5G Communication with a Heterogeneous, Agile Mobile network in the Pyeongchang Winter Olympic competitioN

Grant agreement n. 723247

Deliverable D4.1
Operator grade NFV-based and SDN-enriched EPC environment at 5GTN

Date of Delivery:	31 May 2017 (Contractual)	31 May 2017 (Actual)
Editor:	Jari Moilanen	
Associate Editors:		
Authors:	Jussi Pajunpää, Petri Uosukainen, Olli Liinamaa, Jari Moilanen, Ijaz Ahmad, Mika Ylianttila, Madhusanka Liyanage, TaeYeon Kim, Wouter Tavernier	
Dissemination Level:	PU	
Security:	Public	
Status:	Final	
Version:	V1.0	
File Name:	5GCHAMPION_D4.1_Final.pdf	
Work Package:	WP4	

	Title:	Delivrable 4.1: Operator-grade NFV-based and SDN-enriched EPC at 5GTN
	Date: 31-05-2017	**Status:** Final
	Security: PU	**Version:** V1.0

Abstract

This deliverable provides a description of operator grade NFV based and SDN-enriched EPC environment at 5GTN environment. This deliverable contains the design, implementation and deployments of European vEPC, and Korean vEPC for 5G Champion. It describes also interconnectivity of European 5GTN mobile core testbed, and Korean mobile core testbed. It also contains a study on the existing security solutions and the security threats on future virtualized mobile networks. It defines the KPIs for security and possible validation ideas.

Index terms

EPC, vEPC, SDN/NFV, MANO, interconnectivity, security.

Contents

List of Acronyms

5G	5th Generation
5GTN	5G Test network
CLI	Command Line Interface
DNS	Domain Name System
EPC	Evolved Packet Core
GUI	Graphical User Interface
HSS	Home Subscriber Server
LB	Load Balancer
MAF	MME Application Function
MG	Mobile Gateway
MIF	MME Interface Function
MME	Mobility Management Entity
MPH	MME Packet Handler
NCIO	Nokia Cloud Infrastructure OpenStack
NFV	Network Functions Virtualization
NFV-MANO	Network Functions Virtualization Management and Orchestration
NFVI	Network Function Virtualization Infrastructure
OAM	Operation and Maintenance
QoS	Quality of Service
PCEF	Policy and Charging Enforcement Function
PCRF	Policy and Charging Rules Function
PGW	Packet Data Network Gateway / PDN Gateway
SAM	Service Aware Manager
SDN	Software Defined Networking
SGW	Serving Gateway
VM	Virtual Machine
VNF	Virtual Network Function
VNFC	Virtual Network Function Component
VMG	Virtualized Mobile Gateway
VMM	Virtualized Mobility Manager

Table of Figures

1 Introduction

D2.1 describes European vEPC Architecure, and Korean vEPC Architecure. This deliverable will describe the design, implementation and deployments of European vEPC, and Korean vEPC.

The document is structured further as follows. Section 2 describes European vEPC design, implementation and deployment. European vEPC is implemented using Nokia commercial products: MME is called 9471 Wireless Mobility Manager (WMM), and SGW/PGW is called 7750 Virtualized Mobile Gateway. This vEPC is used for interoperability with Korean core network.

Section 3 describes Korean mobile core testbed design, implementation and deployment. Korean setup contains 3 types of testbeds. One testbed led by ETRI is for PoC and demonstration of use cases defined in D2.2 focused on interoperability with European core network. Other 2 types of testbeds are focused on Multi-RAT SDI Edge Cloud and Distributed Mobility Management as the university federation test bed for academic research.

Section 4 describes interoperability connectivity of European 5GTN mobile core testbed, and Korean mobile core testbed. Section 4 describes also Quality of Service (QoS) issues, and network monitoring issues.

Section 5 describes security architecture for virtualized mobile core network. Section 5 contains SDN based Virtualized Core Network Architecture, and security of SDN based Virtualized Core Network Architecture. It contains discussion of Data Link Security, Control Channels Security, and Control Plane Security.

Section 6, concludes the document, by capturing the main outcomes of WP4 so far, and sketches the open points and future work remaining in the project.

2 5GTN mobile core testbed

2.1 Design

2.1.1 Software Defined Networks in Mobile Networks

Mobile network operators are facing a growing challenge thanks to the explosive increase in data traffic due to the prevalence of smartphones and streamed audio and video services. In this new paradigm, the operators need to manage the traffic load, and meet rising consumer and enterprise expectations for excellent performance while providing ubiquitous broadband connectivity. Operators must also roll out new services and applications rapidly to maintain a competitive edge. Slow service rollouts are no longer acceptable. Finally, in every competitive market there is constant pressure to become more efficient; in other words, to maintain or improve performance at a lower operational cost. Existing mobile networks struggle with limitations such as stationary and expensive equipment, complex control protocols, and heterogeneous configuration interfaces.

5G network will apply SDN principles within the mobile networking environments namely SDMN (Software Define Mobile Networks) to be able to address these current limitations. SDN decouples control and data planes leveraging standard protocols enabling remote management and operation of data planes to third-party elements. A synchronization protocol is required for communicating both planes; one such protocol is OpenFlow. The benefits of SDN seem obvious in the area of cloud computing networking; however, the application to the mobile paradigm requires further study.

There are several papers describing the integration of SDN in mobile networks [1-5]. In those papers the proposal consists of adding SDN agents in the mobile network elements. The first paper SoftRAN [1] proposes a centralized architecture as an alternative to the distributed control plane currently implemented in LTE networks. It abstracts out all the base stations deployed in a geographical area as a virtual big-base station while considering all the physical base stations as just radio elements with minimal control logic. These radio elements are then managed by a logically centralized entity which makes control plane decisions for all the radio elements in the geographical area. We call this logically centralized entity, the controller of the big base station. The controller maintains a global view of the radio access network and provides a framework on which control algorithms can be implemented. The second paper CellSDN [2] pushes fine-grained packet classification to the access switches, which can be implemented easily in software (e.g., using OpenvSwitch). These access switches apply fine-grained rules, specified by the controller, to map UE trace to the policy tags and hierarchical addresses. To ensure control-plane scalability, a local agent at the base station caches the service policy for each attached UE. The third paper [3] defines that each base station has an access switch that performs fine-grained packet classification on trace from UEs. Access switches can be software switches (such as OpenvSwitch) that run on commodity server hardware. The server can also run a local agent that caches service policies for attached UEs, to minimize interaction with the central controller. The rest of the cellular core consists of core switches, including a few gateway switches connected to the Internet. These core switches perform multi-dimensional packet classification at high speed, but only for a few thousands or tens of thousands of rules. We assume that the packet-processing hardware can perform arbitrary wildcard matching on the IP addresses and TCP/UDP port numbers. The IETF also defined some problem statements [6] for Mobile Service Providers (MSP) where one of the identified statements is the following. Some MSPs operate over multiple geographies and couple infrastructure from different MSPs, SPs and

possibly SaaS offerings. In these cases, it is important to provide the MSP that offers the ultimate service to the customer with a clean, consistent and efficient interface to all of the infrastructure it relies on.

There are couple of in-depth scientific contributions dealing with mobile network architectures that combine the concepts of cloud computing, SDN and NFV. First architecture proposals [1-5] - especially in the context of Cloud-RAN - include the mapping of the network functions that are required for the integration of mobile networks with SDN technology. These functions are only the mobile network control functions, i.e., MME, HSS, PCRF and the control planes of S/P-GW. Additional functions include transport, load balancing, security, policy, charging, monitoring, QoE or resource optimization. These functions run on the Mobile Network Cloud as SDN applications and enforce the desired function by means of SDN technology. With this approach, the user plane is only composed by strategically located SDN capable switches and regular switches. SDN switches could either replace partly or entirely the current mobile transport network. This consolidated architecture is shown in Figure 1.

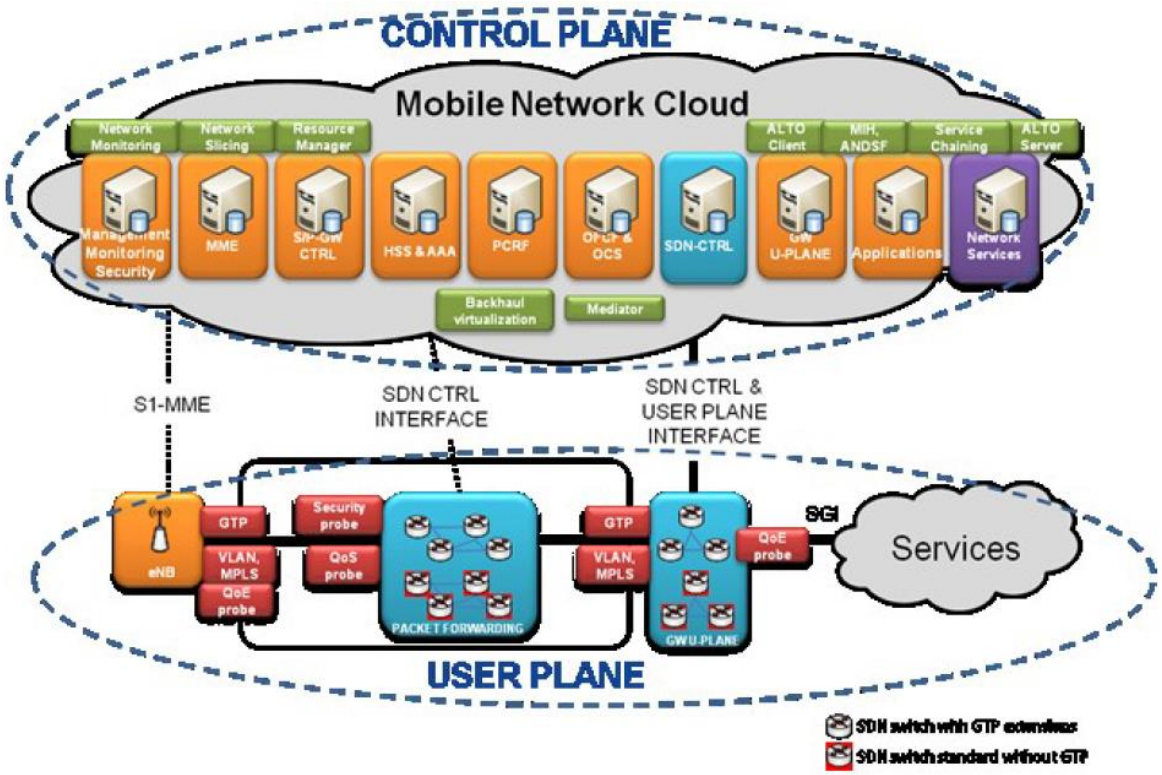

Figure 1 - SDN based consolidated architecture towards 5G [5]

The required EPC network elements run on the cloud to benefit from virtualization. Latency constrains could affect the deployment location of some compute nodes running virtual. Some strategic functions could be placed close to the eNBs or even on some switches, creating a decentralized cloud. In this architecture, the EPC network elements maintain

current 3GPP interfaces to favour migration from legacy mobile networks. That will allow a seamless migration for a new architecture.

Starting from late 1990's the 3GPP has been taking steps towards a clear separation of data and control planes and the respective elements in the architecture. We propose to take this concept to the next level following the SDN paradigm. Figure 2 presents the 5G network control as a group of SDN applications. They are the Base Station App, Backhaul App, Mobility Management App, Monitoring App, Access App, and Secure Service Delivery App. The network applications are orchestrated via the Controller Northbound API. Multiple SDN applications operate without conflicts.

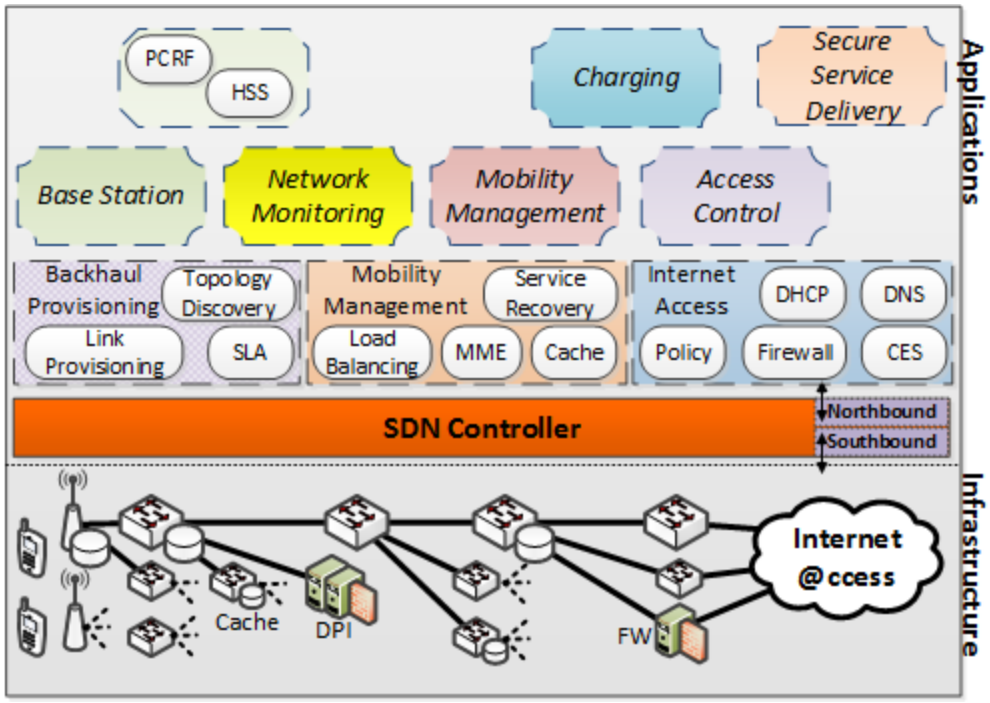

Figure 2 – 5G as a group of SDN Apps [7].

The Base Station App runs the control software that is now vertically integrated with the eNB. The physical base stations under its control consist of an antenna, a band pass filter and an Ethernet card for backhaul connectivity [7]. The Mobility Management App (MM App) implements mobility as a service (MaaS) and incorporates the MME. In addition, it needs to manage the Quality of service for each user, balance the load among the alternative paths across the aggregation network and to route the user to a cache, when possible. The MM App also chooses the path for a device. The load balancing decision is made based on input from the Network Monitoring App. In any case, it is desirable that the point of attachment of a mobile to the Internet is fixed while it stays under the coverage of the current mobile network [7].

In one physical mobile network there may be many Access Apps. In that case, an Access App is owned and operated by a particular Mobile Virtual network Operator (MVNO). Putting

mobility aside, it is the Access App that is responsible for providing the data services to mobile users. Key properties of the Access App include providing Internet access, firewalling unwanted traffic and providing access to premium content [7].

2.1.2 Network Functions Virtualization (NFV) and Network Functions Virtualization Management and Orchestration (NFV-MANO)

NFV is a network architecture concept that proposes using IT virtualization related technologies to virtualize entire classes of network node functions into building blocks that can be connected or chained, to create communication services.

