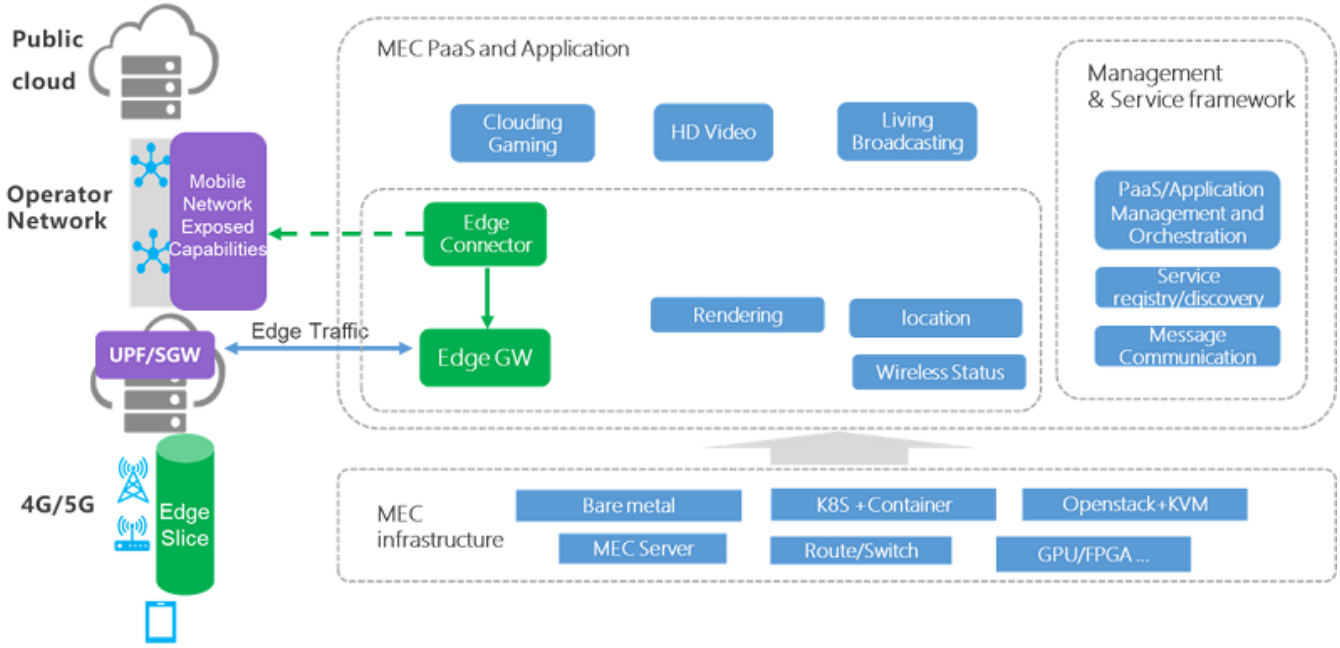
R3 - Architecture Documentation

5G MEC/Slice BP Architecture



The 5G MEC BP mainly consists of two network elements,

- Edge connector, deployed in the cloud to enable flexible traffic offloading from the aspects of control interaction with mobile network exposed capabilities, and to subscribe the edge slice between UE and edge application.
- Edge gateway (GW), deployed close to the 4G/5G network edge to enable the traffic offloading from the aspects of data plane with local traffic routing, traffic management, etc.

