# MigraMEC: Hybrid Testbed for MEC App Migration

Prachi V. Wadatkar
prachi.v.wadatkar@uis.no
University of Stavanger
Stavanger, Norway

Rosario G. Garroppo
rosario.garroppo@unipi.it
University of Pisa
Pisa, Italy

Gianfranco Nencioni
gianfranco.nencioni@uis.no
University of Stavanger
Stavanger, Norway

## ABSTRACT

Multi-access Edge Computing (MEC) enhances the capabilities of 5G by enabling the computation closer to the end-user for real-time and context-aware services. One of the main challenges of MEC is the migration of the MEC application in the presence of user mobility. MigraMEC is a hybrid testbed that simulates the network scenario and user mobility and emulates the MEC framework by using AdvantEDGE. Moreover, MigraMEC implements two physical MEC Hosts (MEHs) by using an extended version of Kubernetes (K8s). Finally, the MigraMEC controller interacts with both the emulative and experimental environments to ensure an efficient migration of the MEC application. Based on network information, the MigraMEC controller not only enforces the MEH where the MEC application is running but also the Point of Access (PoA) to which the user is connected. In our demonstration, the MEC application is a video streaming service, and the results highlight the need for multiple MEHs and efficient migration to maintain a high user experience.

## CCS CONCEPTS

• **Networks** → Network experimentation; **Cloud computing**.

## KEYWORDS

MEC, Migration, AdvantEDGE, Kubernetes

**ACM Reference Format:**
Prachi V. Wadatkar, Rosario G. Garroppo, and Gianfranco Nencioni. 2023. MigraMEC: Hybrid Testbed for MEC App Migration. In *Proceedings of 29th Annual International Conference On Mobile Computing And Networking (ACM MobiCom '23).* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3570361.3614069


*ACM MobiCom '23, Oct 02–06, 2023, Madrid, Spain*

ACM ISBN 978-1-4503-9990-6/23/10…$15.00
https://doi.org/10.1145/3570361.3614069

## 1 INTRODUCTION

The Fifth Generation (5G) of mobile networks enables new advanced services, such as Ultra-Reliable Low-Latency Communications (URLLC), where the end-to-end delay between the user and the application must be very low [4]. Multi-access Edge Computing (MEC) allows to achieve low latency, real-time awareness of the local environment, cloud offloading, and the reduction of the traffic congestion [5].

ETSI is standardizing MEC [6]. Among the tools available in the ETSI MEC Ecosystem, AdvantEDGE [8] empowers application developers to test MEC services by creating network models and facilitates the study of network characteristics and their impact on applications and services.

This work focuses on the availability of a MEC application running on a Multi-access Edge Host (MEH) considering the mobility of the User Equipment (UE). In such mobile scenario, the migration from one MEH to another MEH may become necessary and can be accomplished by using containers or pods, which are units used for deploying containerized applications [7, 10]. This work considers the pod migration, since it is preferred over container migration due to the orchestration and management capabilities provided by Kubernetes (K8s) [13]. However, the default K8s framework does not support manual pod migration; a migration is considered only when nodes face resource constraints or health issues. Pods can be migrated by using an extended K8s [12] that has been implemented in our testbed platform, called MigraMEC, to enable a seamless MEC application migration. Moreover, MigraMEC includes a controller that retrieves the information from the ETSI-MEC APIs implemented by AdvantEDGE, such as ETSI MEC 013 [3] Location and ETSI MEC 012 [2] Radio Network Information, and applies the decisions to the MEHs by using K8s.

The MigraMEC has been evaluated by using the Multi-objective Dijkstra Algorithm (MDA) to select the best Point of Access (PoA) and MEH based on various metrics. Once the MEH is selected, the MigraMEC controller facilitates the migration of the MEC application between MEHs.

## 2 SYSTEM DESIGN

Figure 1 presents a schematic overview of the MigraMEC testbed, showing the interaction between the controller, AdvantEDGE, and K8s. The NUCs have an Intel i7 with (32/512)

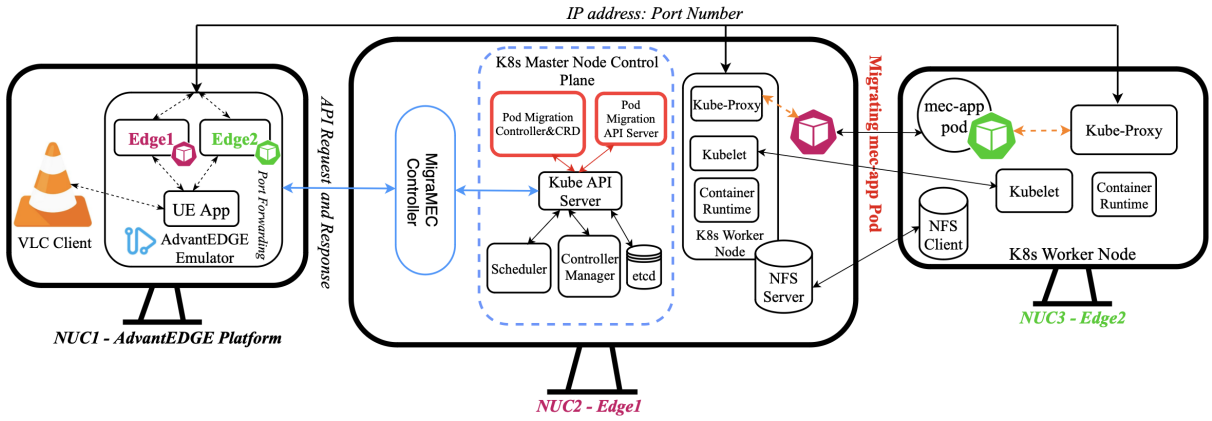


**Figure 1: Schematic Representation of the MigraMEC Testbed**

GIGABYTE. The AdvantEDGE platform is deployed on NUC1, which also serves as the host for a user VideoLAN Client (VLC) [9]. Initially, the MEC application, named `mec-app` and which is a VLC server, is deployed on NUC2 (MEH named `Edge1`) and later migrated to NUC3 (MEH named `Edge2`) based on the location of the UE. NUC2 also serves as the master node within the K8s cluster. Instead, NUC3 is a worker node within the same cluster.