NFV relies upon, but differs from, traditional server virtualization techniques such as those used in enterprise IT. A virtualized network function (VNF) can consist of one or more virtual machines running different software and processes, on top of industry standard high volume servers, switches and storage, or even cloud computing infrastructure, instead of having custom hardware appliances for each network function. NFV reference architecture is shown in Figure 3.

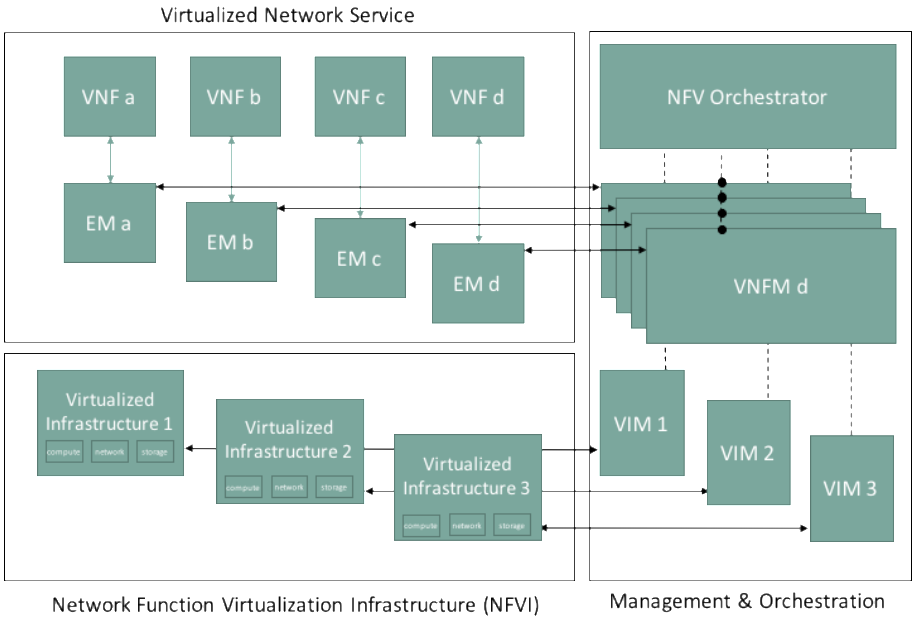

Figure 3 – NFV reference architecture

The NFV framework consists of the following main components:

• NFV Infrastructure (NFVI) is the totality of all hardware and software components that build up the environment in which VNFs are deployed. The NFVI can span across several locations. The network providing connectivity between these locations is regarded to be part of the NFVI.

• Network Functions Virtualization Management and Orchestration Architectural Framework (NFV-MANO) is the collection of all functional blocks, data repositories used by these functional blocks, and reference points and interfaces through which these functional blocks exchange information for the purpose of managing and orchestrating NFVI and VNFs.

The building block for both the NFVI and the NFV-MANO is the NFV platform. In the NFVI role, it consists of both virtual and physical processing and storage resources, and virtualization software. In its NFV-MANO role, it consists of VNF and NFVI managers and virtualization software operating on a hardware controller. The NFV platform implements carrier-grade features used to manage and monitor the platform components, recover from failures and provide effective security — all required for the public carrier network.

Nokia follows the principles of ETSI Network Function Virtualization (NFV) reference architecture for virtualized environments. Nokia cloud products support NFV management and orchestration.

Network function virtualization infrastructure (NFVI) includes all software and hardware components, providing the environment in which VNFs are deployed, managed, and executed. Typically, this is achieved by deploying virtualization layer (hypervisor, for example, Linux/KVM) over legacy IT hardware. In the scope of implemented Cloud, the used NFVI is Nokia Cloud Infrastructure on OpenStack (NCIO).

2.1.3 Cloud Run-time Architecture

Virtualization is the act of creating virtual versions of HW platform, operating system, storage, and networking resources. Virtualization transforms the physical requirements for the execution of an application into virtual ones, and enables more than one applications to run on the same platform.

Virtualization is supported by the following layers:

- HW, a physical HW infrastructure, providing the actual execution environment
- Host Operating system (Host OS), an operating system running on a the physical HW
- Hypervisor, a virtual operating platform for the execution of guest operating systems and in turn the guest applications. Hypervisor is responsible for allocating HW resources to VMs
- Guest Application and Guest OS, encapsulated inside a Virtual Machine (VM) which provides the runtime environment for them

Figure 4 below depicts a generic, layered cloud run-time architecture.

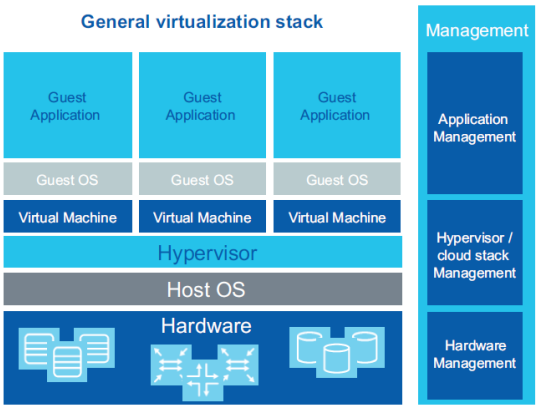

Figure 4 – Virtualization stack

Cloud stack management groups HW into three logical resources:

- Compute nodes, providing computing capacity for the use of the Guest Applications
- Controller nodes, providing supervision functions for the Compute nodes and networking as required for the Guest Applications
- Network, composed of the Network Interface Cards (NIC) hosted by Compute nodes, Controller nodes, and the virtual and physical switches and routers that interconnect these nodes.

Virtualization optimizes HW utilization efficiency by abstracting the hardware layer from the software layer.

2.2 Implementation

2.2.1 Nokia Cloud Infrastructure Software

The main software building blocks in the Nokia Cloud Infrastructure are:

- OpenStack
- Red Hat Enterprise Linux
- HostCC
- OpenStack node types

2.2.1.1 OpenStack

OpenStack is a powerful, open source platform for cloud-based applications. It is an integration of several components which provides services for deploying applications in the cloud. Nokia Cloud Infrastructure includes Red Hat Enterprise Linux OpenStack Platform® 6 (OSP6).

2.2.1.2 Red Hat Enterprise Linux

Nokia Cloud Infrastructure relies on Red Hat Enterprise Linux 7.1 as host operating system. Red Hat Enterprise Linux 7.1 is a leading OS for enterprise application, with a wide install base in IT and Telco applications. It has been fully integrated and tested within the Nokia Cloud Infrastructure solution.

2.2.1.3 HostCC

Nokia Cloud Infrastructure provides a cloud host environment, based on Red Hat Enterprise Linux 7.1 with additional functionality developed by Nokia, to properly host Nokia Telco cloud applications. Nokia Cloud Infrastructure also includes Open Source Software, and selected third party components that provide additional functionality.

Host Cluster Controller (HostCC) is the management entity responsible for the hardware and cloud lifecycle. Some of the key functions of HostCC are:

- Enabling Nokia Cloud Infrastructure deployment from bare metal hardware.
- Monitoring Nokia Cloud Infrastructure platform components (hardware and software), and produces alarms when needed.
- Providing automatic evacuation of VMs in case of hardware or software failure.

HostCC provides the functionality of Virtualized Infrastructure Manager in the NFV reference architecture framework.

A simplified diagram of a Nokia Cloud Infrastructure with the specific location of HostCC is provided in the Figure 5 below.

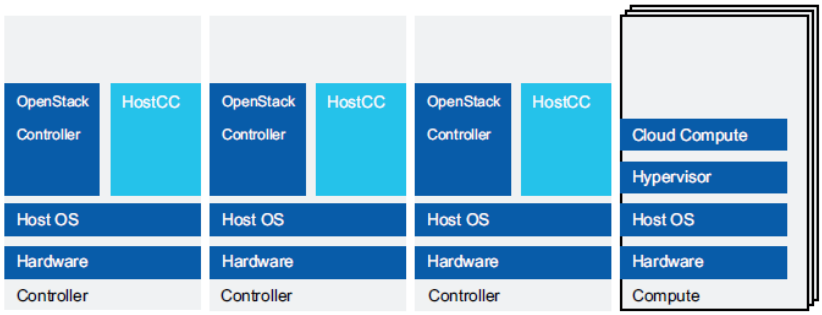

Figure 5 – HostCC in Nokia Cloud Infrastructure Controller blades

2.2.1.4 OpenStack Node Types

Cloud architecture is structured with four basic types of resources:

- Compute
- Controller
- Network
- Storage

A concrete cloud is defined by the specific combination of those nodes as resources. Such resources are created, deployed and undeployed according to the actual service needs.

Compute nodes provide computing capacity for the use of the actual applications that are expected to be hosted in the cloud. In the Nokia Cloud Infrastructure context, a compute node provides computing power for Nokia Telco applications.

Controller nodes provide supervision functions. They also provide networking as required for the cloud applications.

2.2.2 Virtualized Mobility Manager (Nokia VMM)

2.2.2.1 9471 VMM Implementation as VNF

Nokia virtualized MME implementation is called 9471 VMM.

The 9471 VMM is implemented as VNF consisting of following VNF components (VNFCs): OAM (OAM Server), MIF (MME Interface Function), MAF (MME Application Function), and MPH (MME Packet Handler). Each VNFC is deployed as Virtual Machine (VM).

VMM implementation is show in Figure 6.

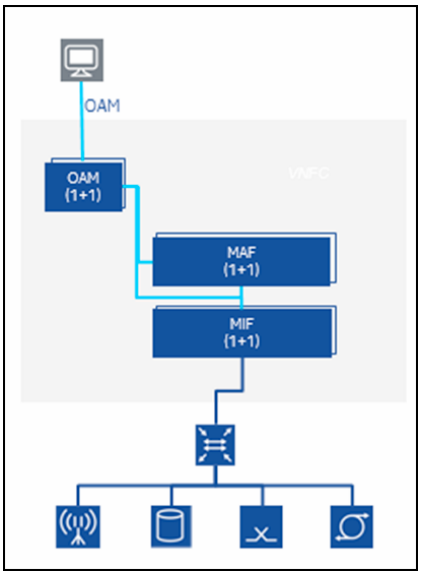

Figure 6 – VMM implementation

2.2.2.2 MME management

Nokia portfolio has SAM 5620 (Service Aware Manager) product. 5620 SAM implements NFV Management and Orchestration (NFV-MANO) functions to monitor and manage the VMM virtual network-function components and coordinates with the Virtualized Network Function Manager to provide lifecycle management of underlying VMM VMs.

VMM life cycle management includes the following operations:

- VNF instantiation
- Scaling: scale-in/de-scaling
  - Growing a VM host
  - Degrowing a VM host
- Healing — rebuilding a virtual machine
- Re-instantiating VNF components

However 5620 SAM is not yet utilized in European 5GTN test bed so far. MANO tasks are done manually with OpenStack user interface, and Command Line Interface (CLI) tools.

MME management is done using CLI (Command Line Interface) and MI-Agent GUI. MI-Agent GUI is a Web based tool.

OA&M functions on the MI-Agent GUI are organized and displayed according to FCAPS (fault, configuration, accounting, performance, security). High-level functions include:

- Fault Management:
  - Collect, store, display, and clear alarms and events
  - Monitor the system
- Configuration management:
  - Discover new or deleted subnetwork elements (SNEs).
  - Software administration including backup and restore, software updates, and growth and degrowth.

- Basic provisioning not covered by the SAM GUI (such as the provisioning of the EMS northbound interface)
- State and status management

- Performance management:
  - Collect, schedule, and display performance measurements
  - Run call trace for a specific UE
  - Overload monitoring and control
- Security management
  - View users roles
  - Manage passwords
  - Manage SNMPv3 interface security

Figure 7 shows a snapshot of MI-Agent GUI view.

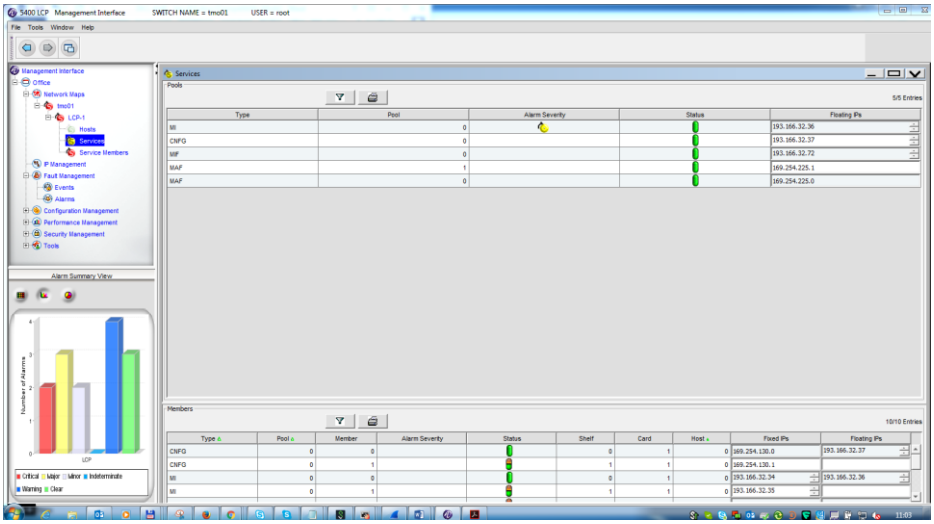

Figure 7 – MI-Agent GUI view

2.2.2.3 Hardware Requirements

The hardware for 9471 VMM must be x86-64 Intel processors based with hyper-threading and VT technology, support KVM hypervisor, and provide a Data Plane Development Kit (DPDK)-supported Ethernet interface of either Virt-IO igbe/ixgbe or a SR-IOV 82599 ixgbe driver.

The 9471 VMM also provides driver support for the Linux Guest O/S for the Mellonax X4 NIC (MCX4121A-ACAT).

2.2.2.4 Security

Networking for the 9471 VMM implementation will vary based on the capabilities of the underlying cloud network infrastructure.

With the 9471 VMM, the service provider has the networking responsibility and should be aware of the following security considerations:

- How to separate and secure their internal network connectivity between 9471 WMM VMs (OAM, MIF, etc.) from other vEPC elements (vSGW, vPGW, etc.) and from

external networks. It is highly recommended that the customer separate signaling, management, and internal networks using dedicated and separate Layer 3 IP subnets.

Also, it is recommended that the customer separate signaling, management and internal networks using Layer 2 separation using VLANS where possible.

- It is highly recommended that the customer uses firewalls to isolate external IP networks from any internal IP network subnets.

2.2.2.5 Reliability — Availability Zones

Reliability of the 9471 VMM is commensurate with the underlying cloud. Assignment of VMs to hardware is a function of the cloud orchestration use of availability zones.

