2017 IEEE 7th International Symposium on Cloud and Service Computing

Application-aware Traffic Redirection: A Mobile Edge Computing Implementation toward Future 5G Networks

Shih-Chun Huang, Yu-Cing Luo, Bing-Liang Chen, Yeh-Ching Chung and Jerry Chou

Department of Computer Science
National Tsing Hua University
Hsinchu 300, Taiwan
Email: {sjhuang,ycluo,cbl920809}@sslab.nthu.edu.tw
{ychung, jchou}@cs.nthu.edu.tw

***Abstract*—With the development of network technology, there are billions of devices accessing resources and services on the cloud through mobile telecommunication network. A great number of connections and data packets must be handled by the mobile network. It not only consumes the limited spectrum resources and network bandwidth, but also reduces the service quality of applications. To alleviate the problem, the concept of Mobile Edge Computing (MEC) has been proposed by European Telecommunications Standard Institute (ETSI) in 2014. MEC suggests to provide IT and cloud computing capabilities at the network edge to offer low-latency and high-bandwidth service. The architecture and the benefits of MEC have been discussed in many recent literature. But the implementation of underlying network is rarely discussed or evaluated in practice. In this paper, we present our prototype implementation of a MEC platform by developing an application-aware traffic redirection mechanism at edge network to reduce service latency and network bandwidth consumption. Our implementation is based on OAI, an open source project of 5G SoftRAN cellular system. To the best of our knowledge, it is also one of the few MEC solutions that have been built for 5G networks in practice.**

Keywords*—*Mobile Edge Computing, Application-aware, Traffic Redirection, SoftRAN

I. INTRODUCTION

In the past few decades, mobile devices have seen a dramatic rise due to the advent of revolutionary technological developments in the field of electronics. It was predicted by Cisco that about 500 billion devices will be connected to the Internet by 2030 [1]. As smart mobile devices become our daily needs, how to deal with the huge mobile data traffic is getting more attentions. Based on the existing Mobile Cloud Computing (MCC) architecture, many services are deployed in remote data centers to serve a large amount of users [2]. The mobile data traffic to and from mobile devices has to be transmitted by Radio Access Network (RAN). This also implicates the Mobile Network Operators (MNOs) are facing enormous pressure on mobile communication networks due to the rapid growth of network load and demand of mobile network bandwidth.

Moreover, Internet of Things (IoT) is an upcoming paradigm in the next-generation communication. It will result in a massive increase in data traffic due to the frequent network communications from a great number of IoT devices. Owing to the resource-constrained nature of IoT device, like the limited battery life and the lower processing and storage capability, these IoT devices do not have the same performance compared to the desktop or laptop. Therefore, there are a lot of computational offloading studies [3] have been proposed which offload the computational tasks to the public cloud, thereby prolonging battery life and reducing power consumption. However, computational offloading from end devices to the remote resourceful cloud may result in long propagation delays and high bandwidth consumption for data exchange in mobile network.

To overcome aforementioned problems, there are two ways to afford these tremendous mobile data traffic. One way is to upgrade the load capacity and increase the hardware equipment of mobile communication network. However, the total cost of ownership in mobile network is increasing significantly and the operator-billed revenues stay flat or even decrease over time. Revenues for MNO may be exceeded by network cost if no remedial action are taken [4]. Therefore, purchasing and upgrading the network equipment is really a heavy burden for MNOs. The other solution is to design a proper network architecture to handle these huge amounts of traffics. Mobile Edge Computing was firstly proposed by ETSI in 2014, which was defined as a new network architecture concept that provides IT and cloud-computing capabilities within the RAN in close proximity to mobile subscribers [5]. MEC network architecture is regarded as a key enabler towards to achieve the high-bandwidth and delay-sensitive requirements of future RAN architecture.

MEC architecture can provide real time radio information, such as network load and status, to the content and application developers. Moreover, MEC increases the edge processing capability and allows the computation of applications to be hosted at the edge, which reduces the network latency and bandwidth consumption of backhaul network. For MNOs, MEC architecture brings a new business model to increase their revenues by providing mobile edge platform to third-party partners. For developers, application can be deployed on the MEC platform to provide the low-latency and data intensive computing service for end user, such as tactile internet and

978-1-5386-5862-8/17 $31.00 © 2017 IEEE
DOI 10.1109/SC2.2017.11

augmented reality.

