Service Mobility in Mobile Networks

Hany Assasa
IMDEA Networks Institute, Madrid, Spain
Universidad Carlos III, Madrid, Spain
Email: hany.assasa@imdea.org

Srinivasa Vinay Yadhav and Lars Westberg
Cloud Core and Architecture
Ericsson Research AB, Stockholm, Sweden
Email: {vinay.yadhav, lars.westberg}@ericsson.com

***Abstract*—In the current mobile network architecture, network traffic between user equipment (UE) and services deployed on the public cloud is tromboned towards the anchor point which could lead to network congestion. Deploying services closer to the UE, for example near the eNodeB, is a potential solution. The services are deployed on small scale data centers connected to, or collocated with the eNodeB, called 'eNodeB-Cloud' (eNBC). Mobility of UEs presents a challenge for deploying services in an eNBC. When the UE is handed over from one eNodeB to another, seamless migration of UE context between the service instances running in different eNBCs needs to be ensured.**

In this paper, we propose a Platform as a Service framework to enable UE context migration between eNBCs. The architecture consists of handover signaling mechanism, TCP session migration technology, context transfer protocol and a set of APIs towards the service. An evaluation of the prototype implementation shows that on an average the time taken to migrate a UE context between two eNBCs is in the order of 12 ms, which is within the limit of handover interruption time between two eNodeBs.

***Index Terms*—eNodeB-Cloud, UE Context Migration, Service Mobility.**

I. Introduction

In recent times, there has been an increase in the number of people using mobile devices to access a variety of services on the internet. For example, traffic related to video content accessed on the mobile devices has increased rapidly, and the trend is expected to continue [1]. Increase in the number of mobile devices and network traffic generated or consumed by them presents new challenges in effectively delivering the services to the mobile devices. In the current mobile network architecture, the network traffic to and from the user equipment (UE) is tromboned towards the anchor point. Anchor points are network entities located at fixed locations in the network hierarchy [2], [3]. These anchor points maintain UE connections and relay traffic from and to them. Anchor points ease mobility management of mobile UEs, but on the other hand they constitute a potential bottleneck in the network. With the expected increases in network traffic from UEs, network congestion at anchor points due to tromboned traffic can cause a degradation in the quality of service perceived by UEs.

Different solutions have been proposed to mitigate the issues of both traffic concentration and single data plane emphasized by centralized anchor points in legacy mobile networks. These solutions depend on the distribution of the anchor points as proposed in [4], [5]. Although these solutions mitigate the problem of traffic tromboning and provide optimal data routing, they do not reduce End-to-End delay due to the inevitable long path followed to fetch contents. In addition, the distributed anchor points approach is not part of the 3GPP standard and it requires considerable amount of efforts in order to be accepted by the industrial community. Finally, handling large number of anchor points poses difficulties for operators and result in prohibitive costs of OPEX and CAPEX. In our paper, we propose a new solution that aims at deploying services closer to the UE by leveraging cloud computing capabilities. Cloud platforms allow computing resources become ubiquitous, elastic, and self-configuring. Our solution co-exists with the current 3GPP mobile network architecture.

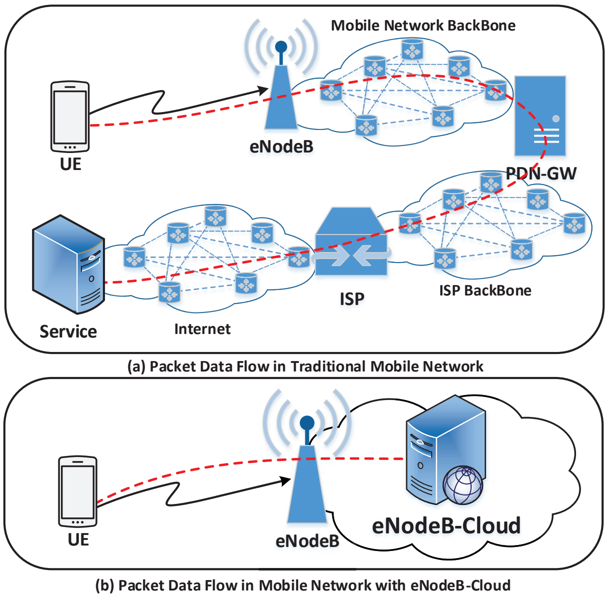

Fig. 1. Quality of Service Improvement

In our solution we bring the services closer to the UEs and near the eNodeBs. The services can be hosted on small scale data centers connected to or collocated with the eNodeB,

The author Hany Assasa did the work discussed in this paper at KTH Royal Institute of Technology, Sweden.

