open EDGE computing

Interconnect Work Stream Report

Phase 1: Virtual Living Edge Lab

CONTENTS

EXECUTIVE SUMMARY

In January 2020, the Open Edge Computing Initiative (OEC) [1] identified a key challenge in edge computing networks. Interconnection points (IXP) between different carriers were designed and implemented prior to the emergence of edge computing. These designs did not account for the need for low latency computing between users and edge computing nodes (aka *cloudlets [2])*.

To help carriers understand the tradeoffs of different network positioning of IXPs in the context of edge computing, the OEC established the Interconnect Workstream and chartered the Living Edge Lab [6] to investigate these tradeoffs. This report presents the results of phase one of this investigation.

In this phase, we created a simulation environment, referred to as the *virtual Living Edge Lab (vLEL)* that emulated multiple mobile wireless networks and IXP positions and enabled the measurement of key application and network performance characteristics over the course of the simulation. While this experiment cannot be considered a general study of the IXP placement problem, it does provide some key insights and conclusions that are likely to be broadly applicable.

The business success of edge computing depends on addressing limitations imposed by the industry's legacy approach to carrier interconnect. These limitations render many edge-native applications unusable in the many scenarios. Our workstream results show that viable edge computing requires:

- Regardless of IXP location, edge computing "cloudlets" must be located in the same metro/region as the application users. Without this, end-to-end latency becomes unacceptable for many applications.
- Once metro cloudlets are deployed, IXPs must be established within the metro area and networks engineered to prevent user to cloudlet data paths outside of the metro. Since cloudlets will often be hosted on wired metro networks, metro IXPs will increase significantly in importance.
- Within the metro area, the marginal performance benefit to moving IXPs closer to the user (e.g., to the cell tower) is small and may not justify cost. This conclusion, however, depends on the value and requirements of the full set of edge applications to be deployed. For example, edge apps like augmented and virtual reality games that rely on very fast user and display responses require very low and consistent round-trip times to be acceptable. Achieving this will necessitate moving the edge closer to the gamer. IOT sensor applications requiring real-time or near real-time control responses will also likely need closer placements.
- Metro third-party neutral host IXPs will provide equivalent performance to direct carrier-to-carrier IXPs with potentially lower complexity.
- While application performance is the main criteria for IXP placement decisions, other requirements like lawful intercept and data geofencing also need to be considered. For example, many widely distributed IXPs make it more difficult for carriers to assure full compliance with lawful intercept regulations.

Given long planning and implementation cycles, carriers should begin work immediately to enable edge computing by deploying metro IXPs with other carriers as soon as possible.

INTRODUCTION

In January 2020, the Open Edge Computing Initiative (OEC) [1] identified a key challenge in edge computing networks. Interconnection points between different carriers were designed and implemented prior to the emergence of edge computing. These designs did not account for the need for low latency computing between users and edge computing nodes (aka *cloudlets [2])*.

This problem was first defined by Gerszberg [3] in his January 2020 blog. Carrier inter-exchange points (IXP) are physical locations at which carriers transfer user and network data that must move between carriers. In traditional networks, IXPs are often at centralized locations far from users and cloudlets. Data passing from a user on one carrier network to a cloudlet on another will need to travel across the first network to an IXP connecting to the second network before it can reach the cloudlet. Return packets will traverse a similar path in reverse. For multi-user interactive applications, traffic between users on different carrier networks will also need to pass through an IXP. Depending on the locations of the users, cloudlet and IXP, the added end-to-end latency can be 10s to 100s of milliseconds.

To help carriers understand the tradeoffs of different network positioning of IXPs in the context of edge computing, the OEC established the Interconnect Workstream and chartered the Living Edge Lab [6] to investigate these tradeoffs. This report presents the results of phase one of this investigation.