In order to shift paradigm from mobile cloud computing to mobile edge computing, and evaluate one such solution in future 5G networks, we designed and implemented a MEC framework with the OpenAirInterface (OAI), an open source project of SoftRAN for future 5G networks. Our MEC framework is a threshold-based application-aware traffic redirection mechanism for mobile edge computing to reduce end-to-end latency and backbone bandwidth consumption by enabling cloud services within the close proximity of mobile subscribers. We evaluated our 5G networks MEC prototype framework in a real-world testbed by running a streaming face detection applications from local end devices to public cloud services. The results show the application traffic can be redirected to servers at edge networks in few seconds without service interruption.

The remainder of this paper is organized as follows. In Section 2, we discuss the related work about current research of MEC, as well as software radio access network. Our prototype system overview is presented in Section 3.The design and implementation of our platform and traffic redirection mechanism are described in Section 4. Section 5 shows the setup of our experiment environment and performance evaluation. Finally, we summarize and conclude our work in Section 6.

II. Related Work

Mobile Edge Computing aims to alleviate the pressure of the core network and reduce the end-to-end latency as much as possible. It is a promising solution proposed colocating computing and storage resources at network edge. In this section, we have a review on current research results in Mobile Edge Computing at first. Furthermore, then we show the related work about the software radio access network approaches which are widely used in establishment of cellular platform.

A. The State-of-the-Art Research of MEC

Although the MEC architecture brings a lot of benefits, there is no clear definition of where MEC server should be located. Choose the location of MEC server is the first step to build up a real MEC system. FemtoCells [6] allows us to deploy computational resources in the small cell base station, which can be considered as the MEC server. But it is difficult to handle compute-intensive tasks due to the computational capability is much smaller than the MEC server which is in the macro base station. The MEC server also can be next to the Packet Data Network Gateway (P-GW). It can handle more user traffic because its higher level and larger coverage in this case. But the latency and variability will be higher than deployed at base station. Based on the above reasons, we deploy MEC server at software LTE base station.

Abdelwahab et al. proposed REPLISOM [7] which is a mobile edge cloud architecture and aims at cutting down on the fronthaul network throughput, power consumption of device, and so on through the proposed algorithm. Traditionally, the IoT devices replicate a small size of memory block to the edge cloud through wireless radio front-end of LTE environment. REPLISOM architecture suggests that the edge cloud near the base station can provide the specific virtual machine with well-defined storage and network resource for dedicated IoT application. In REPLISOM, IoT devices will regularly transmit its updated memory object to the specific IoT device, and edge cloud pulls the new memory blocks from the specific IoT device. And then, edge cloud will distribute these memory blocks to respective virtual machine. Consequently, the LTE-optimized memory blocks convergence and distribution protocol in REPLISOM can fulfill the target of both cost and throughput saving via reducing the communication times.

Beck et al. proposed Mobile Edge Computing enabled Voice over LTE (ME-VoLTE) [8] architecture which can offload the video-encoding workload to MEC edge server so that the energy consumption of mobile devices can be saved during video streaming communication. The video encoding will determine the quality of image output and is a compute-intensive work that costs a great amount of battery power. In ME-VoLTE, device uses a simple video encoding methodology with lower compression rate to encode the video in order to consume less battery power. Then, MEC server will forward video stream to remote end user after transcoding video with higher compression rate.

Nunna et al. proposed Mobile Edge Computing architecture based on 5G technologies for collaborative context-aware real time application [9]. It might be a potential challenge for collaboration with time-sensitive application owing to the high latency of wireless communication. MEC server is deployed close to each eNodeB so that the authors can realize the collaboration through the proximity service and context aware computing. The middleware MEC collaborative platform in MEC server targets to collect the important information such as location through standard API which will take the just-in-time notifications in traffic accident, remote device control scenario and so on. In this collaborative computing architecture, the key factors are low latency and synchronization of fifth generation network technology. However, the proposed architecture is just a theoretical concept because the existed limitations have not been solved.

The Mobile Edge Computing is still a novel concept that the detailed framework components in MEC server is not clearly specified and not yet realized in real-world infrastructure. The existing studies mainly focus on the benefit of application perspective and proposed theoretical notions and algorithms to verify the MEC approaches. In this paper, we aim to build up a prototype of mobile edge computing platform based on a real LTE software implementation.