Availability zones enable the arrangement of compute hosts into logical groups and provides a form of physical isolation and redundancy from other availability zones, such as by using separate power supply or network equipment. An availability zone is defined so that a specified compute host resides locally on each server. An availability zone is commonly used to identify a set of servers that have a common attribute. For instance, if some of the racks in a data center are on a separate power source, the servers in those racks in their own availability zone. Availability zones can also help separate different classes of hardware. When users provision resources, they can specify from which availability zone they want their instance to be built. This allows cloud consumers to ensure that their application resources are spread across disparate machines to achieve high availability in the event of hardware failure.

The 9471 VMM supports 2, 4, and 8 availability zone configurations.

Number of availability zones	Services per zone
2	all
4	OAM, non-OAM
8	OAM, MIF, MAF/IPPU

Table 1 – Availability zone configurations

2.2.3 Virtualized Mobile Gateway (Nokia VMG)

2.2.3.1 VMG Overview

The Virtualized Mobile Gateway (VMG) supports mobile gateway functionality that can be deployed on a generic computing infrastructure in a cloud environment. The VMG can support multiple GW functions including PGW, GGSN, SGW, SAE-GW (combined SGW/PGW/GGSN and ePDG). A VMG instance consists of multiple virtual machines (VMs) running on a generic computing infrastructure such as x86 servers. Each VM is dedicated to a specific set of functions that can be replicated across many similar VMs. A group of VMs is represented as a single instance of an application as they operate in-sync with other similar VMs in the group to support a network function. The ability to add multiple VMs for each function allows the VMG to scale horizontally and support a scaling range of a few thousand

to several million devices. The VM, within a VMG instance, is agnostic of other virtual machines present in the shared server environment.

2.2.3.2 VMG Implementation as VNF

The VMG architecture is comprised of the following virtual network function components (VNFC), as shown in Figure 8:

- OAM-VM
- Load Balancer VM (LB-VM)
- Mobile Gateway VM (MG-VM)

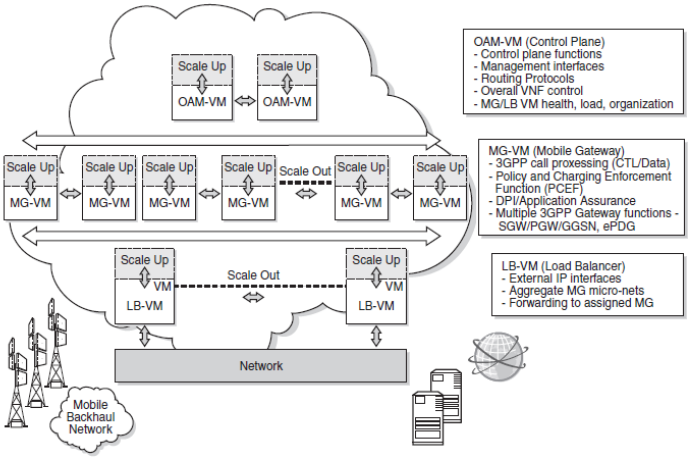

Figure 8 - VMG VNFC Architecture

OAM-VM—The OAM-VM component performs control plane functions that include VNF-VNFC management, routing protocols, management interface - SNMP/TELNET/SSH/CLI for the configuration, KPI-KCI periodic XML report generation, and so on.

LB-VM—The LB-VM component provides network connectivity to the mobile gateway function and load distribution across the MG-VMs. It also forwards the GTPC/GTP-U and UE addressed packets to the MG-VM. The LB-VM can provide a single common IP for network interfacing elements (MME, eNodeB, S/PGW, TWAG, ePDG).

Alternatively, each signaling and data plane interface can be configured on separate IP addresses. The ability to separate individual functions or merge multiple functions on individual interfaces allows maximum flexibility for various deployment cases of a VMG instance.

MG-VM—The MG-VM services include 3GPP call processing (Control and Data Plane), Policy and Charging Enforcement Function (PCEF), Application Assurance (PCEF enhanced with ADC for application detection and control and with L7 service classification for policy charging control). The MG-VM supports all 3GPP gateway functions such as SGW, PGW/GGSN, SAE-GW (S/PGW/GGSN), ePDG, TWAG and so on. The supported service functions depend on the configurable personality of the MG-VM. As SGW, the MG-VM provides bearer management services and the mobility anchor point. In a MG-VM that is configured as a PGW, the VM provides bearer management, IP anchor point, lawful interception functionality, charging functionality, QoS enforcement and advance DPI

functionality. When configured as a SAE-GW (combined SGW and PGW), all the SGW and PGW functions are supported on a single MG-VM.

2.2.3.3 VMG System Architecture

The VMG system architecture has three system components, as shown in Figure 9:

- x86 host
- virtual machine (VM)
- VMG guest operating system (OS).

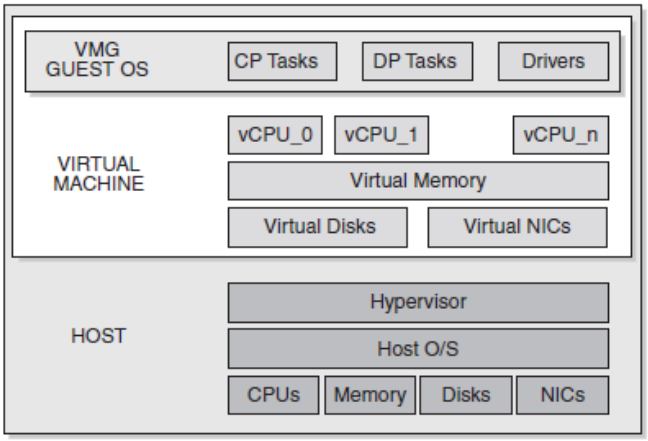

Figure 9 - VMG System Architecture

2.2.3.3.1 VMG Guest OS

The VMG is built on the SR OS, which in this context is the guest OS. The VMG guest OS is a true 64-bit operating system, capable of accessing and using more than 4Gb of virtual machine memory.

The VMG supports symmetric multi-processing (SMP) on the control plane, which distributes the processing workload over the available vCPUs for optimum performance.

The VMG has the following functional components:

- control plane software modules—to perform control plane functions of the VMG, including routing and signaling protocols, subscriber management, OAM functions, and higher-level ISA functions
- forwarding plane software modules—to perform data forwarding functions of the VMG
- management plane software modules—to allow in-band and out-of-band management of the node
- drivers—to improve data processing. The VMG software includes the VirtIO paravirtualized driver that inter-operates with the KVM hypervisor in order to optimize the data path between the VMG and the input and output ports on the x86 host.

2.2.3.4 VMG Deployment Model

In a distributed VMG deployment model, a VMG instance consists of multiple VMs that may be hosted across multiple distributed servers. Each VM within a VMG instance communicates with other participating VMs through an internal (inter-VM) network.

The distributed VMG provides full flexibility to deploy individual VMs on a generic server platform independent of infrastructure, including compute server, host OS, hypervisor, cloud management and compute/network orchestration. The VMG can be deployed on Intel x86 servers, independent of specific vendor brand or product.

The OAM-VM requires a management interface to support SNMP/SSH/CLI and other OAM functions for network connectivity. Participating VMs (OAM-VM, LB-VM and MG-VM) communicate with each other using dedicated inter-VM control and the data networks. Multiple data networks can be created for the following:

- to forward inter-VM data between the LB-VM and MG-VM
- to support high-throughput applications
- to provide protection against failure of physical links and switches of the underlying network between the LB-VM and MG-VM

2.2.3.5 VMG Redundancy Model

The VMG supports the following redundancy protection models:

- intra VNF 1:1 redundancy
- inter VNF 1:1 redundancy

2.2.3.5.1 Intra VNF 1:1 Redundancy

Figure 10 shows an example of a VMG that supports intra VNF 1:1 (active/standby) redundancy protection.

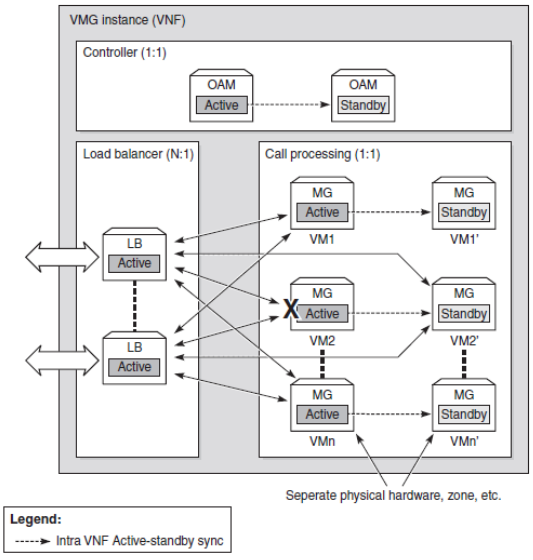

Figure 10 - Intra-VNF 1:1 (active-standby) Redundancy Protection

MG-VMs (call-processing VMs) and OAM-VMs (controller VMs) are configured in 1:1 (active/standby) pairs. The load balancer/IO VM (LB-VMs) use the N:1 method and operate in an all-active mode.

Active MG-VMs participate in call setup and data path forwarding, and also synchronize all bearer/service data flow (SDF) /charging information with the standby VMs. Standby MG-VMs do not take part in call setup or data path forwarding but are available for service in hot state.

OAM-VMs operate in 1:1 (active/standby) redundant mode. Active OAM-VMs take part in management and routing protocols processing and synchronize information to the standby VM. When it detects failure on an active OAM-VM, the standby OAMVM takes over for the failed VM with no control plane outage.

An OAM-VM monitors all active MG-VMs. It triggers a switchover to the standby MGVM in the event of active VM failure. Standby MG-VMs maintain active session states and take over the call-processing of existing sessions with no control plane impact and minimum data plane outage. VMG VMs can also be spread on separate hardware platforms and in separate locations, assuming low latency and high bandwidth on the network connecting the VMs.

2.2.3.5.2 Inter VNF 1:1 Geo-Redundancy

Figure 11 shows an example of a VMG that supports inter VNF 1:1 (active/standby) geo-redundancy protection across different locations.

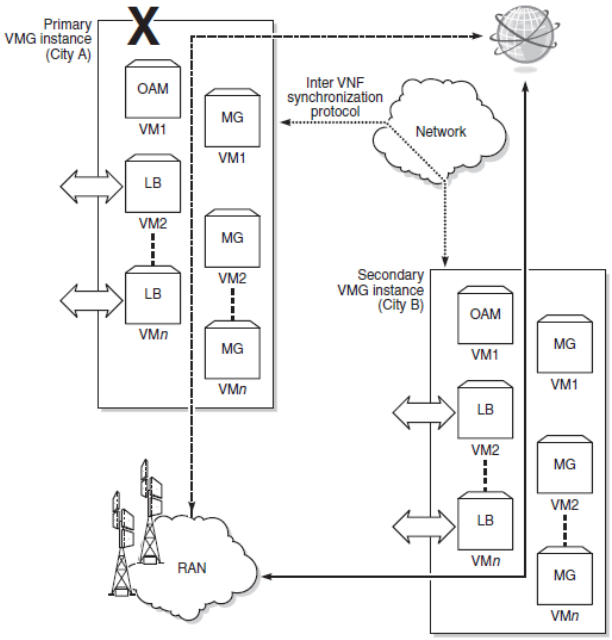

Figure 11 - Inter-VNF 1:1 (active-standby) Geo-Redundancy Protection

In this protection scheme, two independent VMG instances (primary and secondary) are configured in a 1:1 (active/standby) pair. The active or primary VMG instance participates in call processing and data path forwarding.

The Active VMG instance synchronizes all bearer/SDF flow/charging information to the standby or secondary VMG instance. The standby VMG does not take part in call setup or data path forwarding. However, among other functions, synchronization of bearer/SDF/charging is available for service in hot state, thus assuring minimal outage

Routing protocols are used (low metric towards active and high metric towards standby) to attract control plane and data plane traffic to the active VMG instance. Plus, inter VNF 1:1 geo-redundancy can be deployed with or without intra VNF 1:1 level redundancy. The synchronization and election of the active and standby VMG uses a unique Nokia Inter-VNF Synchronization protocol to ensure that bearer information is synchronized on a per MG-VM basis.

2.2.3.6 VMG Requirements

2.2.3.6.1 Physical CPU Considerations for VMs

The maximum number of vCPUs that can be made available to the VMs running on the host application depends on the attributes of the host CPUs. Each of the following attributes affects the number of vCPUs that you can configure for the VM applications:

- the number of physical CPUs
- the number of cores per CPU
- hyperthreading capabilities

Although the VMG does not require multiple vCPUs to support basic operations, Nokia recommends that you assign multiple vCPUs to each VMG VM. All vCPUs for a VM should be associated with the same physical CPU (that is, NUMA zone).

2.2.3.6.2 Linux Server OS, Hypervisor and Cloud Orchestration

The hypervisor software allows you to create, configure, and run VMs on the host machine. Table 2 lists the OS, hypervisor, and software tools that are required and supported.

Software	Version Supported
Host OS [1]	CentOS Linux
	RHEL
	ESXi
Hypervisor	KVM
	ESXi
KVM Tools	Libvirt toolset, which includes virsh toolset to create VMs
Cloud Orchestration	Openstack • Minimum: OpenStack Juno • Recommended: OpenStack Mitaka VMware • Minimum: vSphere 5.5 • Recommended: vSphere 6.0 Require CPU NUMA zone and CPU pinning support.

Table 2 - Server OS and Hypervisor Requirements

2.2.3.6.3 x86 Host

The x86-based host layer comprises third-party, industry-standard hardware and software components:

- the physical hardware includes the following components:
  - central processing units (CPUs)—systems with one or more CPU sockets are supported. The minimum processor requirement is the Intel Ivy Bridge CPU. Later-generation Haswell and Broadwell CPUs can also be used. Only Intel processors are supported.
  - memory— the host requires a minimum of 24 Gb of memory. This memory must be available on top of the memory requirements of the VMs hosted on the compute node.
  - disk storage—sufficient disk storage must be available for the OS, VM image files and VM storage requirements. Recommended VM storage space is 32G (it depends on the duration of CDR storage capacity required).
  - network interface cards (NICs)—physical network interface cards that are installed in the x86 host

2.2.3.6.4 VM

The VM emulates the server compute hardware on which the VMG runs. The VM, has the following components:

- virtual disks—the VMG virtual disk devices are equivalent to the compact flash devices on a physical router that store the system image and the boot files.
- virtual network interface cards (vNICs)—the vNICs map to the VMG ports, as required by the virtual application.
- memory—the recommended minimum memory for OAM-VM and LB-VM is 16 Gb and for MG-VM 32 Gb.
- virtual CPU (vCPU) cores—the recommended number of vCPUs is 8 cores for each VM.

2.2.3.7 VMG in OpenStack Environment

VMG is implemented using OpenStack. VMG VNFC Architecture is shown in Figure 12.