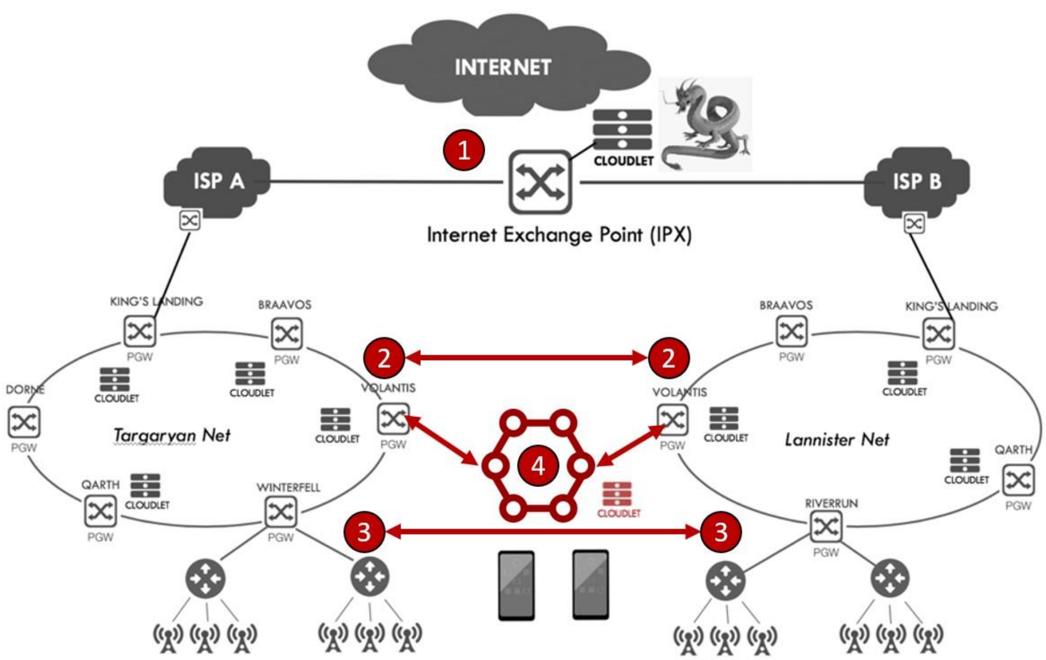

Figure 1: Potential Interexchange Points

BACKGROUND

Edge computing [2] brings the promise of enabling new edge-native applications [4] that need low latency and high bandwidth connections to mobile and wired edge devices to achieve acceptable user experience. It has been shown [5] that without edge computing, end-to-end network latency can exceed 150ms. Many connected user devices such as cell phones, cameras, vehicles, etc., referred to as user equipment (UE) by the telecommunications industry, produce data volumes that exceed the viable economic costs and acceptable transport times to transfer from the UE to a remote cloud. The traditional approach to managing these challenges has been to deploy application functionality on the UE that mitigates the need to transfer data to the cloud. For example, traditional mobile gaming typically implements the majority of game functionality on the UE with only time-insensitive tasks implemented in the cloud. Similarly, smart cameras may perform cropping, down sampling and encoding functions on incoming streams to reduce the transferred bitrate.

These techniques can meet the needs of many applications but, for others, the application experience quality can be inadequate. In gaming, for example, lighting effects may be of low quality due to the processing limitations of the UE graphics processing unit (GPU). A computer vision application may become less accurate when a highly compressed bitstream is sent to the cloud for processing.

Edge computing offers a solution to these problems by reducing the physical network distance and the number of network hops from the device to nearby application computing resources. An *edge-native* application is one that is designed to take advantage of attributes that this closer placement provides: low latency, bandwidth scalability, privacy-preservation, and wide area network failure resiliency. [4] defines the concept of edge-native applications and points to several specific edge-native applications. Most edge-native applications fall into one of the following four categories. See Figure 2.

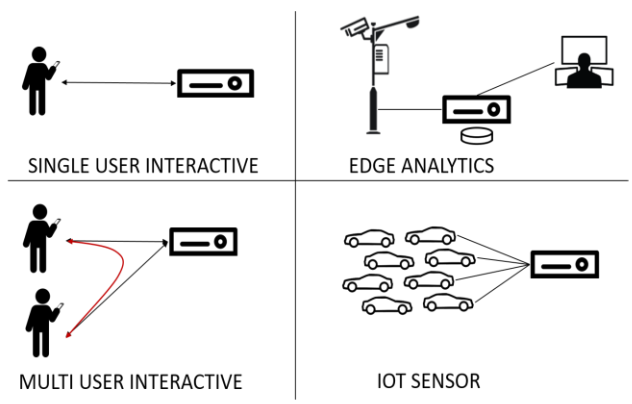

Figure 2: Edge-native Applications

1. Single User Interactive – These applications involve a single user interacting through a mobile UE with a distributed application service. Although many users may use the service simultaneously, interaction between users is negligible. Examples include many augmented reality applications like wearable cognitive assistance [7] and virtual desktop infrastructure. The user experience for these applications is generally measured by response time and visual quality.
2. Multi-User Interactive – These applications retain many of the characteristics of single user interactive applications but add significant interaction between users. Examples include multi-player gaming and video conferencing. User experience is still measured by response time

and visual quality but delivering acceptable performance is complicated by the potential for collaborating users to be serviced by different cloudlets and mobile carriers.

3. Multi-User Interactive – These applications retain many of the characteristics of single user interactive applications but add significant interaction between users. Examples include multi-player gaming and video conferencing. User experience is still measured by response time and visual quality but delivering acceptable performance is complicated by the potential for collaborating users to be serviced by different cloudlets and mobile carriers.
4. Edge Analytics – These applications involve data collection and processing from distributed UEs to gain understanding and insight that can drive operational action. Often, transferring raw collected data to a centralized location is cost or transfer latency prohibitive or is unacceptable due to privacy concerns. Examples include intelligent processing of surveillance videos and distributed federated machine learning [8]. User experience is driven by the cost and time to insight from gathered data.
5. Internet of Things (IOT) Sensor – These applications aggregate connections from many distributed sensor and actuator UEs to provide control or control-assist and data analysis and collection functions. Examples include autonomous vehicles and distributed traffic monitoring services. User experience is driven by the response time for control functions and the cost and time to insight for analytics functions.