B. Software Radio Access Network

To deal with the huge increase of data traffic in the fifth generation mobile network, how to keep the balance between computing performance and operational expenditure has become a critical issue for RAN deployments. The software RAN has come to the fore as the first step toward a true RAN deployment which is one of the key architecture concept to achieve a flexible, programmable and centralized controlled implementation. SoftRAN [10] is a concept which abstracts base station in a geographical area with centralized controller located in the cloud central and remote radio elements closer to end devices. SoftRAN can realize the proof of concept approach and perform greatest efficiency benefits for reconfigurable system architecture.

Karthikeyan Sundaresan et al. from NEC Labs has proposed FluidNet [11] which is a non-open source project. FluidNet is a Cloud-RAN [12] based small network with a dynamic configurable of network. It can realize centralized baseband unit (BBU) processing and simplified Radio Access Unit (RAU) design for simulation. The benefits of FluidNet are including that it can deal with diverse user and traffic profile, optimize energy saving and customize configurations for operators and technologies.

On the other hand, OpenBTS [13] is an open source software combined with Universal Software Radio Peripheral (USRP) radio front-end responsible for receiving and transmission I/Q sampling. And it can substitute the dedicated hardware provided by telecommunication operators for other general-purpose processor. However, OpenBTS only supports GSM/UMTS network technologies.

OpenAirInterface (OAI) [14] is also an open source project created by EURECOM which aims to realize the deployment of 5G cellular network stack, simplify the network upgrade, reduce the CAPEX/OPEX and increase the flexibility. The OAI platform provides the full software implementation of 4th generation mobile cellular system compliant with full protocol stack of 3GPP LTE standards and both for the RAN and the core network. It includes real-world testbed and both simulation and emulation constructions. In our work, we target to use OAI as our software RAN approach.

III. System Overview

Our mobile edge computing platform consists of three main components as shown in Fig. 1, end devices, mobile edge cloud and public cloud. In the device domain, there are not only smartphones and tablets but also various types of IoT device which connect to the eNodeB via radio interface. In the vision of 5G communication, mobile edge computing will be deployed in mobile network, but the location of MEC server is not fully specified. In our system, the MEC server is co-located with the eNodeBs pool in the base station. There are a couple of reasons for this deployment strategy. Compared to deploy the MEC server in the core network, the base station is more closer to the mobile subscribers so that we can provide lower latency between user and MEC server. The bandwidth consumption between base station and core network is also alleviated. The other reason is that this deployment of MEC architecture will be able to integrate with the BBU pool of Cloud-RAN which is an innovative architecture of 5G by centralizing the baseband unit of the base station. The public cloud offer the computing services by third-party providers over the internet, such as Amazon Elastic Compute Cloud (EC2), Google Cloud Platform and Microsoft Azure.

We use OpenAirInterface to build up the mobile communication network, including the radio access network and the core network. OAI platform is composed of OpenAirInterface5G and OpenAirCN, which are the project name for the eNodeB and the core network. OpenAirInterface5G implements a complete 3GPP LTE protocol stack of radio access network, which includes PHY, MAC, PDCP, RLC and RRC layer. Traditionally, mobile communication network only have the responsibility of forwarding user data packets. Mobile devices or IoT devices connect to the eNodeB and core network

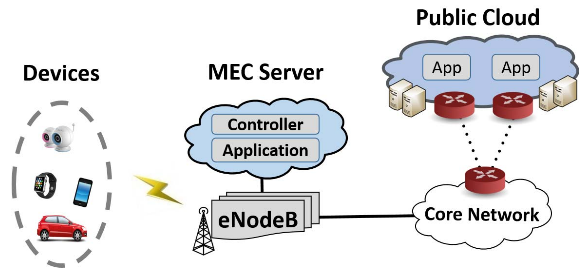

Fig. 1. The overview of our system.

through the radio interface after finish random access and attach procedure. After that, user equipment can access the application which in the public data center or somewhere else via internet. The eNodeB in LTE system is an element that take responsibility for all radio resource management functions, such as dynamic radio resource allocation and packet scheduling. When eNodeB receive user data from end device, packet will be forwarded to serving gateway (S-GW) of core network through S1-U interface by means of GPRS Tunneling protocol. We also implement an extra function in the eNodeB to redirect the user data to the application in MEC server. The core network, known as Evolved Packet Core (EPC) in LTE system, is composed of Serving Gateway and Packet Data Network Gateway for data signal processing, Mobility Management Entity (MME) for control signal processing, and Home Subscriber Server (HSS) for storing the information about all mobile subscribers of the MNO.