called 'eNodeB-Cloud' (eNBC). The difference between the traditional service deployment and the deployment of services on an eNBC is depicted in figure 1. Deploying services on eNBCs would reduce the number of network elements the UE traffic has to traverse, which results in optimal delivery of services towards the UE and could improve the quality of service. Though we can address the issue of tromboned traffic by deploying services on eNBCs, it raises the issue of seamless UE mobility without any service disruption. In the absence of a mobility system for migrating the UE contexts, when a handover of a UE occurs, the UE session associated with the service running on the eNBC is disrupted, which adversely impacts the quality of service. There are other approaches and ideas that have been proposed like Fog Computing [6], [7] in which services are pushed to the edge of the network in close proximity to the end user. While they outline the importance of service mobility, these approaches do not outline a architectural solution to address the issue.

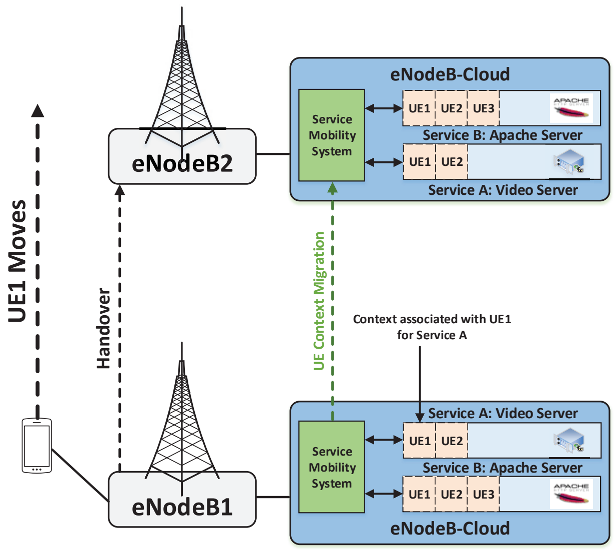

Fig. 2. Illustration of Service Mobility

Figure 2 illustrates the concept of service mobility. Each eNodeB is connected to an eNBC. The eNBC consists of several virtual machines running various services such as Video Streaming service, Apache Web service, etc. When the UE is handed over from eNodeB1 to eNodeB2, all UE contexts related to the services should migrate with the UE from the eNBC connected to eNodeB1 towards the eNBC connected to eNodeB2. The entire service mobility process is transparent to the end user, and there is no discontinuity in the services being delivered to the UE. A UE context consists of network session between the service and the UE, as well as the context related to the UE in the service. For example, in case of HTTP video server, the service context associated with a UE consists of the requested video file name and the current offset in the file. The network session in the example relates to the TCP connection between the UE and the HTTP video server.

Figure 3 shows the proposed deployment scenario for the service mobility framework, where cloud computing capabilities and networking are integrated in the next generation mobile networks architecture. In this deployment scenario, the cloud deployments are arranged in a tree topology structure. The tree structure has a root node which is the Primary Cloud-Site, intermediate nodes (Hub-Cloud), and leaves (eNodeB-Clouds). Primary Cloud-Site is the main data center where the services are hosted. Having such an architecture can help delivering services to the UE in an optimal way depending on its location.

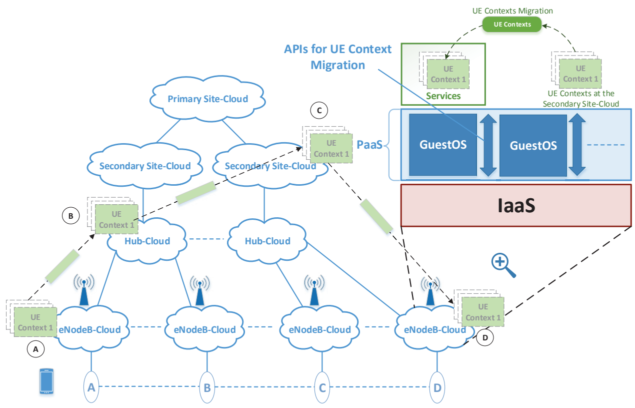

Fig. 3. Service Mobility Deployment Scenario

In the following sections, we first describe the issues in the current mobile network architectures. We then introduce a cloud based architecture, then we briefly talk about different technologies involved in our framework design. Subsequently, we propose our service mobility framework. Finally, we summarize

II. Background Study

A. *IEEE 802.21 Media Independent Handover*

The main purpose of the IEEE 802.21 is to allow the optimization of handover between heterogeneous access network technologies (including IEEE 802 and cellular technologies) without or with minimum service interruption, hence improving mobile user experience. IEEE 802.21 provides the missing and technology-independent abstraction layer capable of providing a common unified interface to the upper layers protocols, thus hiding media layer technology-specific primitives and details. This abstraction can be exploited by any upper layer protocols like IP stack to interact with the underlying technologies, thus leading to improved handover performance. The MIH provides the upper layers such as mobility management protocols with the required functionality to perform an enhanced handover. These new functions are provided by an entity called the MIH Function (MIHF) [8], [9], [10].

