersed throughout the physical world (outside of or scattered across managed local data centers), increasingly isolated in separate local edge clouds or data silos. Thus, there needs to be a standard way to find it. New and existing protocols will need to be identified/developed/enhanced for these purposes. Being able to discover distributed data at the edge or in the middle of the network will be an important component of Edge computing.

1.2. Background

Several IETF T2T RG Edge Computing discussions have been held over the last couple years. A comparative study on the definition of Edge computing was presented in multiple sessions in T2T RG in 2018 and an Edge Computing I-D was submitted early 2019. An IETF BEC (beyond edge computing) effort has been evaluating potential gaps in existing edge computing architectures. Edge Data Discovery is one potential gap that was identified and that needs evaluation and a solution. The newly proposed COIN RG highlights the need for computations in the network to be able to marshal potentially distributed input data and to handle resultant output data, i.e., its placement, storage and/or possible migration strategy.

1.3. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

1.4. Terminology

- o Edge: The edge encompasses all entities not in the back-end cloud. The device edge represents the very leaves of the network and encompasses the entities found in the last mile network. Sensors, gateways, compute nodes are included. Because the things that populate the IoT can be both physical and/or cyber, in some solutions, particularly in software-defined or digital-twin contexts, the device edge can include logical (vs physical) entities. The infrastructure edge includes equipment on the network operator side of the last mile network including cell towers, edge data centers, cable headends, POPs, etc. See Figure 1 for other possible tiers of edge clouds between the device edge and the back-end cloud data center.
- o Edge Computing: Distributed computation that is performed near the network edge, where nearness is determined by the system requirements. This includes high performance compute, storage and network equipment on either the device or infrastructure edge.
- o Edge Data Discovery: The process of finding required data from edge entities, i.e., from databases, file systems, and device memory that might be physically distributed in the network, and providing access to it logically as if it were a single unified source, perhaps through its namespace, that can be evaluated or searched.

- o ICN: Information Centric Networking. An ICN-enabled network routes data by name (vs address), caches content natively in the network, and employs data-centric security. Data discovery may require that data be associated with a name or names, a series of descriptive attributes, and/or a unique identifier.

2. Edge Data Discovery Problem Scope

Our focus is on how to define and scope the edge data discovery problem. This requires some discussion of the evolving definition of the edge as part of a cloud-to-edge continuum and in turn what is meant by edge data, as well as the meta-data about the edge data.

2.1. A Cloud-Edge Continuum

Although Edge Computing data typically originates at edge devices, there is nothing that precludes edge data from being created anywhere in the cloud-to-edge computing continuum (Figure 1). New edge data may result as a byproduct of computation being performed on the data stream anywhere along its path in the network. For example, infrastructure edges may create new edge data when multiple data streams converge upon this aggregation point and require transformation (e.g., to fit within the available resources, to smooth raw measurements to eliminate high-frequency noise, or to obfuscate data for privacy).

Initially our focus is on discovery of edge data that resides at the Device Edge and the Infrastructure Edge.

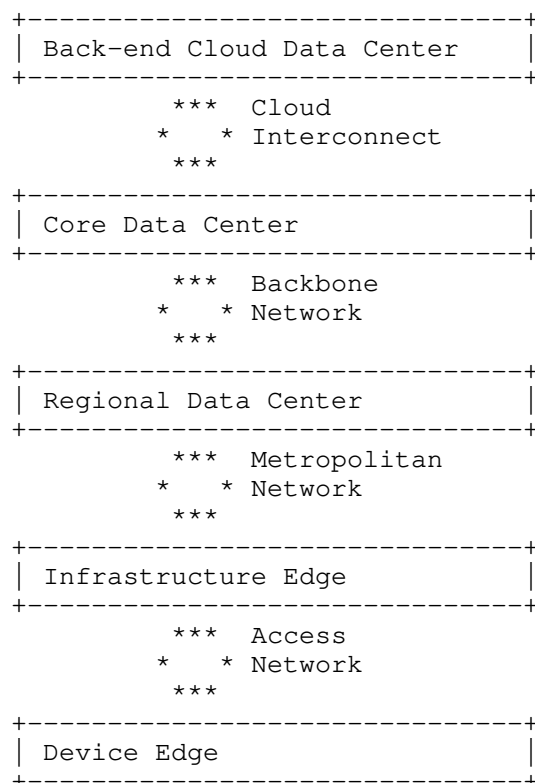


Figure 1: Cloud-to-edge computing continuum

2.2. Types of Edge Data

Besides classically constrained IoT device sensor and measurement data accumulating throughout the edge computing infrastructure, edge data may also take the form of higher frequency and higher volume streaming data (from a continuous sensor or from a camera), meta data (about the data), control data (regarding an event that was triggered), and/or an executable that embodies a function, service, or any other piece of code or algorithm. Edge data also could be created after multiple streams converge at an edge node and are processed, transformed, or aggregated together in some manner.