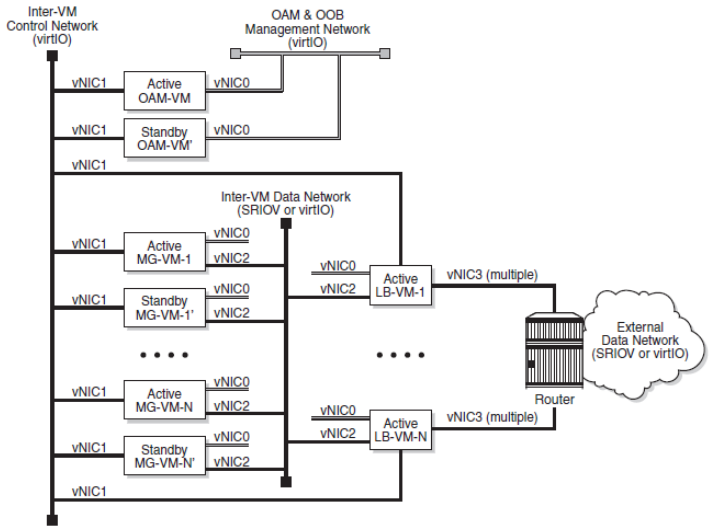

Figure 12 - VMG VNFC Architecture

2.3 Deployment

2.3.1 Network Topology

Figure 13 below shows European 5GTN Network Topology.

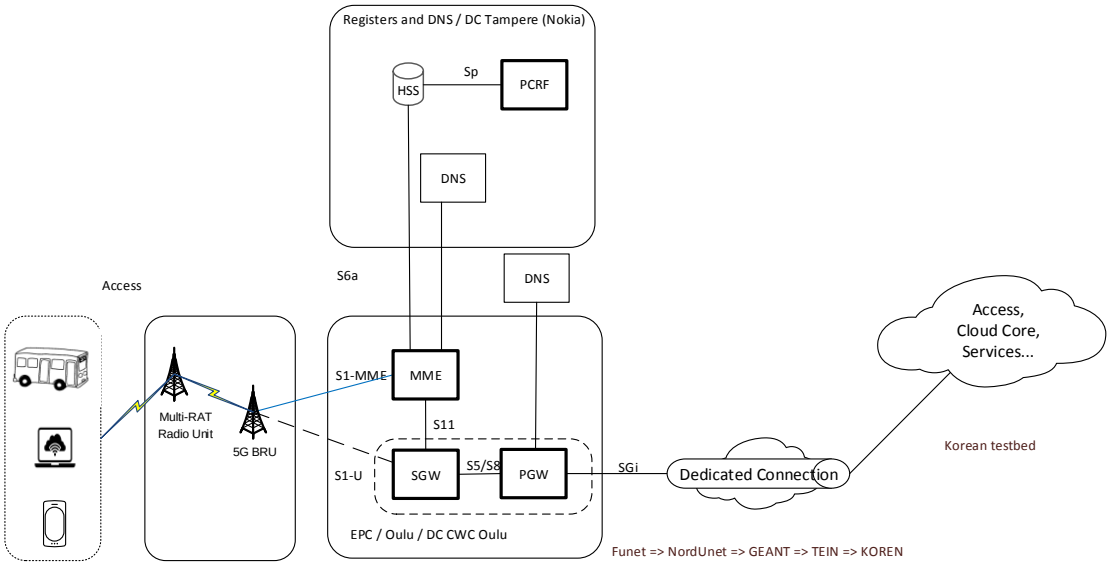

Figure 13 – European 5GTN Network Topology

vEPC functions MME (Mobility Management Entity), SGW (Serving Gateway), and PGW (PDN Gateway) are physically located in Oulu. HSS, PCRF, and DNS (for MME) are physically located in Tampere. PGW is using DNS from UOulu network.

Figure 14 below shows more detailed network diagram.

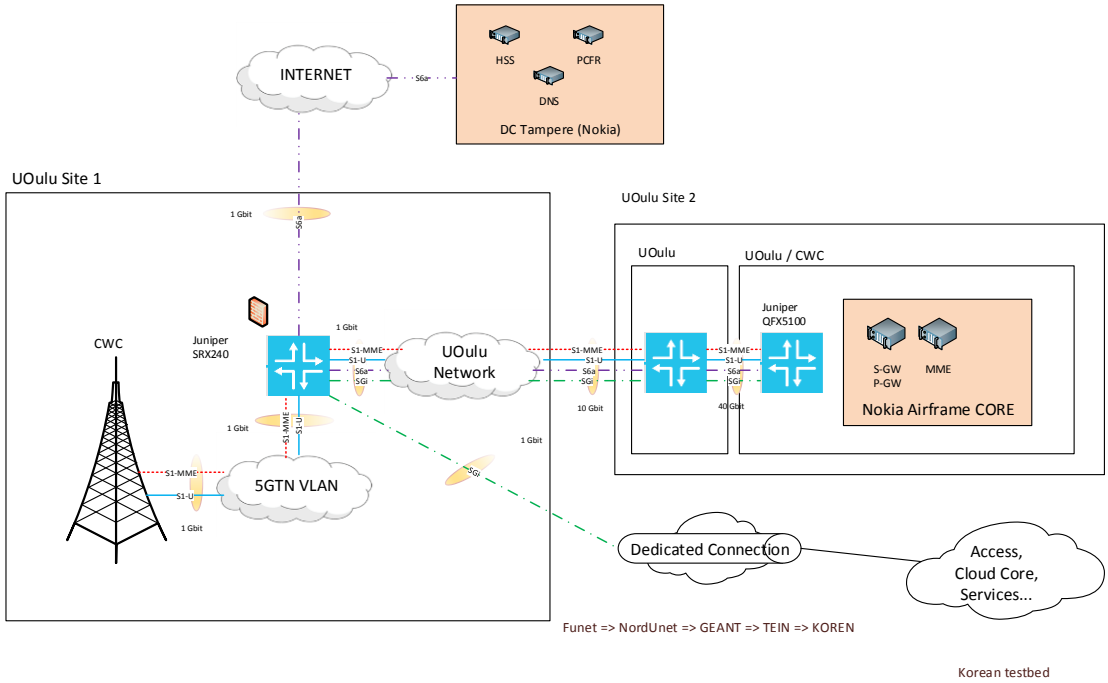

Figure 14 – European 5GTN Network Diagram

5GTN Network Elements in Oulu are physically located in two different sites. eNB's and Juniper SRX240 are located in Site 1. Juniper SRX240 is connecting external entities to 5GTN. DC Tampere is connected via L2VPN connection. Korean entities are also connected using L2VPN connection. Interconnection from Oulu to Korea is made through following research networks: Funet ⇔ NordUnet ⇔ GEANT Open ⇔ TEIN ⇔ KOREN.

EPC is physically located in Site 2, and connected through University of Oulu (UOulu) switches to radio access network.

2.3.2 Nokia Cloud Infrastructure Hardware (AirFrame)

The Nokia Cloud Infrastructure is based on a standard, commercial IT hardware platform from 3rd party which is productized as the Nokia Cloud Infrastructure Hardware. This Nokia reference hardware is called AirFrame.

Nokia Cloud Infrastructure is the common virtual hardware platform for cloud deployments.

The hardware configuration consists of one or two racks. European vEPC solution uses single rack configuration as shown in Figure 15.

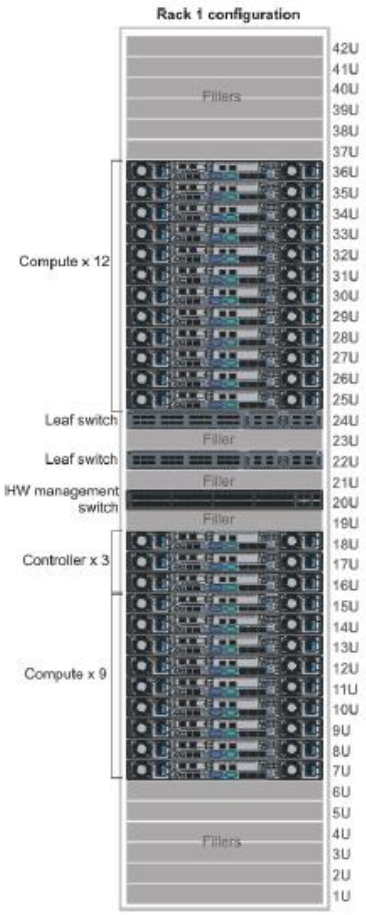

Figure 15 - Single rack configuration

European testbed vEPC has 11 servers, of which 3 servers are used as controller nodes and 8 servers are used as compute nodes, one HW management switch, and two leaf switches.

5GTN EPC AirFrame hardware is shown in Figure 16:

- Server type: 11 x Quanta B51BP-1U, manufactured by Quanta Computer Inc.
- HW management switch type: 1 x Quanta LB9, manufactured by Quanta Computer Inc.
- Leaf switches type: 2 x Juniper QFX5100-24Q switches each having 2 x QFX-EM-4Q expansion modules, manufactured by Juniper Networks Inc.

Figure 16 – 5GTN EPC AirFrame hardware

Three uppermost servers are controller nodes. Remaining 8 servers are compute nodes. Below servers is the management switch, and below it are the leaf switches.

2.3.3 VMM Deployment

VMM is implemented using 4 availability zones, each of which is having one server. VMM services are allocated as shown in Table 3 below:

Server	Availability Zone	Network Element	Services
Compute-3	zone1	VMM	VMM-OAM-0
Compute-4	zone2	VMM	VMM-OAM-1
Compute-7	zone3	VMM	VMM-MIF-00, VMM-MAF-00, VMM-MAF-02
Compute-8	zone4	VMM	VMM-MIF-01, VMM-MAF-01, VMM-MAF-03

Table 3 – VMM service allocation

VMM configuration for vCPU, Memory and Disk usage is shown in Table 4 below.

TYPE	QTY	DISK GB	vCPU	MEM GB	Total Disk	Total vCPU	Total MEM
QAM	2	76	10	32	152	20	64
MIF	2	30	20	32	60	40	64
MAF	4	30	10	32	120	40	128
Total	8				332	100	256

Table 4 – VMM configuration (vCPUs, Memory, and Disk)

2.3.4 VMG Deployment

VMG is implemented using 2 availability zones, each of which is having two servers. VMG services are allocated as shown in Table 5 below:

Server	Availability Zone	Network Element	Services
Compute-1	zoneMG1	VMG	reserved for future expansion
Compute-2	zoneMG1	VMG	VMG-MG-3, VMG-OAM-A, VMG-LB-1
Compute-5	zoneMG2	VMG	VMG-OAM-B, VMG-LB-2
Compute-6	zoneMG2	VMG	VMG-MG-4

Table 5 – VMG service allocation

VMG configuration for vCPU, Memory and Disk usage is shown in Table 6 below.

TYPE	QTY	DISK GB	vCPU	MEM GB	Total Disk	Total vCPU	Total MEM
OAM	2	4	8	16	16	16	32
LB	2	4	8	16	8	16	32
MG	2	11	8	16	22	16	32
Total	6				46	48	96

Table 6 – VMG configuration (vCPUs, Memory and Disk)

3 Korean mobile core testbed

There are three types of testbed for Korean core networks. One testbed led by ETRI is for PoC and demonstration of use cases defined in D2.2 focused on interoperability with European core networks. Other two types of testbed are focused on Multi-RAT SDI Edge Cloud and Distributed Mobility Management as the university federation test bed for academic research.

The testbeds for Korean 5G core network has two distinct requirements.

- Network Virtualization - In 5G system, mobile network/service functions run as SW on general-purpose servers. Dedicated networks such as disaster proof network, IoT network, and traffic network are virtualized as SW solutions on universal and common HW equipment. This means a new network can be easily deployed through modifying the configuration of the network using SW on the existing network and even customized as a dedicated network that meets an operator's business needs and facilitates introduction of application services.
- Network distribution and SW-based control architecture - In 5G where capacity per cell reaches 20 Gbps, traffic must be distributed before they can be processed. Servers (i.e. edge clouds) are distributed across the nation through SW virtualization, and network functions are placed on to the servers, effectively distributing traffic data. Resource allocation for virtualized network functions on the edge clouds are controlled by the orchestrator in the central location through SW.

3.1 Mobile Core PoC testbed

3.1.1 Design

Mobile Core Infrastructure

MCI (Mobile Core Infrastructure) provides environment for VNFs, including resources for computation, storage, and networking, is deployed by operators. MCI networks interconnect the computing and storage resources contained in the MCI PoP (Point-of Presence). This may include specific switching and routing devices to allow external connectivity. It delivers the actual physical resources and corresponding software on which VNFs can be deployed.

NFV decouples network services from the hardware that delivers them. As a result, functions for EPC (Evolved Packet Core) service can be delivered in software and deployed on general purpose appliances. This gives organizations a lot more flexibility in the way they design, deploy and manage their network services

Virtualization technology is necessary to allow any instance of one IT resource hosting multiple other IT resources, including applications, servers, clients, storage capacities, or networks. Virtualization therefore enables to get more value out of finite resource. With it, we can run more software and complete more processes with the same amount of hardware.

NFV enabled infrastructure

The evolved 5G network will be characterized by agile resilient converged fixed/mobile networks based on NFV technologies and capable of supporting network functions and applications encompassing different domains.

- Cloud-NFV Orchestration
  - Cloud NFV Integrated Management/Orchestration
  - Real-time Orchestration
- High Performance Multi-technology(Fixed, Mobile, Media, ...) Virtual Functions
  - Cloud NFV/MEC/5G/IoT Platform
- Fault Tolerant Virtual Infrastructure
  - Real-time monitoring & Fast Isolation/Recovery
- Data/Control Plane Deep Acceleration
  - High Availability & Performance Data/Control Plane Acceleration

SDN based infrastructure

The SDN paradigm provides a new capability for fast service provisioning. It started with limited networking environment such as cloud data centers and enterprise networks and widen its coverage into wide area transport network and wireless/wireline integrated multi-domain networks. Instead of applying it as a standalone network control tool, it is now used with NFV (Network Function Virtualization) and as a component of an end-to-end orchestration solution. It provides an intelligent knowledge plane to make control decisions via traffic steering, traffic engineering, and flexible service chaining for latency sensitive and reliability seeking applications. It can be used in efficient communications among distributed core functional components. Our SDN based infrastructure takes full advantage of such an integrated control as a sub-component of end-to-end orchestration system. Figure 17 shows the high-level architecture of SDN/NFV-based Korean distributed virtualized EPC.