This list excludes operator and operations related applications such as firewalls, traffic control and routing and other virtual network functions (VNF) [9]. It instead focuses on value-added services where an external consumer or business user gains a tangible and visible benefit from use of the service.

The physical and network placement of cloudlets and the interconnection of user and cloudlet networks is a complex trade-off between costs and achieving the user performance requirements for the diverse applications described above. In this report, we present the results of a simulation experiment conducted in the CMU Living Edge Lab that looked at the placement of network IXPs and the impact of that placement relative to cloudlets and various UE for a specific single user interactive application.

To execute this experiment, we created a simulation environment, referred to as the *virtual Living Edge Lab (vLEL)* that emulated multiple mobile wireless networks and IXP positions and enabled the measurement of key application and network performance characteristics over the course of the simulation. While this experiment cannot be considered a general study of the IXP placement problem, it does provide some key insights and conclusions that are likely to be broadly applicable.

The rest of this report is structured as:

- The Virtual Living Edge Lab Simulation Environment
- Baselining the Real World
- Interconnect Simulation Scenario and Results
- Conclusions and Next Steps

THE VIRTUAL LIVING EDGE LAB

Our Virtual Living Edge Lab simulation framework is depicted physically in Figure 3 and logically in Figure 4.[1] This section describes this framework.

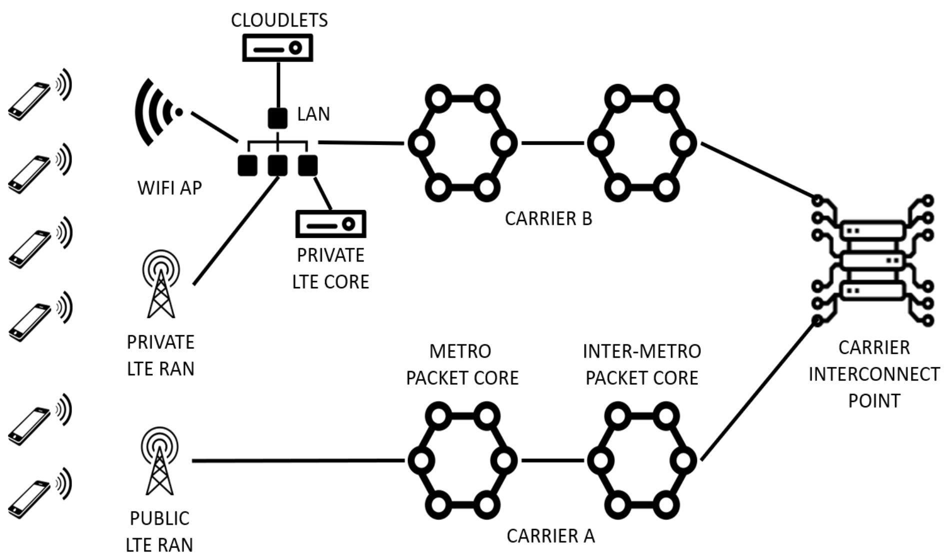

Figure 3: Framework Physical Architecture

Physical Infrastructure

The physical infrastructure for the simulation framework is built on the Living Edge Lab [6] infrastructure at Carnegie Mellon University in Pittsburgh, Pennsylvania. It consists of three independent but interconnected wireless networks, a cloudlet and a set of mobile UEs. The three wireless networks are:

1. An in-building WiFi network connected to a wired LAN network. The cloudlet is also connected to the same wired LAN.
2. Local commercial public LTE networks from AT&T and T-Mobile that are connected through a remote commercial interexchange point to the LAN.
3. An outdoor private LTE network that is directly fiber connected to the wired LAN. The Private LTE wireless core resides on the same wired LAN.

Due to technical issues, the simulations described below were done using the WiFi and public LTE networks. The WiFi network was used for the purely emulated testing as it introduced the minimum real network latency to the end-to-end application pipeline (<3ms). The public LTE network was used for the real-world measurement cases. Using this network had the advantage of providing

---

[1] In this report, the term *simulation* means the execution of a test scenario on the framework described in this section. *Emulation* means the use of AdvantEDGE platform to emulate a mobile wireless network.

measurements from an operational commercial network, however the lack of a metro IXP meant that traffic between client and cloudlet travelled out of the metro area and, therefore, experienced an additional 20-40ms of one-way end-to-end network latency.

The cloudlet is a single node Intel® Core™ i7-6700 CPU @ 3.40GHz with an NVIDIA GeForce GTX 1060 3GB GPU. The server-side simulation framework and test application run on the cloudlet. The client UEs are android smartphones including a Samsung Galaxy S8 and an Essential PH-1.