Regardless of edge data type, a key problem in the Cloud-Edge continuum is that data is often kept in silos. Meaning, data is often sequestered within the Edge where it was created. A goal of this discussion is to consider the prospect that different types of edge data will be made accessible across disparate edges, for example to enable richer multi-modal analytics. But this will happen only if

data can be described, searched and discovered across heterogeneous edges in a standard way. Having a mechanism to enable granular edge data discovery is the problem that needs solving either with existing or new protocols. The mechanisms shouldn't care to which flavor cloud or edge the request for data discovery is made.

3. Edge Scenarios Requiring Data Discovery

1. A set of data resources appears (e.g., a mobile node hosting data joins a network) and they want to be discoverable by an existing but possibly virtualized and/or ephemeral data directory infrastructure.
2. A device wants to discover data resources available at or near its current location. As some of these resources may be mobile, the available set of edge data may vary over time.
3. A device wants to discover to where best in the edge infrastructure to opportunistically upload its data, for example if a mobile device wants to offload its data to the infrastructure (for greater data availability, battery savings, etc.).

4. Edge Data Discovery

How can we discover data on the edge and make use of it? There are proprietary implementations that collect data from various databases and consolidate it for evaluation. We need a standard protocol set for doing this data discovery, on the device or infrastructure edge, in order to meet the requirements of many use cases. We will have terabytes of data on the edge and need a way to identify its existence and find the desired data. A user requires the need to search for specific data in a data set and evaluate it using their own tools. The tools are outside the scope of this document, but the discovery of that data is in scope.

4.1. Types of Discovery

There are many aspects of discovery and many different protocols that address each aspect.

Discovery of new devices added to an environment. Discovery of their capabilities/services in client/server environments. Discovery of these new devices automatically. Discovering a device and then synchronizing the device inventory and configuration for edge services. There are many existing protocols to help in this discovery: UPnP, mDNS, DNS-SD, SSDP, NFC, XMPP, W3C network service discovery, etc.

Edge devices discover each other in a standard way. We can use DHCP, SNMP, SMS, COAP, LLDP, and routing protocols such as OSPF for devices to discover one another.

Discovery of link state and traffic engineering data/services by external devices. BGP-LS is one such solution.

The question is if one or more of these protocols might be a suitable contender to extend to support edge data discovery?

4.2. Early Stage of Discovery

The different types of discovery may involve mobile devices, which can be the source, or target, of discovery operations. Mobile devices may have an influence on discovery in COIN, and early stage discovery may be necessary in some scenarios.

In many cases (e.g. crowds, drones or vehicular scenarios), multiple networks, or attachment points, are available to a mobile device. This type of device needs to efficiently select among multiple interfaces, or multiple attachment points, which one(s) to use for discovery. An early discovery stage should provide enough information to perform such a selection and therefore reduce power consumption, service latency, and impact on network usage.

To select among (already attached) multiple interfaces, we can leverage provisioning domains, router advertisements, DHCP, etc. to convey information about service or data. To select among multiple attachment points, pre-attachment discovery (e.g. 802.11aq, or obtaining provisioning domains through a control plane) or a discovery protocol over a control plane (e.g. as described in 3GPP edge computing) can be used.

What are suitable protocols to extend to support this early stage of discovery? There is also a tradeoff between the amount of exposed information and the limited resources available at this early stage. Trust and privacy are also important early stage discovery factors.

4.3. Naming the Data

Information-Centric Networking (ICN) RFC 7927 [RFC7927] is a class of architectures and protocols that provide "access to named data" as a first-order network service. Instead of host-to-host communication as in IP networks, ICNs often use location-independent names to identify data objects, and the network provides the services of processing (answering) requests for named data with the objective to finally deliver the requested data objects to a requesting consumer.

Such an approach has profound effects on various aspects of a networking system, including security (by enabling object-based security on a message/packet level), forwarding behavior (name-based forwarding, caching), but also on more operational aspects such as bootstrapping, discovery etc.

The CCNx and NDN (<https://named-data.net>) variants of ICN are based on a request/response abstraction where consumers (hosts, application requesting named data) send INTEREST messages into the network that are forwarded by network elements to a destination that can provide the requested named data object. Corresponding responses are sent as so-called DATA messages that follow the reverse INTEREST path.

