5G Mobile Platform with P4-enabled Network Slicing and MEC

Wilson Wang
National Chiao Tung University

About Me

- Wilson Wang
  - NCTU PhD candidate
  - ITRI engineer
  - Cat person
  - ONOS deployment brigade member

國立交通大學 National Chiao Tung University X ITRI Industrial Technology Research Institute

Outline

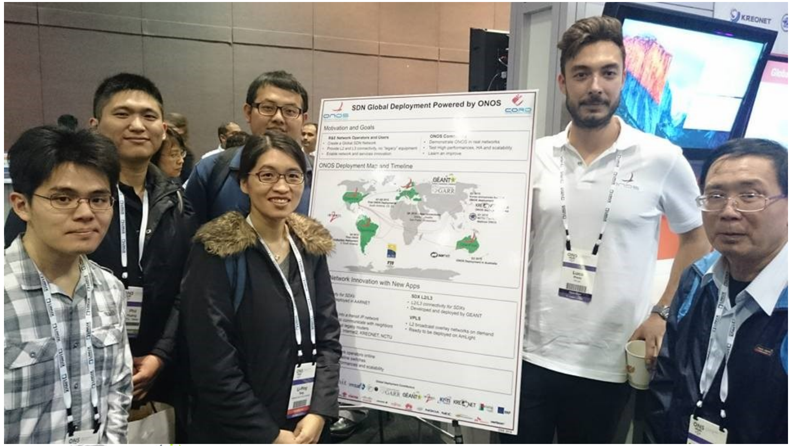
- What We Have Done
- 5G Mobile Platform with free5GC
- Reduce Loading in MEC with P4 Switch
- P4-enabled Network Slicing

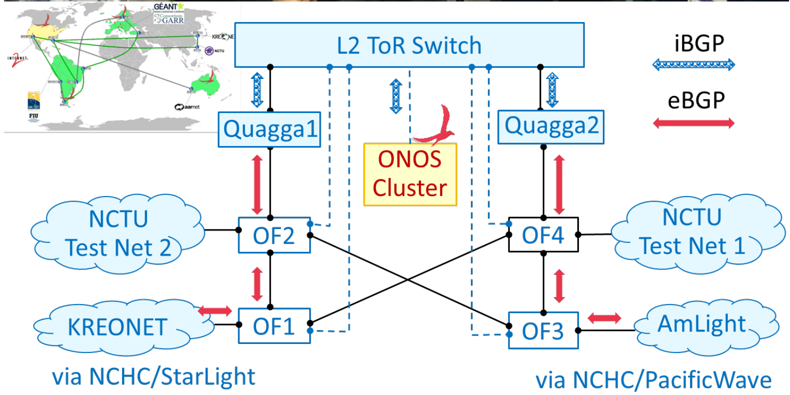
ONS 2016

- Build an **SDN-IP** site in NCTU
  - Connect with KREONET and AmLight
- Live DEMO

42u Rack Cabinet

	42, 41U
L2 ToR Switch (1G)	40U
L2 ToR Switch (1G)	39U
	38-35U
NCTU-of1 Switch (10G)	34U
NCTU-of2 Switch (10G)	33U
NCTU-of3 Switch (10G)	32U
NCTU-of4 Switch (10G)	31U
	30-27U
ONOS cluster	26U
NCTU Quagga 2 (AS 65113)	25U
NCTU Quagga 1 (AS 65113)	24U
NCTU BGP 1 (AS 65110)	23U
NCTU BGP 2 (AS 65120)	22U