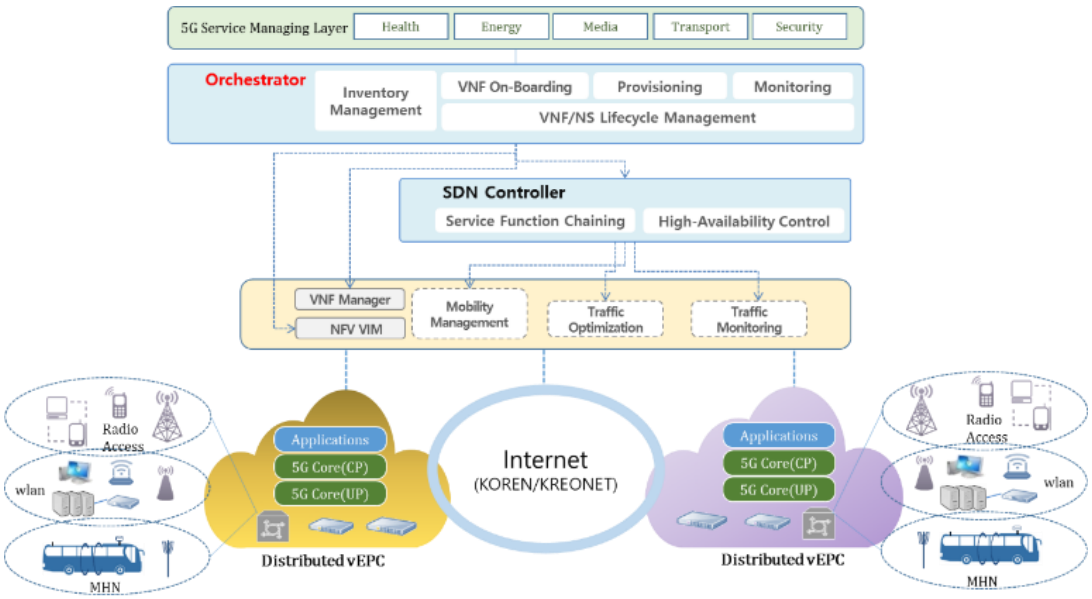

Figure 17 - High level architecture of Korea's Distributed virtualized EPC

Mobile Core Functions

Virtualized EPC supports following requirements to satisfy flexible 5G service characteristics.

- vEPC Support VNFs which follow cloud-native design principles. Assumptions are that such VNFs can be decomposed into many lightweight components and common platform services, are designed following a component-based software engineering style, and are built for quick restoration under failure conditions.
- Enhancements to the NFV ISG defined interfaces (MANO and NFVI) to provide flexible choices for the designers of VNFs may thus be needed, for example, in the area of provisioning common platform services, management of many dependent VNF components, and communication between many VNF components.
- Virtualised implementations of 5G network functions may be instantiated, terminated and updated more rapidly than traditional physical implementations, for example, to support the on-demand nature of network slicing.

3.1.2 Implementation

Mobile Core Infrastructure

MCI (Mobile Core Infrastructure), the hardware platform for 5G mobile core in Korea, creates a virtualization layer that sits right above the hardware and abstracts the HW resources, so they can be logically partitioned and provided to the VNF to perform their functions. NFV has a feature of delivery of network functionality via software running on industry-standard commercial off-the-shelf (COTS) hardware. The main advantages are that it can provide networking needs of a service provider or enterprises' application on standard server and storage infrastructures. New services do not require new hardware infrastructure – simply software installation.

The MCI was developed with the Brahmaputra release of OPNFV (Open Platform for NFV) which facilitates the development and evolution of NFV components across various open source ecosystems in order to create a reference NFV platform. OPNFV builds NFV Infrastructure (NFVI) and Virtualized Infrastructure Management (VIM).

Equilement	Specification					
	CPU	RAM	HDD	OpenStack	SW Ver.	OPNFV
NFVO Server	X86	64G	2TB	-	Ubuntu 14.04 LTS	-
VNFM Server	X86	64G	2TB	-	Ubuntu 14.04 LTS	-
VIM Server	X86	64G	2TB	Liberty	Ubuntu 14.04 LTS	Brahmaputra
NFVI Server	X86	64B	4TB	Liberty	Ubuntu 14.04 LTS	Brahmaputra
Hardware Common Platform	X86	64G	512G	Mitaka	Ubuntu 14.04 LTS	-

Switch	Interfaces	remarks
ToR	1GbE L2 Ethernet Switch	
Management Switch	10GbE L2 Ethernet Switch	
Data Swtich	10GbE L2 Ethernet Switch	
Storage Switch	10GbE L2 Ethernet Switch	

Table 7 – MCI specification

Virtualization SW (OS/ Network/ Hypervisor)

- Host OS: Linux (Ubuntu 16.04)
- Host Network: 10 Gigabit Ethernet enabling SR-IOV on 3400/8400/4500 Ethernet Adapters and configuring SR-IOV on hosts
- Host hypervisor: Type 1 (native or bare-metal) hypervisor => QEMU (with KVM as hardware acceleration)

NFV enabled infrastructure

NFV enabled infrastructure needs to create network services and manages them properly during their lifetime. The ETSI ISG NFV defined NFV management and orchestration (MANO) framework to essentially manage the network services in the NFV architecture. Our infrastructure conforms to the MANO framework.

The MANO framework is broken up into three functional blocks: NFVO, VNFM, and VIM.

- NFV Orchestrator (NFVO) is responsible for on-boarding of new network services (NS) and virtual network function (VNF) packages; NS lifecycle management; global resource management; validation and authorization of network functions virtualization infrastructure (NFVI) resource requests.
- VNF Manager (VNFM) oversees the lifecycle management of VNF instances; coordination and adaptation role for configuration and events reported from VNFs.
- Virtualized Infrastructure Manager (VIM) controls and manages the NFVI compute, storage, and network resources

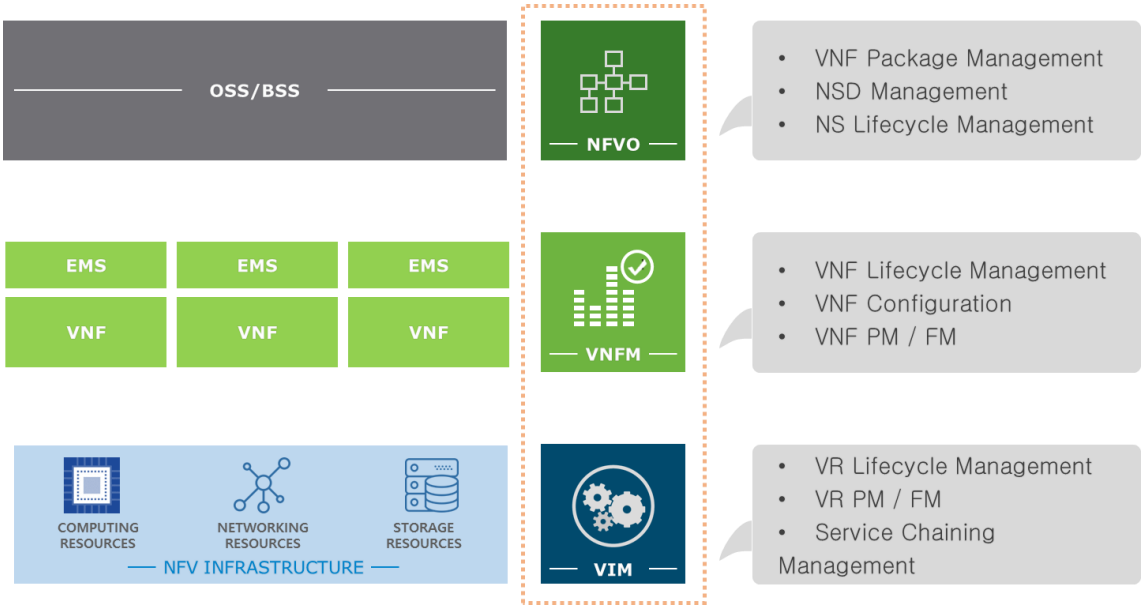

Figure 18 - NFV enabled infrastructure

We have implemented the NFVO from scratch. Our NFVO system provides a web-based user interface which enables administrators to easily handle NFV services. The developed NFVO system provides the following system features:

- multi-VIM support
- transport-SDN based inter-VIM networking
- VPN based inter-VIM networking
- Neutron port chain based service function chaining
- OpenStack (Liberty release) based VIM

On the contrary to NFVO, VNFM and VIM were implemented with existing open source projects. The VNFM was developed with Tacker, which is an official OpenStack project building a Generic VNFM and NFVO. In particular, we have expanded the Tacker source in order to support indirect mode of NFV. Since our hardware platform was built with the OPNFV, OpenStack was selected as the VIM.

Figure 19 shows the system architecture of the NFV enabled infrastructure. While the NFVO and VNFMs are centralized, VIMs and NFVIs are distributed. The single NFVO supports multiple VIMs and there can be various types of VNFMs. The two distributed VIMs and NFVIs can be connected by two types of networking services: VPN over Internet and T-SDN.

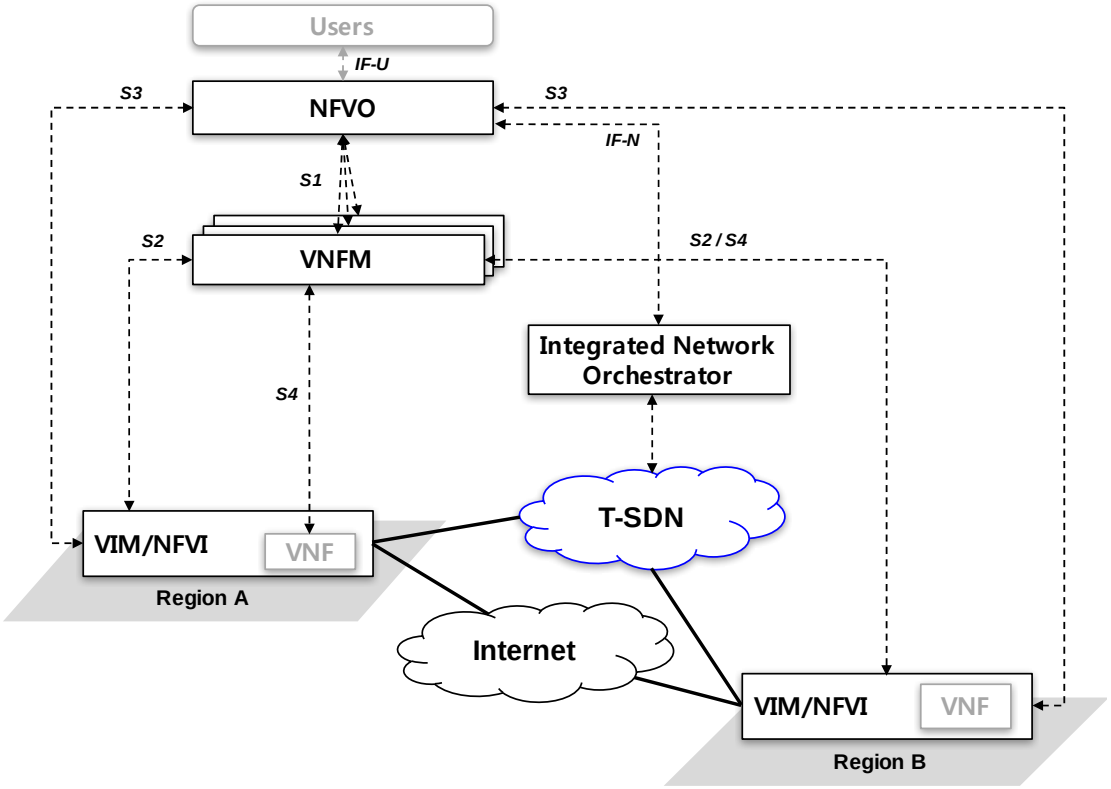

Figure 19 - System architecture of the NFV enabled infrastructure

Our NFV enabled infrastructure supports MANO system functions and interfaces as follows

- NFVO
  - NS (Network Service) lifecycle management
  - General virtualised resource management
  - VNF-related resource management in indirect mode
  - NS-related resource management performed by the NFVO
  - Virtualised resource performance management
  - Virtualised resource information management
  - NFP (Networking Forwarding Path) management
  - VNF lifecycle management
  - VNF instantiation
  - VNF termination
  - NS lifecycle management

- NS instantiation
- NS termination
- VNF Package management
- VNF instance information management
- NSD management
- Software image management
- NFV acceleration management

- VNFM
  - Virtualised resource management
  - VNF lifecycle management
  - VNF instantiation
  - VNF termination
  - VNF configuration management
  - VNF information management
  - VNF instance information management
  - VNF performance management
  - VNF indicator management

- VIM
  - Virtualised resource management
  - Virtualised resource performance management
  - Virtualised resource information management
  - NFP (Network Forwarding Path) management
  - NFV acceleration management (SR-IOV)

MANO system interfaces

- Interfaces based on ETSI NFV specifications
  - S1: based on ETSI GS NFV-IFA 007
  - S2: based on ETSI GS NFV-IFA 006
  - S3: based on ETSI GS NFV-IFA 005
  - S4: based on ETSI GS NFV-IFA 008

SDN based infrastructure

SDN capability is implemented as a part of VIM. It consists of two main functionalities: control and monitoring. For control, it responds to either VNFM or Orchestrator's network path(s) setup requests. The target network of the control is the one which interconnects various vEPC VNFs. It provides traffic steering and traffic engineering capabilities to optimize delay and resource usage. For monitoring, it collects various traffic status of the target network and analyzes the monitored data for performance and fault assurance purpose. Our current implementation provides the latter and we are planning to complete control functionality in the second phase of the project. Figure 20 illustrates a simple experimentation architecture of the SDN monitoring functionality of vEPC which is realized in a single host with multiple VMs for proof of concept. We are planning to provide a complete implementation of SDN capabilities including both control and monitoring in the second phase of the project.

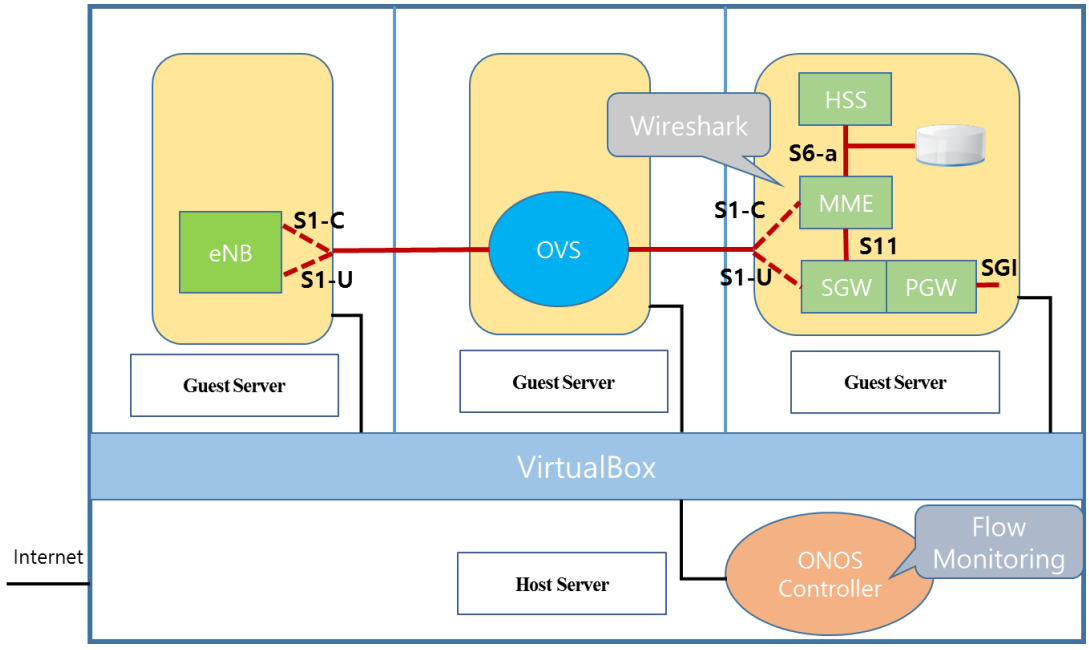

Figure 20 - SDN based traffic monitoring

vEPC (Mobile Core Functions)

vEPC consists of vMME, vSGW-CU, vPGW-CU, vSGW-DU, and vPGW-DU. Traditional EPC has combined SGW and PGW but vEPC has divided control plane and data plane in order to take better control over data flow. In particular, our vEPC uses DPDK with SRIOV to achieve desirable performance.