Network Emulation Platform (AdvantEDGE)

The simulation framework is centered around the AdvantEDGE platform [10]. AdvantEDGE is a mobile edge emulation platform that runs on Docker and Kubernetes. AdvantEDGE provides an emulation environment that enables experimentation with edge computing technologies, applications, and services. The platform facilitates exploring edge deployment models and their impact on applications and services in short and agile iterations.

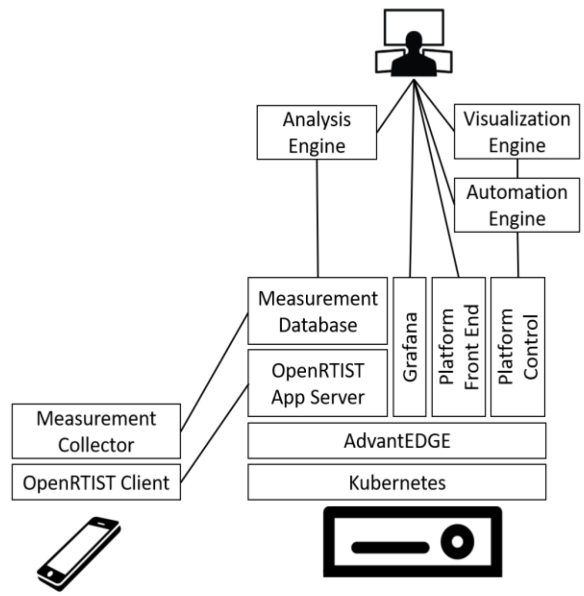

Figure 4: Framework Logical Architecture

AdvantEDGE enables the user to define scenarios that include:

- A network topology of cloudlets, clients, wireless points of access, zones and UE
- Network characteristics for each element including latency, jitter, packet loss and throughput
- Network and mobility events to change network characteristics and the location of UE and cloudlets during simulation run time

It allows the connection of real cloudlet and UE applications so that simulation can capture the impact of network design on application performance. It also supports event scripting, collection of measurements in an offline InfluxDB time series database and real time Grafana dashboards. This combination makes it a powerful platform for edge network simulation. These capabilities were all used in the scenarios discussed below.

Instrumented Client Application

Figure 5: OpenRTIST Style Transfer

In edge-native applications, the client application running on the UE directly provides the user experience. It has visibility to application experience degradation caused by end-to-end latency, jitter, packet loss and bandwidth constraints. It also has access to network and location information that can be highly valuable for network analysis. In addition, the nature of the application itself defines the user experience metrics of importance. For example, latency only matters if the delay is perceptible by a user and materially affects the experience. A test application for assessing edge computing networks must have a real user experience and be able to measure important quantitative experience metrics.

In this work, we used a modified version of the OpenRTiST [11] application as our test application. OpenRTiST is a simple single user interactive application that captures user video, transfers that video to the cloudlet, performs image style transfer [12] on the video and sends the styled video back to the UE for display (see Figure 5). Network degradation impacts the experience by making the styled video appear jerky and lagging behind real time. This experience can be quantitatively measured using two metrics that are captured by the client, *round trip time (RTT)* and *framerate (FPS)*. These two metrics are inversely related to each other and are heavily impacted by network latency and packet loss. RTT is measured by time stamping each video frame leaving the client and detecting when the corresponding styled frame is returned to the client. RTT includes the round-trip network latencies, the style processing time at the cloudlet and the transmit and receive times at the client.[2] To prevent frame buffers from overflowing due to these delays, frames are dropped by the application when necessary. This frame dropping reduces the application FPS.

In addition to these two user experience metrics, other data for use in monitoring and analysis of the system was captured including network route, location, phone and cell tower characteristics.

Automation Engine

To simulate a realistic mobile network, the configuration and characteristics of that network need to vary over the course of a simulation. This need requires an automation engine to script these variations for the specific simulation. The AdvantEDGE platform allows the creation of *mobility events* and *network characteristic events* as simulation building blocks.

Mobility events allow the movement of UEs from one Point of Access to another, cloudlets from one zone to another and cloudlet services from one cloudlet to another. *Network characteristic events* allow the characteristics of individual nodes to be changed. The primary controllable network characteristics are *latency, latency variation a.k.a. jitter, throughput* and *packet loss*.

[2] Times for camera frame capture and frame display on screen are not included in the RTT.

Data Management and Analysis

The value of a simulation is derived primarily from insights from the data collected. As mentioned above, the AdvantEDGE platform and the instrumented client load data into an InfluxDB time series database. AdvantEDGE stores network and event measurements from deployed scenarios. The instrumented client stores the measurements described above. The database is accessible for:

- Display on AdvantEDGE and Grafana dashboards
- Extraction, analysis and visualization by any number of external analytics engines

BASELINING THE REAL WORLD

Prior to running the specific model simulations, we baselined three attributes of our environment:

Application Latency with an Unconstrained Network

The OpenRTiST application has a UE execution component and a cloudlet execution component. These components add latency to the total RTT the user experiences. This application-specific latency is mostly independent of the specific network characteristics of the mobile network. To measure this application latency, we set our network emulation to a "null" network – no network latency, jitter, packet loss and infinite bandwidth. Under these conditions, the two components of the application introduce a latency of 41ms to the overall RTT as shown in Figure 6.