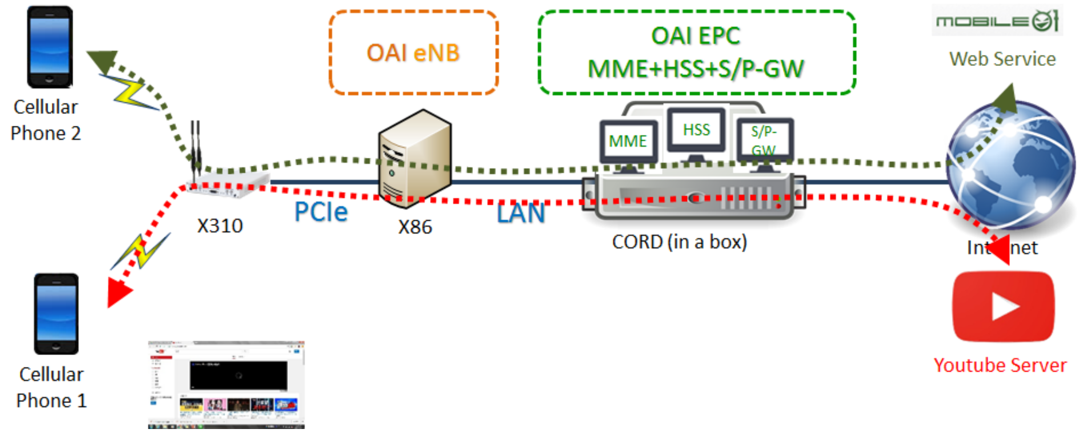


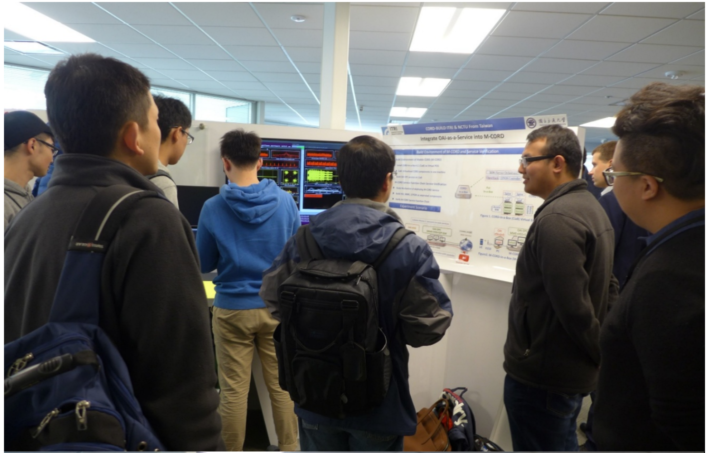


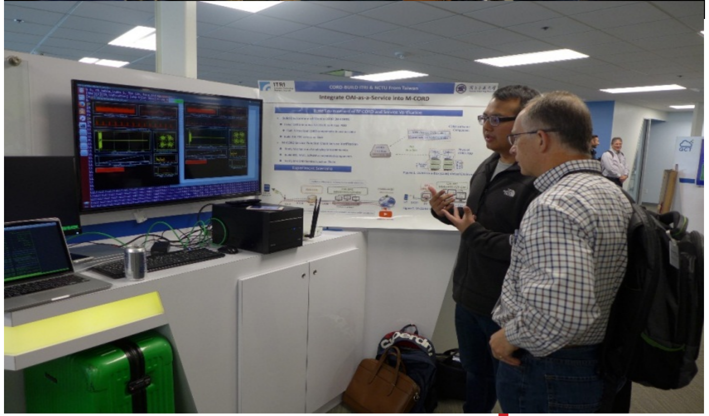
ONF CONNECT

CORD Build 2017

- **Integrate OAI-as-a-Service into M-CORD**
  - Presentation and live DEMO

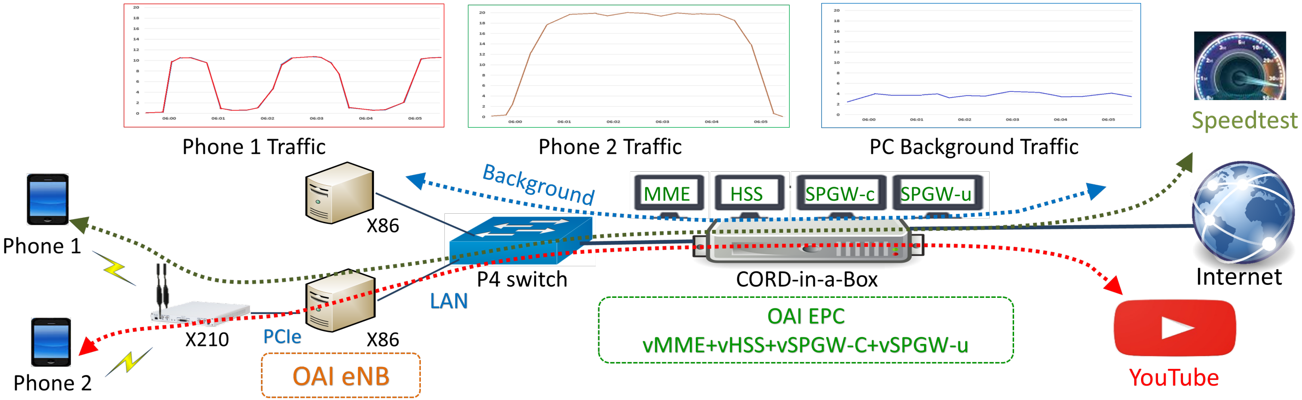






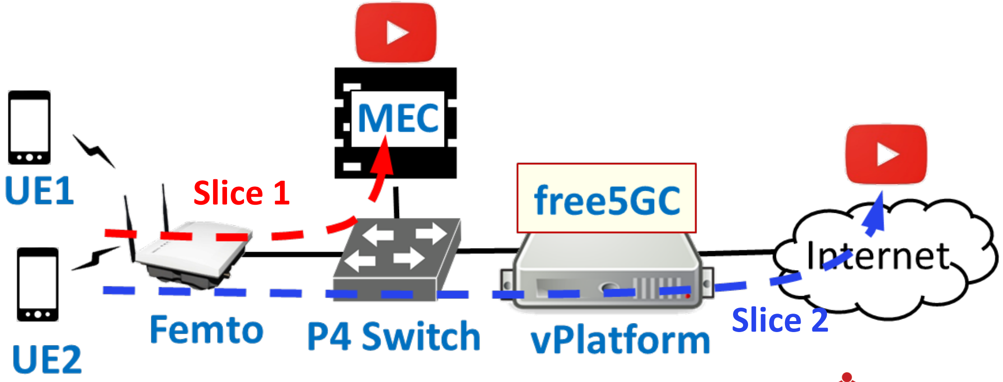
ONF Connect 2018

- **OAI M-CORD Platform with P4-enabled Network Slicing**
  - Live DEMO



ONF Connect 2019

- 5G Mobile Platform with P4-enabled Network Slicing and MEC
  - Compliant with **ETSI MANO**
  - NCTU **free5GC**
  - Loading Reduction in **MEC** with **P4 Switch**
  - P4-enabled **network slicing**



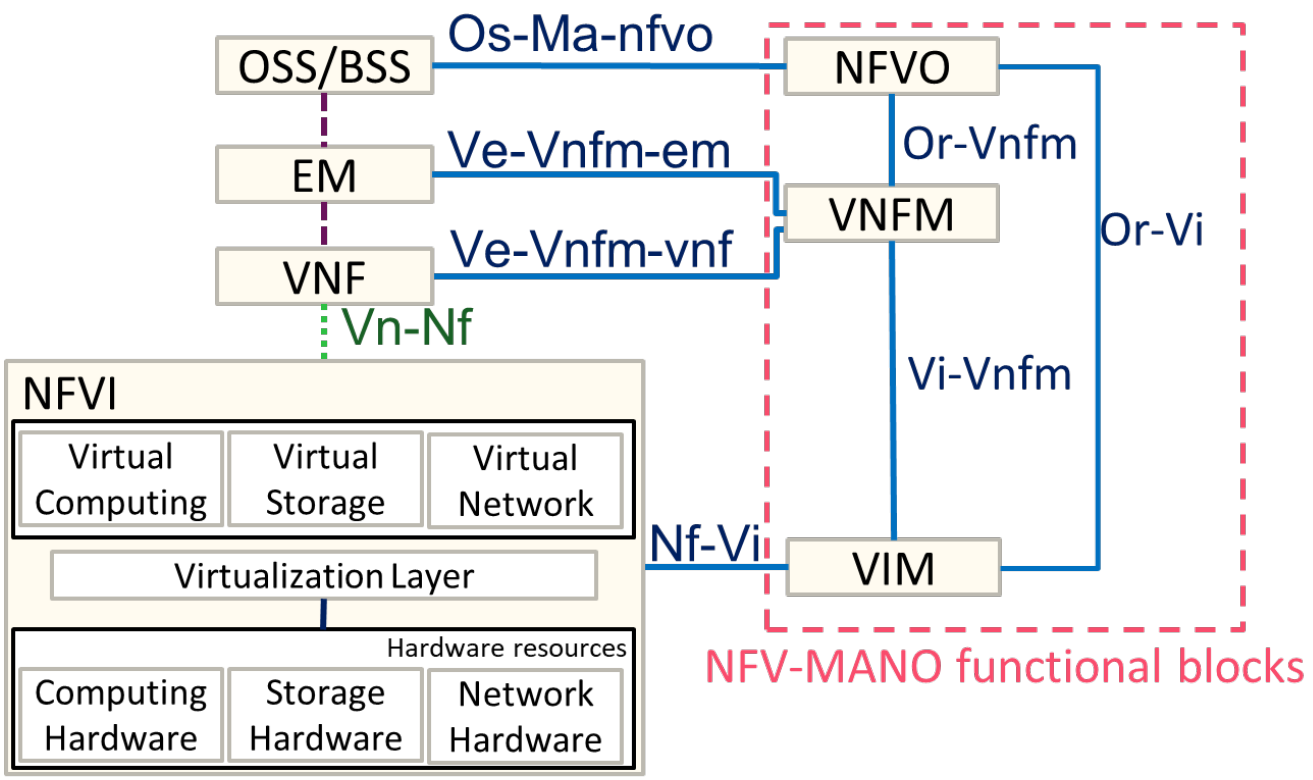
Introduction and Motivation for Our 5G Mobile Platform

- 5G need **Virtualized Network Functions (VNFs)**
  - Flexible and efficient network
- **Cloud-Native VNFs (CNF)**
  - VNFs based on Cloud-Native containerization technology
  - Lower overhead and higher performance
- ETSI proposes **NFV Management and Orchestration (NFV-MANO)** architecture
- Many existing NFV-MANO projects
  - Complex service development
  - Insufficient support of CNF orchestration
  - High resource usage, e.g. CPU, memory, disk …
- Need a **5G Lightweight NFV-MANO platform**

So We Want to

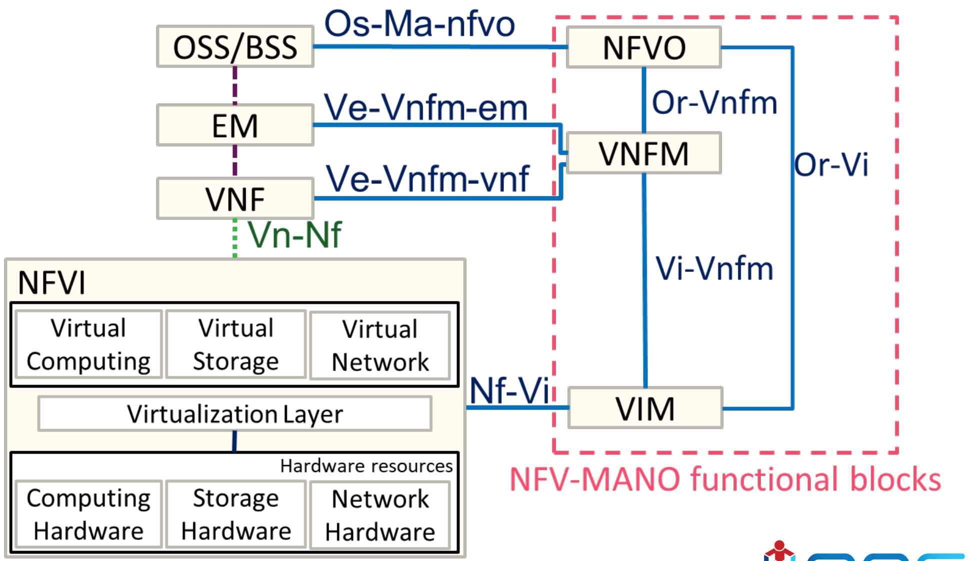
- Propose a **5G Lightweight NFV-MANO Mobile Platform**
  - Utilize SDN, NFV, Cloud to provide 5GC flexibility and scalability
  - All open sources
    - Kubernetes, ONOS, free5GC …
  - NFV functionality
    - Scalable free5GC CNFs
  - Cloud functionality
    - Agile orchestration
  - SDN functionality
    - Flexible underlay network