- Features: vEPC features separately installed in vm as VNF. This allows for more flexibility, scalability, and agility as resource status changes. Also, vEPC data plane is separated from control plane which enables higher capacity of data flow. DPDK and SRIOV is main enabler for such higher data throughput.

- Architecture: vEPC has five main components and two simulator components with EM. Five components are vMME, vSGW-CU, vPGW-CU, vSGW-DU, and vPGW-DU which provide main functionality of vEPC. Two simulators are HSS and PCRF for authentication. Each component is installed in an individual VM and controlled by EM. All components are connected to an internal network and, at the same time, MME, SGW-DU and PGW-DU are also connected to an external network for external access such as eNB and PDN as shown in the Figure 20.
- Functions: The goal of vEPC is to guarantee 10G end-to-end throughput. vEPC's separated data plane with DPDK - SRIOV allows VM performance and scalaiblity.

DPDK drivers have to be installed in each data plane VM. Once DPDK drivers are installed, each VM's conditions such as KVM version have to be configured. Next step is to install matching driver to host (compute node) with a dedicated physical 10G port device. By installing driver, VM is capable to connect to physical port by creating VF (virtual function). Once VF is created, PF (physical function) needs to be connected to VF individually so that VF can connect directly to the physical port.

When package gets downloaded from a repository, SOP agent will start process for each VNF. VNF agent will run when pre-made image is running, and it will operate as specified in the standard such as instantiation and monitoring performance. VNFM can request or send messages to VNF agent for instantiation and monitoring performance. The Figure 21 shows an example of instantiation message from VNFM to VNF agent. At the bottom of message 'vnfSpecificData" is where VNF agent will get rpm package from repository to install VNF package for instantiation. After all packages are installed and processes are running, VNF agent will collect monitored data for sending to VNFM. It includes usage of CPU, memory, files, network connection, and running processes for each VM. EM also can view and control process of each individual vEPC components.

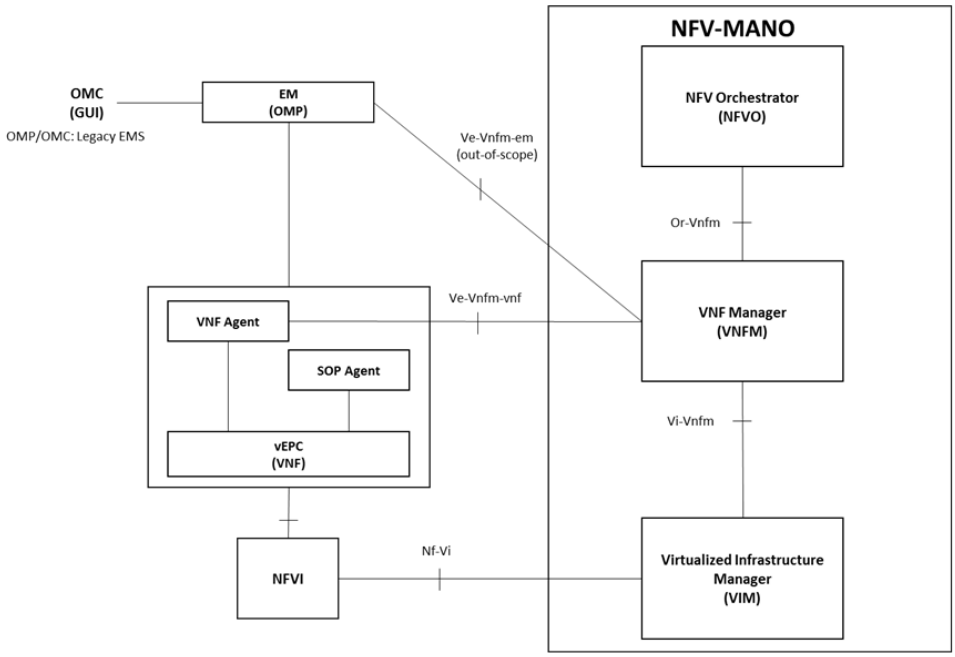

Figure 21 – Management interface of mobile core functions

3.1.3 Deployment

Infrastructure Deployment

There are three possible PoPs in Korea interconnected over KOREN. They are Seoul, Daejeon and Gangneung where PyeongChang Winter Olympic Games take place in 2018. We plan to deploy Mobile Core Infrastructure for 5G networks at those three sites for service deployment.

Virtualization technologies including SDN, NFV and Cloud will be deployed for flexible control and management over the targeted virtual mobile functions and related functions as well as virtual resources to support those functions. Orchestrator for multiple sites sitting in the Daejeon PoP to oversee the distributed PoPs via KOREN in Korea.

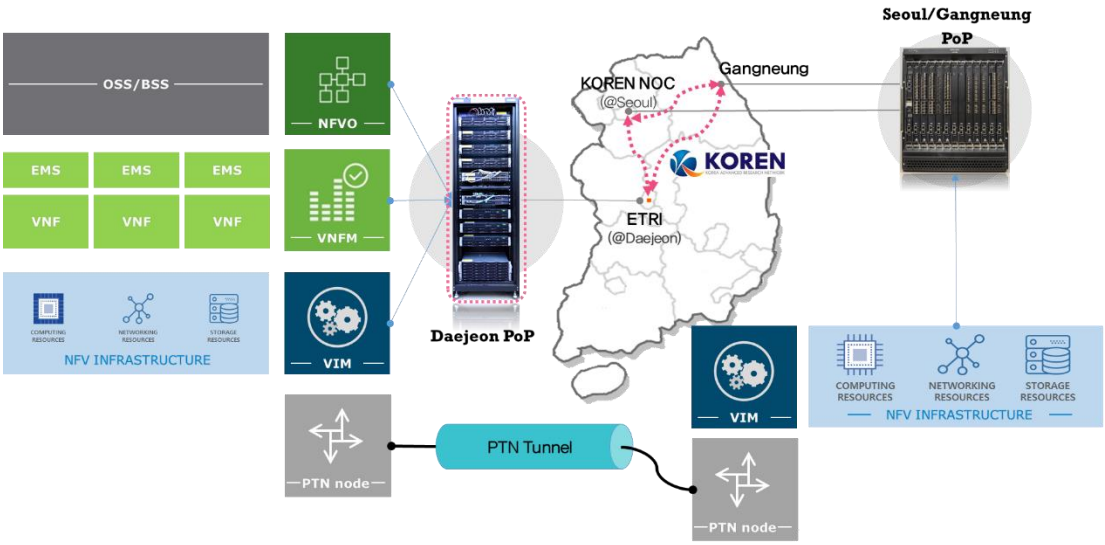

Figure 22 - Multiple mobile core PoPs in Korea

Service Deployment

- Network Service (NS) for vEPC
  - vEPC NS consists MME, virtual S-GW-CU (control unit), virtual S-GW-DU (data unit), virtual P-GW-CU (control unit) and virtual P-GW-DU (data unit).
  - In our deployment, vEPC NS does not cover remaining functionalities for vEPC (i.e., HSS and PCRF).
  - Virtual S-GW-DU and virtual P-GW-DU must have SR-IOV enabled ports in order to enhance their performance.

NFVO, VNFM, and VIM closely interwork with each other to create and manage network services. Figure 23 shows the procedures to instantiate a network service in the NFV enabled infrastructure.

1. OSS/BSS (or Administrators) requests to create a network service (NS) at NFVO by defining a new NS descriptor (NSD) or selecting one.

2. NFVO requests to allocate network resources at VIM, which connect VNFs composing the requested NS. In this step, management network resources are also created for management access.
3. Once the network resources are allocated, NFVO requests VNFM to instantiate VNFs. Since our NFV infrastructure is in the indirect mode, VNFM indirectly requests the VNF resource allocation at NFVO and then the request is sent to VIM.
4. When VNF resources are allocated, VNFM configures VNFs with any VNF specific parameters.

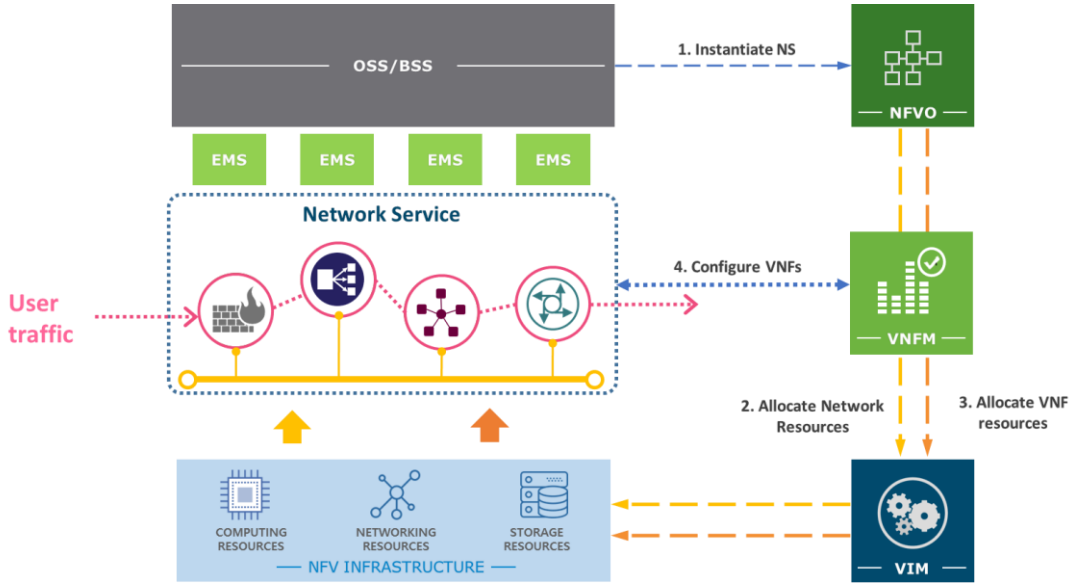

Figure 23 - Network Service Instantiation Procedures

The state of network services is monitored from two aspects. One is to monitor a service utilization aspect and the other is to monitor a resource utilization aspect. NFVO receives monitoring results from VNFM and VIM, and exploits those results to perform another management operations such as a scaling operation.

Figure 24 illustrates the two types of monitoring.

- VNF Monitoring: VNF providers can specify some indication on the VNF behavior and they include those information as a parameter (i.e., VnfIndicator) of VNF descriptor (VNFD). Based on the parameter, VNFM requests the actual value of a given indicator from the VNFs.
- Virtual Resources Monitoring: VIM keeps monitoring the allocated virtualized resources such as virtual compute, virtual storage, and virtual network.

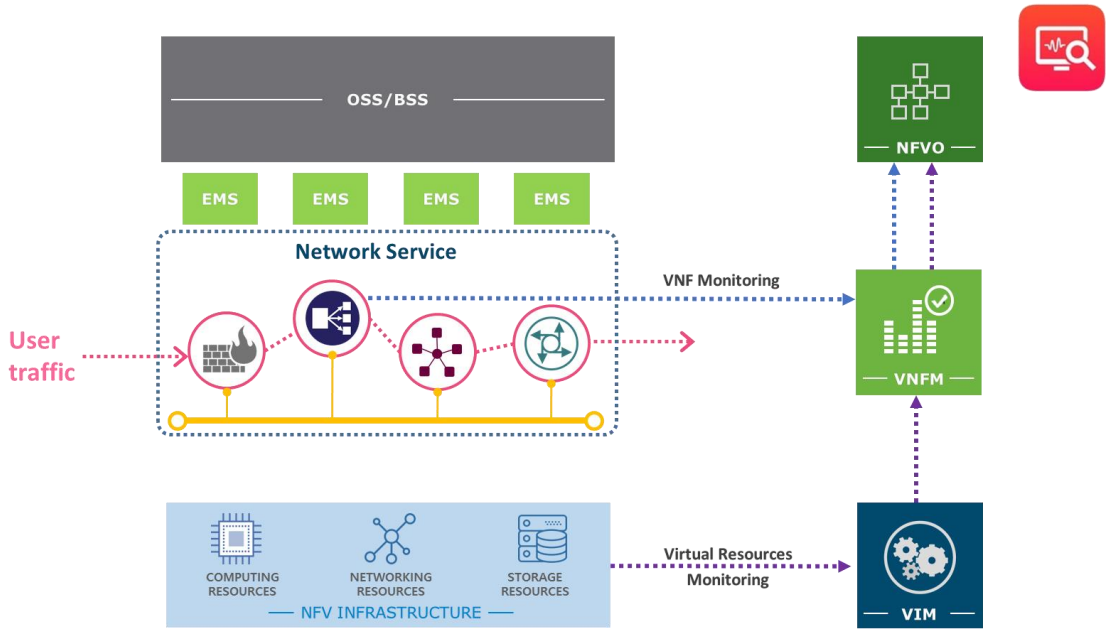

Figure 24 - VNF and Virtual Resources Monitoring

3.2 Multi-RAT Edge Cloud

3.2.1 Design

In addition to the ETRI-led testbed, it aims to build a Multi-RAT SDI Edge Cloud testbed as a university federation test bed for academic research. The Multi-RAT SDI Edge Cloud Testbed is deployed at the multi-site level as an economical small cluster that can demonstrate SDN / NFV / Cloud based software-defined infrastructure technology. The IoT-Cloud Hub for secure and flexible data collection and management for small devices, called IoTs using Wifi / LoRaWAN-based communication interfaces, is responsible for overall control of IoT devices. We are researching multi-RAT demonstration software for UE-based Giga-Ethernet / Wifi / 2G / 3G / 4G / 5G communication interface and will connect with IoT-Cloud Hub of multi-site based K-Cluster to be.