Our MEC server is designed to start and stop the redirection procedure and manage the resource of edge cloud. It consists of a MEC controller and some applications. In our MEC platform, we design a mechanism of threshold-based application-aware traffic redirection to provide a flexible choice to applications. Our redirection mechanism is motivated by the requirement of different types of application. The low-latency or compute-intensive application, such as augmented reality, wants to redirect their user packets to the MEC server as soon as possible. But, some application with high bandwidth consumption, such as live webcast, needs to redirect video streaming request from public cloud to MEC server while there are so much audience under the same eNodeB watching the same streamer or youtuber. At this situation, the bandwidth of backhaul network will be consumed quickly in spite of each person is watching the same video from streaming server via internet. In the MEC architecture, it only keep one video streaming from streaming server to MEC application so that users in the coverage of eNodeB are able to watch the video streaming from MEC application. We let this type application can set a threshold of application bandwidth consumption to start the redirection procedure. These application just need to redirect user requests to MEC server when the traffic is not affordable instead of redirect at the beginning. This also implicates that we can set the lower threshold to start redirection procedure as quickly as possible for the low-latency requirement application.

To realize our threshold-based application-aware traffic redirection mechanism, we design a redirection list at the MEC server for storing the registered information of the third-party application. The format of each entry is like a key-value structure which the key is the destination of application in public cloud and the value is its expected threshold of application throughput consumption. Currently, we use a pair of ip address and port number as the destination in our MEC platform. Once developers want to register their application on MEC system, they need to not only register on the redirection list but also provide the container image so that we can quickly launch their application while the redirection mechanism is triggered. Another feature of our redirection mechanism is that the service will not be interrupted while the redirection procedure is working.

IV. Design and Implementations

There are many various offloading mechanism and MEC architecture proposed by organizations and researchers. However, most of studies focus on the application layer, such as how to split the computational task and proposing theoretical offloading algorithm. There are some issues that needs to be discussed in order to build up a real MEC platform, one of them is the underlying framework of mobile edge cloud. At first, this section describes the architecture of our MEC prototype and the components of our mobile edge cloud. Then, we introduce the implementation of the application-aware function in the radio access network and entire traffic redirection workflow.

A. Architecture

It is difficult to identify the efforts and impacts of establishing transmission path between eNodeB and MEC server since the mobile edge computing architecture is still under development. Therefore, we implement an extra function and create new interfaces in OpenAirInterface to facilitate the realization of a MEC platform. We also deploy a cloud co-located with the base station based on Docker container technique so that we can provide cloud services within the close proximity of mobile subscribers. We had profiled the performance over different virtualization techniques and observed that Docker containers outperforms virtual machines in network and computational latency, and also instance launch time in our previous work [15]. Furthermore, we found that a fine-tuned low-latency kernel may significantly reduce the processing latency and deployed the eNodeB in docker container based on real-time kernel successfully.

To realize the prototype of our system, a visual representation of every components in our mobile edge cloud is shown in Fig. 2. Our mobile edge cloud has the following components: the infrastructure of cloud, the MEC controller and applications in MEC server, the eNodeB with extended functions and the interfaces between eNodeB and MEC server. The infrastructure in our mobile edge cloud provides physical and virtualized resources, such as computation, storage, I/O and network. We deploy the mobile edge cloud by several general-purpose computers, each computer is installed with low-latency kernel and Ubuntu OS. For the purpose of on-demand resource provisioning, we run the eNodeB and MEC application based on container-based platform.

The MEC server is composed by two parts, a MEC controller and MEC applications. The controller of MEC has

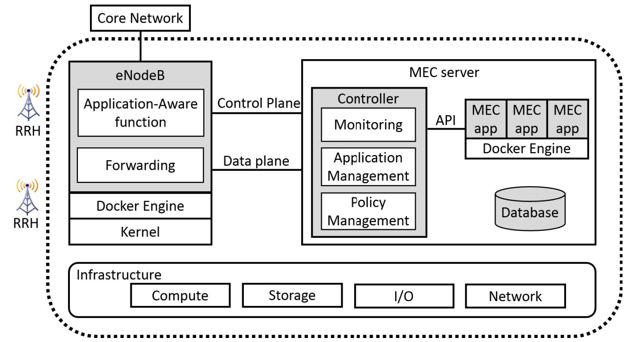

Fig. 2. Components of mobile edge cloud.