B. *Context Transfer Protocol (CXTP)*

CXTP RFC4076 protocol [11] is an experimental protocol proposed to enable authorized context transfers between access routers. Context transfers improve node-based mobility so that a mobile user can experience minimal service interruption. A context transfer happens when an event, such as handover, takes place. Performing a context transfer may have multiple advantages described as following:

- Conducting context transfer in advance may increase handover performance and reduce interruption time.
- Performing context transfer after a handover has occurred, is a better option than having to re-establish all the contexts at the target node from scratch.

C. *Session Migration Mechanism*

Most of the networking services today are built over TCP protocol, which provides reliable service over a non-reliable transport medium. To make a connection, a transport-layer protocol in the TCP suite needs both the IP address and the port number, at each end. This need creates a strong binding between a service and the IP address of the server providing it, during the lifetime of a TCP connection. Due to this, a TCP client will be vulnerable to any adverse conditions that may affect the TCP endpoint of the server or the network in between, after a connection is created: network congestion, failure or server overloaded. One solution to the previous problem could be by migrating the TCP/IP connection from the point of failure to a stable point. Different session migration mechanisms have been introduced to migrate TCP/IP sessions across servers such as SockMi [12], MSOCKS [13], Migratory TCP (M-TCP) [14], Reliable Network Connections [15], TCP Connection Passing (tcpcp) [16], and Service Continuations (SC) [17]. With respect to other TCP migration mechanisms, SockMi solution provides the following features:

- A connection end-point which is not involved in a migration mechanism, is not affected in any way. This implies no cooperation between the two endpoints of a connection is required.
- Ease of deployment, since it works a loadable kernel module and does not require kernel patching.
- It does not require any modification to the current TCP protocol stack.

III. System Design

Figure 4 displays the position of the service mobility system in a cloud platform. Service mobility system is designed to be a part of cloud deployment's Platform as a Service (PaaS) offering. Services that are deployed on cloud platforms that offer service mobility system can utilize its functionality through a set of API calls.

A. *General Design Framework*

Figure 5 shows different design blocks involved in the development of service mobility system. The system is composed of five functional blocks. Each block has a specific set of functions to perform. A common controller is required to allow interaction and communication between these blocks.

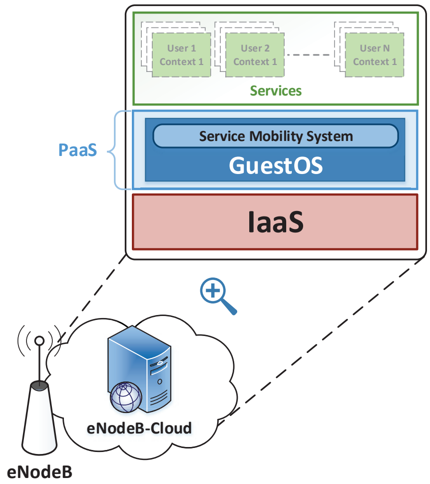

Fig. 4. Service Mobility In Deployment

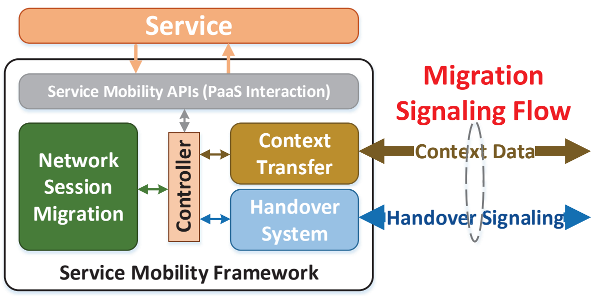

Fig. 5. Service Mobility General Design Framework Diagram

1) Handover Signaling System Block: This block is mainly responsible for initiating context migration signaling procedure. The handover signaling system should provide the following requirements:

- A handover protocol to exchange information regarding the migration procedure and details about the UE whose context is to be migrated. It is desirable to employ a standardized protocol as it facilitates interoperability across vendors.
- A handover decision-making mechanism that allows the eNBC to trigger a service migration process.