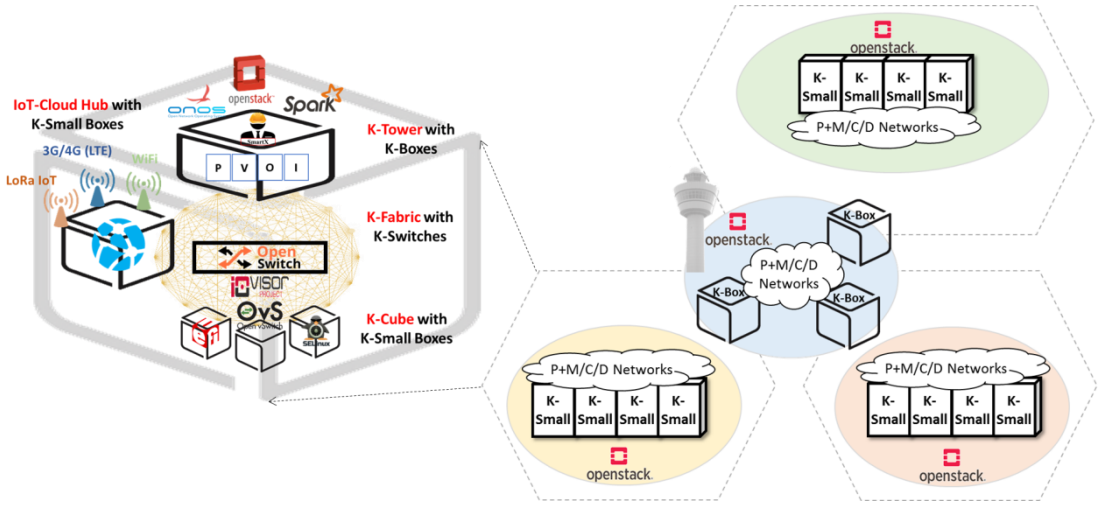

Figure 25 - Multi-site K-Cluster Design

K-Cluster:

K-Cluster defines an economical small cluster that can integrate SDN / NFV / Cloud-based software-defined infrastructure technology. K-Cluster must respond to the edge cloud that provides cloud resource clusters while serving as a middle box that supports SDN / NFV to provide an integrated common development environment. At the same time, flexible networking should be provided to expedite the processing of data originating from the Internet. By configuring the K-Cluster as a convergence white box and automatically managing the entire cluster with the central DevOps tools, a data-centric variety of service demonstrations should be constructed as an economical test bed that is as easy as possible.

- K-Cluster Components:
  - K-Tower: Automated control of the entire K-Cluster using the DevOps tool (Monitoring & Control)
  - K-Cube: Small resource clusters that provide real computing/networking/storage resources
  - K-Fabric: Connect all elements in the center of the K-Cluster and provide fast and reliable connectivity between components
  - IoT-Clod Hub: A bridgehead between the K-Cluster and the IoT, so validate data preferentially. Also control data path between K-Cluster and the IoT and manage the IoT devices and switches located between them.

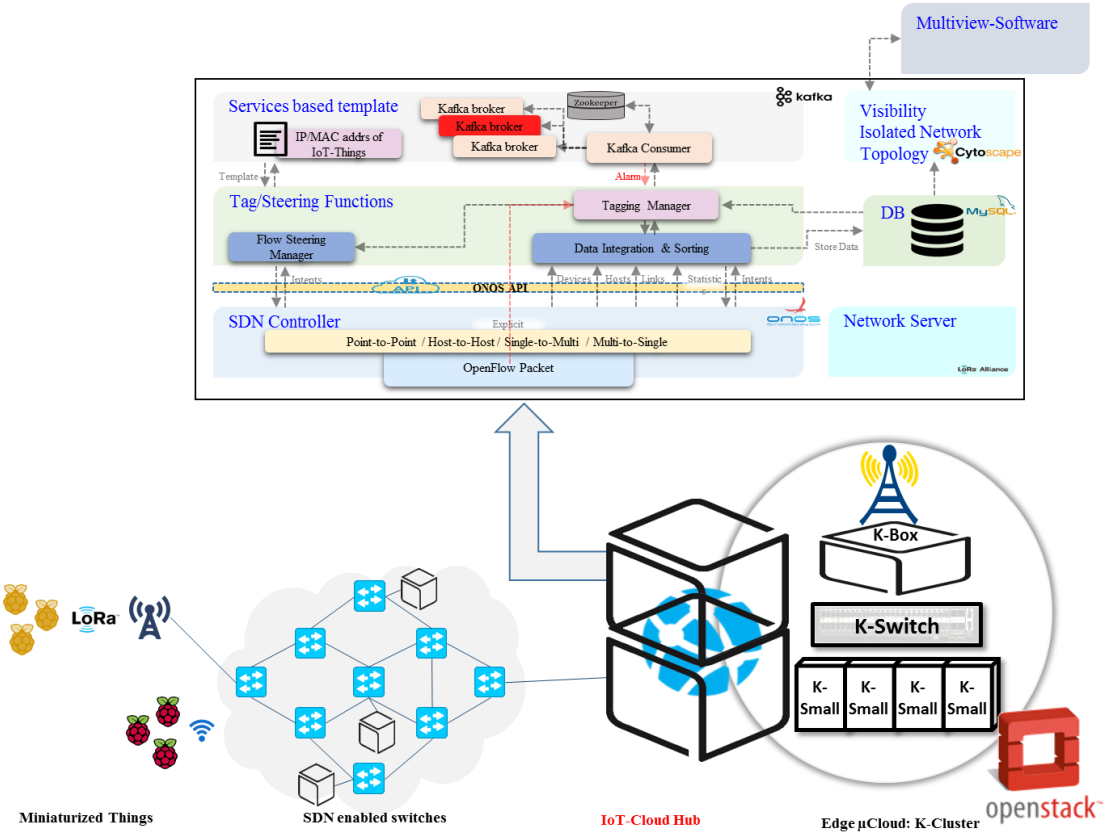

Figure 26 - IoT-Cloud Hub Design of K-Cluster

IoT-Cloud Hub

The IoT-Cloud Hub is a component of the K-Cluster and serves as a bridgehead for IoT things. The IoT-Cloud Hub conducts overall control of the IoT, collecting data from the IoT device and validating the data first. It also controls the data path using SDN to securely collect the data and manages the IoT device and the SDN-enabled switch. IoT-Cloud Hub targets SmartX IoT-Cloud Services for small-scale IoT things. The SmartX IoT-Cloud Service is configured using the Kafka Messaging System, and IoT things transfers data using the wired / wifi / LoRaWAN communication interface and is transferred to the IoT-Cloud Hub via SDN-enabled switches.

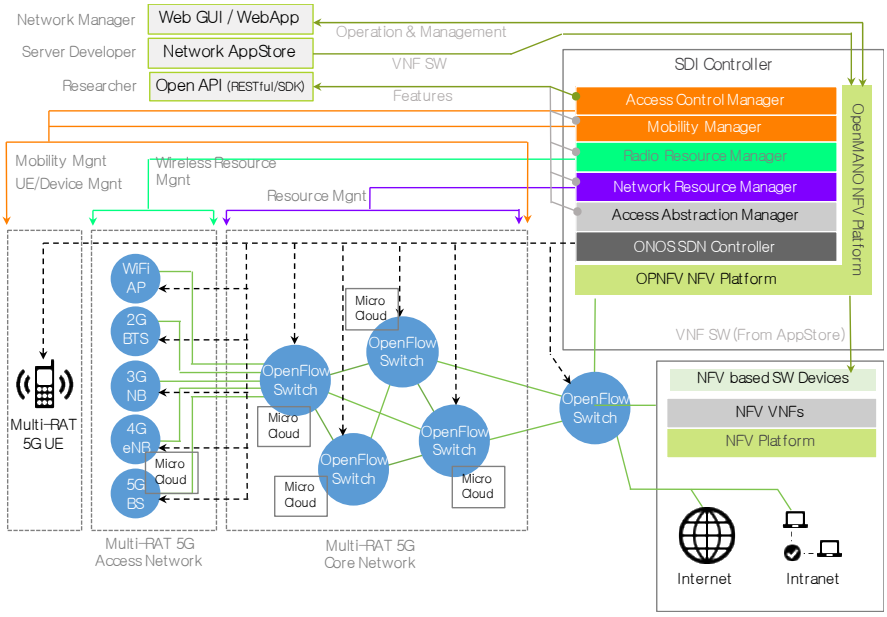

Figure 27 - SDI based Multi-RAT 5G UE/Network/Service testbed.

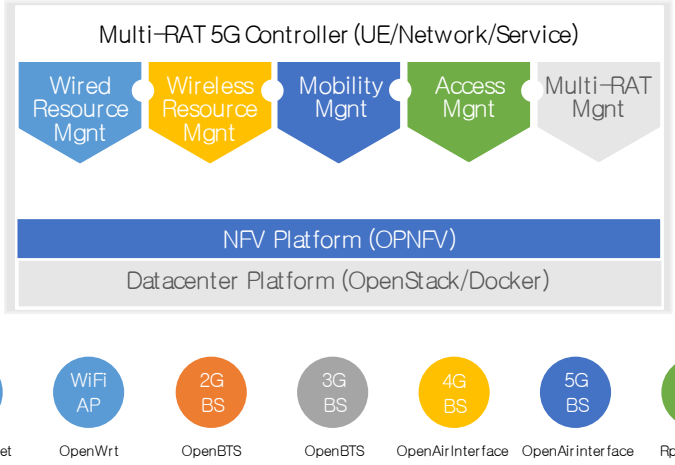

Figure 28 - Open Source Software for Testbed Implementation.

SDI based Muti-RAT 5G UE/Network/Service testbed contains Carrier-Grade Giga-Ethernet/WiFi/2G/3G/4G/5G network Controller SW. It operates as Access Agnostic Network Controller Software. SDI controller is developed based on ONOS / OPNFV global open source software. The testbed also supports Wire/Wireless/UE/Subscriber/Service management technology with independent access technology and network service developers of testers to freely develop and test network functions using an Open API (Open Application Programming Interface). It contributes to global opensource software and is aimed at open eco-system implementation.

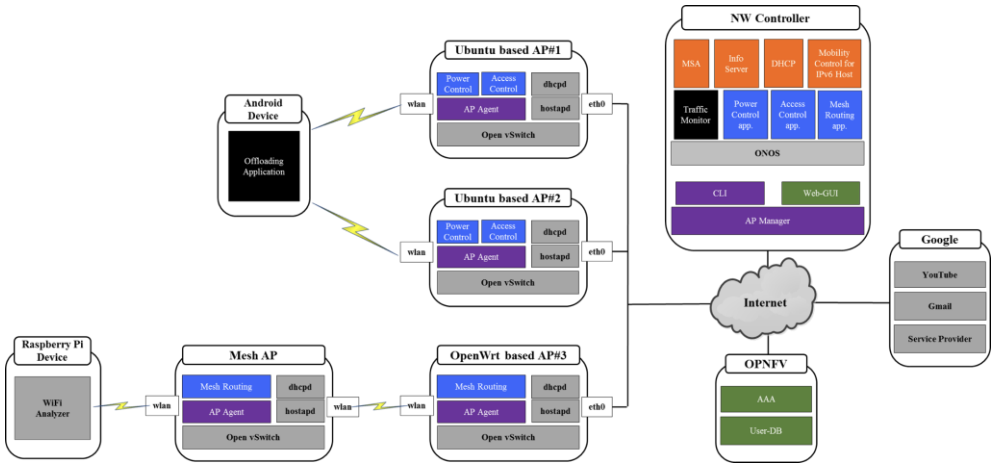

Figure 29. Open Source Softwrae based for Multi-RAT 5G Testbed Implementation.

Figure 29Figure 29. Open Source Softwrae based for Multi-RAT 5G Testbed Implementation. is Open Source Software based for Muti-RAT 5G testbed implementation. It enables Centralized Wired/Wireless Open Sourcing Networking. And it consists of design and implementation of Open Source Software based WiFi UE, Network & Service Testbed. Also 2G/3G/4G air interfaces are intergrated below.

3.2.2 Prototyping

K-Cluster

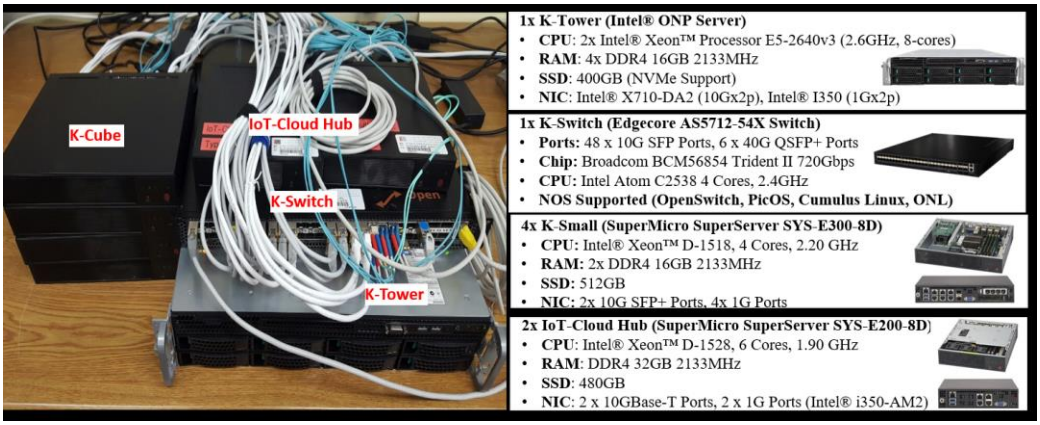

Figure 30 - Single K-Cluster prototype

Figure 30 is the prototype of realized K-Cluster, following the K-Cluster model. The elements consist of Convergence white resource boxes, to form an economical common development environment, which can correspond to SDN/NFV/Cloud-centric diverse SDI (Software-Defined Infrastructure) services. We deployed this prototype to GIST/Korea Univ./Soongsil Univ. and inter-connected via KOREN/KREONET and established K-ONE common development environment and we're operating it.

Fore-mentioned single K-Cluster can do the proof of concept limited to single-site-limited SDN/NFV/Cloud and cannot correspond to the ICT infrastructure's properties, Multi-Box,

Multi-Cluster, Multi-Site, Multi-Domain, all together called MultiX. Therefore, only by utilizing the multi-site common development environment with multiple K-Cluster with distributed deployment and linked together, we can proof the combined concept of MultiX-corresponding SDN/NFV/Cloud.