Each unique data object is named unambiguously in a hierarchical naming scheme and is authenticated through Public-Key cryptography (data objects can also optionally be encrypted in different ways). The naming concept and the object-based security approach lay the foundation for location-independent operation. The network can generally operate without any notion of location, and nodes (consumers, forwarders) can forward requests for named data objects directly, i.e., without any additional address resolution. Location independence also enables additional features, for example the possibility to replicate and cache named data objects. On-path caching is a standard feature in many ICN systems -- typically for enhancing reliability and performance.

In CCNx and NDN, forwarders are stateful, i.e., they keep track of forwarded INTEREST to later match the received DATA messages. Stateful forwarding (in conjunction with the general named-based and location-independent operation) also empowers forwarders to execute individual forwarding strategies and perform optimizations such as in-network retransmissions, multicasting requests (in cases there are several opportunities for accessing a particular named data object) etc.

Naming data and application-specific naming conventions are naturally important aspects in ICN. It is common that applications define their own naming convention (i.e., semantics of elements in the name hierarchy). Such names can often be directly derived from application requirements, for example a name like /my-home/living-room/light/switch/main could be relevant in a smart home setting, and corresponding devices and applications could use a corresponding convention to facilitate controllers finding sensors and actors in such a system with minimal user configuration.

The aforementioned features make ICN amenable to data discovery. Because there is no name/address chasm as in IP-based systems, data can be discovered by sending an INTEREST to named data objects

directly (assuming a naming convention as described above). Moreover, ICN can authenticate received data objects directly, for example using local trust anchors in the network (for example in a home network).

Advanced ICN features for data discovery include the concept of manifests in CCNx, i.e., ICN objects that describe data collections, and data set synchronization protocols in NDN (<https://named-data.net/publications/li2018sync-intro/>) that can inform consumers about the availability of new data in a tree-based data structure (with automatic retrieval and authentication). Also, ICN is not limited to accessing static data. Frameworks such as Named Function Networking (<http://www.named-function.net>) and RICE can provide the general ICN feature for discovery not only for data but also for name functions (for in-network computing) and for their results.

5. Use Cases of Edge Data Discovery

5.1. Autonomous Vehicles

Autonomous vehicles rely on the processing of huge amounts of complex data in real-time for fast and accurate decisions. These vehicles will rely on high performance compute, storage and network resources to process the volumes of data they produce in a low latency way. Various systems will need a standard way to discover the pertinent data for decision making.

5.2. Video Surveillance

The majority of the video surveillance footage will remain at the edge infrastructure (not sent to the cloud data center). This footage is coming from vehicles, factories, hotels, universities, farms, etc. Much of the video footage will not be interesting to those evaluating the data. A mechanism, perhaps a set of protocols, is needed to identify the interesting data at the edge. What constitutes interesting will be context specific, e.g., a video frame might be considered interesting if and only if it includes a car, or person, or bicyclist, or a backyard nocturnal creature, or etc. Interesting video data may be stored longer in storage systems at the very edge of the network and/or in networking equipment further away from the device edge that has access to data in flight as it transits the network.

5.3. Elevator Networks

Elevators are one of many industrial applications of edge computing. Edge equipment receives data from hundreds of elevator sensors. The data coming into the edge equipment is vibration, temperature, speed,

level, video, etc. We need the ability to identify where the data we need to evaluate is located.

5.4. Service Function Chaining

Service function chaining (SFC) allows the instantiation of an ordered set of service functions (SFs) and the subsequent "steering" of traffic through them. Service functions are expected to be deployed at the edge of the network, as a feasible deployment of "Compute In the Network", with multiple types of potential use cases (e.g., fog robotics, Industry 4.0 automation, etc). Service functions provide a specific treatment of received packets, therefore they need to be discoverable so they can be used in a given service composition via SFC. In addition, these functions can be producers and/or consumers of data. So far, how the functions are discovered and composed has been out of the scope of discussions in the IETF. While there are some mechanisms that can be used and/or extended to provide this functionality, more work needs to be done. An example of this can be found in [I-D.bernardos-sfc-discovery].

In an SFC environment deployed at the edge, the discovery protocol may also need the following kind of meta-data information per (service) function:

- o Service Function Type: identifying the category of function provided.
- o SFC-aware: Yes/No. Indicates if the function is SFC-aware.
- o Route Distinguisher (RD): IP address indicating the location of the function.
- o Pricing/costs details.
- o Migration capabilities of the function: whether a given function can be moved to another provider (potentially including information about compatible providers topologically close).
- o Mobility of the device hosting the function, with e.g. the following sub-options:
 - Level: no, low, high; or a corresponding scale (e.g., 1 to 10).
 - Current geographical area (e.g., GPS coordinates, post code).
 - Target moving area (e.g., GPS coordinates, post code).