Based on these requirements, the IEEE 802.21 MIH is chosen to assist in service migration. For the implementation, the Open Dot Twenty-One (ODTONE) [18] is used to realize the functionalities and services offered by the IEEE 802.21 MIH. In this way, the scope of MIH is extended from supporting handover across various wireless access networks to provide handover signaling between cloud deployments in the mobile network architecture.

2) Network Session Migration Block: The block of network session migration is an essential part of service mobility system. It is responsible for obtaining various states and data related to the network session to be migrated during the export phase. Whereas in the import phase, it recreates all states and contents of different queues of the migrated network session in the importing server. The network session migration mechanism should be transparent and seamless from a UE's perspective. In details this means:

- The client's TCP/IP protocol stack should remain the same without requiring any modifications to its functionality.
- The modification of the connection ownership should have a minimum impact on the perceived user's quality.

For the implementation phase, the SockMi [12] session migration solution is used.

3) Context Transfer Block: This block is responsible for migrating UE context between eNBCs involved in a context migration. The block has three main tasks summarized as following:

- Encapsulation of the UE context data in a standardized migration message format.
- Extraction of UE context data from the migration message.
- Serialization and deserialization of migration data message.

The CXTP protocol is used to transmit UE context data. The CXTP provides a set of standardized messages to exchange contexts between nodes. In addition, it allows service developers to define the meaning and specifications of their service context through Feature Profile Type (FPT) codes.

4) Migration Signaling Flow Block: The migration signaling flow block is responsible for defining the sequence of signals required to migrate UE contexts between eNBCs. While defining the sequence of the signals, the following requirements should be kept in mind:

- The signaling flow should deliver information regarding the role of each eNBC involved in a service migration.
- Information regarding the status of the migration operation should be delivered to all parties.

5) Service Mobility APIs (PaaS Interaction) Block: This block provides an abstracted interface of the service mobility system through a set of API calls. Following are the requirements from the API interface:

- The API interface should be agnostic to the underlying technologies used by different blocks.
- It should be easy and straightforward for the service developer to interact with service mobility system.

B. Framework Architecture

Figure 6 shows the detailed design for implementing the proposed service mobility framework. Interactions between the different components are described below:

1) Standard interface between MIHF and MIH User as defined in the IEEE 802.21 specification.

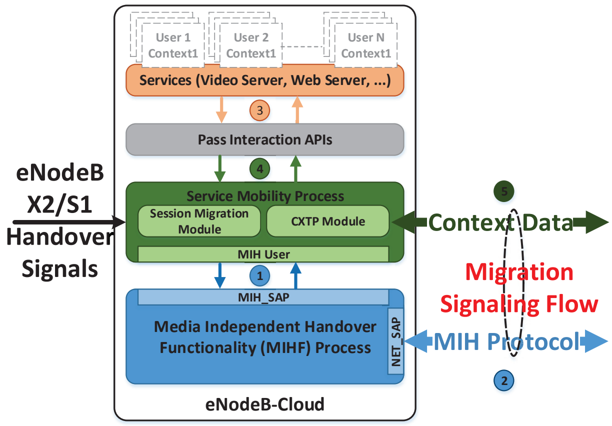

Fig. 6. Service Mobility Framework Architecture

2) Interface between local and remote MIHF entities as defined in the IEEE 802.21 specification.
3) API calls for PaaS Interaction.
4) Interface to interact with the service mobility process.
5) CXTP Protocol interface defined by RFC4076.

C. Migration Signaling Flow

Figure 7 displays the proposed migration messages sequences to carry out a UE context migration.

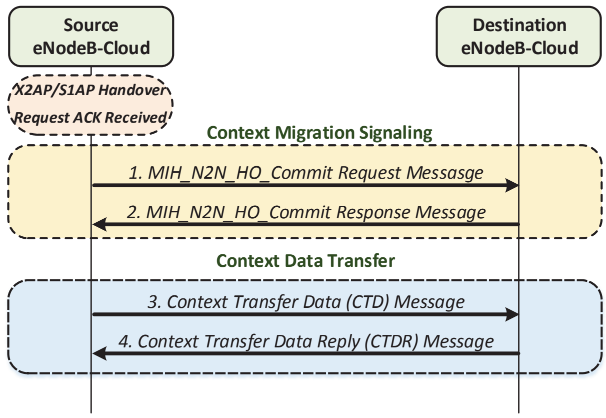

Fig. 7. Migration Signaling Flow

1) The migration process starts when the source eNodeB receives a Handover Request ACK on the X2/S1-C interface [19], [20]. This message is used to trigger the MIH-User part in the source eNodeB-Cloud. Then, the MIH-User informs its local MIHF to serialize an MIH_N2N_HO_COMMIT Request message and sends it to the destination eNBC.
2) In the destination eNBC, upon the reception of the MIH_N2N_HO_COMMIT Request message, the local MIHF sends an MIH_N2N_HO_COMMIT Indication

primitive to its local MIH-User. Later, the MIH-User tells its local MIHF to serialize an MIH_N2N_HO_COMMIT Response message. This message is sent back to the source eNBC indicating whether it can proceed with the UE context migration or not.

