PCEI Use Case Development

User Plane Function Distribution and Local Break-Out

- UPF Distribution/Shunting capability -- distributing User Plane Functions in the appropriate Data Center Facilities with qualified compute hardware for routing the traffic to desired applications and network/processing functions/applications.
- Local Break-Out (LBO) – Examples: video traffic offload, low latency services, roaming optimization.

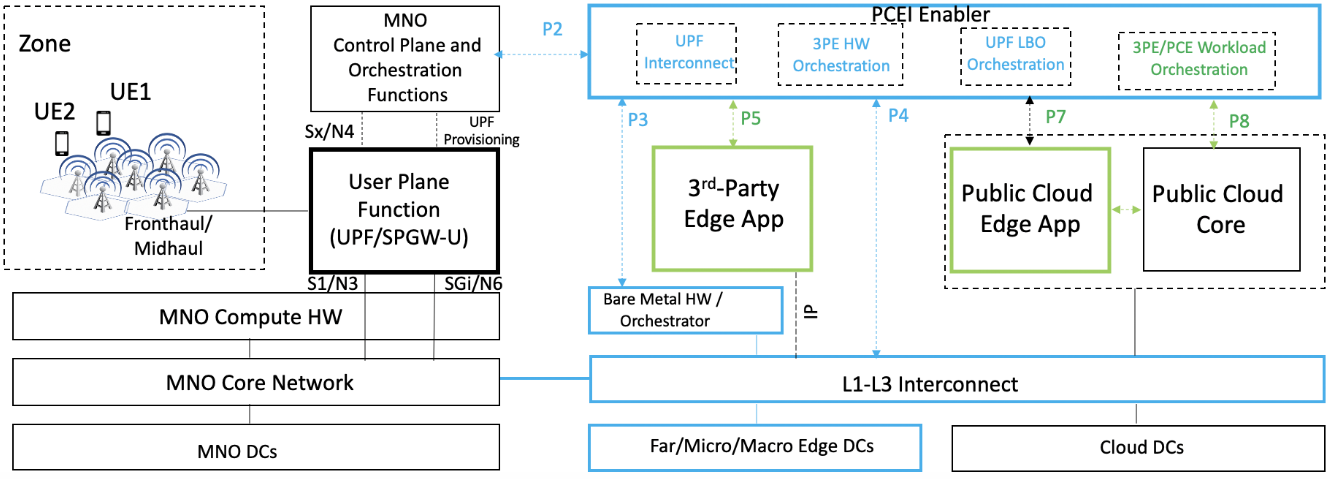
The UPF Distribution use case distinguishes between two scenarios:

- UPF Interconnection. The UPF/SPGW-U is located in the MNO network and needs to be interconnected on the N6/SGi interface to 3PE and/or PCE/PCC.
- UPF Placement. The MNO wants to instantiate a UPF/SPGW-U in a location that is different from their network (e.g. Customer Premises, 3rd Party Data Center)

UPF Interconnection Scenario

Assumptions:

- MNO hosts UPF/SPGW-U in their network.
- MNO provisions all UPF functions.
- MNO may request UPF Interconnection for Sgi/N6 interface in a required Metro to 3PE/PCE via APIs (on P2)
- Bare Metal Orchestration Provider and Interconnect Provider expose Data Center Location / Metro to PCEI Enabler via APIs (on P3 and P4)
- 3PE and PCE providers expose Data Center Location / Metro to PCEI Enabler via APIs (on P5 and P8/P7)
- PCEI Enabler may request Bare Metal Orchestration for Distributed UPF via APIs (on P3)
- PCEI Enabler may request Interconnect for MNO UPF (L2/L3) via APIs (on P4)
- PCEI Enabler may request PCC/PCE and/or 3PE Controller to instantiate workload instances for LBO processing via APIs (on P8/P5)

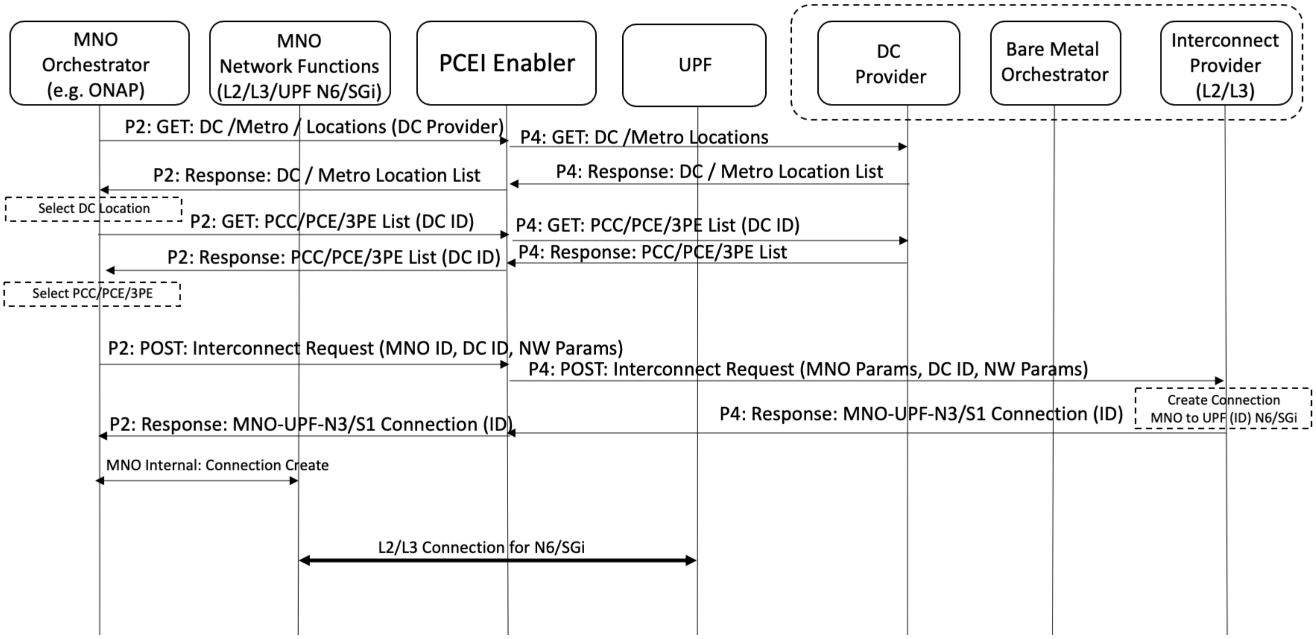


UPF Interconnection (N6/SGi) Call Flow Sketch

Notes:

- **Does not include interconnect to 3PE/PCE/PCC - to be added**
- **Does not include 3PE HW Orchestration - to be added**
- **Does not include UPF LBO Orchestration - to be added**
- **Does not include 3PE/PCE Workload Orchestration - to be added**