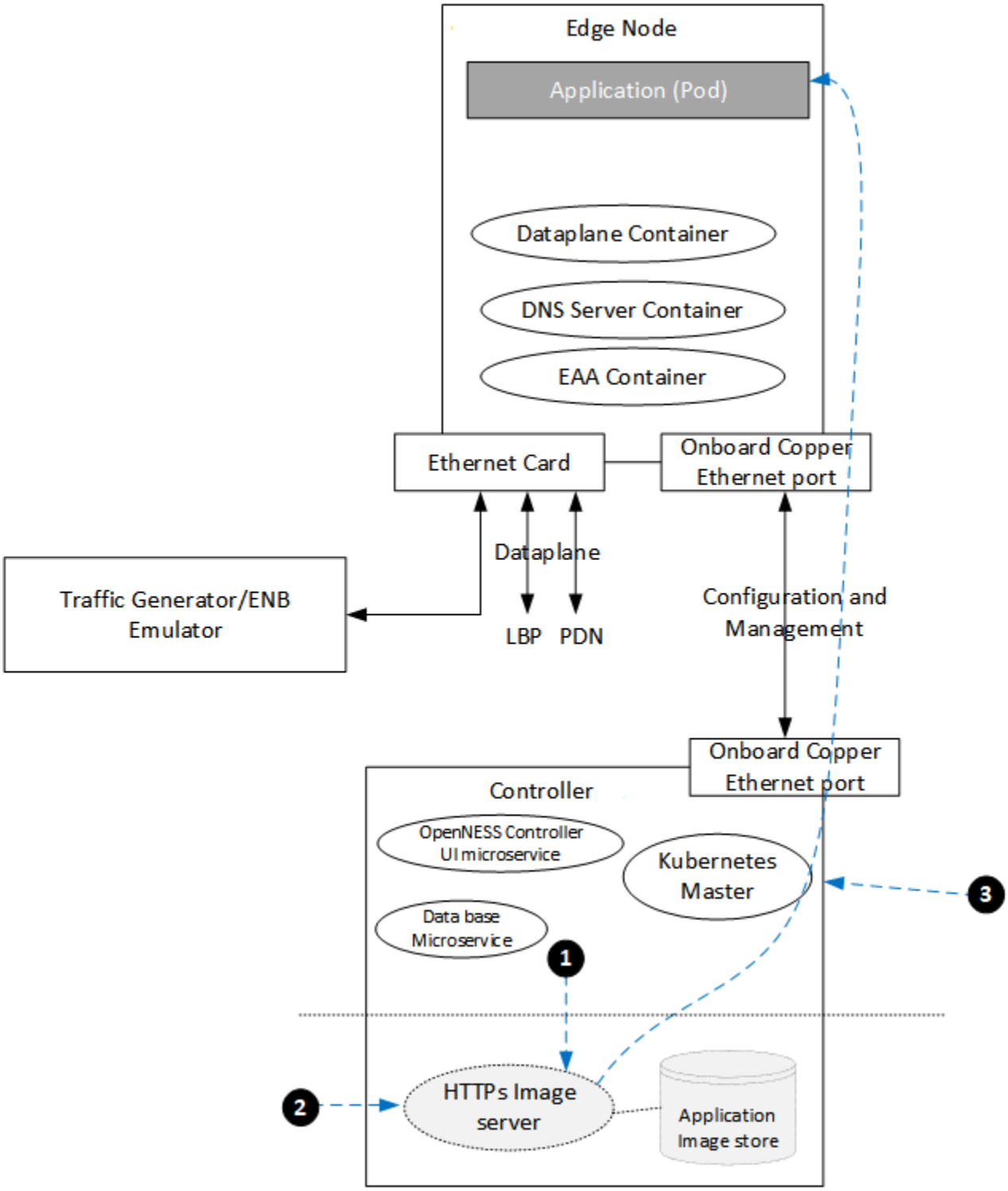
Edge Connector and Edge GW are respectively built on top of OpenNESS Edge Node and OpenNESS Controller Community Edition, with a couple of new components described in the following subsections to meet the requirements of cloud service providers (CSP).

Edge Connector

- **Offloading management**: Inject offloading rules to the cellular network (e.g., LTE/CUPS, 5G) control plane.
- **Telemetry**: Get basic edge compute microservices telemetry from connected Edge GW.
- **Application lifecycle management**: Support applications through their lifecycle:
  - Expose the silicon micro architecture features on CPU, Accelerator, Network interface etc. through Enhanced Platform Awareness (EPA) framework to the applications for lower overhead and high performance execution.
  - Deploy edge compute applications from the image repository
  - Configure the Edge compute application specific Traffic policy
  - Configure the Edge compute application specific DNS policy
- **Slice management**: Subscribe network slice for a particular group of UEs, or any UE accessing a specific edge service.

