COINRG
Internet-Draft
Intended status: Standards Track
Expires: January 14, 2021

M. McBride
Futurewei
D. Kutscher
Emden University
E. Schooler
Intel
CJ. Bernardos
UC3M
D. Lopez
Telefonica

July 13, 2020

Edge Data Discovery for COIN
draft-mcbride-edge-data-discovery-overview-04

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 January 14, 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 3
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. Naming the Data 8
5. Use Cases of Edge Data Discovery 9
5.1. Autonomous Vehicles 9
5.2. Video Surveillance 10
5.3. Elevator Networks 10
5.4. Service Function Chaining 10
5.5. Ubiquitous Witness 12
6. IANA Considerations 13
7. Security Considerations 13
8. Acknowledgement 13
9. Normative References 13
Authors' Addresses 14

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

- 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.

- 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.

- 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.

- 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.

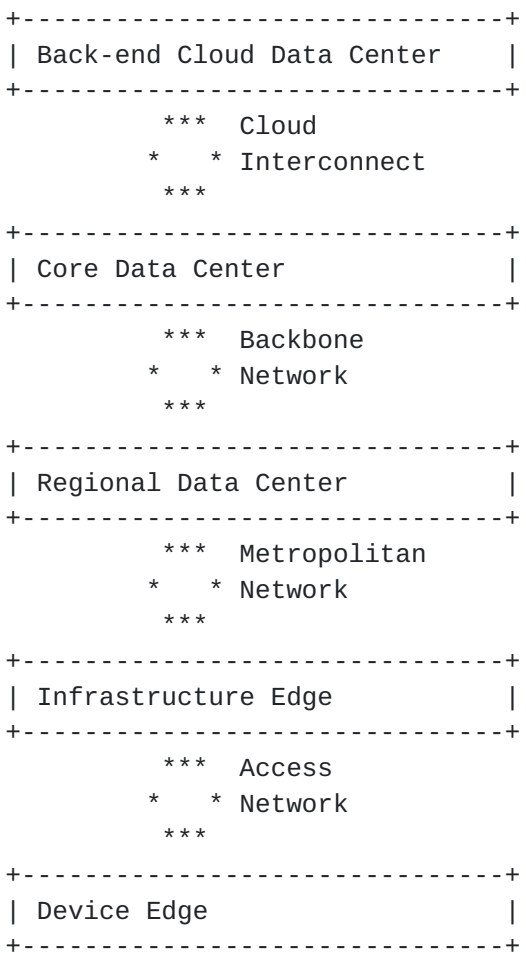


Figure 1: Cloud-to-edge computing continuum

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. 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 evalute 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:

- Service Function Type: identifying the category of function provided.

- SFC-aware: Yes/No. Indicates if the function is SFC-aware.
- Route Distinguisher (RD): IP address indicating the location of the function.
- Pricing/costs details.
- Migration capabilities of the function: whether a given function can be moved to another provider (potentially including information about compatible providers topologically close).
- 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).
- 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 ICNRG 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.

8. Acknowledgement

The authors thank Dave Oran for his detailed feedback on an early version of this draft, as well as inputs from Greg Skinner and Lixia Zhang.

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-03 (work in progress), February 2020.

[I-D.bernardos-sfc-discovery]
Bernardos, C. and A. Mourad, "Service Function discovery in fog environments", draft-bernardos-sfc-discovery-04 (work in progress), March 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. Waehlisch, "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/