Mobile Data Repositories at the Edge

Ioannis Psaras, Onur Ascigil,
Sergi Reñé, George Pavlou
UCL, UK
{*i.psaras,o.ascigil,s.rene,g.pavlou*}*@ucl.ac.uk*

Alex Afanasyev
FIU, USA
aa@cs.fiu.edu

Lixia Zhang
UCLA, USA
lixia@cs.ucla.edu

Abstract

We contend that in a future IoT-dominated environment the majority of data will be produced at the edge, which may need to flow toward the network core. This reverses today's *"core-to-edge"* data flow to an *"edge-to-core"* model and puts severe stress on edge access/cellular links. In this paper, we propose a data-centric communication approach which treats storage and wire the same as far as their ability of supplying the requested data. Because storage is cheaper and scales easier than wires, we argue for enhancing network connectivity with local storage services (e.g., in WiFi Access Points, or similar) at the edge of the network. Such local storage services can be used to temporarily store IoT and user-generated data at the edge, prior to data-cloud synchronization.

1 Introduction

It is being continuously claimed that in the very near future the majority of mobile devices (from wearables to every sensor on automotive) will be connected to the Internet. What is not often discussed is that these mobile devices and sensors will be constantly producing enormous amounts of data. For instance, Internet of Things (IoT) global data is forecasted to exceed 1.6 zettabytes by 2020 [1, 2], In the case of surveillance cameras or car sensors, a constant stream of data is produced from each device. Also, user generated content (e.g., real-time video streaming from user devices to social network applications) will stress the network further and might cause a big *data explosion* that the access network is not able to absorb.

We are therefore starting to see a reverse data-flow forming, according to which, data is produced at the edge and flows towards the core of the network to be stored or processed. This is in stark contrast to today's model, where we largely assume that data resides at the core of the network (in some data-centre or CDN server farm) and flows towards the edge (to users' devices).

The question then becomes, what would be the best way to handle all this data? To whom are they of interest? Consider the biker's helmet-camera or the car's camera that is constantly recording everything as the vehicle is moving around. This data is primarily of interest to insurance companies in case of a collision/accident, but of little use otherwise. According to the current Internet infrastructure, there are a few options of what one can do with such data produced at the edge of the network: i) assume that the helmet or car can apply image-processing functions onboard, and data is transmitted to the insurance company only when a collision is detected after processing the data, or ii) transmit all data through the cell network to the backend-cloud for storage and processing. The first option would require significant amount of processing power on the helmet camera or the car itself, which would in turn increase significantly the price of these devices. The second option, which is straightforward to implement using the existing TCP/IP protocol stack that offers point-to-point connectivity, presents several challenges:

- Cell network would be brought to its knees. Despite increasing capacities of cell towers and last-mile links, it is highly-unlikely that the mobile backhaul network will reach the capacity of broadband connections any time soon. That is, cellular networks may not have the capacity to transfer all this data.

- The current (Internet Service Provider) ISP-relationship business model would be turned on its head. Edge/Eyeball ISPs business is traffic download. However, in case of orders of magnitude more upload traffic produced at the edge, ISPs will have to upgrade their network accordingly. This may pose a tremendous challenge, as it could be a show-stopper for IoT as a whole: the increased costs for edge/eyeball ISPs would push them to increase their charges/subscription costs to end-users and IoT application providers, making it more expensive to actually use the network.

- Mobility has always been a challenge in IP networks [3, 4]. User mobility (both client- and producer-/server-mobility) has traditionally presented a challenge for the IP network. When users move and therefore disconnect from their point of connectivity, the session is temporarily broken until the user connects to the next access point. The session-based, synchronous mode of communication supported by IP is unfit for

purpose called for asynchronous data services needed by edge-produced data.

In an environment where asynchronous data-services (as opposed to synchronous telephony services) dominate mobile communications, a plain connectivity between two end points –as provided today by (Wireless)ISPs (wISP)– is not the end, but only the means to an end. The end-goal in data-intensive environments is access to the information contained in those data. The emerging IoT and edge-computing era requires a data-centric model of communication which enables one to best engineer a network to directly meet the end goal of applications.