In addition to the default K8s cluster framework, NUC2 and NUC3 incorporates extended version of K8s and Container Runtime Interface (CRI) [11], which is supported by CRIU project [1]. Extended K8s and CRI version is necessary to facilitate the seamless pod migration. The `mec-app` is a VLC server application deployed as a pod that runs within a container. By utilizing K8s service and Kube-proxy mapping, the `mec-app` is made accessible to end-users (in this case, mapped to the AdvantEDGE emulator) through an IP address and port number. `Pod-migration controller` and API server are part of the extended K8s framework that controls and tracks the pod migration activities, respectively. Kube API server interacts with K8s elements and operator (in this case, MigraMEC controller). The NFS server and client are used to share the pod checkpoint information during the migration. The controller manager, scheduler, etcd, are default elements of the K8s cluster responsible for node control, pod scheduling activities, and functioning as a database for cluster information, including authentication keys.

## 3 DEMONSTRATION

The considered network scenario consists of two sets of `WiFi`, `5G`, and `4G` PoAs with coverage radius of 200, 500, 1000 meters, respectively. Network characteristics, such as latency, jitter and packet loss, have been set. Furthermore, PoAs and UE have their geographical locations and an actual map representation can be created by AdvantEDGE.

During the demonstration, we will enable the UE mobility by using AdvantEDGE automation. We will show how MDA will select the best PoA and MEH based on the UE location.

Our demonstration focuses on evaluating the performance of the system through two experimental tests. First test corresponds to having only single MEH (i.e., `Edge1`). The absence of an MEH (i.e., `Edge2`) noticeably degrades the video quality, resulting in visible blurriness, while VLC statistical values indicate the loss of frames. Moreover, the absence of an MEH significantly affects the latency and throughput between the MEH and UE, especially when compared to the VLC server throughput. This impact is primarily due to the longer path between the UE and MEH, resulting in increased latency and decreased throughput for MEH-UE communication.

Having two MEHs that support the `mec-app` migration showcased superior performance compared to a single MEH. During UE movement, the `mec-app` was seamlessly relocated to the nearest MEH depending on UE location, resulting in an enhanced Quality of Experience (QoE) for the user. MEH-UE latency and throughput were optimized in comparison to the single MEH test. While migrating the `mec-app` between MEHs, there was a brief buffering period for video frames as the `mec-app` transitioned and restoration took place. However, the average buffering time was less than 3 seconds, considering that the VLC client is typically set with a default buffering value of 1 second.

In the future, the MigraMEC controller will be further developed to not only retrieve network information from AdvantEDGE, but also computing information from K8s.

## ACKNOWLEDGMENTS

This work was partially supported by the NFR through the 5G-MODaNeI project (no. 308909) and the Italian Ministry of Education and Research (MIUR) in the framework of the FoReLab project (Departments of Excellence).

# REFERENCES


[1] CRIU. [n. d.]. A project to implement checkpoint/restore functionality for Linux. https://github.com/checkpoint-restore/criu. (Accessed: 2022-05-07).

[2] ETSI. 2019. GS MEC 012 V2.1.1: Multi-access Edge Computing (MEC); Radio Network Information API.

[3] ETSI. 2019. GS MEC 013 V2.1.1: Multi-access Edge Computing (MEC); Location API.

[4] ETSI. 2020. TS 123 501 V16.6.0: 5G; System architecture for the 5G System (5GS) (3GPP TS 23.501 version 16.6.0 Release 16).

[5] ETSI. 2022. ETSI TS 123 304 V17.3.0: 5G; Proximity based Services (ProSe) in the 5G System (5GS) (3GPP TS 23.304 version 17.3.0 Release 17).

[6] ETSI. 2022. GS MEC 003 V3.1.1: Multi-access Edge Computing (MEC); Framework and Reference Architecture.

[7] Mohammed A. Hathibelagal, Rosario G. Garroppo, and Gianfranco Nencioni. 2023. Experimental comparison of migration strategies for MEC-assisted 5G-V2X applications. *Computer Communications* 197 (2023), 1–11. https://doi.org/10.1016/j.comcom.2022.10.009

[8] InterDigitalInc. 2023. AdvantEDGE on github. https://interdigitalinc.github.io/AdvantEDGE/. (Accessed: 2022-12-02).

[9] Video LAN organization. [n. d.]. Video LAN website. https://www.videolan.org/index.it.html. (Accessed: 2022-05-07).

[10] Jakob Schrettenbrunner. 2020. *Migrating Pods in Kubernetes*. Ph. D. Dissertation. https://doi.org/10.13140/RG.2.2.31821.97762

[11] Schrettenbrunner, Jakob. [n. d.]. containerd-cri. https://github.com/schrej/containerd-cri. (Accessed: 2022-05-07).

[12] SSU-DCN. [n. d.]. podmigration-operator. https://github.com/SSU-DCN/podmigration-operator/blob/main/init-cluster-containerd-CRIU.md. (Accessed: 2022-05-07).

[13] The Kubernetes Authors. [n. d.]. Kubernetes. https://kubernetes.io. (Accessed: 2022-05-07).