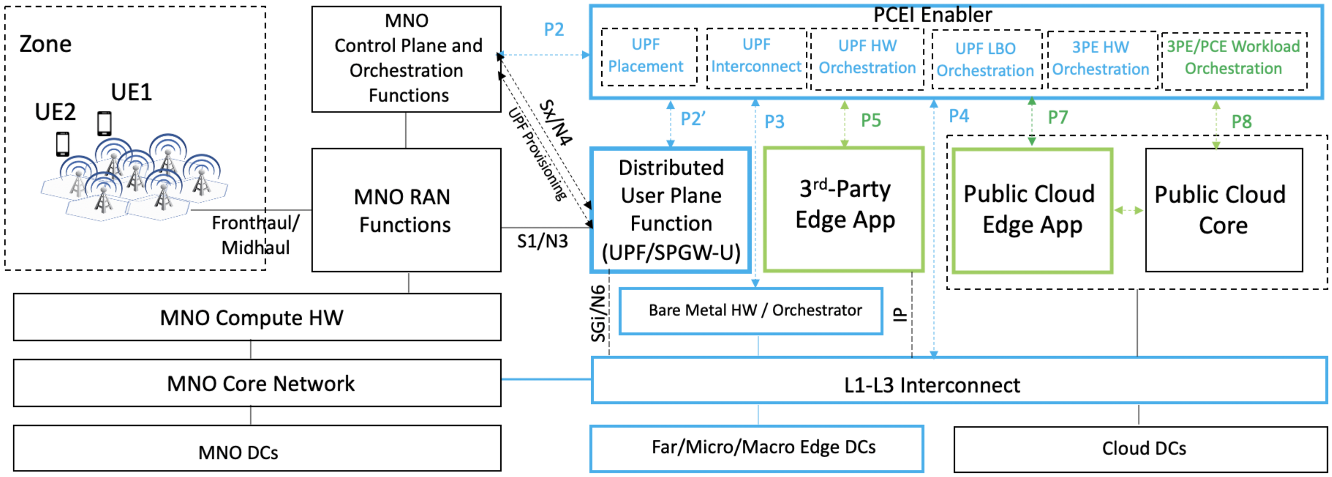
PCEI Facilitated UPF Interconnection on N6/SGi Interface


MNO Orchestrator (e.g. ONAP)
MNO Network Functions (L2/L3/UPF N6/SGi)
PCEI Enabler
UPF
DC Provider
Bare Metal Orchestrator
Interconnect Provider (L2/L3)
P2: GET: DC /Metro / Locations (DC Provider)
P4: GET: DC /Metro Locations
P2: Response: DC / Metro Location List
P4: Response: DC / Metro Location List
Select DC Location
P2: GET: PCC/PCE/3PE List (DC ID)
P4: GET: PCC/PCE/3PE List (DC ID)
P2: Response: PCC/PCE/3PE List (DC ID)
P4: Response: PCC/PCE/3PE List
Select PCC/PCE/3PE
P2: POST: Interconnect Request (MNO ID, DC ID, NW Params)
P4: POST: Interconnect Request (MNO Params, DC ID, NW Params)
Create Connection MNO to UPF (ID) N6/SGi
P2: Response: MNO-UPF-N3/S1 Connection (ID)
P4: Response: MNO-UPF-N3/S1 Connection (ID)
MNO Internal: Connection Create
L2/L3 Connection for N6/SGi

UPF Placement Scenario

Assumptions:

- MNO supports CUPS (5G NSA) and/or UPF (5G SA).
- MNO may request UPF Placement / Metro, and 3PE/PCE Access / Metro, via APIs (on P2)
- Bare Metal Orchestration Provider and Interconnect Provider expose Data Center Location / Metro to PCEI Enabler via APIs (on P3 and P4)
- 3PE and PCE providers expose Data Center Location / Metro to PCEI Enabler via APIs (on P5 and P8/P7)
- PCEI Enabler may request Bare Metal Orchestration for Distributed UPF via APIs (on P3)
- PCEI Enabler may request Bare Metal Orchestration for 3PE via APIs (on P3)
- PCEI Enabler may request Interconnect for Distributed UPF N3/S1 and N6/SGi traffic (L2/L3) via APIs (on P4)
- PCEI Enabler may expose UPF management access to MNO
- MNO may provision the Distributed UPF over management access
- PCEI Enabler may provision UPF connectivity and LBO configuration (based on the UPF provisioning model) over P2’ using appropriate protocols
- PCEI Enabler may request PCC/PCE and/or 3PE Controller to instantiate workload instances for LBO processing via APIs (on P8/P5)