- o Power source of the device hosting the function, with e.g. the following sub-options:

Battery: Yes/No. If Yes, the following sub-options could be defined:

Capacity of the battery (e.g., mmWh).

Charge status (e.g., %).

Lifetime (e.g., minutes).

Discovery of resources in an NFV environment: virtualized resources do not need to be limited to those available in traditional data centers, where the infrastructure is stable, static, typically homogeneous and managed by a single admin entity. Computational capabilities are becoming more and more ubiquitous, with terminal devices getting extremely powerful, as well as other types of devices that are close to the end users at the edge (e.g., vehicular onboard devices for infotainment, micro data centers deployed at the edge, etc.). It is envisioned that these devices would be able to offer storage, computing and networking resources to nearby network infrastructure, devices and things (the fog paradigm). These resources can be used to host functions, for example to offload/complement other resources available at traditional data centers, but also to reduce the end-to-end latency or to provide access to specialized information (e.g., context available at the edge) or hardware. Similar to the discovery of functions, while there are mechanisms that can be reused/extended, there is no complete solution yet defined. An example of work in this area is [I-D.bernardos-intarea-vim-discovery]. The availability of this meta-data about the capabilities of nearby physical as well as virtualized resources can be made discoverable through edge data discovery mechanisms.

5.5. Ubiquitous Witness

Ubiquitous Witness (UW) is the name of a use case that has been presented in past COINRG and ICNRRG meetings at the IETF. It describes what might occur in dense IoT deployments when an anomaly occurs. There are many "witnesses" to report on what happened within a limited region of interest and around an approximate point in time. The use case highlights the need for upstream data discovery and management. It is agnostic to where the dense IoT deployment resides, whether in a factory, home, commercial building, city, entertainment venue, et cetera. For example, as cameras and other sensors have become ubiquitous in Smart Cities, it would be helpful to be able to discover and examine data from all devices and sensors

that witnessed an accident in a city intersection; this could be data from cameras mounted at the intersection itself, on nearby buildings, in cars, and cell phones of individuals on location.

If an anomaly were to automatically trigger independent upstream flows of video data from all of the witnesses (within a proximal vicinity and time window), the data flows would naturally converge at shared collection or aggregation points in the network. These edge nodes might opt to vault any data deemed part of a safety-related anomaly, which would enable interested parties (the car owner, the car manufacturer, an insurance company, a city traffic planner) to investigate the root cause of the anomaly after the fact. The implication however is that enough meta data has been generated alongside the data itself (e.g., a data name, an identifier, or a geo location and timestamp), to allow the retrieval of this distributed data, provided those asking have proper authorization to access it.

The UW streams are contextually-related and as such it can be advantageous also to be able to process them simultaneously, at the time they are first generated. For example if collection nodes could derive that groups of data streams are contextually-related, they could stitch streams together to create a 360-degree view of the anomalous event (e.g., to walk around in the data), or to winnow the set of vaulted data to only the "best" video (e.g., highest resolution, unoccluded views) or to perform compute-in-the-network to enable them to fit within the available resources (e.g., at the receiving node due to the convergence or implosion of upstream data, or over the next hoplink). Ubiquitous Witness data doesn't have to be video data, but video illustrates why one might want to jointly process upstream flows in real-time.

6. IANA Considerations

N/A

7. Security Considerations

Security considerations will be a critical component of edge data discovery particularly as intelligence is moved to the extreme edge where data is to be extracted.

An assumption is that all data will have associated policies (default, inherited or configured) that describe access control permissions. Consequently, the discoverability of data will be a function of who or what has requested access. In other words, the discoverable view into the available data will be limited to those who are authorized. Discovering edge data that is exclusively private is out of scope of this document, the assumption being that

there will be some edge clouds that do not expose or publish the availability of their data. Although edge data may be sent to the back-end cloud as needed, there is nothing that precludes it from being discoverable if the cloud offers it as public.

A trust relationship may be needed between the source and target of a discovery operation to avoid denial of service attacks from a malicious source or target of the operation. And discovery information, which is exposed by a node or network, may need to be protected for privacy purposes, e.g. not leak information in the presence of a certain type of data in a network.

8. Acknowledgement

The authors thank Dave Oran, Greg Skinner and Lixia Zhang for contributing to this document.

