Student thesis proposal: A study on scaling and load balancing for 5G core network using Kubernetes

November 30th, 2019

1 Objective

Investigate and develop new scaling or load balancing strategies for the 5G Core network, in particular its control plane functions (e.g., MME), in a container-based environment orchestrated by Kubernetes.

2 Background

Scaling refers to adding or removing resources from a system to increase the system resource efficiency as well as the system performance. Load balancing refers to efficiently distributing incoming network traffic across a group of back-end servers (aka a server pool). A good load balancing algorithm will help increase the overall system performance. Therefore, these play a crucial role in providing scalability and efficiency of any system such as cloud-based systems.

Taking advantages of virtualization, telco operators and providers have been migrating their network functions from running on dedicated specialized hardware to virtual applications or services on top of cloud infrastructure. The virtual applications or services can be deployed on Virtual Machines (VM) managed by cloud manager tools such as OpenStack [2] or Containers managed by container orchestration systems such as Kubernetes [1]. Currently, containers have become a dominant force in cloud-native development due to its light-weight, less starting time, and native performance over VM-based approaches.

The goal of this thesis is to first examine how well we can do autoscaling and load balancing for network functions of 5G core network in a cloud-native way using Kubernetes and then derive a new improved scaling/load balancing solution.

3 Research Work

- Deploy Open5GCore on Kubernetes (K8s)
- Run experiments with the K8s existing scaling solutions such as horizontal pod autoscaling (HPA), etc.
- Literature review of the existing solutions for container-based horizontal scaling load balancing
- Propose and implement an improved scaling/load balancing of MMEs
- Testing with the Open5GCore benchmarking tool with the control plane traffic

4 Requirements

- Prerequisites: Python programming, basic container/docker, Linux
- Deliverables: scaling/load balancing operations of one of Open5Gore components on K8s
- Duration: 5-6 months

References

[1] Kubernetes. https://kubernetes.io/.

[2] Openstack. https://www.openstack.org/.