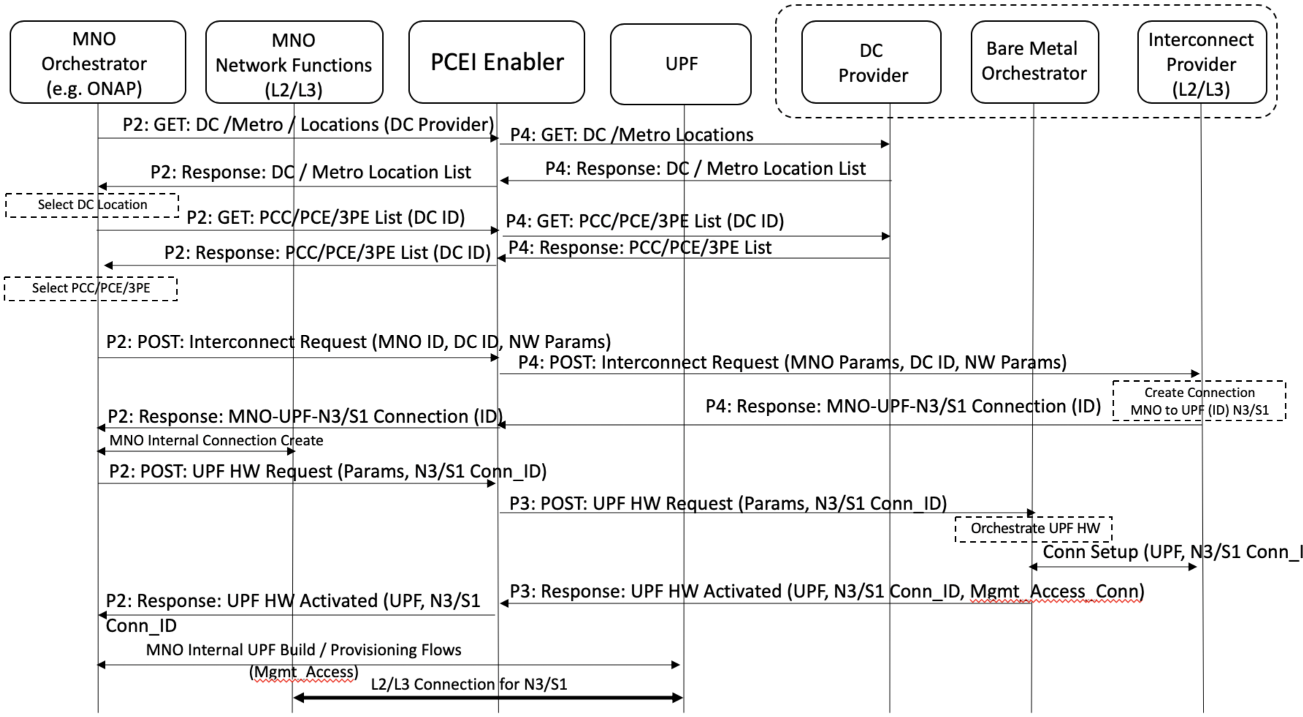


UPF Placement Call Flow Sketch

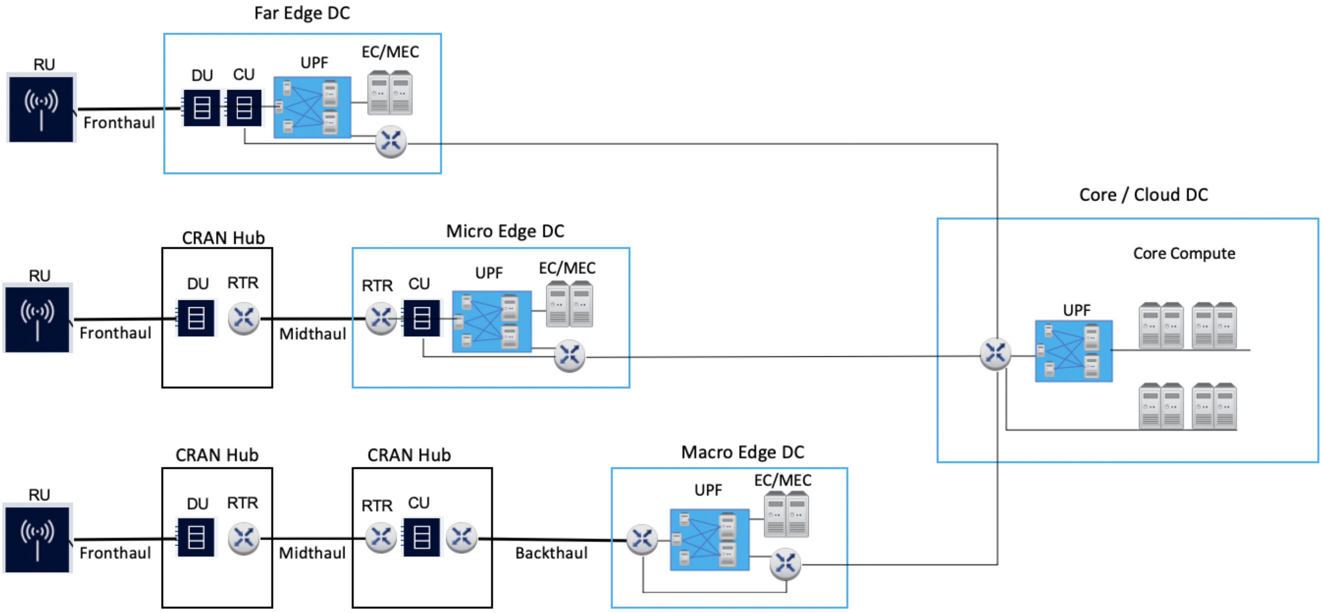
Notes:

- **Does not include interconnect to 3PE/PCE/PCC - to be added**
- **Does not include 3PE HW Orchestration - to be added**
- **Does not include UPF LBO Orchestration - to be added**
- **Does not include 3PE/PCE Workload Orchestration - to be added**

PCEI Facilitated UPF Placement with N3(S1) Interconnect


MNO
Orchestrator
(e.g. ONAP)
MNO
Network Functions
(L2/L3)
PCEI Enabler
UPF
DC
Provider
Bare Metal
Orchestrator
Interconnect
Provider
(L2/L3)
P2: GET: DC /Metro / Locations (DC Provider)
P4: GET: DC /Metro Locations
P2: Response: DC / Metro Location List
P4: Response: DC / Metro Location List
Select DC Location
P2: GET: PCC/PCE/3PE List (DC ID)
P4: GET: PCC/PCE/3PE List (DC ID)
P2: Response: PCC/PCE/3PE List (DC ID)
P4: Response: PCC/PCE/3PE List
Select PCC/PCE/3PE
P2: POST: Interconnect Request (MNO ID, DC ID, NW Params)
P4: POST: Interconnect Request (MNO Params, DC ID, NW Params)
Create Connection
MNO to UPF (ID) N3/S1
P2: Response: MNO-UPF-N3/S1 Connection (ID)
P4: Response: MNO-UPF-N3/S1 Connection (ID)
MNO Internal Connection Create
P2: POST: UPF HW Request (Params, N3/S1 Conn_ID)
P3: POST: UPF HW Request (Params, N3/S1 Conn_ID)
Orchestrate UPF HW
Conn Setup (UPF, N3/S1 Conn_ID)
P2: Response: UPF HW Activated (UPF, N3/S1
Conn_ID
P3: Response: UPF HW Activated (UPF, N3/S1 Conn_ID, Mgmt_Access_Conn)
MNO Internal UPF Build / Provisioning Flows
(Mgmt_Access)
L2/L3 Connection for N3/S1