9. Normative References

[I-D.bernardos-intarea-vim-discovery]

Bernardos, C. and A. Mourad, "IPv6-based discovery and association of Virtualization Infrastructure Manager (VIM) and Network Function Virtualization Orchestrator (NFVO)", draft-bernardos-intarea-vim-discovery-04 (work in progress), September 2020.

[I-D.bernardos-sfc-discovery]

Bernardos, C. and A. Mourad, "Service Function discovery in fog environments", draft-bernardos-sfc-discovery-05 (work in progress), September 2020.

[I-D.irtf-icnrg-ccnxmessages]

Mosko, M., Solis, I., and C. Wood, "CCNx Messages in TLV Format", draft-irtf-icnrg-ccnxmessages-09 (work in progress), January 2019.

[I-D.irtf-icnrg-ccnxsemantics]

Mosko, M., Solis, I., and C. Wood, "CCNx Semantics", draft-irtf-icnrg-ccnxsemantics-10 (work in progress), January 2019.

[I-D.kutscher-icnrg-rice]

Krol, M., Habak, K., Oran, D., Kutscher, D., and I. Psaras, "Remote Method Invocation in ICN", draft-kutscher-icnrg-rice-00 (work in progress), October 2018.

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC7927] Kutscher, D., Ed., Eum, S., Pentikousis, K., Psaras, I., Corujo, D., Saucez, D., Schmidt, T., and M. Waehlich, "Information-Centric Networking (ICN) Research Challenges", RFC 7927, DOI 10.17487/RFC7927, July 2016, <<https://www.rfc-editor.org/info/rfc7927>>.

Authors' Addresses

Mike McBride
Futurewei

Email: michael.mcbride@futurewei.com

Dirk Kutscher
Emden University

Email: ietf@dkutscher.net

Eve Schooler
Intel

Email: eve.m.schooler@intel.com
URI: <http://www.eveschooler.com>

Carlos J. Bernardos
Universidad Carlos III de Madrid
Av. Universidad, 30
Leganes, Madrid 28911
Spain

Phone: +34 91624 6236
Email: cjbc@it.uc3m.es
URI: <http://www.it.uc3m.es/cjbc/>

Diego R. Lopez
Telefonica

Email: diego.r.lopez@telefonica.com
URI: <https://www.linkedin.com/in/dr2lopez/>

Xavier de Foy
InterDigital Communications, LLC
1000 Sherbrooke West
Montreal
Canada

Email: Xavier.Defoy@InterDigital.com

COIN
Internet-Draft
Intended status: Informational
Expires: January 9, 2020

M. Montpetit
Triangle Video
July 8, 2019

In Network Computing Enablers for Extended Reality
draft-montpetit-coin-xr-03

Abstract

Augmented Reality (AR) and Virtual Reality (VR), combined as Extended Reality or XR, challenge networking technologies and protocols because they combine the features of fast information display, image processing, computing and forwarding. This document presents some of these challenges and how adding computing in the network could respond to them.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 9, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
1.1. Requirements Language	3
2. Definitions	3
3. Extended Reality and In-Network Computing	4
3.1. XR Network Requirements	4
3.2. In-Network Computing Advantages in XR	5
4. XR in Data Intensive Services and Applications	6
5. Enabling Technologies	6
5.1. Information Centric Networking (ICN) and Named Data Networking (NDN)	7
5.2. Network Coding	7
5.3. Blockchains and Distributed Trust	8
6. Conclusion	9
7. Acknowledgements	9
8. References	9
8.1. Normative References	9
8.2. Informative References	9
Author's Address	10

1. Introduction

Virtual Reality (VR) and Augmented Reality (AR) taken together as Extended Reality or XR are at the center of a number of technological advances in many different fields, including not only gaming and entertainment but immersive journalism, remote diagnosis and maintenance, telemedicine, manufacturing and assembly and smart cities.

But with the emergence of the edge and the programmability of network elements all the way from the data center to the users the possibility of creating networked, multiparty/multisource and interacting XR comes closer to reality. This document wants to review what is necessary for the current localized and cloud enhanced XR to evolve to a more distributed and edge centric architecture to support advanced immersive application and services. It assumes that network programmability will enable to tailor the network to the XR requirements. This document is about requirements not solutions per se but will mention work that has already been done towards a more networked XR including Information Centric architectures, Artificial Intelligence and in network coding. The networked functionality should enable to supplement local XR services and devices while keeping the very low latency and the very high data rates that are required by XR.