Application onboarding

5G MEC BP users need to use the Kubernetes Master to onboard an application to the Edge Node. BP supports applications that can run in a docker container. Docker image tar.gz. The image source can be a docker registry or HTTPS image repository. The image repository can be an external image server or one that can be deployed on the controller. The figure below shows the steps involved in application onboarding.



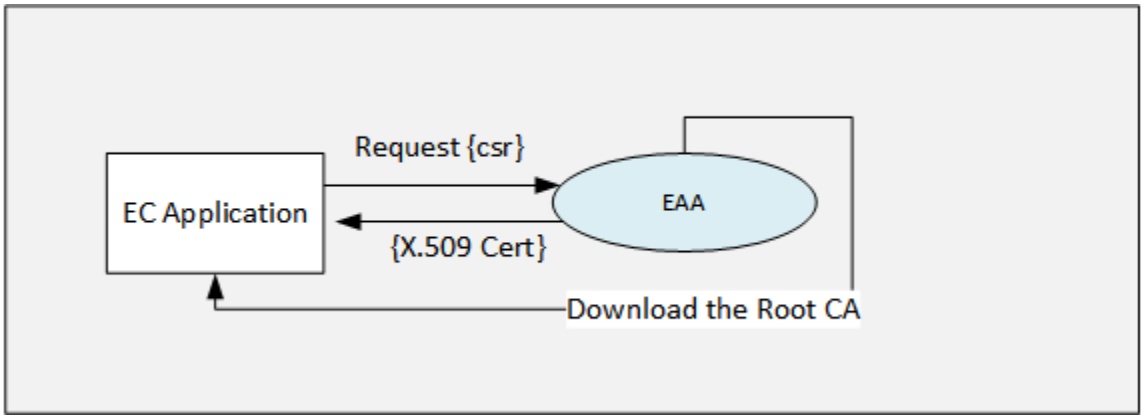
1. User sets up the HTTPS based Application image server / Docker Registry where application container image is stored.
2. User uploads the application image (container tar.gz) to the HTTPs server. User downloads the application container image to the Edge Node.
3. User initiates the Application deploy step using Kubernetes (kubectl).

Edge GW

Edge GW takes advantage of a couple of microservices from OpenNESS, such as EAA. A few new microservices namely Traffic management and Local DNS are introduced to operate and manage edge applications from the perspective of CSP.

Details of Edge GW microservices functionality:

- **Edge Node Enrollment**: Edge Node enrollment is supported using K8s cluster join.
- **Dataplane Service**: Steers traffic towards applications running on the Edge Node or the Local Break-out Port.
    - Using OVN/OVS as Dataplane - recommended data plane when incoming and outgoing flows are based on pure IP
        - Implemented using kube-ovn
        - Provides IP 5-tuple based flow filtering and forwarding
        - Same Interface can be used for Inter-App, management, Internet and Dataplane interface
- **Application Authentication**: Ability to authenticate Edge compute application deployed from Controller so that application can avail/call Edge Application APIs. Only applications that intend to call the Edge Application APIs need to be authenticated. TLS certificate based Authentication is implemented.



- **Local DNS service**: Forward the domain name resolution request to the corresponding authoritative DNS server, and insert accurate location information to facilitate name translation
    - Forwarding of DNS request/response
        - Receive the domain name resolution request from the operator's network, replace the source address with the gateway address, and replace the destination address with the corresponding authoritative DNS address before forwarding
        - Receive a resolution response from the authoritative DNS, exchange the source and destination addresses back to the original address and forward it to the operator's network
    - Parse of DNS response - After receiving the domain name resolution response, do DPI of the response to obtain the mapping of edge application address and the domain name, and then inject traffic offloading rules into the operator network with the help of the Edge Connector
    - Support DNS resolution and forwarding services for the application deployed on the edge compute. DNS server is implemented based on Go DNS library. DNS service supports resolving DNS requests from User Equipment (UE) and Applications on the edge cloud
- **Traffic management**: Count how many bytes go to which edge services, and provide statistics for charging.