ETSI NFV-MANO Architecture



ETSI NFV-MANO Functional Blocks

- **NFVO: NFV Orchestrator**
  - Management of the instantiation of VNFMs where applicable
  - Network Services (NSs) lifecycle management
- **VNFM: VNF Manager**
  - Manage lifecycle of VNF instances
  - Creates, maintains and terminates VNF instances
- **VIM: Virtualized Infrastructure Manager**
  - E.g. OpenStack, Kubernetes, ONOS
- **VNF: Virtualized Network Function**
  - **free5GC**
- **NFVI: NFV Infrastructure**
  - Provide the infrastructure resources
- EM: Element Management
- OSS/BSS: Operation/Business System Support

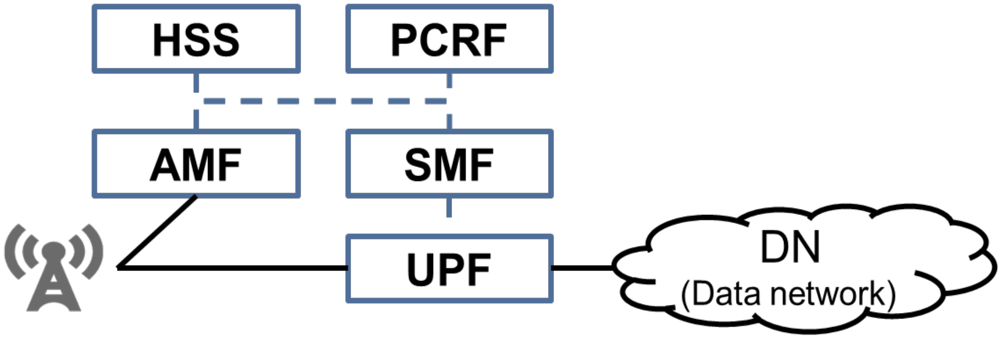


What is free5GC

- The free5GC is an **open-source** project code for **5G generation mobile core network** created by NCTU
- Based on the Rel-13 EPC and migrates into Rel-15 5GC
  - Focus enhance Mobile Broadband (eMBB) feature
  - Ultra-Reliable Low Latency Connection (URLLC) and Massive Internet of Things (MIoT) are not supported yet

free5GC CNFs

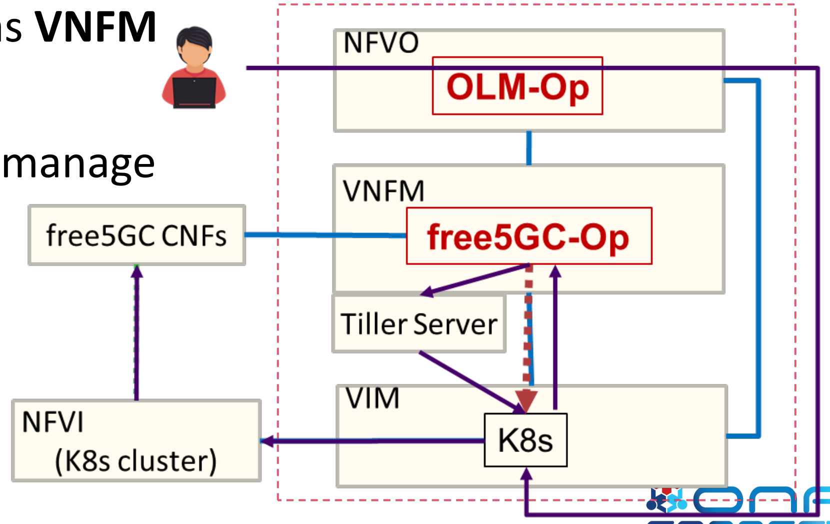
- CNFs
  - AMF: Access Management Function
  - SMF: Session Management Function
  - HSS: Home Subscriber Server
  - PCRF: Policy and Charging Rules Function
  - UPF: User Plane Function
- All CNFs are containerization and running on K8s cluster



Design Concept of 5G Lightweight NFV-MANO Mobile Platform

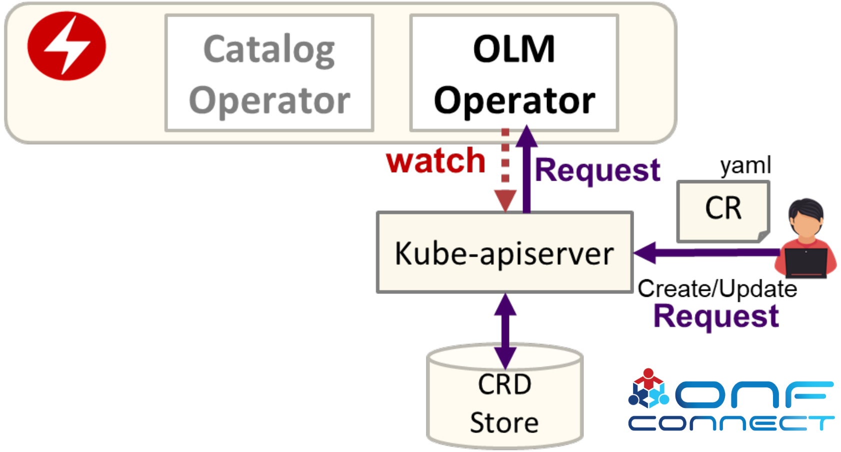
- Each NF of free5GC is a CNF (Base on **SBA**)
- May install/update a group of designated **free5GC CNFs** (Using **Helm)**

- Create Custom Resource Definition for **free5GC CNFs**
- Introduce a free5GC Operator (**free5GC-Op**) as **VNFM** for **free5GC CNFs CR**
- Employ OLM Operator (**OLM-Op**) as **NFVO** to manage **VNFMs** (e.g. free5GC-Op)
- Create Custom Resource Definition for **C-Op**
  - Treat Custom Operator (C-op) as CR allowing dynamic C-Op installation/update



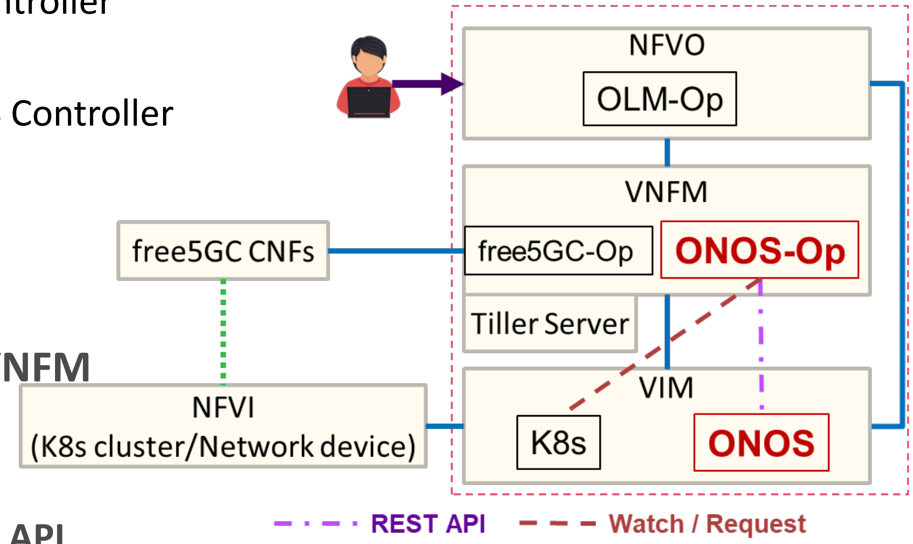
Operator Lifecycle Manager (OLM)

- Open source project hosted by Red Hat
- Create Custom Resource Definition for **Custom Operators (C-Ops)**
  - Treat Custom Operator (C-Op) as CR in K8s
- Employ two operators to manage C-Op CR:
  1. **OLM Operator (OLM-Op)**:
     - Watch **C-Op CR** update request on K8s API Server
     - Perform C-Op installation/modification
  2. **Catalog Operator** (optional)
     - Cache of C-Op custom resource



Design Concept of 5G Lightweight NFV-MANO Mobile Platform (Cont.)