This document is intended as informative to both the networking and application research community. It does not address a specific network layer or protocol but provides architecture and system level specifications and guidelines. For example:

Latency: the physical distance between the XR content cloud of AR/VR and users are short enough to limit the propagation delay to the 20 ms usually cited for XR applications mixed for example with IoT devices and sensors delay reduction for range of interest (RoI) detection.

Applications: better transcoding and use of compression algorithms, pre-fetching and pre-caching and movement prediction.

Monitoring: implement management and metrics gathering closer to where the XR services are consumed.

Network access: push some networking functions in the kernel space into the user space to enable the deployment of stream specific algorithms for congestion control and application-based load balancing based on machine learning and user data patterns.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Definitions

- 360-degree video: 360-degree videos, also known as immersive videos or spherical videos, are video recordings where a view in every direction is recorded at the same time using an omnidirectional camera or a collection of cameras. 360o video is outside the scope of this document.

- AR: Augmented Reality (AR) is a live direct or indirect view of a physical, real-world environment whose elements are augmented by computer-generated input such as sound, video, location or graphical data. It is related to a more general concept called mediated reality [MEDIA], in which reality is modified (diminished or augmented) by computer-generated imagery.

- VR Virtual Reality (VR): uses software-generated realistic imaging, sounds and other sensor inputs to replicate a real or imaginary setting, to simulates a user's physical presence in this environment and provide an immersive experience that enable the user to interact with objects and move within this space.

- XR: extended reality is used to address both AR and VR together.

3. Extended Reality and In-Network Computing

XR is an example of the Multisource Multidestination Problem that combines video, haptics and tactile experiences as well, in interactive or networked mode multiparty and social interactions. Thus, XR is difficult to deliver with a client-server cloud-based solution as it requires a combination of stream synchronization, low delay and delay variations as mentioned above as well as means to cover from losses and provide optimized caching in the cloud and rendering as close as possible to the user at the network edge.

3.1. XR Network Requirements

To deliver the XR experience, there is a need to achieve complete 6 degrees of freedom meaning the 3 axes for body movement (x,y,z) plus pitch, yaw, rotation of the head all of which must be fulfilled in real time. Low delay, low loss and low delay variation are needed to avoid sea sickness symptoms if the image does not follow the movement [CABLE].

But this is not the only difficulty, as there is also the need to provide real-time interactivity for immersive sports, mobile immersive applications with tactile and time-sensitive data and high bandwidth for high resolution images. Since XR deals with personal information and potentially protected content (in entertainment and gaming) XR must also provide a secure environment and ensure user privacy. And of course, the sheer amount data needed for and generated by the XR applications will use recent trend analysis and mechanisms, including machine learning to find these trends and reduce the size of the data sets [INTER].

Shared and global immersive experiences require interconnected, distributed and federated XR nodes. The requirements can be summarized as:

- Allow joint collaboration in VR.
- Provide multi-view AR.
- Add extra streams (IoT) to AR and VR experiences across data intensive services, manufacturing and industrial processes.
- Provide "Social Television" experiences and global viewing and experience rooms.
- Enable multistream, multidevice, multidestination applications.

- Use new Internet Architectures at the edge for improved performance and performance management.
- Integrate with holography, 3D displays and image processing systems [CABLE].

3.2. In-Network Computing Advantages in XR

One aspect of the push of XR to the edge is of course to take advantage of the cloud/edge continuum and functional decomposition to provide lower latency services and enable the development of new applications. While this is very promising the question of the localization of the networking resources in order to provide the service becomes an essential component of the overall architecture. But it is not only finding the best geographical location but also providing the right level of reliability when one or more location is not available especially for mission critical services in medicine or manufacturing. And it does not mean only data laid distribution but also ensuring the availability of the right computational capabilities. The optimization of the location and type of the required resources for the multisource, multidestination, mutiparty, multi-input XR applications can use AI and ML, and advanced load balancing and distributed network principles. There is a need for more research in such resource allocation problems at the edge to enable autonomous node operation and quality of experience [SOL]. These are of course multi-variate and heterogeneous goal optimization problems requiring advanced analysis with fast converging algorithms [MULTI][PACKET]. This is essential for the federation of nodes to provide the required experience.

Of course, image rendering and video processing in XR leverages different HW capabilities combinations of CPU and GPU. Current programmable network entities need to be evaluated to see if they can be sufficient to provide the speed required to provide real-time rendering and execute complex analytics: P4 for example does not support the floating-point operations necessary for advanced graphics.