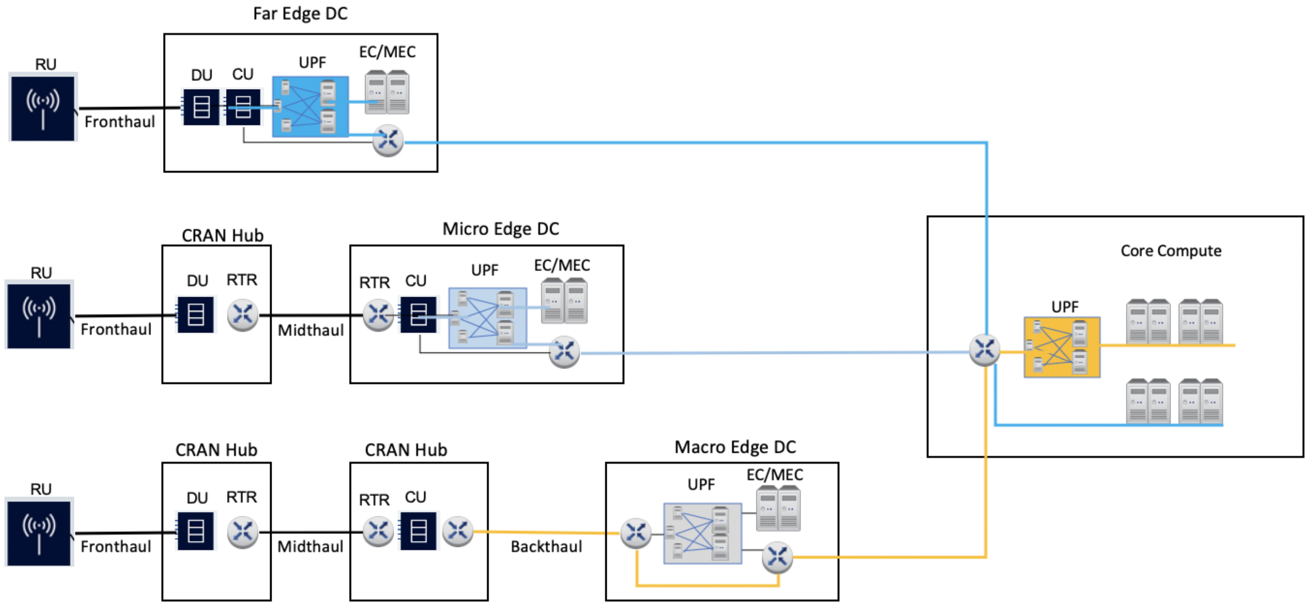
UPF Placement Options



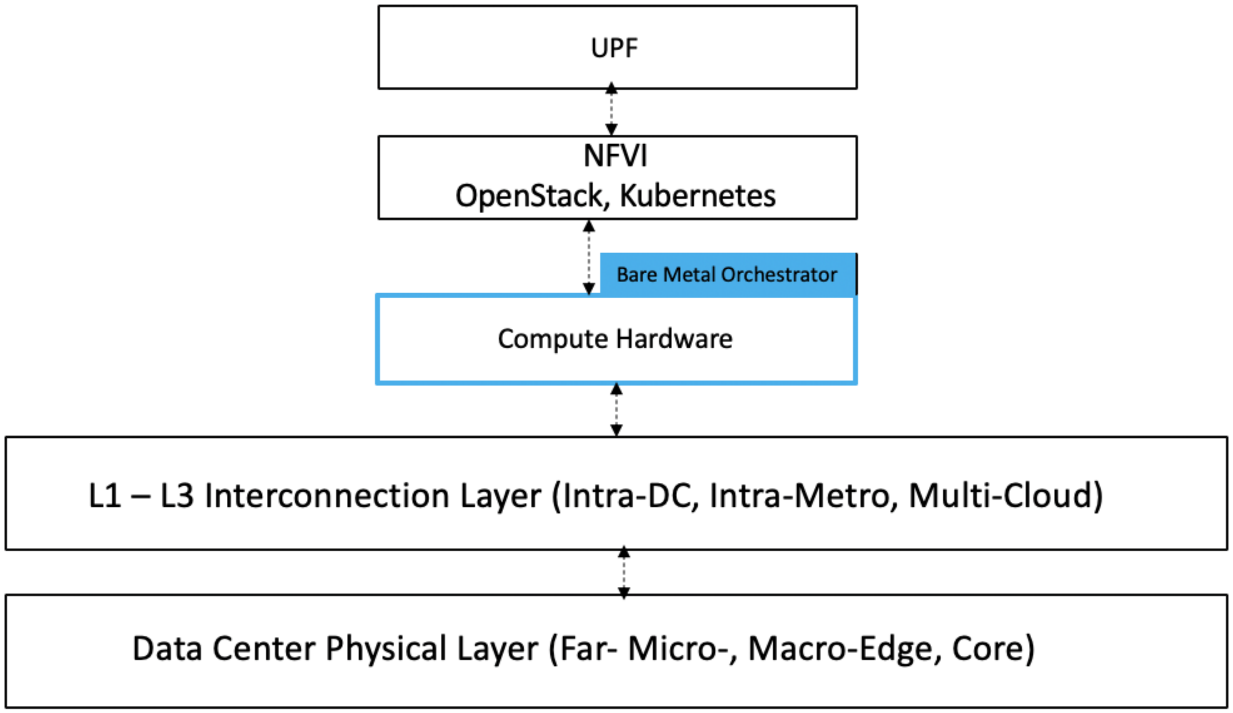
API Structure

To be added

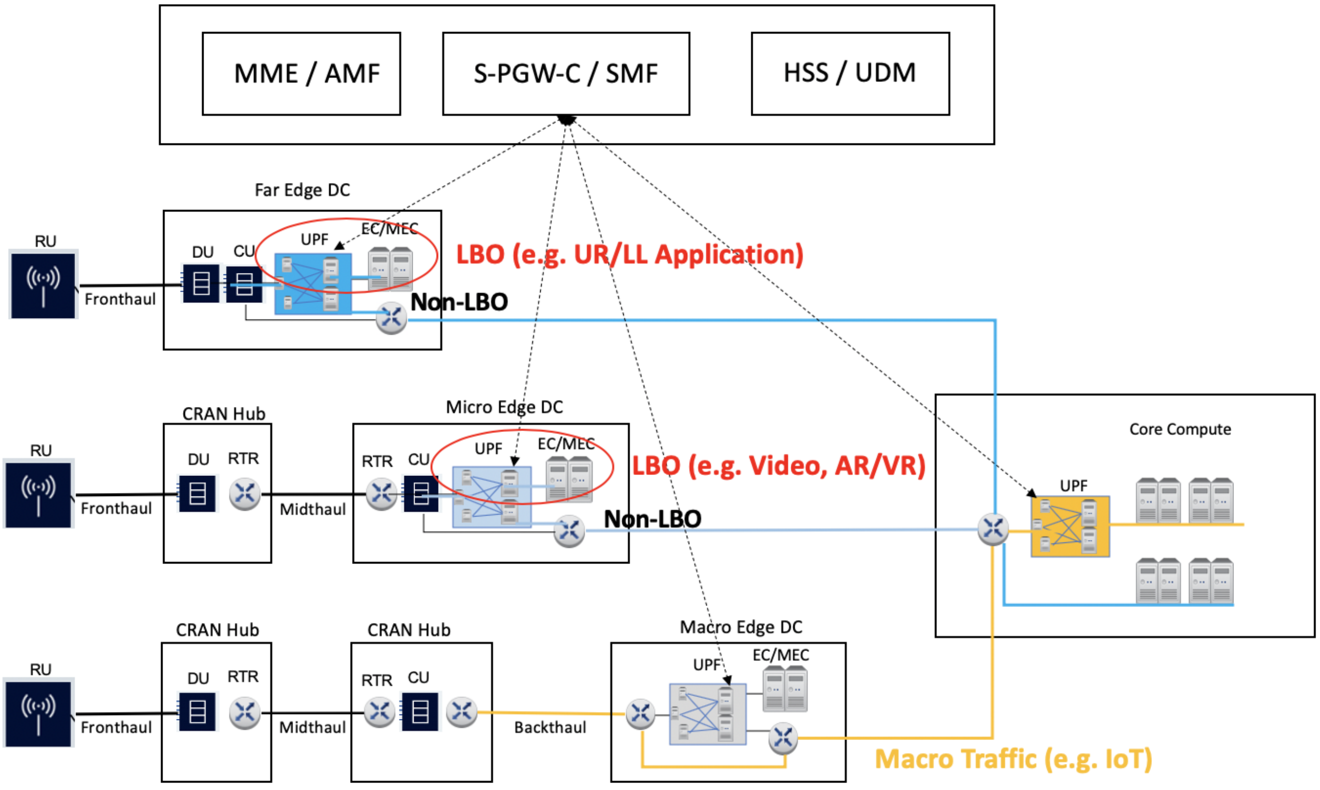
UPF Interconnect Options



UPF Hardware Selection / Orchestration


UPF
NFVI
OpenStack, Kubernetes
Bare Metal Orchestrator
Compute Hardware
L1 – L3 Interconnection Layer (Intra-DC, Intra-Metro, Multi-Cloud)
Data Center Physical Layer (Far- Micro-, Macro-Edge, Core)