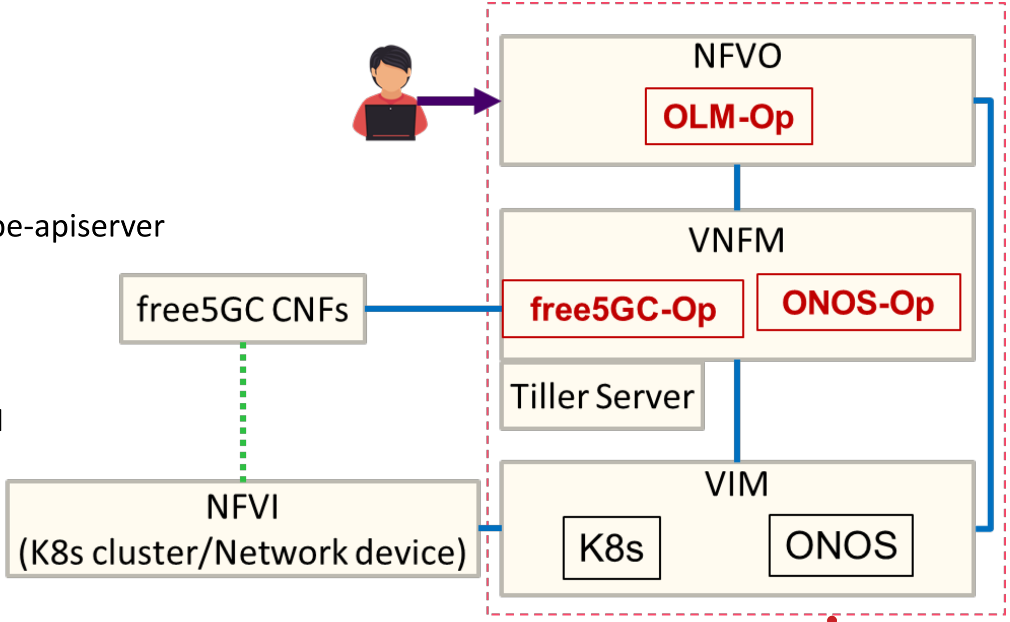
- Use **ONOS** as SDN controller to manage underlying SDN network
- Two **approaches** to interact with ONOS (through **ONOS northbound REST API**)
  1. Modify **OLM-Op** to interact with ONOS Controller directly
  2. Introduce **ONOS-Op** to interact with ONOS Controller (On behalf of OLM-Op)
     - **Need not to modify OLM-Op!**



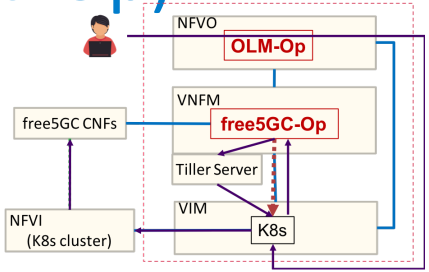
➢ Introduce a ONOS Operator (**ONOS-Op**) as **VNFM**
- Create CRD for **ONOS REST API**
  - Treat ONOS REST API as CR
- Implement a ONOS-Op as **VNFM for ONOS REST API**

Architecture of 5G Mobile Platform

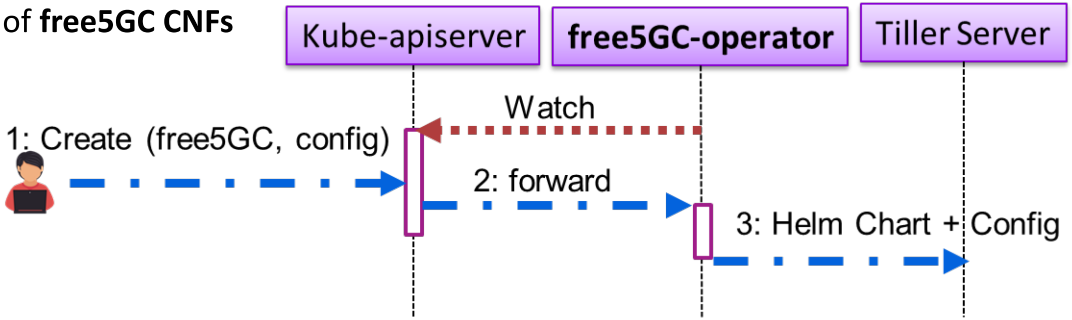
- **NFVO: OLM-Op**
  - Watch **CR** update requests of **C-Op** on Kube-apiserver
  - Install / update C-Op
- **VNFMs:** C-Ops
  - free5GC Operator (**free5GC-Op**)
    - Watch **CR** update requests of **free5GC** on Kube-apiserver
    - Install / update free5GC CNFs
  - ONOS Operator (**ONOS-Op**)
    - Watch **CR** update requests of **ONOS REST API** on Kube-apiserver
    - Call **ONOS northbound REST API**

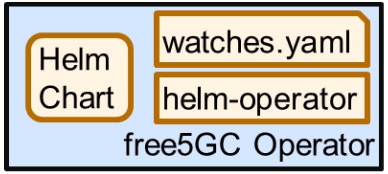


Design of free5GC Operator (free5GC-Op)

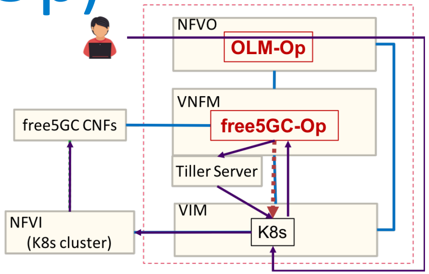


- Responsible for installing/updeting free5GC CNF by using Helm
- Three components:
  - **Helm Chart of free5GC CNFs** (template file)
    - Template for K8s resources definitions of **free5GC CNFs**
  - **watches.yml** (file)
    - CR name: free5GC
    - Helm Chart path
  - **helm-operator** (free5GC-Op core)
    - Get CR name specified in watches.yml
    - Watch **CR** update requests of **free5GC** on kube-apiserver
    - On receiving request, transform **config of request content** into **Helm *Config***
    - Send Helm Chart and Config to Tiller Server

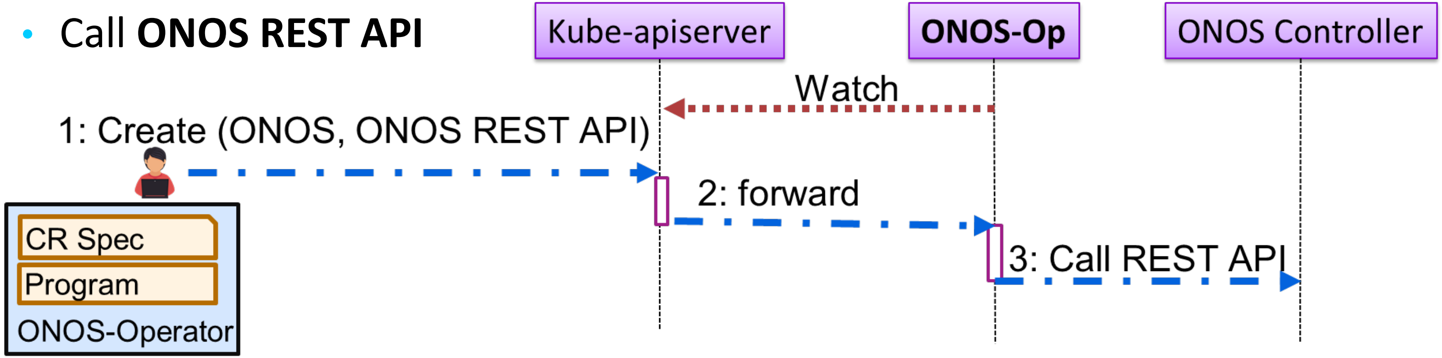




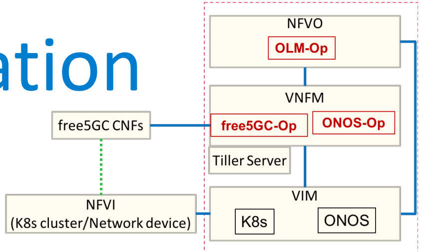
Design of ONOS Operator (ONOS-Op)