3) Upon the reception of the MIH_N2N_HO_COMMIT Response message in the source eNBC, the MIHF entity generates an MIH_N2N_HO_COMMIT Confirm primitive and transmits it to its local MIH-User. Depending on the value of the Status TLV field in the confirm primitive, the MIH-User decides whether to continue with the migration process or abort it. If the migration is possible, the source eNBC serializes a CXTP Context Transfer Data (CTD) message containing UE context data and sends it to the destination eNBC.
4) Finally, the destination eNBC uses the data in the received CXTP CTD message to import the UE context. If the import operation succeeds, the destination eNBC constructs a CXTP Context Transfer Data Reply (CTDR) message and sends it back to the source eNBC, confirming that the migration process has succeeded.

D. Service Mobility Interface APIs

Figure 8 shows the proposed APIs which assist cloud services to perform UE context migration. Six API calls are defined as following:

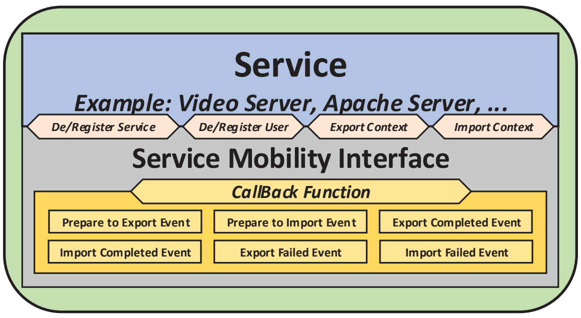

Fig. 8. Service Mobility Interface APIs

1) **Register Service API**: This API registers a callback with the service mobility system. In this way, the service can listen to various events generated by the service mobility system.
2) **De-register Service API**: Calling this API prevents the running service from listening for future events generated by the service mobility system.
3) **Register User API**: When a user connects to the service, the service should register this user with the service mobility system.
4) **De-register User API**: In contrast to Register User API, this API informs the service mobility to remove a user from its internal table.
5) **Export Context API**: This API is used by the service to export context related to a UE and also to inform about the network connections between the UE and the service.
6) **Import Context API**: It is used to import UE context.

IV. Experimental Evaluation

A. Test Setup

Figure 9 illustrates a real-world deployment scenarios of eNBC. In the test setup, we are emulating the real world deployment scenario as shown in figure 10. The test setup consists of two physical machines representing eNBCs connected to eNodeBs. The physical machines are running on Ubuntu 12.04 (Linux kernel version 3.8.0-42) operating system, and the service mobility system is deployed on both physical machines. We are using a simple video server to represent a service running on the eNBC, and we also ensure that both video servers have the same video contents.

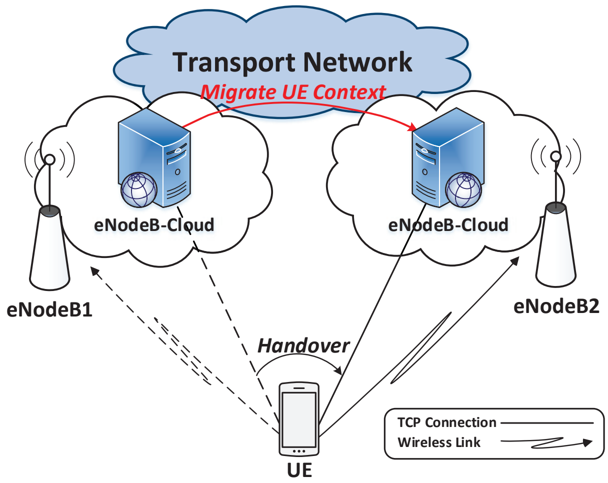

Fig. 9. Real Deployment Scenario

The Ethernet connection between the two bare metals emulates the connectivity link between eNBCs used for MIH signaling and context data transfer over CXTP. Both bare metals are running an instance of the virtual switch inside them. A GRE tunnel (reflected as a port on the virtual switches) is established between the two virtual switches over the wireless interface of the machines. The purpose of the GRE tunnel is to carry the traffic between the UE and the video server running on physical machine 2 after the handover of the UE. A virtual machine running on bare metal 1 is used to emulate the UE. This virtual machine is connected to a virtual switch on the bare metal 1. LENA LTE Simulator [21] based on network simulator (ns3) is used to run an X2-based handover scenario. We tap the X2 interface signals from the simulator during the handover scenario to trigger the service mobility system on bare metal 1.