We, therefore, argue for enhancing network connectivity with local storage services (e.g., in WiFi Access Points, or similar) at the edge of the network, which can be used to temporarily store IoT and user-generated data, prior to data-cloud synchronization.

We envision that wISPs would be motivated to offer edge-network, local (but mobile) storage allowance, together with connectivity service to mobile users. In our vision, storage allowance would be moving together with the user. That is, the user is able to make use of local storage resources (in base stations, wifi access points or even other users' devices) as he moves along to temporarily store IoT-generated data.

Edge storage in WiFi APs forms an ambient edge-cloud where data can be processed, temporarily stored and/or synchronized with the back-end cloud, *only when necessary and following the best strategy depending on the data/application requirements*. That said, edge network functions can better control when to upload the data, in turn being able to shape the upload stream (i.e., volume of upload traffic) according to network conditions, as opposed to the network being merely a path to the cloud. Such an approach cannot but be based on a data-centric communication model.

2 Background

According to today's TCP/IP communication model, upon production, data is transferred from a mobile device to a backend cloud over the cellular network. As argued above, this is not sustainable given the enormous capacity requirements of IoT data being generated. With edge-data repositories, data is offloaded there first and fetched to cloud servers as needed. However implementing edge repositories using the current TCP/IP stack, each data object would have to be mapped to the IP address of the corresponding edge data repository. It is questionable whether one could depend on the DNS system update to make the data immediately accessible, given that DNS update intervals can be very long. Alternatively, the IP address of the edge-data repository can be communicated to the backend part of the application it belongs to. Any subsequent request for this object is redirected from the backend cloud to the edge data repository.

Such an implementation may look straightforward in case of relatively static data generation, i.e., a whole object is generated and offloaded with no end-user mobility involved; the case becomes more complex when the end-user/IoT device is moving and connecting to different edge repositories as it goes. Picture for example the case where video is streamed from a mobile user and is being consumed live by remote users, an application scenario supported by Periscope. Every time the producing device is changing point of its attachment, it has to acquire a new IP address from the new AP and update its global identifier pointing to this. This disrupts the point-to-point connection, which is identified in the current IP-based model with the 5-tuples (i.e., address and port pairs and the protocol), until the producer reestablishes them at its new attachment point after some delay. Such delays due to handover can be prohibitive for any near real-time communication.

Instead, in an edge data repository environment illustrated in Figure 1, data produced by the mobile device are immediately pushed to or pulled by edge APs. Remote users that request access to the real-time stream are served from the AP hiding producer mobility: APs act as the stable in-network rendezvous points for the consumers and producers, decoupling the act of sending packets by the producers from the act of receiving packets by the consumers. Furthermore, given that the data itself is named at the granularity of packets (i.e., *chunks*) and are not bound to a connection between two endpoints, the network simply performs *name resolution* to forward request packets towards the AP which stored the intended data packets. We further discuss name resolution and mobility in Section 3.2.

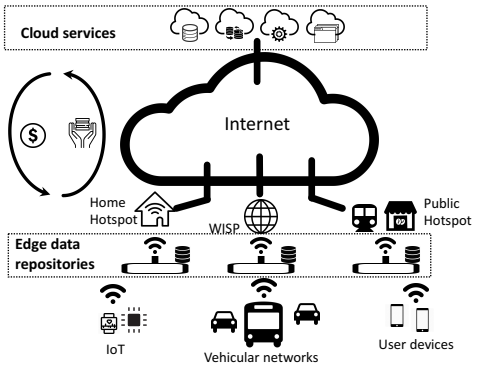

Figure 1: Edge data repositories

The concept of Reverse-CDN was first introduced in [5] for edge-services, in order to improve real-time video analytics. The authors proposed moving the processing logic to the edge of the network using cloudlets, saving bandwidth and reducing latency. Furthermore,

in [6] and [7], the authors propose a solution based on Mobile Edge and Fog Computing in order to process IoT data locally. Closer to our work, in [8], the authors discuss a design vision for a data-centric IoT. They propose the use of both Fog Computing and Information-Centric Networking combined in order to turn IoT devices to smart objects leveraging the handle of upstream data flows.