local ip address and plays the character of the management agent between MEC application and eNodeB. It makes the application developers only need to focus on specific purpose of their applications rather on the detail functionality of radio access network. There are several functions in MEC controller, monitoring, application management and traffic redirection policy management. Monitoring module provides an abstract view of radio access network by extracting the parameters of interest from eNodeB and monitor the network status of edge cloud. Moreover, it can monitor the real-time throughput consumption of each application by the statistics on received indication from eNodeB. Traffic redirection policy management module takes responsibility for maintain the redirection list and makes the decision of starting the redirection procedure. Since the eNodeB will have a replica of redirection list used by application-aware function, policy management module also has a function to update the redirection list via control plane interface while a new entry is inserted in the database. When the throughput consumption of application exceeds the threshold, this module will start the traffic redirection mechanism. Applications management module launch the corresponding container images while an event is triggered by traffic redirection policy management module. This module also take the responsibility for delete the container to free the resource when the container of application is have been a long time in idle status. The database of MEC server store the redirection list and container images of applications. In the other words, the applications of MEC in our prototype system is regarded as a container image that provided by the third-party application developers in advance. And redirection trigger function of policy management module just need launch the container of applications while the redirection mechanism is started.

We implement an application-aware function and forwarding module of the eNodeB to extend the functionality based on the original OpenAirInterface. It has the capability to recognize the destination of every packet through the eNodeB and forward user packets to the corresponding application. However, currently our mobile edge computing platform can only redirect all packets of the application when the redirection procedure is finished. Ideally, the application-aware function needs to determine the specific task of the application should be redirected or not because there are so many applications which split in different tasks in practice. In the future, we will enrich the capability of our MEC system to fulfill these requirements.

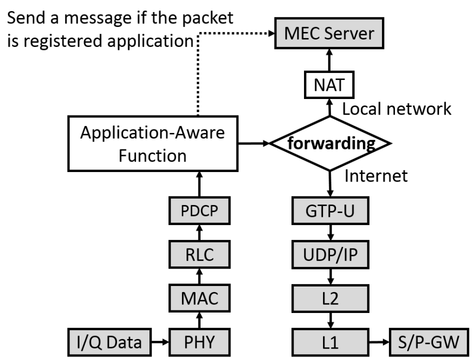

Fig. 3. Application-aware function in eNodeB.

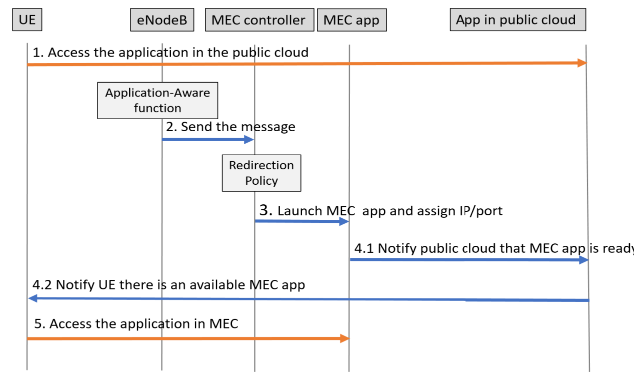

Fig. 4. Redirection workflow.

The communication interface between the radio access network and MEC server is separated in control-plane and data-plane interface in our system. Control plane interface is responsible for the transmission of the packet event message from eNodeB to MEC controller and providing the direct access for the real-time radio network information. Data plane interface is designed to forward user data packets and bring a local ip service end-point to the mobile edge cloud applications.

B. Redirection Mechanism

How to identify user data packets should be redirected or not is an open issue in mobile edge computing. In this section, we will introduce how we realize the capability of application-aware in the eNodeB. We briefly introduce the original data flow of eNodeB in OpenAirInterface from antenna to S/P-GW. We use USRP B210 as the antenna of base station with USB 3.0 connect to PC. When user equipment send a request to eNodeB, it is received by USRP through the radio interface and the radio signals is transformed in I/Q data. The eNodeB of OAI get these I/Q data through USB 3.0 port and then process from PHY layer to PDCP layer. Basically, PHY layer in charge of demodulation and CRC calculation. MAC layer take the responsibility for decomposing the MAC PDUs and de-multiplexing of data. Concatenation, segmentation and reassembly of RLC SDUs is the mainly function of RLC layer. PDCP layer provide header compression, de-compression and ciphering. Finally, the eNodeB add a GTP-U header to the front of each user packets and send it to the S/P-GW based on UDP/IP protocols.