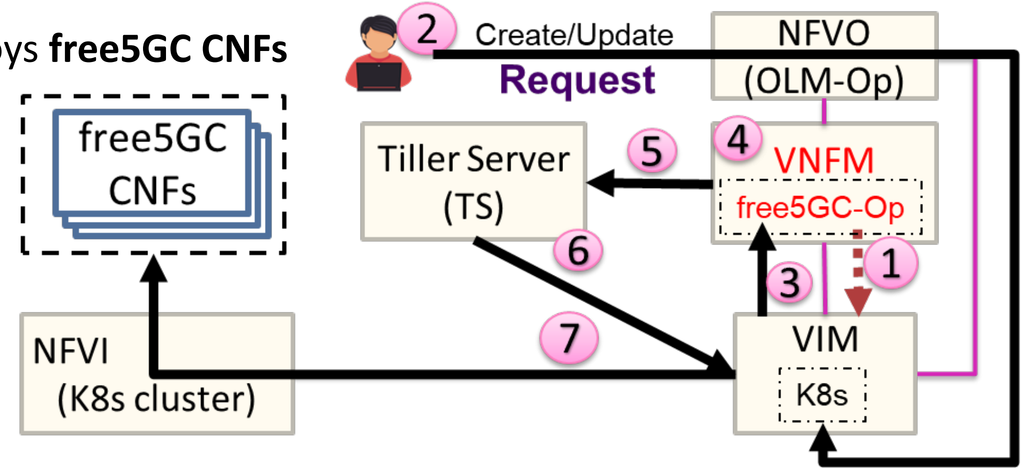
- Responsible for interacting with ONOS Controller
- Two components:
  - Spec of **ONOS REST API**
  - ONOS-Op core:
    - Watch **CR** update requests of **ONOS REST API** on kube-apiserver
    - On receiving request, transform **ONOS REST API CR content** into **REST API format**
    - Call **ONOS REST API**



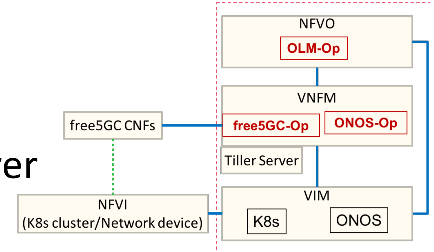
free5GC CNFs Installation / Modification



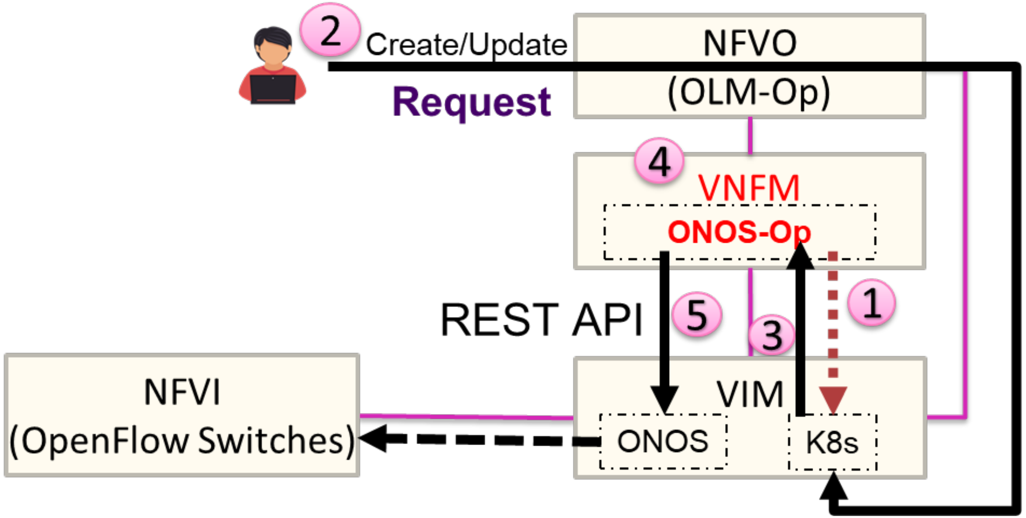
1. free5GC-Op watches **free5GC CR** update request on Kube-apiserver
2. User requests Kube-apiserver to create/update **free5GC CR**
3. Kube-apiserver forwards request to free5GC-Op
4. free5GC-Op transforms **config of request content** into **Helm *Config***
5. free5GC-Op sends **Helm *Chart*** and ***Config*** to Tiller server (TS)
6. TS combines Helm *Chart* and *Config*, sends data to Kube-apiserver
7. Kube-apiserver deploys **free5GC CNFs**



Interact with ONOS

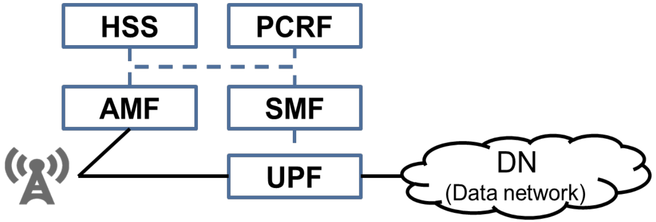


1. ONOS-Op watches **ONOS REST API CR** update request on Kube-apiserver
2. User requests Kube-apiserver to create/update **ONOS REST API** CR
3. Kube-apiserver forwards request to ONOS-Op
4. On receiving request, ONOS-Op transform **request content** into **ONOS REST API format**
5. ONOS-Op call **ONOS northbound REST API**



Data Network: Multus + Calico + SR-IOV

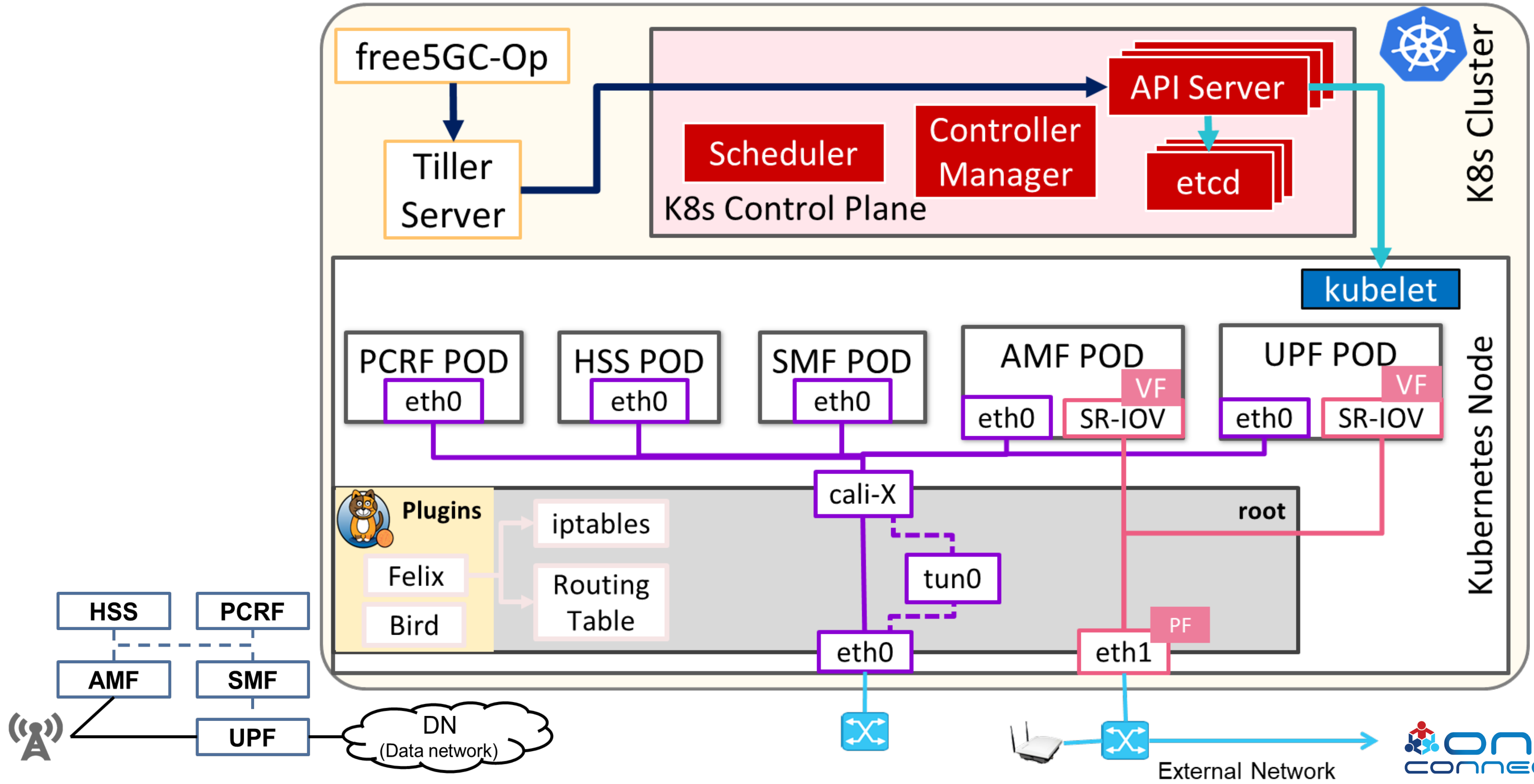
- Multi-interface of free5GC CNFs
  - eth0 of CNFs: for 5G Core Network functions interaction
  - eth1 of AMF and UPF: for connect to eNodeB



Why Multus + Calico + SR-IOV?