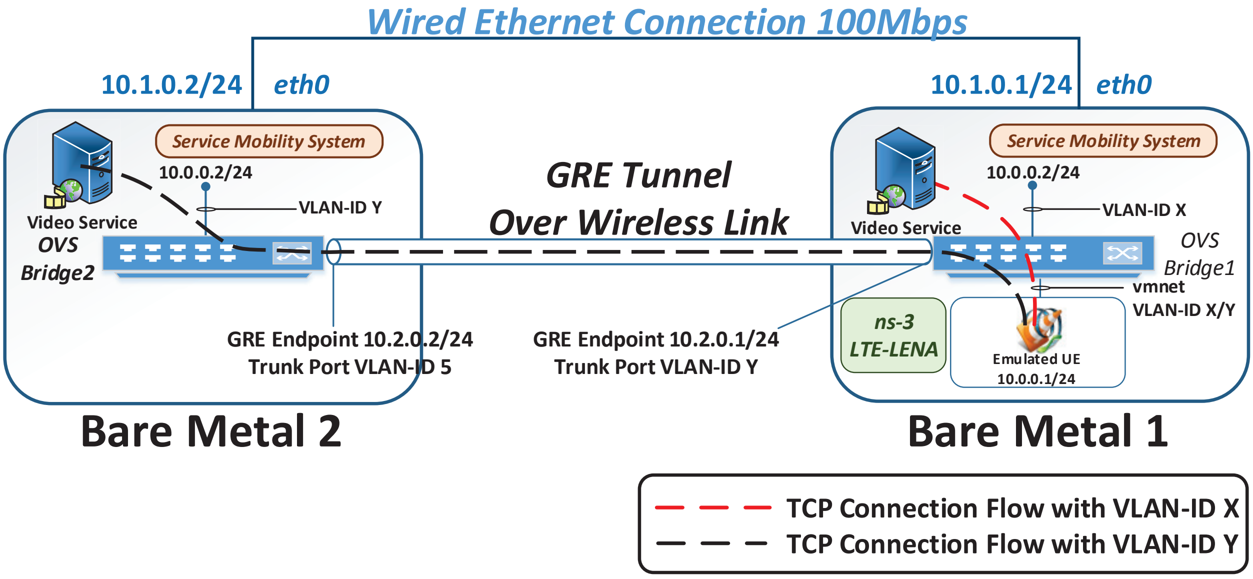

Fig. 10. Test Setup

B. Test Scenario Execution

The test is started by requesting a video file residing on the video server on bare metal 1, from a video client running inside the virtual machine (Emulated UE). An X2-based handover simulation scenario is started on the LENA simulator. When the simulator generates the X2 Handover Request ACK event in its simulation sequence, the service mobility system is triggered. This initiates a migration of video server context and TCP session context related to the UE from bare metal 1 to bare metal 2. At the same time, the traffic path from the UE is switched over the GRE connection in the virtual switch on bare metal 1, emulating a handover of the UE from one eNodeB to another. After the completion of the UE context migration, the video client on the UE continues to receive the video streaming service from the video server on bare metal 2.

C. Analysis

In our evaluation, we have measured the total time taken for migrating a UE context from one service instance to another. We have also measured the time taken by different activities involved in UE context migration. The statistics were collected over 50 runs of the test scenario described earlier. Figure 11 shows a histogram of total context migration time. Table I summaries the mean and standard deviation of the total time as well as the time taken by each of the activities involved.

From the results, we see that the overall time taken is around 12 ms on average. Requirements according to ITU-R state that the UE handover interruption time to be between 27.5 ms to 60 ms [22]. According to 3GPP requirements for support of radio resource management, a maximum interruption time of 130 ms during UE handover is suggested [23]. When an eNodeB hands over a UE to another eNodeB, the context migration times obtained in our evaluation, are within the limits of UE handover interruption times in LTE networks. This illustrates the practical feasibility of deploying services over eNBCs without significant interruption in the service during UE handover. This allows a make-before-break migration of UE context.