As shown in Fig. 3, we add a new function to identify the application of user packet whether is registered on the redirecting list or not. First, the eNodeB get the user original data packet as the normal after the process of PDCP layer, then the application-aware function extract the destination ip address and port of packets by decomposing user packets. The second step is comparing the pair of destination ip and port to the redirecting list of registered application which stored in database. If the application of user request packet is registered, the eNodeB should send a notification to the MEC controller through the control plane. Otherwise, the function do nothing and send it to the next component.

Since we provide a local cloud service in the base station, the destination of user packet could be the local network or the Internet. But, there is no forwarding component between PDCP layer and GTP-U layer in the original eNodeB. All of the user packets send to S/P-GW through GTP tunnel if we do not anymore modification in the eNodeB. Therefore, we enable a forwarding component in the eNodeB by means of the extracting result of application-aware function. If the destination IP address of packet is local network, then the forwarding component forward packet to the MEC server through the data plane interface of mobile edge cloud. Otherwise, the forwarding components do nothing and send the packet to the GTP-U stack as usual. We also need to do Network Address Translation (NAT) function while the packets forward to the MEC applications. The eNodeB act as a router to forward the packets, all the packets receive from MEC applications need to process by the eNodeB process, then send to UE through USRP instead of Network Interface Cards. For this purpose, we implement a NAT function to replace the field of packet header.

The entire traffic redirection workflow of our system is shown in Fig. 4. Originally, the device sends the request to the remote public cloud. The corresponding application will handle the request and return the result back to the device as usual. In our prototype, every user data packet is processed by the application-aware function of eNodeB, it will indicate and send related parameters to the MEC controller if the application of packet request is registered on the redirecting list. Furthermore, the controller of MEC can monitor the throughput usage of each application by the statistics on received indication from eNodeB. A redirection policy component in MEC controller is to make the decision of start or stop the redirection procedure of specific application depend on the monitoring result. Once the network usage of application exceeds the predefined threshold, the application management component of MEC controller will launch the instance of application container on the edge cloud platform and assign an IP address to MEC application container. MEC Controller also provides port numbers to map the port number in the container. Then, the application in MEC need to send its information to the cloud application so that application in public cloud can tell the UE there is an available application in mobile edge. At the end, UE can directly access MEC application after finish redirection procedure.

For the purpose of getting the more flexible resource provisioning, the container of application should be turned off and release the resource while the application in idle status for a long time. When user leave the coverage of the eNodeB or shutdown the device, the network usage of application would reduce to zero in the result of monitoring component. In our prototype, we can set a threshold of idle time for closing the container to increase the resource use efficiency of mobile edge cloud.

V. Performance Evaluation

In this section, we present the setup of our real experimental environment. To show the performance of the proposed mobile edge cloud computing platform, we apply an IoT scenario to our system. Then analysis the performance of offloading procedure in our prototype.

A. Experimental Environment

The prototype of our experimental environment is set up from public cloud to end devices. We combine a scenario of face detection to our prototype, end device acts as a sensor or camera and sends photo to the application of face detection which in the public cloud. As the eNodeB recognize the packet of face detection continuously, the traffic redirection procedure is started by MEC controller.

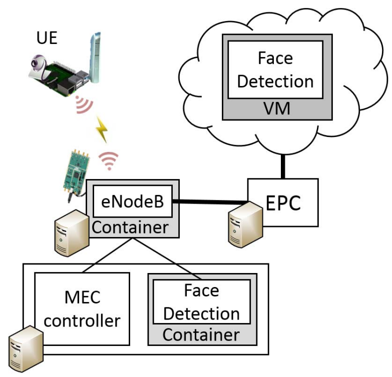

Fig. 5. Setup of our experimental environment.

As shown in Fig. 5, we deploy our face detection application in the following places as public cloud, NTHU, Seoul, US west and US east. The public cloud platform in NTHU is built on OpenStack and the others is based on the Amazon EC2. The camera is a Raspberry Pi 2 single board with a camera module and connect to the USRP of eNodeB via Huawei E3276 4G LTE dongle. To facilitate the deployment, eNodeB, MEC server and EPC are separated in three physical machines connected through a network switch. Each computer equipped with AMD A10-7850K APU at 3.7GHz (4 CPU cores) and 6GB RAM. We use Ubuntu 14.04 with low-latency kernel, and maximize the CPU frequency for performance improvement. We also turned off the power management features to achieve better stability.