There has been no previous proposal for a global edge, data-centric repository architecture that enables any and all application to off-load data. In such an architecture, edge data repositories are given full control of the data (in a secure, encrypted manner), extending the cloud to the edge for storage but also offers opportunity for computing capabilities.

The concept of distributed edge repository storage is similar in rationale to peer-to-peer content-addressed storage systems [9]. In such systems, the producers simply push data to a distributed storage system which guarantees availability (e.g., through replication and up-to-date tracking of chunk locations) even with producers being intermittently connected. Providing availability despite disconnections by the producers has the inherent advantage of supporting producer mobility. However, being an application-level solution, such systems still suffer from the drawbacks of host-to-host communications, while a data-centric communication provides a native solution to the mobility problem.

3 Edge Data Repositories: Technical Challenges and Directions

In the current connectivity-based networking model, asynchronous access to data is managed by centralized (i.e., cloud-based) storage services. Consequently, the edge-data production currently relies on these centralized storage services in that the edge-data production is often followed by pushing of the data to a storage location. This leads to an increasing burden on the last-mile networks, which have to deal with the increasing rate, and thus the cost of reverse (i.e., towards upstream provider) data flows. In this paper, we argue that it is highly unlikely for the cell or last-mile ISPs to be able to cope with the increasingly large amounts of edge-data production with the current connectivity-based communication model where data is simply treated as bits in the wire.

3.1 Edge Access Points and Storage

As a solution to the increasing data production at the edges, we propose a data centric communication model that enables asynchronous access to data within the network. This model allows the producers to simply push their data to the network, and let the network manage the storage and access to data. All this is done without requiring the data producers to establish connectivity with an endpoint (e.g., cloud server) and handle the transfer of data as in the current connectivity-based model. In order to better support the data-centric model, we envision a set of distributed data repositories (i.e., large caches) possibly deployed and managed by the edge networks—e.g., at WiFi access points, which are already widespread.

In cases where the produced edge-data can be consumed or processed with a delay as in the case of batch processing, the data can be stored temporarily at a local edge repository. The ability to store data at the edges can lead to cost saving opportunities in terms of bandwidth usage: i) data can be pre-processed locally within the edge-network to significantly reduce the amount sent upstream, and ii) the transfer of data to cloud can be scheduled over longer period of time to reduce the upstream traffic rate, and thus costs. There may be cases where data is only relevant to local consumers and only for a short time. In addition to cost-saving opportunities for the edge networks, an edge repository infrastructure with data-centric communication would enable heavy data producing applications, which would otherwise be impossible to deploy as the constant stream of reverse data is not sustainable.

Having data stored in the distributed edge-repositories requires the network to implement data resolution mechanisms in order to provide immediate access to data. We discuss this issue in the next section.

3.2 Data Resolution Mechanism and Producer Mobility

Once data is stored at an edge repository (e.g., at an access point), the network then must also manage access to data. In a data-centric communication model that names data at the granularity of packets (i.e. chunks), access to data packets can be enabled through a name resolution mechanism. An *in-network* name resolution mechanism is possible given a name-based routing/forwarding architecture such as the *Named Data Networking (NDN) Internet architecture* [10].

In our system, once data chunks are received, the edge repository assumes the role of a producer of the named data packets. In the case of a moving producer, the name resolution mechanism is more challenging as the producers push their data chunks to different AP locations while moving. The name resolution must forward requests for data chunks to the up-to-date AP locations to be able to handle mobility. Therefore, in order to handle producer mobility for real-time applications (e.g., a producer streaming video and consumers watching), the name resolution is required to enable immediate access to data chunks. In the case of an in-network name resolution, this requires updating of the forwarding state of the

nodes in the network, possibly through the convergence of a routing process.

Given that the routing convergence process takes time, we propose an indirection-based name resolution instead. In particular, the APs can inform the current default destination of data chunks—which may be another AP or a remote cloud server—of a routing hint (i.e., a locator) [11]. This indicates the up-to-date location of the named data chunks. In this case, the interest (i.e., request) packets indicate the name of the requested data packets together with their locators as a routing hint for the network. The name resolution is performed through indirection, whereby the consumer traffic is routed first towards the current default destination (through routing) that redirects the traffic to the correct AP by appending the AP's locator to the consumer request packets and forwarding them on. As a further optimization, the APs can replicate the data at both the current location and the former (i.e. current default) location of the data, until the in-network name resolution converges and starts forwarding consumer traffic to the new AP.