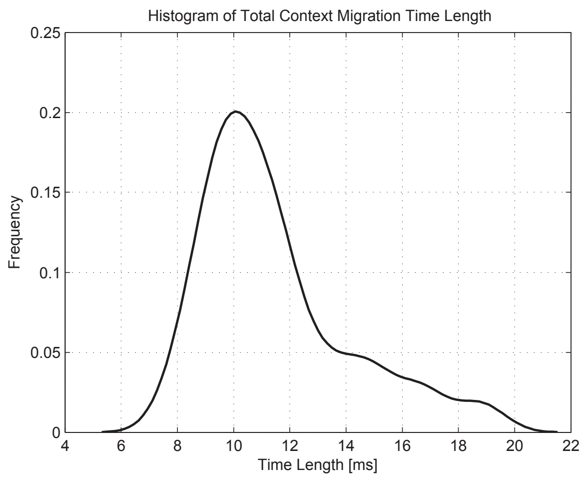

Fig. 11. Histogram of Total Context Migration Time

From Table I, it can be seen that the CXTP transfer activity consumes most of the time during a UE context migration. This activity involves creating the CXTP CTD message and transmitting it from source eNBC to destination eNBC over a single TCP connection. The context data transmitted in a CTD message comprises of network session data and service

Performance Value/Statistics	Mean [ms]	STD [ms]
MIH HO Commit Transaction Time	3.4935	0.6552
Retrieving Application Data Time	0.4156	0.0839
Retrieving Sockets Data Time	1.1765	0.3094
CXTP Transaction Time	6.2195	2.3600
Total Migration Time	11.4726	2.6538

TABLE I
STATISTICS RESULTS FOR THE TOTAL UE CONTEXT MIGRATION COMPONENTS

context data.

In our evaluation, we have also measured the size of the UE context data to be migrated, which consists of both the network session data and service context data. The video streaming service used in the evaluation has a fairly small service context data size and a relatively large network session data size per UE session. The sizes in each of the two part of the UE context data depends on the service. Services requiring high network bandwidth like streaming could have large network session data and services that have complex internal representation of states could have large service context data. The graph in Fig 12 depicts the histogram of total migration data length. On average, the total length of a single UE context data that needs to be migrated is around 75 KB, and the maximum length, is around 100 KB. These statistics can vary based on the services that are hosted on the eNBC and it is important to consider them while dimensioning the bandwidth on the connectivity link between the eNBCs.

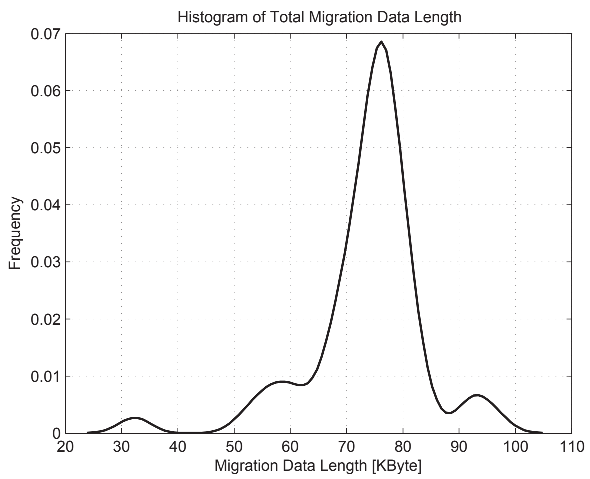

Fig. 12. Histogram of Total Migration Data Length

V. CONCLUSION

In this paper, we have presented the concept of Service Mobility, which enables seamless migration of UE context from one service instance to another. Service Mobility System is designed to be offered as a part of the cloud platforms. Service providers can leverage on the system to deploy services on eNodeB-Clouds in order to bring the services closer to the UE. We have discussed the architecture of the Service Mobility System and evaluated a prototype implementation of the same. The design of the system uses standard interfaces such as MIH and CXTP, which allows interoperability between different vendors and, as a result, eases its adoption and proliferation. The results obtained from the evaluation indicate the feasibility of deploying services on eNodeB-Clouds along with the Service Mobility System. The results show that the UE context migration times between service instances are within the limits of UE handover interruption times.

VI. FUTURE WORK

In this paper, the service mobility framework was evaluated using simple HTTP video server and having one client with one network session. However, a broad range of complex services could be deployed inside eNBCs including but not limited to HTTP proxy service, Apache web service, augmented reality, and online gaming. We plan to adapt some of these complex applications mentioned above to run over the developed service mobility system inside eNBCs. We are currently working on providing multi-language bindings for the service mobility interface APIs which make it easy for applications written in different programming languages like Python and Java to interact with our framework. In addition, the future activities will try to replace the Linux Kernel Module used in this framework with built-in native tools available with latest version of Linux kernel such as crtools [24]. Finally, the impact on UE context migration time when multiple services are hosted on eNBCs needs to be studied.