B. Application Result Response Time

In our scenario, the camera receive a result of face detection with every frame. We set frame rate in 10 frames per seconds and the resolution of camera image is set up to 1920 x 1080 pixels. And the threshold of traffic redirection policy is set to 0 because this requirement of this application needs to be redirected to MEC server as quick as possible. As shown in 6, our system is successfully redirect user packets of the face detection application to our MEC application. The traffic redirection procedure takes around 3.1 seconds, 4.6 seconds, 5.3 seconds and 5.9 seconds with different place of public cloud which is NTHU, Seoul, US west and US east, respectively. The redirection procedure time is obviously related to the location of public cloud. After the redirection procedure, the application response time is significantly decreased from hundreds of millisecond to tens of millisecond so that the user have the better quality-of-experience. The traffic redirection also indicates that the network bandwidth consumption of backhaul network is reduced because the image does not need to be transmitted to core network.

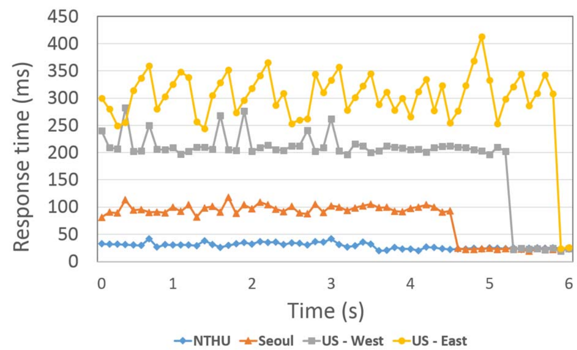

Fig. 6. Application result response time.

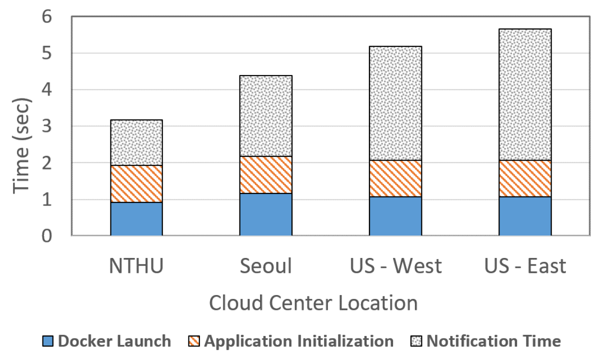

Fig. 7. Redirection procedure time.

C. Redirection Procedure Time

The time of entire traffic redirection procedure time is defined from starting redirection procedure by MEC controller to the UE can directly access MEC application. Apparently, the results of application result response time shows that the farther location of public cloud will result in a longer procedure time. As shown in Fig. 7, we analyze the traffic redirection procedure time. It is composed of three parts, the Docker container launch time, the application initialization time and the time of notification. The startup time of Docker container is almost a constant that take around hundreds of milliseconds. And the application initialization time is depend on the internal operation of each application, it almost takes around 1 second

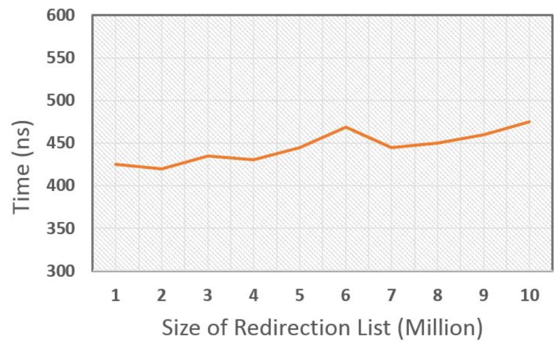

Fig. 8. Application-aware function time.

in our face detection scenario. The notification time is starting from step 4.1 to step 4.2 in redirection workflow, it is highly related to transmission distance from MEC server to cloud, we can see that takes a few seconds in our experiment.

D. Application-Aware Function Time

Since we implement an extra function in the eNodeB, every packet need to be processed by application-aware function. This function process time could be considered as an overhead in our MEC prototype. As shown in Fig. 8, we measure the execution time of application-aware function with each packet under the different redirection list size. Basically, the application-aware function executes 400 to 500 ns for each packet. Since we use hash table to store the redirection list, the processing time of checking the existence of the key usually is a constant. With the growth of the redirection list size, there is a slight increase in execution time. This is caused by the collision while inserting a large size of redirection list.