In order to provide immediate global availability for edge-data, one can utilize the indirection mechanism described above. In particular, a service (possibly running within the cloud) can be configured as the default location of a data object, by associating the name of the data with the server's locator. The consumers then include the locator of the service in their interest packets (as a routing hint), which are forwarded initially to the service as a result. The service then redirects the consumer requests to the AP currently holding the data, after being informed by the AP with the locator of the data.

3.3 Push Communications

In the absence of architectural support for push communications in the network, an entity within the edge-network must initiate the communication with the producers. One way to achieve this is by instantiating lightweight versions of applications inside the edge repositories, e.g., lightweight version of a dropbox-like application to store personal videos The only task of the instances would be to pull the producer data and store them into the edge repository. The deployment of virtual application instances within an edge network can be realized through an edge-computing infrastructure [12, 13].

3.4 ISP Relationships

In the current customer-provider business relationship model, customer ISPs typically commit to a certain rate of traffic (in Mbps) and depending on the committed rate, they are charged per Mbps for the 95th percentile rate, i.e., excluding the bursts. With increasing edge-data production, the volume of data that needs to be sent to higher-tier ISPs, and the rate of requests for the stored data will affect the transit costs of eyeball ISPs. This is likely to cause a "tussle" [14] between last-mile networks and data producing applications in a similar way to the on-going tussle between overlay (i.e., peer-to-peer) routing applications and ISPs due to violation of the ISP routing policies by the overlay traffic.

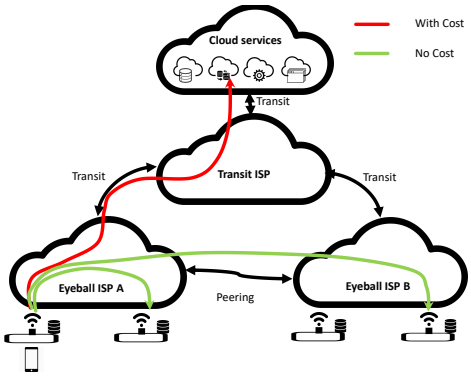

Figure 2: Data consumption scenarios

In general, data might be requested: i) locally from within the same domain, in which case there is no transit cost, ii) from peering domains, in which case there is also no transit cost involved, or iii) from higher-tier ISPs, in which case the eyeball ISP has to pay upstream transit costs, as shown in Figure 2. With a data centric communication model, the eyeball ISPs can monitor access to data, and control the movement of data upstream complying with the policies of the applications expressed as intents [15]. As an example, edge ISPs can take advantage of the unused capacity at certain off-peak times and shift/schedule the transfer of data towards upstream ISPs at those off-peak times. Such transfers can be "free" as long as the ISP can keep the aggregate rate of upstream traffic during off-peak times below the 95th percentile, as proposed for delay-tolerant bulk data transfers [16].

3.5 Business Model

To achieve a successful deployment of edge data repositories in the access network, an important investment is required by WISPs, that need to upgrade the hardware or deploy new one. However, by using edge repositories, WISPs can reduce their transit costs, similar to [17], reducing traffic peaks to avoid being overcharged by transit ISPs.

But not only WISPs can be incentivised to deploy edge data repositories. Cloud service providers (application owners) will require make use of this technology in order to use all the available data for their services. Otherwise, some of this data will be lost because either WISPs would not provide enough bandwidth, or WISPs would transfer this cost to end-users that would not synchronize all this data to the cloud in order to save money.

However, it is not necessary that cloud service providers will be the ones who directly deploy these repositories. Other scenarios are possible such as third-party entities, e.g., a "reverse" CDN with upload/download servers, could own and manage such an infrastructure and in turn pay the eyeball ISP for hosting the infrastructure. Cloud service providers could directly, or indirectly via micropayments, pay for the "reverse" CDN service. There exists some existing projects based on distributed ledger technologies, such as blockchain, that provide distributed storage systems rewarded via cryptocurrencies, e.g.,[18, 19, 20], that share the same idea.