Finally, dynamic network programmability could enable the use of joint learning algorithms across both data center, edge computers and goggles or glasses to allocate functionality and the creation of semi permanent datasets and analytics for usage trending. In the end, the use of computing or networked XR will enable the allocation of control, forwarding and storage resources and related usage models when needed by the application. This may mean re-evaluating the distribution of functionalities between datacenter and edge with less critical elements rendered in the cloud combined with a better understanding of the operational decomposition of the XR experience

to allow the use of novel data structures, three-dimensional modeling and image processing algorithms.

Other advantages of adding computing to networked XR include:

- Multicast distribution and processing as well as peer to peer distribution in bandwidth and capacity constrained environments.
- Evaluation of local caching and micro datacenters with local or cloud-based pre-rendering.
- Trend or ML based congestion control to manage XR sessions quality of service.
- Higher layer protocols optimization to reduce latency especially in data intensive applications at the edge.
- Trust, including blockchains and smart-contracts to enable secure community building across domains.
- Support for nomadicity and mobility (link to mobile edge).
- Use of 5G slicing to create independent session-driven processing/rendering.
- Performance optimization by tunneling, session virtualization and loss protection.

4. XR in Data Intensive Services and Applications

In-network computing is essential for data reduction and multistream low latency services at the edge where moving the data to the cloud is either requiring too much bandwidth or adding unacceptable latency. Examples of these services included industrial processes monitoring, AR-aided design and fabrication and AR/VR supported medical interventions.

5. Enabling Technologies

This section presents some salient research that will lead to in-network computing becoming a major enabler of networked XR.

NOTE: more information and added sub-sections will be added in future versions of the draft with the collaboration of co-authors in the specific research areas.

5.1. Information Centric Networking (ICN) and Named Data Networking (NDN)

The Named Data Networking (NDN) architecture, one architecture of ICN, is particularly well suited for the multisource multi-destination architecture of XR because it allows to create the content experiences based on their components names not a location or pointer to a location hence provides a natural functional decomposition. ICN allows content delivery to evolve from single, context-independent streams to context-dependent Information components that can adapt dynamically to the changes necessary to maintain the immersive nature of the experience and be delivered efficiently. The combination of interest messages to signal what content is needed combined with the data responses help to coordinate the different streams and multiple users (pull mechanisms). The ICE-AR [ICE] project already mentions a concept of acceleration as a service: the exploration of the design and the usage of computation at the edge including the wireless edge.

For XR, ICN also allows to develop robust and resilient networking while allowing application developer to continue using known programming model [RICE]. This is important for the XR developers community that come from the entertainment, gaming or other non networking specific industries and could enable ICN and XR to coexist in user devices (the ultimate edge). NDN concepts are already integrated to distributed video distribution with trust mechanisms (see section below) such as smart contracts on the blockchain to proof of origin and destination sent along with interest messages [HUITX].

5.2. Network Coding

Networked XR requires the synchronization of multiple streams but with its delay sensitivity the use of buffering schemes to achieve this synchronization is impractical. At the same time the need to maintain high data integrity means that packet losses also need to be limited. Network coding has proven very useful to achieve both these goals in commercial streaming services like Netflix, is being added to protocols like QUIC, multi-stream services such as Social Television [SOCIAL] and other data-centric low latency applications. This avoids being reliant on complex synchronization algorithms. The many XR services are constrained in latency and loss budgets especially in mission critical applications hence even the delay due to encoding and decoding operations needs to be minimized. Hence the idea of in-network coding and re-encoding to adapt to dynamic network conditions, not just end to end, can be used to ensure on time packet delivery with loss recovery. In network encoding needs the type of programmability that COIN provides and the development of

programmable switches and the P4 language may allow to create very efficient in-network coding even considering the limitations of the language Note: references to be added.

5.3. Blockchains and Distributed Trust

If XR is to be integrated at the edge of the network to provide the required delay and loss guarantees, then relying on centralized mechanisms for trust is non-realistic. Traditional centralized mechanisms to discover and admit nodes to the network, to provide access right and name resolution need to be updated to be used in the dynamic XR environment. Blockchain technology, with operations performed at the edge and in a decentralized way is fast becoming a major scalable means of providing trust and validate provenance in a large number of applications including those on the XR portfolio.