UPF Configuration Orchestration - Local Break-Out Options



3PE/PCE Workload Instance Distribution

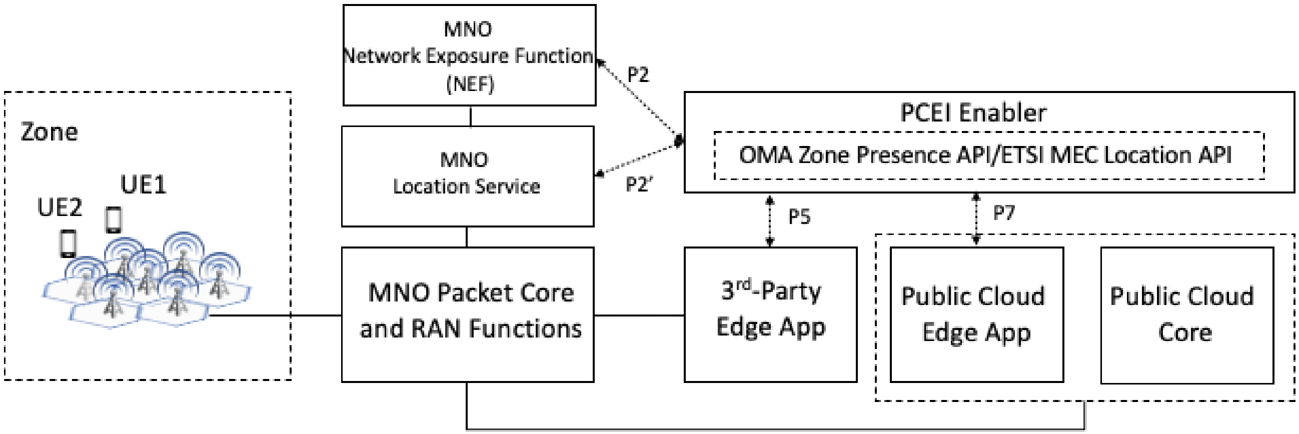
To be added

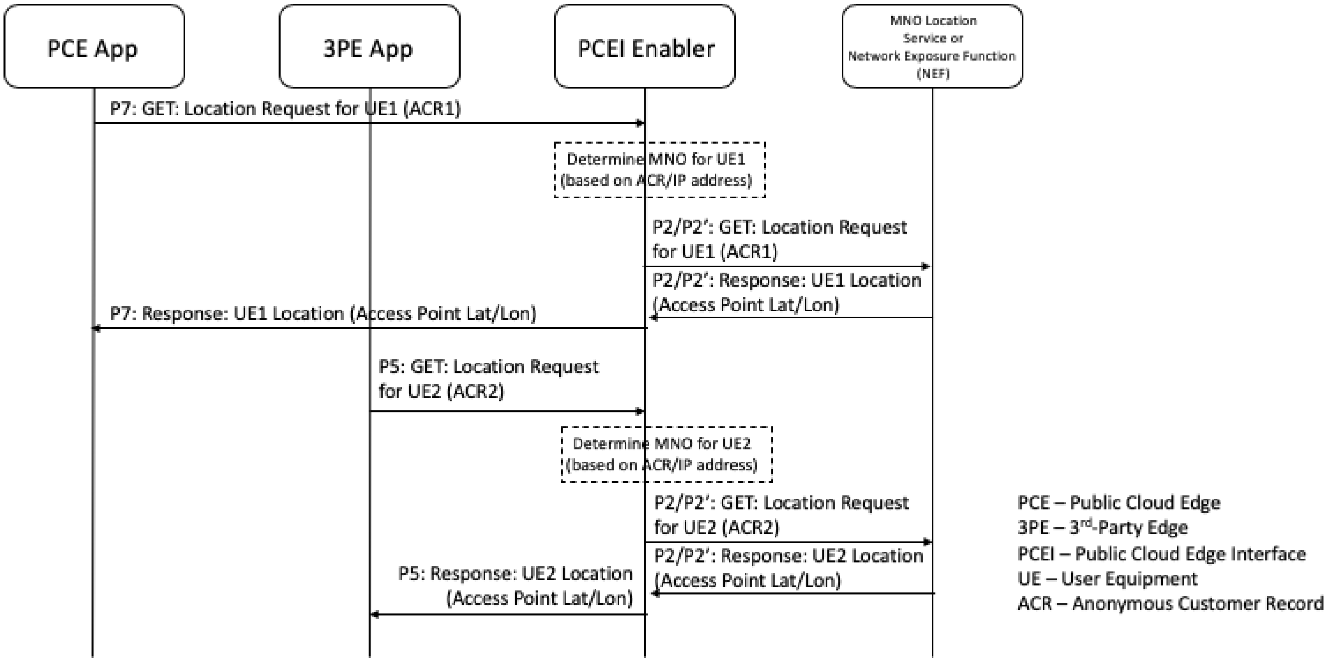
Location Services

- Location Services -- location of a specific UE, or identification of UEs within a geographical area, facilitation of server-side application workload distribution based on UE and infrastructure resource location.