4 Data Management

In the data-centric communication model, the network can be informed of the *intent* [15] of the applications in terms of how their data is to be treated through naming - effectively adding application semantics in network-layer content names/addresses. For instance, if the data is to be processed locally, the hierarchical name of the data can be prefixed with `/exec`, or if the data is to be eventually stored or processed at a remote cloud service within a certain deadline, then the deadline can be included within the name itself. In general, each piece of data may be associated with attributes such as the deadline (see Section 4.1 for an extended list), and these can be expressed with a list of tags or keywords [21, 22] as part of a separate component of the data names. However, this requires data producers to indicate their intentions and policies on the management of their data as attributes. In this section, we first discuss possible attributes of data in Section 4.1 and then describe possible data management strategies in Section 4.2.

4.1 Data Attributes

There is a set of data attributes that are of interest, when managing the transfer of stored data. The data attributes are related to both the semantics of the application(s) that consume them and producer preferences.

- *Persistent Name:* A mandatory attribute of a data is a persistent name, which does not change with mobility. This name may also be used by the network to locate, replicate, cache and access the data.
- *Destination:* Data objects that require transfer to a particular end-point, e.g., for storage, or computation, would be required to provide a locator or a name associated with the destination end-point.

  The rest of the attributes are optional:

- *Shelf-life:* A data object may be no longer relevant for any consumers beyond a certain expiration time. After this point, the data may be safely discarded.
- *Access Scope and Urgency:* The expected origin of Interests for a data object may be strictly local (i.e., originate from the users within the domain), strictly global, or a mix of local and global. The access to data can be immediate or delayed. In the case of delayed access, a deadline can be provided as explained below.
- *Deadline:* For data objects that require certain time-sensitive actions such as access, computation or relaying to a destination, a deadline may be specified.

These and possibly other additional attributes can be desirable for the edge-networks to manage data. These attributes can be expressed as part of the name of the data in a name-based routing architecture.

4.2 Data Management Strategies

A data management strategy dictates how an eyeball network coordinates the management of edge-data stored at a repository. Below are a list of possible strategies:

- *Proactive Strategy:* In this strategy, the local repositories transfer the incoming data pro-actively to their intended destination (e.g., cloud storage) immediately at the rate permitted by the outbound capacity of its link to the domain's upstream provider. This scenario uses the storage at a local repository only in the case when upstream link capacity is below the rate at which data is produced.
- *Reactive Strategy:* The repository shares a routing hint for each stored content object with the data object's destination, which can then pull the data when necessary. This way, the edge repositories handle data transfers in a "lazy" manner, i.e., transfers data only when necessary.
- *Data-specific strategy:* In this strategy, the domain makes use of data attributes such as scope, shelf-life and deadline to determine actions specific to each data object. For example, the edge domain can store the data locally in case the data is short-lived (i.e., shelf-life expiring) or has only local access. Alternatively, the domain can monitor access to data objects and transfer them to their intended destinations in case of heavy global access.

5 Conclusions and Future Work

We have presented the skeleton of data-centric architecture that, combined with Network Functions and storage capacity deployed in the edge of the network, will allow great flexibility in order to manage, process and/or temporarily store data locally, without the necessity of instantly moving all data to the cloud. Mobile Data Edge Repositories will reduce the pressure on over-dimensioning of cloud datacenters, ISP and cellular networks, which would be required otherwise.

References

[1] D Evans. The internet of things: How the next evolution of the internet is changing everything. 1:1–11, 01 2011.

[2] D. Shey A. Markkanen. Edge analytics in iot. 04 2015.

[3] C. Perkins. IP Mobility Support for IPv4. RFC 3344 (Proposed Standard), August 2002. Updated by RFC 4721.

[4] S. Gundavelli, K. Leung, V. Devarapalli, K. Chowdhury, and B. Patil. Proxy Mobile IPv6. RFC 5213 (Proposed Standard), August 2008.

[5] M. Satyanarayanan, P. Simoens, Y. Xiao, P. Pillai, Z. Chen, K. Ha, W. Hu, and B. Amos. Edge analytics in the internet of things. *IEEE Pervasive Computing*, 14(2):24–31, Apr 2015.