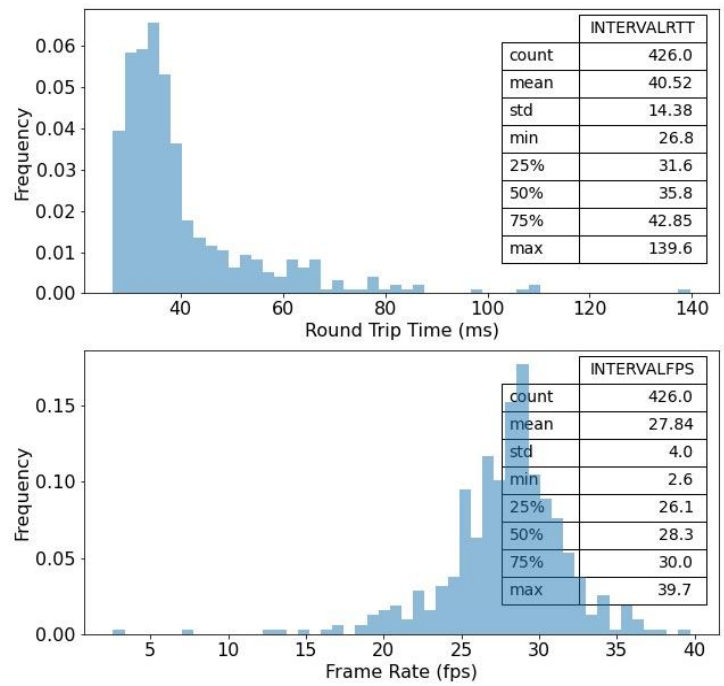

Figure 6: Application Only Round-Trip Time and Framerate

Quantitative Impact of Network Latency Increases on Application Performance

To understand the impact of incremental network latency, we simulated monotonically increasing network latency in steps of 5ms with a range from 0ms to 50ms. Figure 7 shows the result of this simulation. As expected, the overall RTT increases linearly with additional network latency with a y-intercept around 41ms as we described above.

We also observed the effect of increasing latency visually on the application itself – as the network latency increases, the displayed video rapidly becomes choppy and delayed. The user experience becomes visibly degraded with an RTT more than 150ms and framerate less than 10 FPS. These thresholds are clearly subjective and application specific but, for the purposes of this report, were used as cut off thresholds for acceptable application performance.

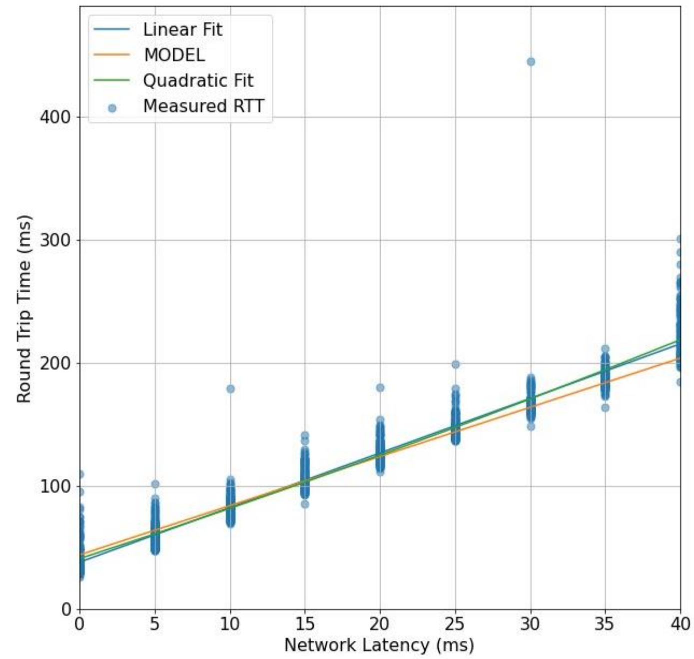

Figure 7: Expected Application Performance in Increasing Network Latency

Application Performance on A Commercial Mobile LTE Network

We also baselined an expected RTT for a typical commercial LTE wireless network to assure that our network characteristics were in line with real world measurements. The data provided by carrier OEC members provided a starting point for our model characteristics; our next step connected a mobile UE to the local Pittsburgh T-Mobile LTE network. The connection traversed T-Mobile to a distant carrier inter-exchange point (IXP) where it returned to Pittsburgh through the Verizon FIOS wired access network. The route of travel for a typical session is shown in Figure 8. We collected application performance data for this connection. The mean RTT was 237ms and the mean FPS was 8 FPS – unacceptable application performance when compared with our acceptability criteria of 150ms and 10 FPS.