Assumptions:

- MNO provides a Location Service (LS) compliant with OMA Zonal Presence API (OMA-TS-REST_NetAPI_ZonalPresence)
- MNO may expose the Location Service via the Network Exposure Function (NEF)
- A Public Cloud Edge (PCE) instance is associated with a Zone (collection of Access Points such as small cells) provided by an MNO
- A 3rd-Party Edge (3PE) instance is associated with a Zone (collection of Access Points such as small cells) provided by an MNO
- An application/workload in the PCE requires Location Information (e.g. coordinates of the Access Point) for the UE/subscriber
- An application/workload in the 3PE requires Location Information (e.g. coordinates of the Access Point) for the UE/subscriber
- PCEI Enabler facilitates Zonal Presence API Request/Response routing between PCE and the MNO LS and between the 3PE and the MNO LS

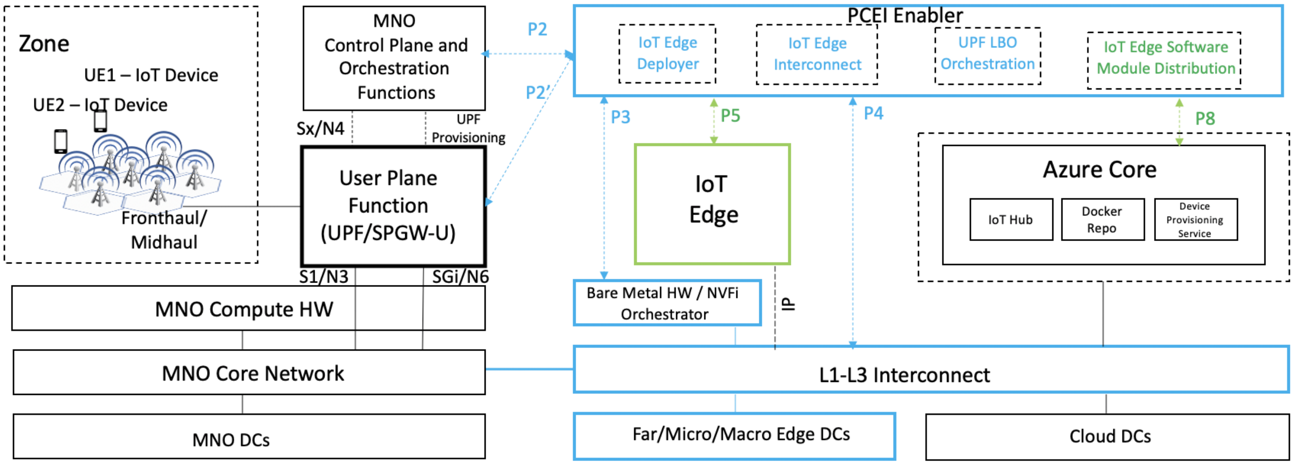


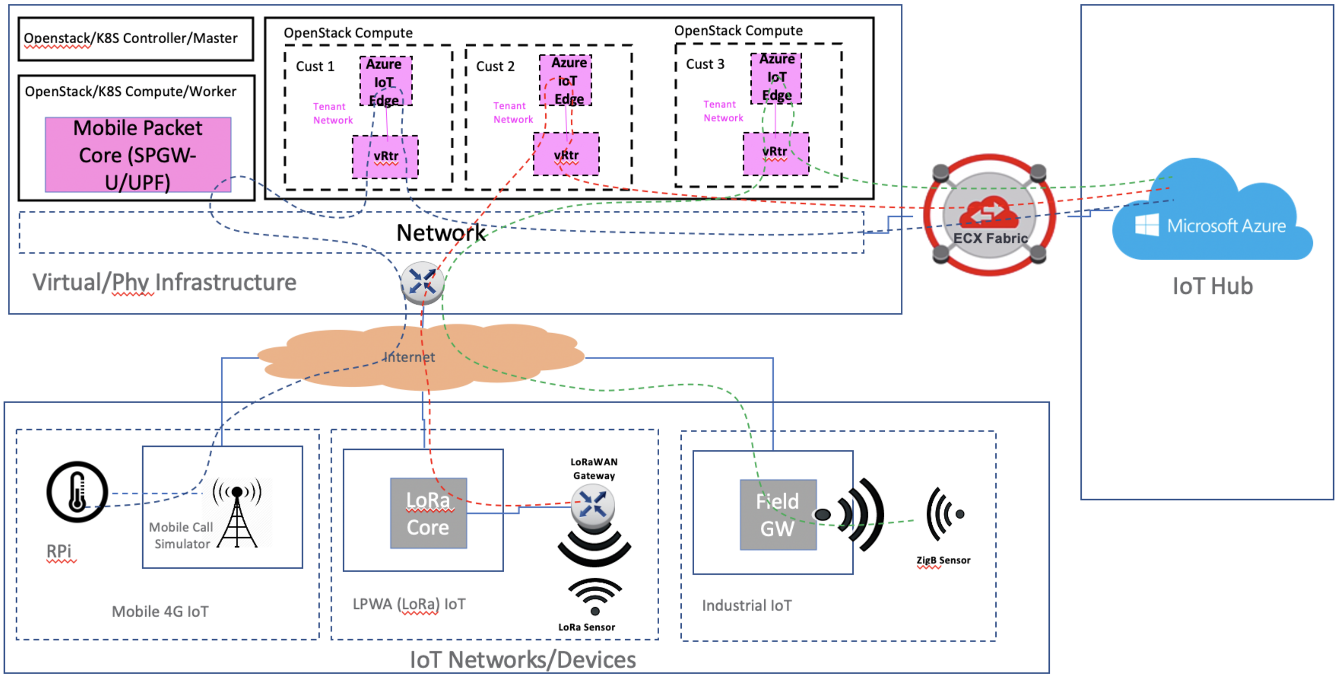

PCE App
3PE App
PCEI Enabler
MNO Location
Service or
Network Exposure Function
(NEF)
P7: GET: Location Request for UE1 (ACR1)
Determine MNO for UE1
(based on ACR/IP address)
P2/P2': GET: Location Request
for UE1 (ACR1)
P2/P2': Response: UE1 Location
(Access Point Lat/Lon)
P7: Response: UE1 Location (Access Point Lat/Lon)
P5: GET: Location Request
for UE2 (ACR2)
Determine MNO for UE2
(based on ACR/IP address)
P2/P2': GET: Location Request
for UE2 (ACR2)
P2/P2': Response: UE2 Location
(Access Point Lat/Lon)
P5: Response: UE2 Location
(Access Point Lat/Lon)
PCE – Public Cloud Edge
3PE – 3rd-Party Edge
PCEI – Public Cloud Edge Interface
UE – User Equipment
ACR – Anonymous Customer Record

Azure IoT Edge

Assumptions:

- MNO hosts UPF/SPGW-U in their network.
- MNO provisions all UPF functions.
- MNO may request Deployment of IoT Edge on Bare Metal or NFVi via APIs (on P2)
- PCEI Enabler IoT Edge Deployer may request HW or Virtual resources (e.g. VM/Container) (on P3)
- PCEI Enabler IoT Edge Deployer may provision IoT Edge (install runtime, deploy standard modules, register with IoT Hub) via APIs/Deployer Code (on P8/P5)
- PCEI Enabler may request Interconnect for IoT Edge between MNO and Azure (L2/L3) via APIs (on P4)
- PCEI Enabler may request UPF LBO for MNO to direct customer traffic to IoT Edge (on P2/P2')
- PCEI Enabler may facilitate deployment of custom IoT Edge modules