Smart contracts (on the blockchain) supply a mechanism to provide the trust and validation for XR edge nodes. A new XR participant node is admitted after it has committed to a smart contract that contains the rules and mechanisms to distribute content via this node in a trusted and secure way. This constitutes its proof of validity. After a node is admitted, it will can then provisioned with the appropriate software to become fully operational to provide the XR experience. Newly admitted nodes will be inserted in the general ledger on the blockchain enabling other nodes to discover them, and hence, to form a trusted network. A name resolution authority can also be provided by the blockchain to manage and validate the origin of the content, the proof of origin, and to provide the ability to search such content. The proof of origin can also be used to prevent some content from reaching one or more nodes and implement content filtering based on trusted authorities. This is useful not only for content packets but also for packets capable of modifying the node operations. Finally, when some content reaches a specific destination, it can be verified against the content rules of the reached node even and before it is sent to the application; this allows to provide a proof of delivery for the content and enable to generate statistics, performance metrics and enable the nodes to adapt to the XR requirements.

All of the above assumes that the nodes can implement the functions needed by the blockchain hence once again infers that there is enough computing power in the nodes to perform these operations. At this point both proof of concept and proof of every are limited due to the added overhead and the size of the blockchain. As distributed blockchains and COIN continue to evolve this should continue to be a field of interest for the development of secure and private XR experiences.

6. Conclusion

More and more applications and service are being developed and deployed that use or will use combinations of AR and VR, XR, along with extra stream from sensors and IoT devices. And many of these applications require to be deployed over a network because of their interactive or multiparty nature. In that context, it not uniquely necessary to move functionality to the network but to carefully evaluate which elements to locate in network nodes, where these nodes are and what computational support they need to support the XR experience. Hence, it is believed that a great enabler of networked XR is the capability to co-locate programmable elements in the XR network node to respond to the dynamics of the services in an efficient, resilient and secure manner.

7. Acknowledgements

The author would like to thank Jeffrey He, Dirk Kutscher, Cedric Westphal and Weiguang Wang for their contributions to the presentation that lead to this draft.

8. References

8.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.

8.2. Informative References

- [CABLE] Hinds, A., "The Near Future of Immersive Experiences: Where We Are on the Journey, What Lies Ahead, and What It Takes to Get There.", SIGCOMM 2018 Workshop on AR/VR. <http://conferences.sigcomm.org/sigcomm/2018/workshop-arvr.html>, August 2018.
- [HUITX] "8X: ICN Based Video Distribution.", 8XLab Web Site. <https://www.8xlabs.com>, 2018.
- [ICE] Burke, J., "ICN-Enabled Secure Edge Networking with Augmented Reality: ICE-AR.", ICE-AR Presentation at NDNCOM. <https://www.nist.gov/news-events/events/2018/09/named-data-networking-community-meeting-2018>, September 2018.

- [INTER] Bastug, E., "Towards Interconnected Virtual Reality: Opportunities, Challenges and Enablers.", IEEE Communications Magazine, Volume 55 , Issue: 6. <https://arxiv.org/pdf/1611.05356.pdf>, June 2017.
- [MEDIA] Wikipedia.org, "Mediated Reality.", 2018, <https://en.wikipedia.org/wiki/Computer-mediated_reality>.
- [MULTI] Batalla, J., "Evolutionary Multi-objective optimization algorithm for multimedia delivery in critical applications through Content-Aware Networks.", The Journal of Supercomputing. <https://link.springer.com/article/10.1007/s11227-016-1731-x>, March 2017.
- [PACKET] Jeyakumar, V., "Millions of Little Minions: Using Packets for Low Latency Network Programming and Visibility.", Proceedings of SIGCOMM 2014. <http://conferences.sigcomm.org/sigcomm/2014/program.php>, August 2014.
- [RICE] Krol, M., "RICE: Remote Method Invocation in ICN.", Proceedings of the ACM Conference on Information-Centric Networking 2018. <http://conferences.sigcomm.org/acm-icn/2018/proceedings/icn18-final9.pdf>, September 2018.
- [SOCIAL] Montpetit, M. and M. Medard, "Social Television: Enabling Technologies and Architectures.", Proceedings of the IEEE, Volume 100, pp. 1395-1399. <http://proceedingsoftheieee.ieee.org>, May 2012.
- [SOL] Heorhiadi, V., "Simplifying Software-Defined Network Optimization Using SOL.", 13th USENIX Symposium on Networked Systems Design and Implementation. <https://www.usenix.org/system/files/conference/nsdi16/nsdi16-paper-heorhiadi.pdf>, March 2016.
- [VRSICK] LaViola, J., "SA Discussion of Cybersickness in Virtual Environments.", ACM SIGCHI Bulletin 32(1):47-56. <http://www.eecs.ucf.edu/~jjl/pubs/cybersick.pdf>, January 2000.

Author's Address

Marie-Jose
Triangle Video

Email: marie@mjmontpetit.com