- Multus: Enabling attach **multiple network** interfaces to PODs
- Calico: Good performance for deliver native Linux networking dataplane
  - No packets encapsulation, direct packets natively by BGP routing mechanism
  - Minimize overall CPU usage and occupancy by Calico's control plane and policy engine
- SR-IOV: **Lowers latency** and boosts **throughput** to satisfy CNF data plane needs
  - Hardware based virtualization technology that improve performance and scalability

Design of Data Network

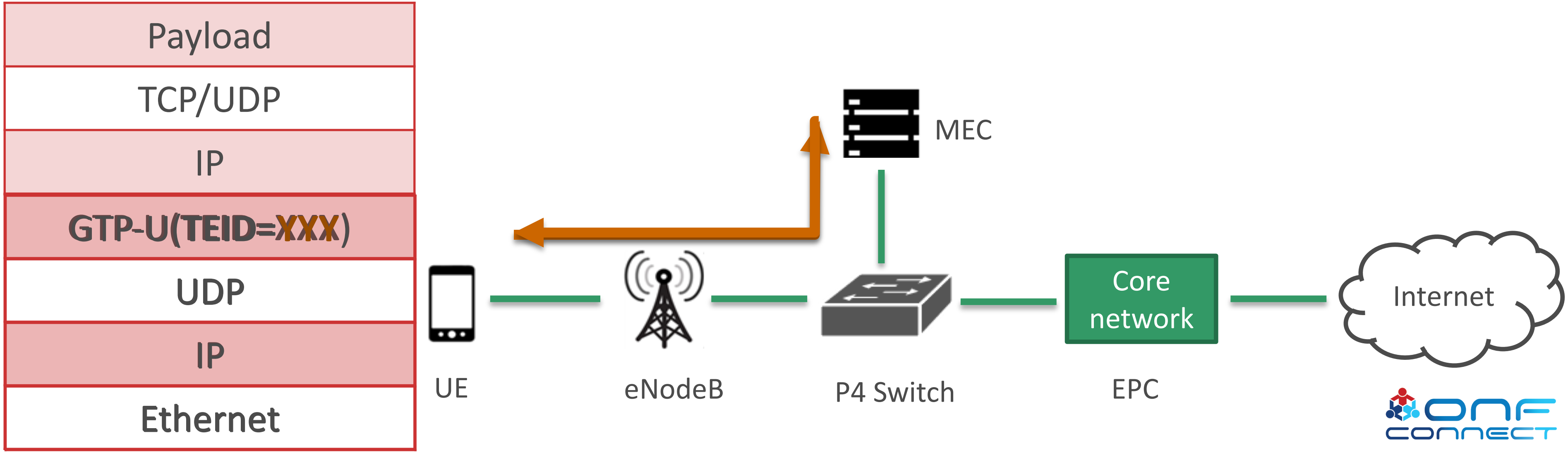


Reduce Loading in MEC with P4 Switch

- Propose a P4-based MEC network
  - Network feature
    - Provide better packet I/O with P4 switch
    - Reduce MEC loading from packet encapsulation and decapsulation
    - Redirect DNS

Stateful GTP packet tracking

- Decapsulate GTP-U header before sending it MEC
- Encapsulate packet with GTP-U header before sending it to UE
  - Tracking mapping between **UE IP** and **downlink** **TEID**



Reduce Loading in MEC with P4 Switch

- Two approaches
  - Packet-in downlink GTP-U packets
  - Packet-in SCTP packets

Packet-in downlink GTP-U packets

Switch

Payload
TCP/UDP
IP
GTP-U
UDP
IP
Ethernet

UE session state

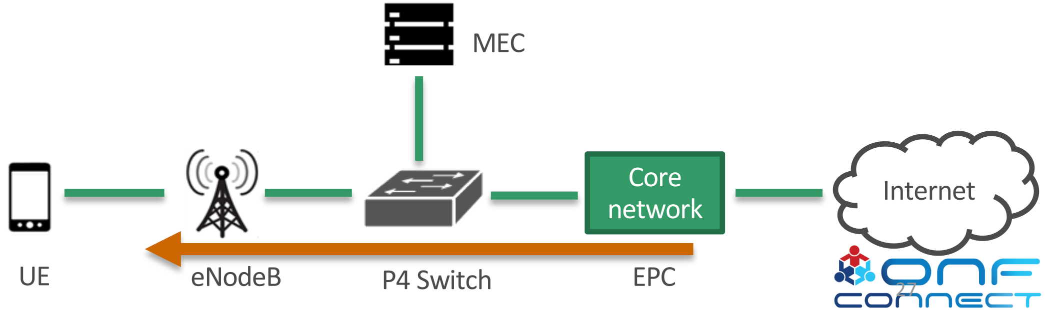
Match	Action
192.168.3.2	NoAction

UE encapsulation

Match	Action
192.168.3.2	set_gtp_header(777)