VII. ACKNOWLEDGMENT

The authors would like to thank Azimeh Sefidcon, Daniel Turull and András Zahemszky from Ericsson Research for their comments and feedback.

REFERENCES

[1] Ericsson. Ericsson Mobility Report On The Pulse Of The Networked Society, 2014. http://www.ericsson.com/res/docs/2014/ericsson-mobility-report-june-2014.pdf.

[2] 3GPP. Network architecture. TS 23.002, 3rd Generation Partnership Project (3GPP), September 2014.

[3] 3GPP. General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) access. TS 23.401, 3rd Generation Partnership Project (3GPP), September 2014.

[4] A. Yegin, Jungshin Park, Kisuk Kweon, and Jinsung Lee. Terminal-centric distribution and orchestration of ip mobility for 5g networks. *Communications Magazine, IEEE*, 52(11):86–92, Nov 2014.

[5] W. Hahn. 3gpp evolved packet core support for distributed mobility anchors: Control enhancements for gw relocation. In *ITS Telecommunications (ITST), 2011 11th International Conference on*, pages 264–267, Aug 2011.

[6] Cisco. Fog computing, ecosystem, architecture and applications. http://www.cisco.com/web/about/ac50/ac207/crc_new/university/RFP/rfp13078.html.

[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] Antonio de la Oliva, Albert Banchs, Ignacio Soto, Telemaco Melia, and Albert Vidal. An overview of ieee 802.21: media-independent handover services. *IEEE Wireless Commun.*, 15(4):96–103, 2008.

[9] Ieee standard for local and metropolitan area networks - media independent handover services. *IEEE Std 802.21-2008*, pages 1–0, Jan 2009.

[10] P.A Shah and M. Yousaf. End-to-end mobility management solutions for tcp: An analysis. In *Computer and Information Technology, 2008. ICCIT 2008. 11th International Conference on*, pages 696–701, Dec 2008.

[11] J. Loughney, M. Nakhjiri, C. Perkins, and R. Koodli. Context transfer protocol (cxtp). RFC 4067 (Proposed Standard), July 2005.

[12] Massimo Bernaschi, Francesco Casadei, and Paolo Tassotti. Sockmi: a solution for migrating tcp/ip connections. In *PDP*, pages 221–228. IEEE Computer Society, 2007.

[13] Pravin Bhagwat, David A. Maltz, and Adrian Segall. Msocks+: an architecture for transport layer mobility. *Computer Networks*, 39(4):385–403, 2002.

[14] Florin Sultan, Kiran Srinivasan, Deepa Iyer, and Liviu Iftode. Migratory tcp: Connection migration for service continuity in the internet. In *ICDCS*, pages 469–470, 2002.

[15] Victor C. Zandy and Barton P. Miller. Reliable network connections. In Ian F. Akyildiz, Jason Yi-Bing Lin, Ravi Jain, Vaduvur Bharghavan, and Andrew T. Campbell, editors, *MOBICOM*, pages 95–106. ACM, 2002.

[16] Werner Almesberger. Tcp connection passing.

[17] Florin Sultan, Aniruddha Bohra, and Liviu Iftode. Service continuations: An operating system mechanism for dynamic migration of internet service sessions. In *SRDS*, pages 177–186. IEEE Computer Society, 2003.

[18] Daniel Corujo, Carlos Guimraes, Bruno Santos, and Rui L. Aguiar. Using an open-source ieee 802.21 implementation for network-based localized mobility management. *IEEE Communications Magazine*, 49(9):114–123, 2011.

[19] 3GPP. Evolved Universal Terrestrial Radio Access Network (E-UTRAN); X2 Application Protocol (X2AP). TS 36.423, 3rd Generation Partnership Project (3GPP), September 2014.

[20] 3GPP. Evolved Universal Terrestrial Radio Access (E-UTRA) ; S1 Application Protocol (S1AP). TS 36.413, 3rd Generation Partnership Project (3GPP), September 2014.

[21] CTTC. Lena: Lte-epc network simulator. http://networks.cttc.es/mobile-networks/software-tools/lena/.

[22] ITU-R. Requirements related to technical performance for IMT-Advanced radio interface(s). Technical Report M.2134, International Telecommunication Union (ITU), 2008.

[23] 3GPP. Evolved Universal Terrestrial Radio Access (E-UTRA); Requirements for support of radio resource management. TS 36.133, 3rd Generation Partnership Project (3GPP), October 2014.

[24] CRIU: Checkpoint/Restore In Userspace. http://criu.org/Main_Page.