PCEI Enabler IoT Edge Deployer Functions

- Deploy a base Ubuntu VM
- Download and Install Microsoft GPG public key to apt config
- Install moby engine and cli
- Install azure iot edge
- Modify iot edge config file using customer defined parameters (azure hub url, keys, scope id, etc)
- Deploy iot edge agent
- Deploy iot edge hub module
- Install certificates if edge is used as a gateway
- Restart iot edge

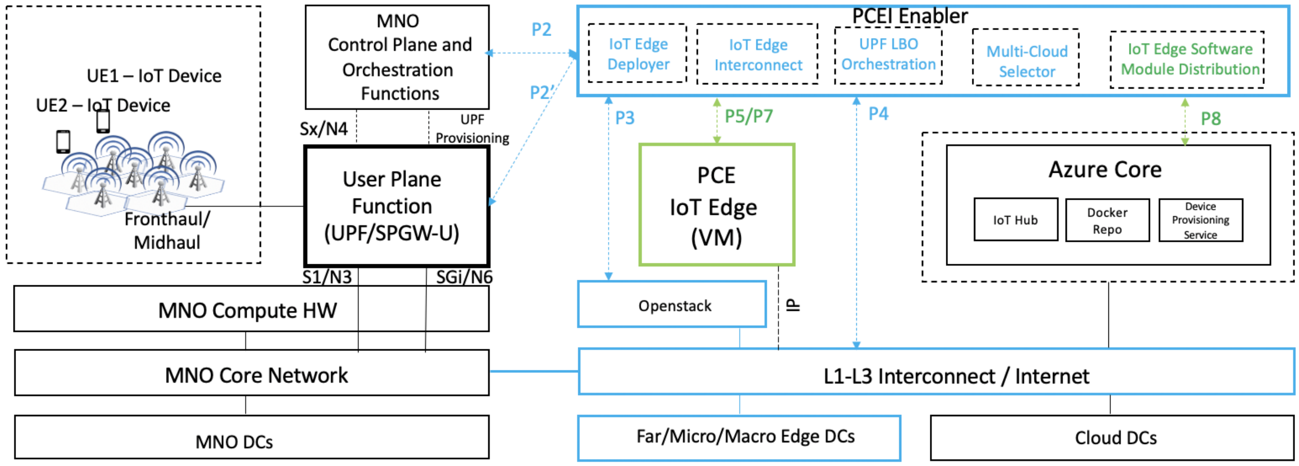
PCEI Enabler IoT Edge Interconnect Functions

- Request virtual connectivity to MNO
- Request virtual connectivity to Azure (e.g. ExpressRoute)

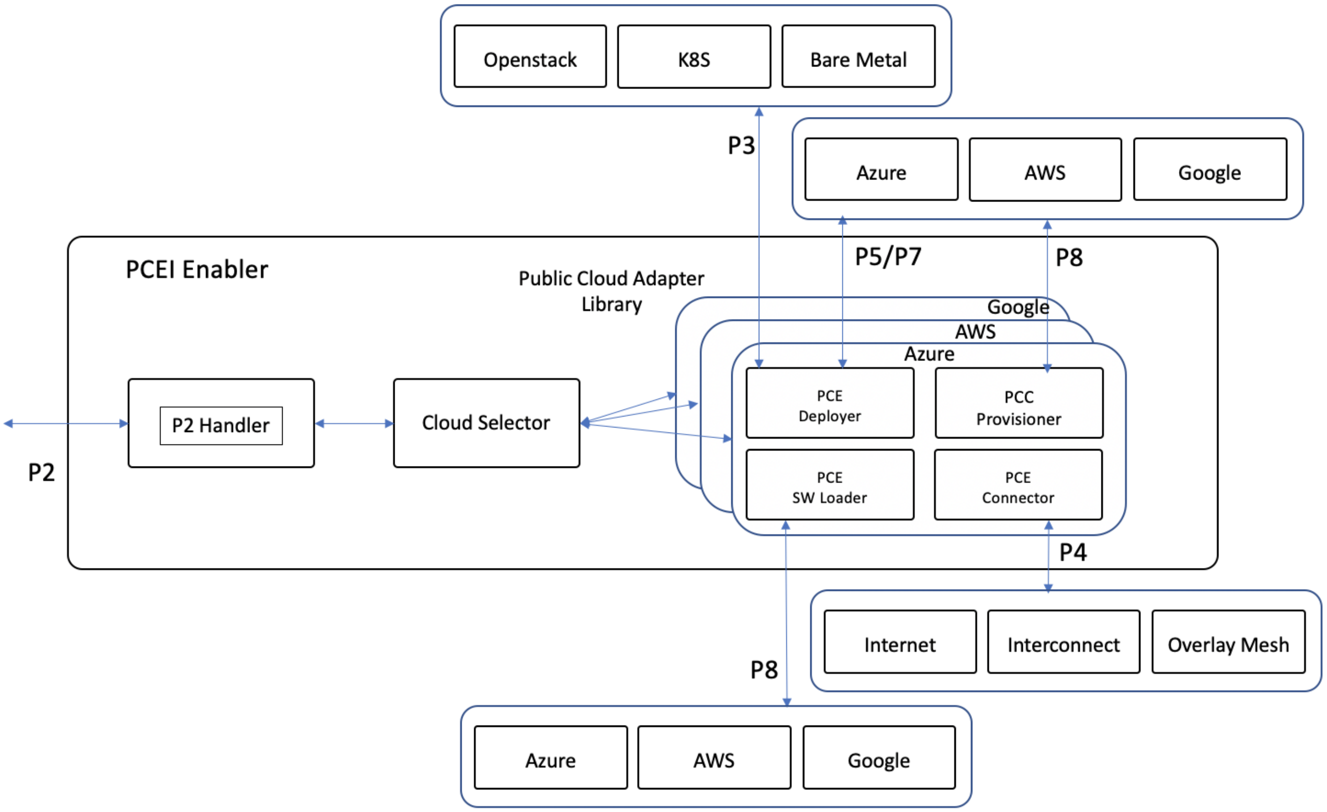
PCEI Enabler IoT Edge Software Distribution Functions

