1. Follows Quan’s TUTORIAL ON SETUP KUBEEDGE ENVIRONMENT ON CLOUD AND EDGE

-------------------------------------------------------------------------------------------------------

2. Use Phuc’s scheduler-extender and scheduler-extender-config.yaml, scheduler-extender-policy.json file and replace the kube-scheduler

2-1. Since scheduler extender is implemented as a webhook call, so first we need to create a server which serves for the extender calls from the kube-scheduler whenever new pod is going to be scheduled.

2-2. Execute the thpa-config/k8s-scheduler-extender file to start the server: ./k8s-scheduler-extender

2-3. copy files scheduler-extender-config.yaml and scheduler-extender-policy.json in thpa-config directory to /etc/kubernetes/

2-4. copy file thpa-config/kube-scheduler.yaml to /etc/kubernetes/manifests/

2-5. Restart kube-scheduler pod on master node.

3. Refer to <https://edgemesh.netlify.app/> , and install edgemesh, follows manual install, not helm install

4-1. In testbed, Use example yaml file in <https://github.com/kubeedge/edgemesh/tree/main/examples> = Round-robin load balancing

4-2. Local scheduling uses different yaml files to provide different services for each node to be distributed. For example, hostname-lb-roundrobin 1,2,3.yaml

5. Refer to <https://github.com/rakyll/hey> , <https://hub.docker.com/r/williamyeh/hey/> , and use a docker traffic generator by hey tool

6. Do experiments. Distribute the hostname-lb-roundrobin 1,2,3 files from (4-2.) to each edge node, and then transform the number of simultaneous users per node with a docker traffic generator from (5.) to obtain throughput and latency.

\*If the topology configuration is unstable due to an error, implement the topology again step by step, resetting it in the order you installed it recently.