Controller

UE Addr.	DL TEID
192.168.3.2	777



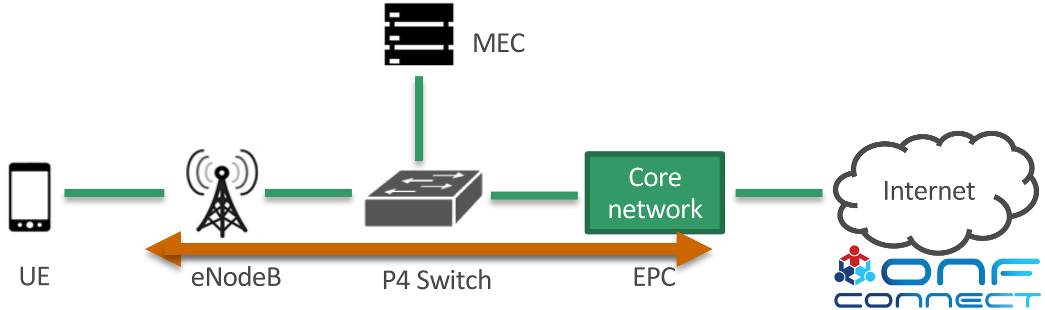
Packet-in downlink GTP-U packets

Switch

Initial Context Setup Response
S1-AP
SCTP
IP
Ethernet

Controller

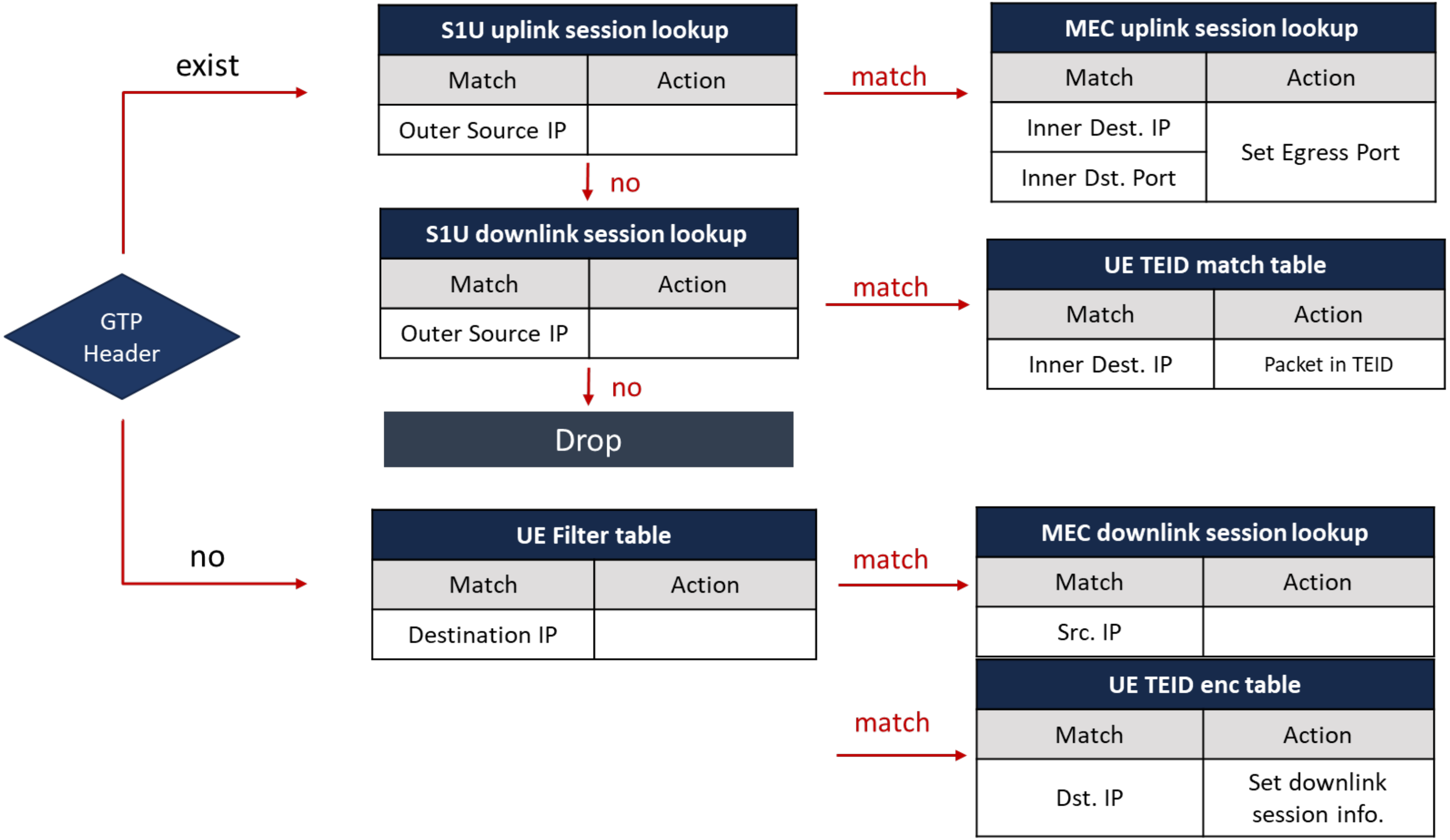
MME-UE-ID	SGW Addr.	ENB Addr.	DNS Addr.	UE Addr.	DL TEID	UL TEID
112233	10.0.9.2	10.0.9.100	8.8.8.8	192.168.3.2	777	1



DNS traffic redirection

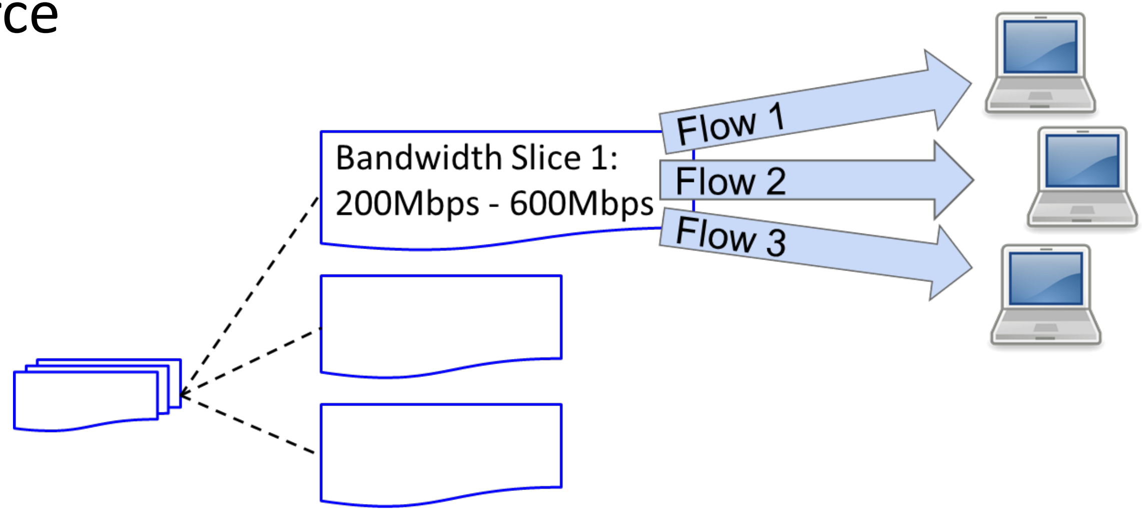
1. UE send DNS requests to ask for a specific service on the internet
2. Switch redirect the DNS query to MEC
    - Target service can be provided by MEC
        - Response the request by MEC address
    - Target service cannot be provided by MEC
        - Response the request by real service address
3. UE send normal traffic to service

Implementation of mec-spgw.p4



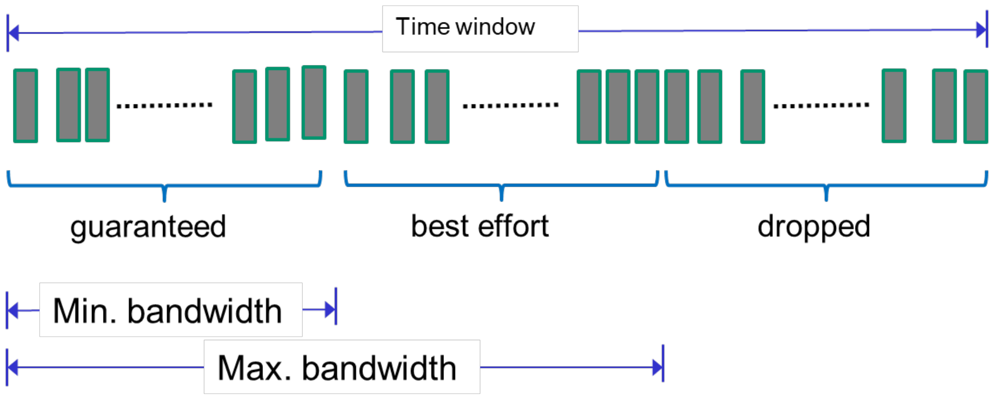
Design Concept for Bandwidth Slice Management

- Bandwidth slice
  - Contain disjoint traffic flows identified from user-defined field
  - Reach isolation of bandwidth resources by priority forwarding
- Aggregated traffic flow in a slice will share the bandwidth resource



Policy of Bandwidth Management

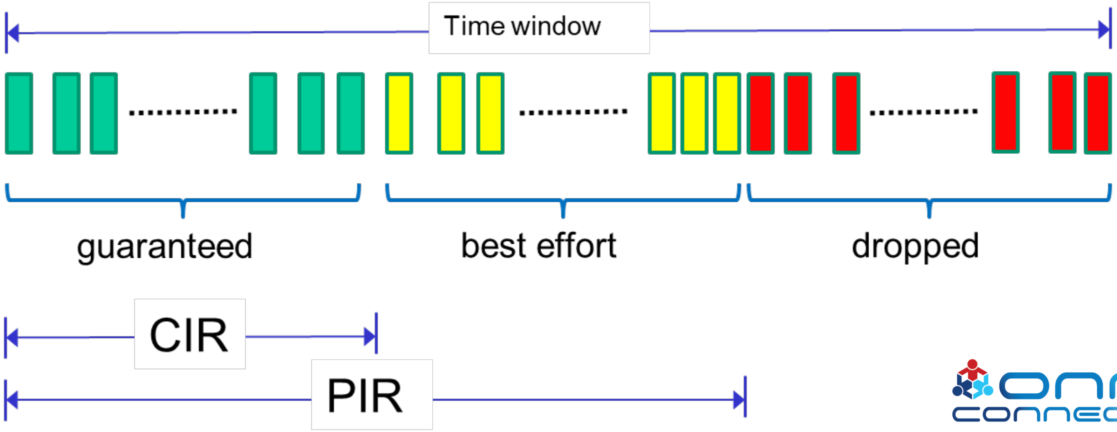
- Slice Traffic (aggregated traffic flows)
  - Guarantee minimum bandwidth
  - Best effort delivery without any guarantee
  - Limit maximum bandwidth