[6] X. Sun and N. Ansari. Edgeiot: Mobile edge computing for the internet of things. *IEEE Communications Magazine*, 54(12):22–29, December 2016.

[7] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. Fog computing and its role in the internet of things. In *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, MCC '12, pages 13–16, New York, NY, USA, 2012. ACM.

[8] E. M. Schooler, D. Zage, J. Sedayao, H. Moustafa, A. Brown, and M. Ambrosin. An architectural vision for a data-centric iot: Rethinking things, trust and clouds. In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pages 1717–1728, June 2017.

[9] Juan Benet. Ipfs-content addressed, versioned, p2p file system. *arXiv preprint arXiv:1407.3561*, 2014.

[10] Lixia Zhang, Alexander Afanasyev, Jeffrey Burke, Van Jacobson, Patrick Crowley, Christos Papadopoulos, Lan Wang, Beichuan Zhang, et al. Named data networking. *ACM SIGCOMM Computer Communication Review*, 44(3):66–73, 2014.

[11] Alexander Afanasyev, Cheng Yi, Lan Wang, Beichuan Zhang, and Lixia Zhang. Snamp: Secure namespace mapping to scale ndn forwarding. In *Computer Communications Workshops (INFOCOM WKSHPS), 2015 IEEE Conference on*, pages 281–286. IEEE, 2015.

[12] Pedro Garcia Lopez, Alberto Montresor, Dick Epema, Anwitaman Datta, Teruo Higashino, Adriana Iamnitchi, Marinho Barcellos, Pascal Felber, and Etienne Riviere. Edge-centric computing: Vision and challenges. *ACM SIGCOMM Computer Communication Review*, 45(5):37–42, 2015.

[13] Michał Król and Ioannis Psaras. Nfaas: named function as a service. In *Proceedings of the 4th ACM Conference on Information-Centric Networking*, pages 134–144. ACM, 2017.

[14] David D Clark, John Wroclawski, Karen R Sollins, and Robert Braden. Tussle in cyberspace: defining tomorrow's internet. In *ACM SIGCOMM Computer Communication Review*, volume 32, pages 347–356. ACM, 2002.

[15] Yehia Elkhatib, Geoff Coulson, and Gareth Tyson. Charting an intent driven network. In *Network and Service Management (CNSM), 2017 13th International Conference on*, pages 1–5. IEEE, 2017.

[16] Nikolaos Laoutaris, Georgios Smaragdakis, Pablo Rodriguez, and Ravi Sundaram. Delay tolerant bulk data transfers on the internet. In *ACM SIGMETRICS Performance Evaluation Review*, volume 37, pages 229–238. ACM, 2009.

[17] Nikolaos Laoutaris, Georgios Smaragdakis, Pablo Rodriguez, and Ravi Sundaram. Delay tolerant bulk data transfers on the internet. In *Proceedings of the Eleventh International Joint Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '09, pages 229–238, New York, NY, USA, 2009. ACM.

[18] Protocol Labs. Filecoin: A decentralized storage network. White paper available at `https://filecoin.io/filecoin.pdf`, 2017.

[19] Luke Champine David Vorick. Sia: Simple decentralized storage. White paper available at `https://sia.tech/sia.pdf`, 2014.

[20] Josh Brandof James Prestwich Gordon Hall Patrick Gerbes Philip Hutchins Chris Pollard Shawn Wilkinson, Tome Boshevski. Storj: A peer-to-peer cloud storage network. White paper available at `https://filecoin.io/filecoin.pdf`, 2016.

[21] M Ascigil, Sergi Reñé, George Xylomenos, Ioannis Psaras, and George Pavlou. A keyword-based icn-iot platform. In *Proceedings of the 4th ACM Conference on Information-Centric Networking*, pages 22–28. ACM, 2017.

[22] I. Psaras, S. Reñé, K. V. Katsaro, V. Sourlas, G. Pavlou, N. Bezirgiannidis, S. Diamantopoulos, I. Komnios, and V. Tsaoussidis. Keyword-based mobile application sharing. In *Proceedings of the Workshop on Mobility in the Evolving Internet Architecture*, MobiArch '16, pages 1–6, New York, NY, USA, 2016. ACM.