PCEI Enabler for Azure IoT Edge Implementation

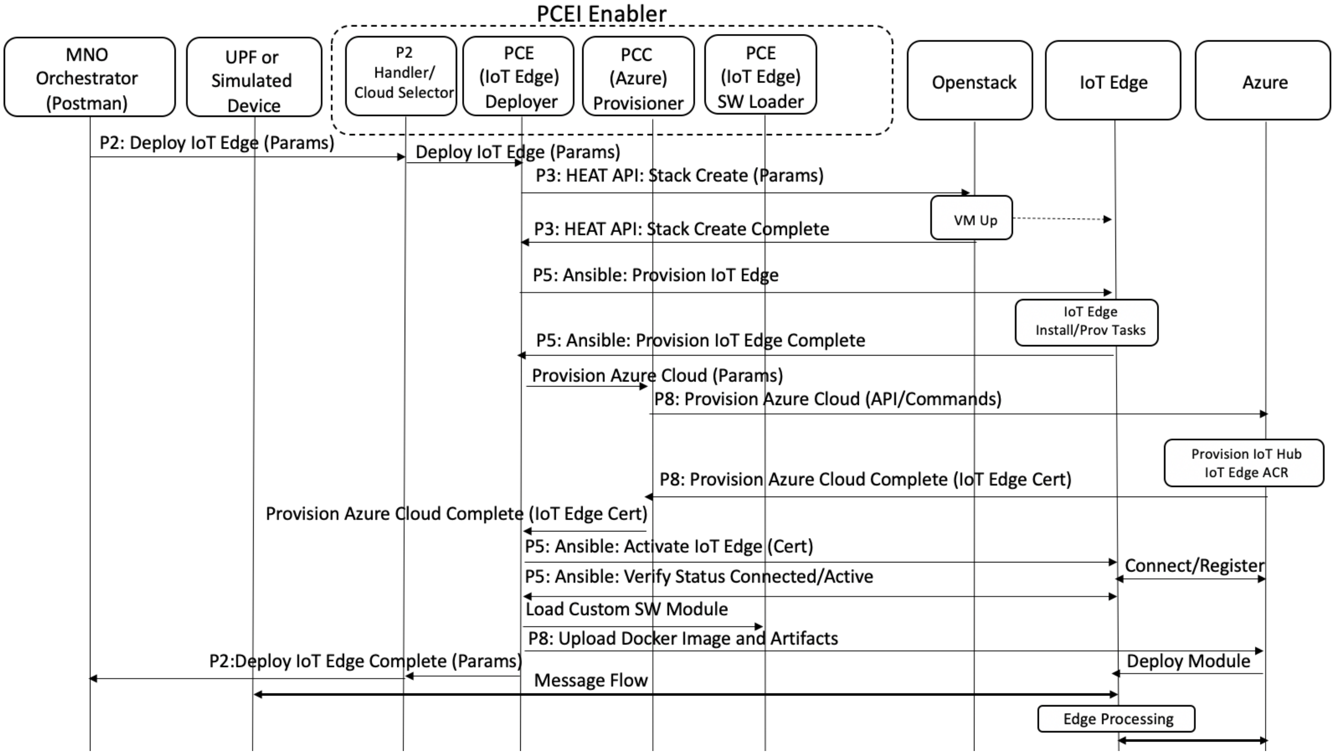
Architecture and Interfaces



PCEI Enabler Structure



PCEI for Azure IoT Edge Call Flow (High-level)


PCEI Enabler
MNO Orchestrator (Postman)
UPF or Simulated Device
P2 Handler/ Cloud Selector
PCE (IoT Edge) Deployer
PCC (Azure) Provisioner
PCE (IoT Edge) SW Loader
Openstack
IoT Edge
Azure
P2: Deploy IoT Edge (Params)
Deploy IoT Edge (Params)
P3: HEAT API: Stack Create (Params)
VM Up
P3: HEAT API: Stack Create Complete
P5: Ansible: Provision IoT Edge
IoT Edge Install/Prov Tasks
P5: Ansible: Provision IoT Edge Complete
Provision Azure Cloud (Params)
P8: Provision Azure Cloud (API/Commands)
Provision IoT Hub IoT Edge ACR
P8: Provision Azure Cloud Complete (IoT Edge Cert)
Provision Azure Cloud Complete (IoT Edge Cert)
P5: Ansible: Activate IoT Edge (Cert)
Connect/Register
P5: Ansible: Verify Status Connected/Active
Load Custom SW Module
P8: Upload Docker Image and Artifacts
P2:Deploy IoT Edge Complete (Params)
Deploy Module
Message Flow
Edge Processing

Openstack HEAT Example

HEAT Template and Environment Files

Template (pcei_base.yaml)

```
#####

heat_template_version: 2013-05-23

description: Heat template that deploys PCEI IoT Edge VM in Openstack


##############
# #
# PARAMETERS #
# #
##############

parameters:
pcei_image_name:
type: string
label: Image name or ID
description: Image to be used for compute instance
pcei_flavor_name:
type: string
label: Flavor
description: Type of instance (flavor) to be used
os_private_net_id:
type: string
label: os management network name or ID
description: Private network that connects os components and the VNF
os_private_subnet_id:
type: string
label: os management sub-network name or ID
description: Private sub-network that connects os components and the VNF
os_private_net_cidr:
type: string
label: os private network CIDR
description: The CIDR of the protected private network
pcei_private_ip_0:
type: string
label: VNF IP Address
description: IP address that is assigned to the IoT Edge
pcei_name_0:
type: string
label: VNF name
description: Name of the vPacketGenerator
key_name:
type: string
label: Key pair name
description: Public/Private key pair name
pub_key:
type: string
label: Public key
description: Public key to be installed on the compute instance

#############
# #
# RESOURCES #
# #
#############

resources:
random-str:
type: OS::Heat::RandomString
properties:
length: 4

my_keypair:
type: OS::Nova::KeyPair
properties:
name:
str_replace:
template: base_rand
params:
base: { get_param: key_name }
rand: { get_resource: random-str }
public_key: { get_param: pub_key }
save_private_key: false
```

```
# Instance behing vRouter
pcei_private_0_port:
type: OS::Neutron::Port
properties:
network: { get_param: os_private_net_id }
fixed_ips: [{"subnet": { get_param: os_private_subnet_id }, "ip_address": { get_param: pcei_private_ip_0 }}]

pcei_0:
type: OS::Nova::Server
properties:
image: { get_param: pcei_image_name }
flavor: { get_param: pcei_flavor_name }
name: { get_param: pcei_name_0 }
key_name: { get_resource: my_keypair }
networks:
# - network: { get_param: os_private_net_id }
- port: { get_resource: pcei_private_0_port }
user_data_format: RAW
user_data: |
#cloud-config
password: pcei
chpasswd: { expire: False }
ssh_pwauth: True
runcmd:
- [ sh, -xc, "sed -i 's,#UseDNS yes,UseDNS no,' /etc/ssh/sshd_config" ]
- systemctl restart sshd.service
```

Environment (pcei_base.env)

```
#########

parameters:

pcei_image_name: ubuntu1604

pcei_flavor_name: l3

os_private_net_id: provider

os_private_subnet_id: provider

os_private_net_cidr: 10.121.11.0/24

pcei_private_ip_0: 10.121.11.91

pcei_name_0: PCEI-IOT-EDGE

key_name: maskey1

pub_key: ssh-rsa AAAAB3NzaC1y
```

Openstack HEAT Command

```
openstack stack create -t pcei_base.yaml -e pcei_base.env
```