VI. Conclusion

In this paper, we build up a real mobile edge computing platform on top of a container-based environment. We also introduce the design of our mobile edge and traffic redirection workflow and the implementation of application-aware function in eNodeB. Our prototype of mobile edge computing provides computing capabilities and storage spaces and coordinates with eNodeB in close proximity to end users. Consequently, some complicated tasks of application and the large data can be forwarded to MEC server. At the end, we evaluate the performance of traffic redirection mechanism in a IoT scenario under the real environment. The result shows that it can reduce the latency of user application and it also implicates the burden of backhaul network is alleviated.

There are still many challenging issues in the MEC field that need to be solved, such as mobility management, scalability, security, standardization, and so on. Our work is just the beginning of mobile edge computing and we will enrich the functionality of our MEC system in the future.

Acknowledgement

This study was conducted under the Advanced Communication Technology Research and Laboratory Development project of the Institute for Information Industry, which is subsidized by the Ministry of Economic Affairs of the Republic of China.

References

[1] Cisco, "Internet of things." [Online]. Available: https://www.cisco.com/c/dam/en/us/products/collateral/se/internet-of-things/at-a-glance-c45-731471.pdf

[2] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: architecture, applications, and approaches," *Wireless communications and mobile computing*, vol. 13, no. 18, pp. 1587–1611, 2013.

[3] K. Kumar, J. Liu, Y.-H. Lu, and B. Bhargava, "A survey of computation offloading for mobile systems," *Mobile Networks and Applications*, vol. 18, no. 1, pp. 129–140, 2013.

[4] M. Mnakri, "The impact of otts from an operators perspective global and local dynamics." [Online]. Available: https://www.itu.int/en/ITU-T/studygroups/2013-2016/03/Documents/2016-02-miniworkshop/02-presentation-mnakri.pdf

[5] M. ETSI, "Mobile-edge computing," *Introductory Technical White Paper*, 2014.

[6] J. G. Andrews, H. Claussen, M. Dohler, S. Rangan, and M. C. Reed, "Femtocells: Past, present, and future," *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 3, pp. 497–508, 2012.

[7] S. Abdelwahab, B. Hamdaoui, M. Guizani, and T. Znati, "Replisom: Disciplined tiny memory replication for massive iot devices in lte edge cloud," *IEEE Internet of Things Journal*, vol. 3, no. 3, pp. 327–338, 2016.

[8] M. T. Beck, S. Feld, A. Fichtner, C. Linnhoff-Popien, and T. Schimper, "Me-volte: Network functions for energy-efficient video transcoding at the mobile edge," in *Intelligence in Next Generation Networks (ICIN), 2015 18th International Conference on*. IEEE, 2015, pp. 38–44.

[9] S. Nunna, A. Kousaridas, M. Ibrahim, M. Dillinger, C. Thuemmler, H. Feussner, and A. Schneider, "Enabling real-time context-aware collaboration through 5g and mobile edge computing," in *Information Technology-New Generations (ITNG), 2015 12th International Conference on*. IEEE, 2015, pp. 601–605.

[10] A. Gudipati, D. Perry, L. E. Li, and S. Katti, "Softran: Software defined radio access network," in *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*. ACM, 2013, pp. 25–30.

[11] K. Sundaresan, M. Y. Arslan, S. Singh, S. Rangarajan, and S. V. Krishnamurthy, "Fluidnet: A flexible cloud-based radio access network for small cells," *IEEE/ACM Transactions on Networking*, vol. 24, no. 2, pp. 915–928, 2016.

[12] C. Mobile, "C-ran: the road towards green ran," *White Paper, ver*, vol. 2, 2011.

[13] P. Pace and V. Loscri, "Openbts: a step forward in the cognitive direction," in *Computer Communications and Networks (ICCCN), 2012 21st International Conference on*. IEEE, 2012, pp. 1–6.

[14] N. Nikaein, M. K. Marina, S. Manickam, A. Dawson, R. Knopp, and C. Bonnet, "Openairinterface: A flexible platform for 5g research," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 5, pp. 33–38, 2014.

[15] C.-N. Mao, M.-H. Huang, S. Padhy, S.-T. Wang, W.-C. Chung, Y.-C. Chung, and C.-H. Hsu, "Minimizing latency of real-time container cloud for software radio access networks," in *Cloud Computing Technology and Science (CloudCom), 2015 IEEE 7th International Conference on*. IEEE, 2015, pp. 611–616.