- Unspecified Traffic
  - Best effort delivery without any guarantee

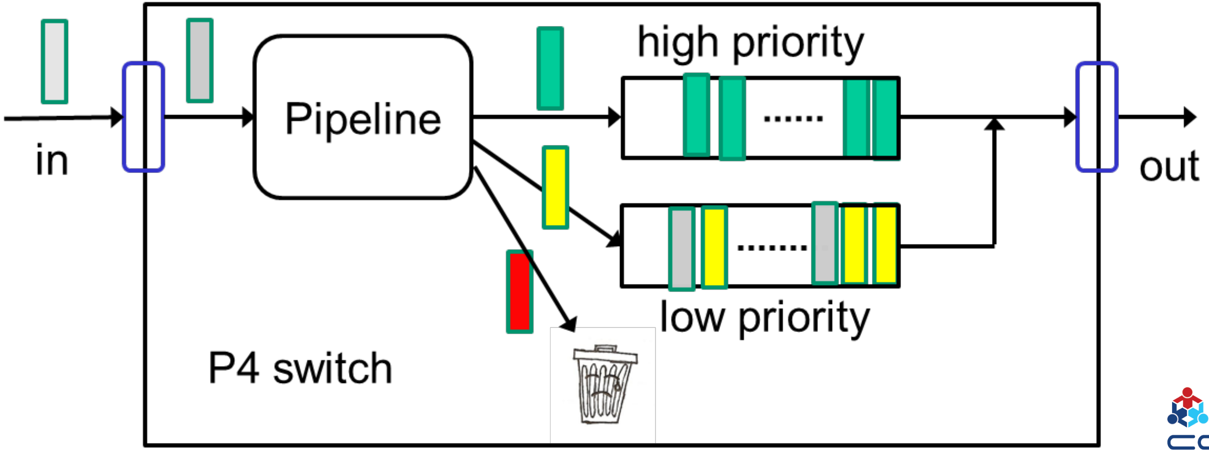
Packet Classification

- P4 Meter with Two Rate Three Color Marker classification
  - minimum bandwidth: Committed Information Rate (CIR)
  - maximum bandwidth: Peak Information Rate (PIR)
- Color result
  - Green: Guarantee traffic
  - Yellow: Best Effort traffic
  - Red: Abandon traffic



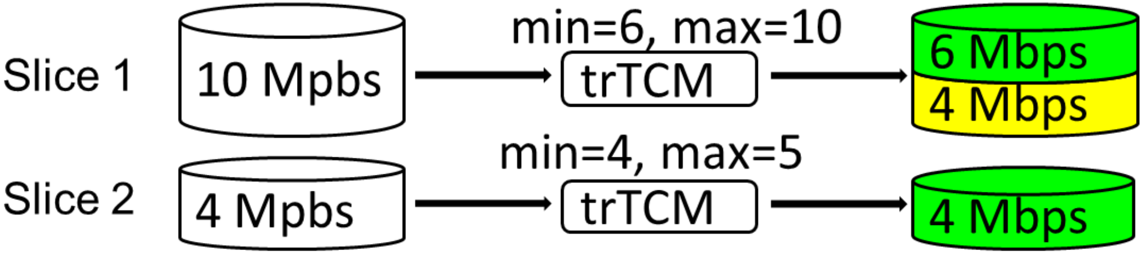
Priority Forwarding

- Guarantee traffic
  - Request bandwidth cannot exceed link available bandwidth
- Best Effort traffic
  - Contain unspecified packets
  - Deliver by residual bandwidth
    - Maximize bandwidth utilization
- Abandon traffic

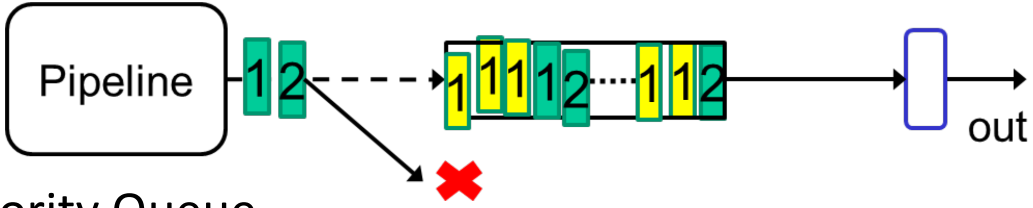


Priority Forwarding - Two-Level Priority Queue

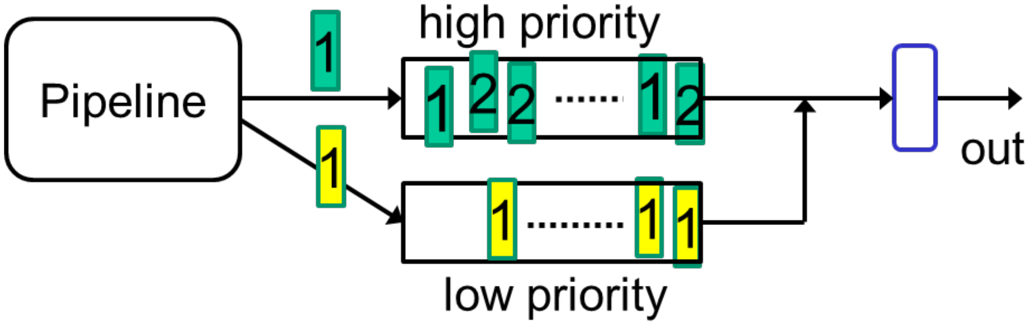
- For example:



- Single Queue: Best effort packet interference



- Two-Level Priority Queue



Implementation of BW-Slicing.p4

- Extension from ONOS Basic pipeline
  - Provides fundamental data-plane functionalities of the switch

mec-spgw.p4 pipeline → Basic.p4 pipeline → BW-Slicing.p4 pipeline

Slicing Table

Match	Action
Whatever	tag_slice_id
Match	
Fields	
You	
Like	

(Unspecified fields treated as wildcard)

→

Classifier Table

Match	Action
slice_id	set_color
	set_uncolor

→

Policer Table

Match	Action
packet_color	set_priority
	drop
	…

→

Guarantee Forwarding Table

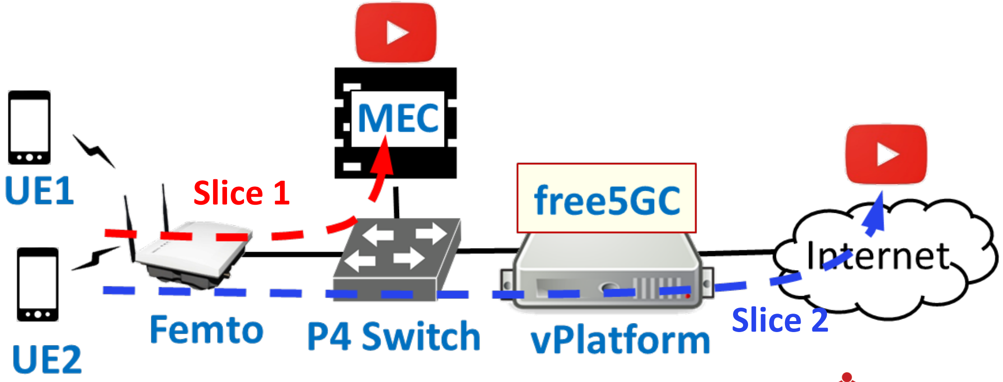
Match	Action
…	…

Best Effort Forwarding Table

Match	Action
…	…

ONF Connect 2019

- 5G Mobile Platform with P4-enabled Network Slicing and MEC
  - Compliant with **ETSI MANO**
  - NCTU **free5GC**
  - Loading Reduction in **MEC** with **P4 Switch**
  - P4-enabled **network slicing**



Thank You

Follow Up

willson6688@gmail.com