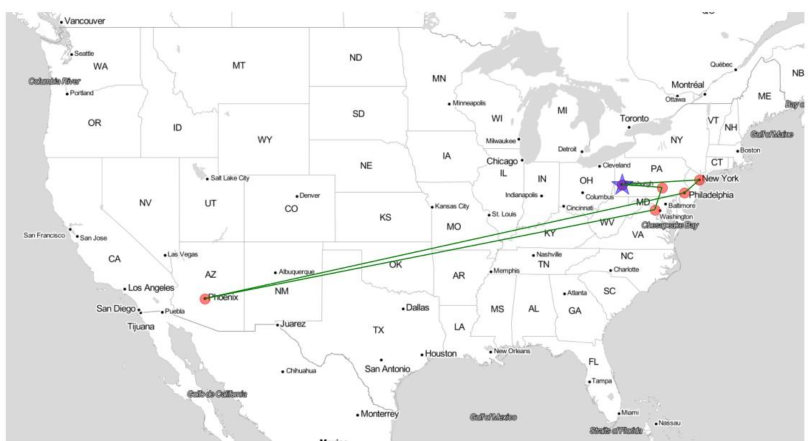

Figure 8: Mobile Wireless Traffic Route

INTERCONNECT SIMULATION RESULTS

Gerszberg [3] defined four potential IXP positions for assessment as shown in Figure 1.

1. An IXP position geographically remote from the user and the cloudlet. This position is currently typical of many existing mobile networks and what we saw in the network baseline shown in Figure 8. In the simulation, we assumed that the cloudlet was connected to the network at the radio access network (RAN), requiring packets from the cloudlet to traverse RAN, the metro core and the Wide Area Network (WAN) to reach the IXP.
2. An IXP position within the same metropolitan area (aka serving area) as the user and cloudlet. This position would typically be somewhere in the metro core infrastructure. In the simulation, we looked at two IXP locations in the metro core, a) at a point in the metro core far from the UE and cloudlet and b) at the edge of the RAN near the UE and the cloudlet.
3. An IXP position in the radio access network (RAN). This position puts the IXP very close to the UE and the cloudlet.
4. Two IXPs within the metro area core. In this case, the IXPs are provided by a third-party neutral host who transfers the data between the two carriers. Based on partner input, we estimated that traversing two IXPs within the neutral host's metro network had <1ms impact on the latency as compared to Model 2.

Our simulation goals were to measure the application user experience with users and cloudlets on different carrier networks given each of these IXP positions. We assume that costs increase as the IXP is moved closer to the network edge. This cost increase derives from the increased number of IXPs and increased transport infrastructure required to connect to the IXPs. Therefore, the optimal IXP position is the location furthest from the edge where the application user experience meets the minimum acceptable requirement. These criteria are obviously application specific and are complicated in multi-user interactive applications where the relative positions of users and cloudlets can be very complex.

To implement this simulation, we used the OpenRTiST instrumented client and created an AdvantEDGE network scenario that reflected the network topology and characteristics provided by OEC carrier members, especially Vodafone and VaporIO. The topologies and network characteristics are shown in Figure 9.

Distant IXP *(Base Case)*

We calibrated the Distant IXP simulation to align with the baselined commercial LTE environment. We were able to calibrate our simulation to the observed commercial LTE RTT of 237ms by setting the *out-of-area WAN* latency to 35ms and left the other characteristics unchanged.

Metro Core and 3rd Party Metro IXP

The metro core model (Model 2) has two sub-cases. Case 2a places the IXP in the metro core near the connection to the out-of-area WAN. This IXP location would be typical as metro interconnect often occurs at the same physical location as carrier interconnect to the wide area internet. The metro core case 2b places the IXP in the metro core near the connection between the RAN and the metro core.

The difference in the two cases is an incremental 5ms network latency between case 2b and case 2a. The 3rd party metro model (Model 4) is simulated with the same configuration as case 2a.

Near RAN IXP

The near RAN (Model 3) assumes that the IXP is placed very near the edge of the RAN such that the latency between the radio and cloudlet is less than 2ms. This model is the most expensive case as IXPs and cloudlets would necessarily be widely distributed and the network infrastructure to create many distributed IXPs may be cost prohibitive.

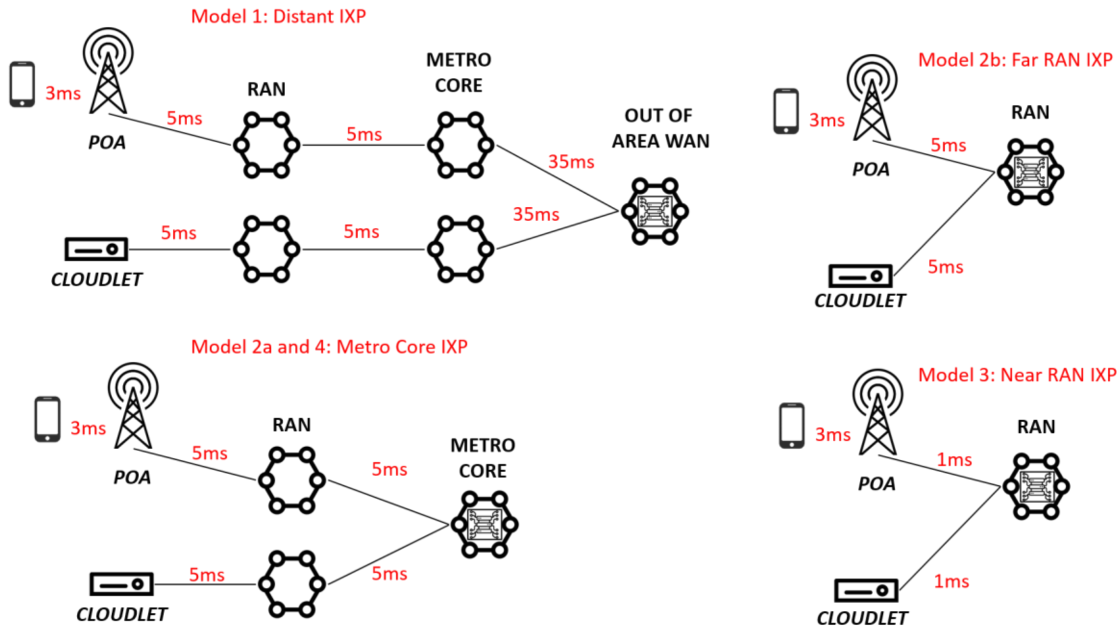

Figure 9: Simulation IXP Topologies

Simulation Summary

The RTT and FPS measurements for all four models are shown in Figure 10. As we would expect from our baselining exercise, our Distant IXP simulation gives application RTT and FPS performance below acceptable per our criteria above. From this data and for this application, we can see that the other models achieve acceptable application performance.

For this application, IXP placement closer to the UE than the metro core gives an incremental improvement in user experience. However, that improvement is slight and the application experience is already acceptable with any placement in the metro core. There may be applications (e.g., real time sensitive, safety critical IoT applications) where the added cost of a closer placement can be justified.

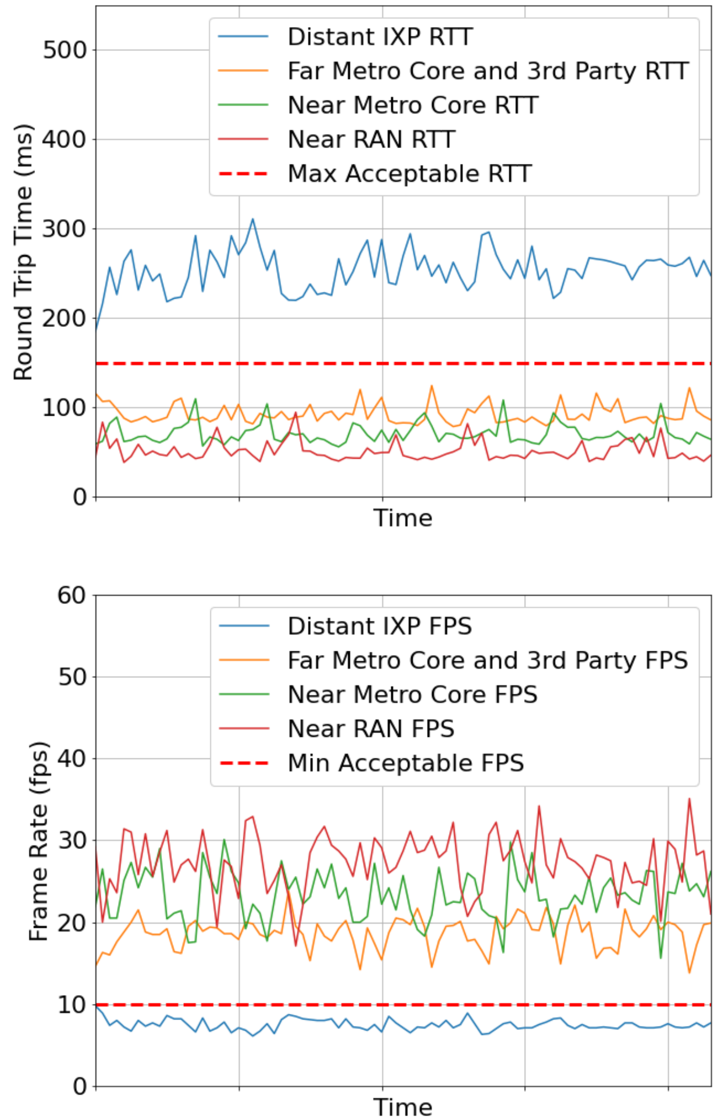

Figure 10: IXP Simulation Results

CONCLUSIONS AND NEXT STEPS

This report presented the results of the first phase of the OEC Interconnect Workstream, the *Virtual Living Edge Lab*. The intent of this work was to gain insights into the optimal placement of carrier interconnect points in support of edge-native application deployment. We do not present generalized conclusions from the results due to the limitations of the simulations:

- The work was completed using a single application, OpenRTiST. While OpenRTiST is an excellent representative of single user interactive applications, it does not adequately represent all applications of all classes. Future work should expand on the representation with a particular focus on broadening to other application classes.
- The network characteristics used in the simulations were derived from real network measurements provided by our carrier partners. However, this data was limited in scope and, accordingly, simulations were run with simple scenarios. Going forward, we hope to replicate the experiments on real networks with local interconnect.

The business success of edge computing depends on addressing limitations imposed by the industry's legacy approach to carrier interconnect. These limitations render many edge-native applications unusable in the many scenarios. Our workstream results show that viable edge computing requires:

- Regardless of IXP location, edge computing "cloudlets" must be located in the same metro/region as the application users. Without this, end-to-end latency becomes unacceptable for many applications.
- Once metro cloudlets are deployed, IXPs must be established within the metro area and networks engineered to prevent user to cloudlet data paths outside of the metro. Since cloudlets will often be hosted on wired metro networks, metro IXPs will increase significantly in importance.
- Within the metro area, the marginal performance benefit to moving IXPs and cloudlets closer to the user (e.g., to the cell tower) is small and may not justify cost. This conclusion, however, depends on the value and requirements of the full set of edge applications to be deployed. For example, edge apps like augmented and virtual reality games that rely on very fast user and display responses require very low and consistent round-trip times to be acceptable. Achieving this will necessitate moving the edge closer to the gamer. IOT sensor applications requiring real-time or near real-time control responses will also likely need closer placements.
- Metro third-party neutral host IXPs will provide equivalent performance to direct carrier-to-carrier IXPs with potentially lower complexity.
- While application performance is the main criteria for IXP placement decisions, other requirements like lawful intercept and data geofencing also need to be considered. For example, many widely distributed IXPs make it more difficult for carriers to assure full compliance with lawful intercept regulations.

Given long planning and implementation cycles, carriers should begin work immediately to enable edge computing by deploying metro IXPs with other carriers as soon as possible.

This research was supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001117C0051 and by the National Science Foundation (NSF) under grant number CNS-1518865 and the NSF Graduate Research Fellowship under grant numbers DGE1252522 and DGE1745016. Additional support was provided by Intel, Vodafone, Deutsche Telekom, Crown Castle, InterDigital, Seagate, Microsoft, VMware and the Conklin Kistler family fund. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the view(s) of their employers or the above funding sources.

REFERENCES

1. The Open Edge Computing Initiative. https://openedgecomputing.org/. 2020.
2. M. Satyanarayanan. "The Emergence of Edge Computing". In: *Computer* 50.1 (2017), pp. 30–39.
3. Tomasz Gerszberg. Shared Edge Experience. https://www.linkedin.com/ pulse/shared-edge-experience-tomasz-gerszberg/. 2019.
4. M. Satyanarayanan et al. "The Seminal Role of Edge-Native Applications". In: *2019 IEEE International Conference on Edge Computing (EDGE)*. 2019, pp. 33–40.
5. M. Satyanarayanan et al. "Cloudlets: at the leading edge of mobile-cloud convergence". In: *6th International Conference on Mobile Computing, Applications and Services*. 2014, pp. 1–9.
6. Carnegie Mellon University. *The Living Edge Lab*. https://openedgecomputing. org/living-edge-lab/. 2020.
7. Kiryong Ha et al. "Towards Wearable Cognitive Assistance". In: *Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services*. MobiSys '14. Bretton Woods, New Hampshire, USA: Association for Computing Machinery, 2014, pp. 68–81. ISBN: 9781450327930. DOI: 10.1145/2594368.2594383. URL: https: //doi.org/10.1145/2594368.2594383.
8. Latif U. Khan et al. *Federated Learning for Edge Networks: Resource Optimization and Incentive Mechanism*. 2019. arXiv: 1911.05642[cs.DC].
9. S. Mehraghdam, M. Keller, and H. Karl. "Specifying and placing chains of virtual network functions". In: *2014 IEEE 3rd International Conference on Cloud Networking (CloudNet)*. 2014, pp. 7–13.
10. Michel Roy, Kevin Di Lallo, and Robert Gazda. *AdvantEDGE: A Mobile Edge Emulation Platform (MEEP)*. https://github.com/InterDigitalInc/AdvantEDGE. 2020.
11. S. George et al. "OpenRTiST: End-to-End Benchmarking for Edge Computing". In: *IEEE Pervasive Computing* (2020), pp. 1–9. DOI: 10.1109/MPRV.2020.3028781.
12. Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. "Image Style Transfer Using Convolutional Neural Networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016.