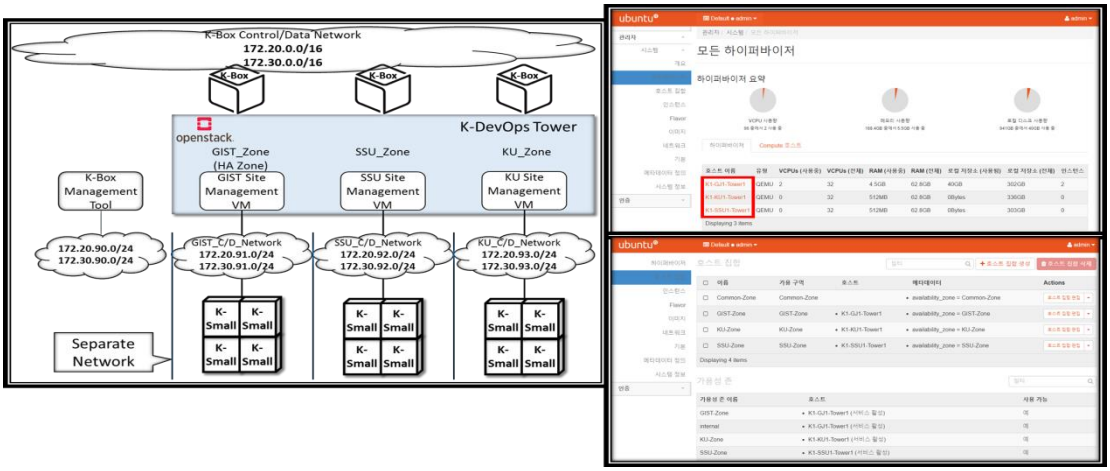

Figure 31 - Multi-site K-Cluster Deployment

- Distributed deployment of K-Cluster on GIST, SSU and KU and forming single OpenStack cloud via K-Tower deployed at each site
- Operating K-Cube in K-Clusters on each site appropriately with SDN/NFV experiment
- Policy-based configuration of VM network on remote sites is available
- Administrator can use Horizon dashboard of OpenStack cloud built on multi-site K-Tower base to create VMs and their distribution to the desired site of multi-site K-Tower
- Building diverse SDN/NFV proofing environment by employing K-Cluster, which is Multi-site SDI-based testbed

IoT-Cloud Hub

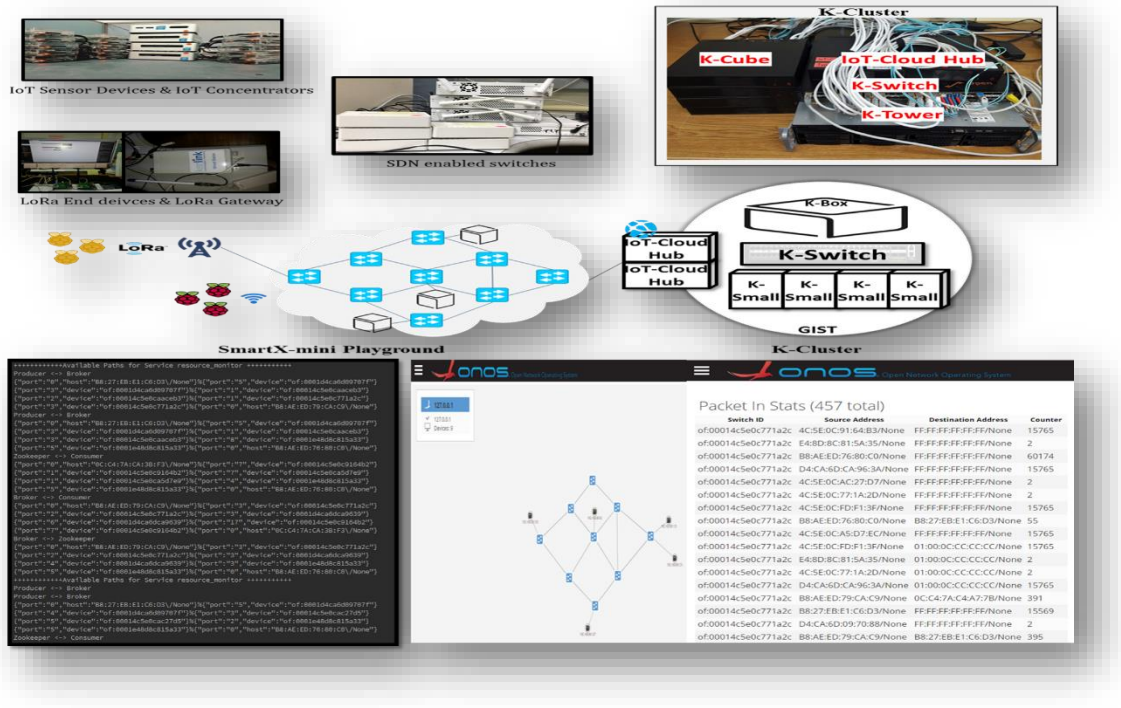

Figure 32 - Single-site K-cluster with IoT-Cloud Hub testbed.

To conducts overall control of the IoT and collecting data from the IoT device through IoT-Cloud Hub, we made IoT-SDN-Cloud Testbed like Figure 32. The K-Cluster on GIST is combined with IoT-Cloud Hub elements and will be used as an IoT-SDN-Cloud testbed. For IoT things with small establishments, Raspberry Pi 2 models will be used, and to configure the topology above, Mikrotik switches with OpenFlow 1.0 support will be used for the support of WiFi connection and SDN-enabled switch. For the support of LoRaWAN Network, the kerlink gateway will be connected to the SDN-enabled switch and network server components will be installed into IoT-Cloud Hub.

The ONOS SDN Controller on IoT-Cloud Hubs manages IoT things and SDN-enabled switches and also provides Flow Steering function for sensor data from IoT things. Flow Steering is controlling IoT service's data path based on service configuration file. Service Developer write the file and then, IoT-Cloud Hub parse the file and check network resources. After, checking resources, IoT-Cloud Hub make data path per each service using ONOS SDN intent framework. ONOS SDN Intent framework abstracts flow, so it is easy to make and install flow rules to switches. We used 3 intents (Point-to-Point, Multi-to-Single and Single-to-Multiple). IoT-Cloud Hub decides what intent have to be installed for data path.

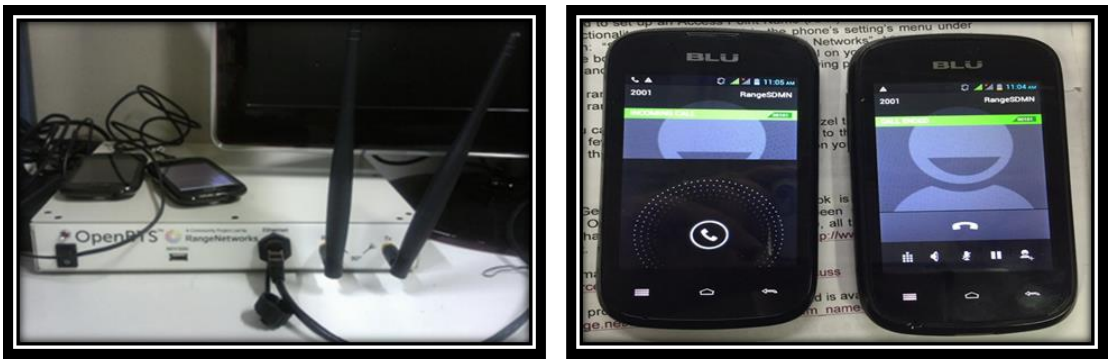

Figure 33 - 2G access network and voice communication test built on Open

OpenBTS-based 2G Access network construction technology

We tested to verify support of voice call and GPRS service of terminal after building 2G access network based on OpenBTS. We built and tested 2G access network with a single OpenBTS at the lab-level.

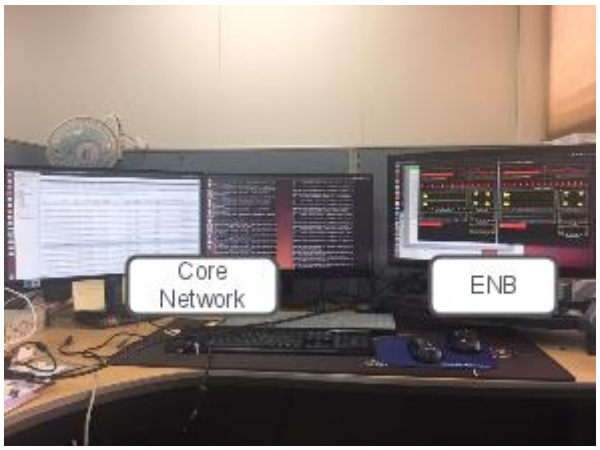

Figure 34 - 4G access network and data communication test build on OpenAirInterface.

OpenAirInterface-based 4G access network construction technology

We performed data communication test using OAI's Core Network, eNB and UE. We built and tested 4G access network with a single OpenAirInterface at the lab-level.

Figure 35 - eNB agent and controller operation.

Implementation of eNB agent, core network agent and controller

We developed ONOS controller and OpenAirInterface interworking technology to control eNB and UE. It consists of an Implementation of eNB configuration technology, user subscription and termination management technology.

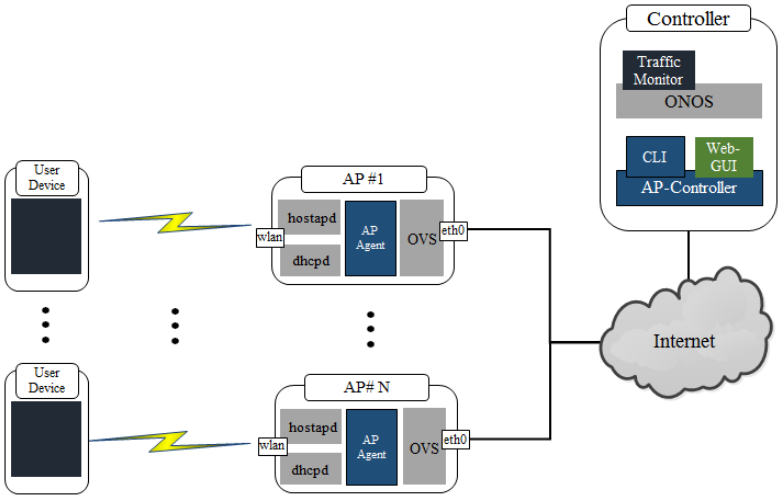

Figure 36 - Wifi AP Agent and Controller SW testbed.

Implementation of Wi-Fi AP agent and controller

We developed Hostapd based Wi-Fi AP construction technology. We Implemented the SW using YANG Modeling and RPC-based AP agent and controller based for AP control / management. The testbed consists of small wireless access network and controller using Raspberry-pi. The SW can perform QoS control, SSID management, authentication / access control, and SSL communication function between AP and controller.

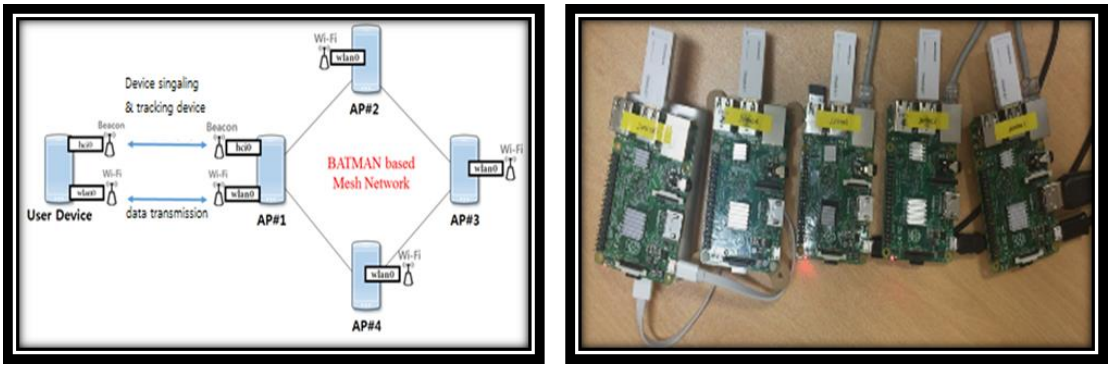

Figure 37 - BATMAN based mesh network testbed.

BATMAN-based mesh network construction and network structure visualization technology

We have acquired mesh / wireless relay network construction technology using open source BATMAN. We built and tested small mesh / wireless relay network using Raspberry. We developed mesh / wireless relay network structure visualization SW built on BATMAN.

3.3 Distributed Mobility Management

3.3.1 Design

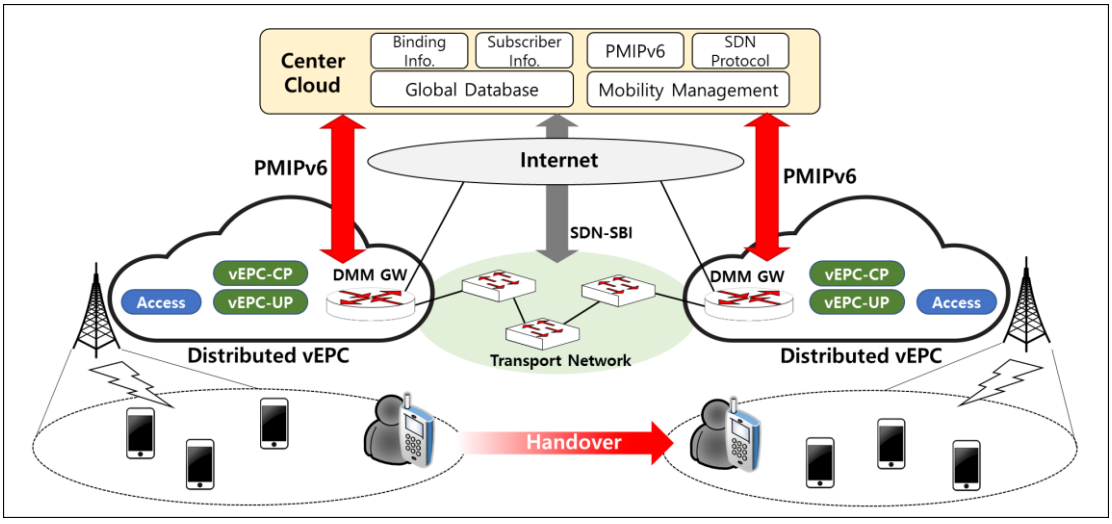

Figure 38 - PMIPv6/SDN based DMM Testbed Design

Distributed Mobility Management (DMM) standardized in the IETF has been focused on split traditional IP anchor function into control/data plane and distributed them to the edge of the network. The DMM can provide per-flow IP mobility management and on-demand mobility management. Our DMM testbed design is illustrated in Figure 38. The DMM testbed is divided three part. First part is distributed vEPC located in the edge cloud and contained all vEPC components. Access network functions are also located in distributed vEPC cloud. For supporting DMM, the DMM-GW is located in the distributed vEPC cloud and each GW has a role of distributed IP anchor for each cloud. It means that the DMM-GW owns a unique Ipv6 prefix pool and assigns IP prefix to each UE on link. The DMM-GW acts as a plain router for prefix when UE on-link and, when the UE moves to another place and attaches to the different DMM-GW, it exchanges PMIPv6-based signaling with the center cloud and uses PMIPv6-based tunneling to routed prefix when the UE moves. Between distributed vEPC clouds, there are SDN-based transport network for managing data packets and optimizing their path. The Center Cloud is the central control plane for overall network and managing mobility between distributed clouds. In the center cloud, there are several mobility control functions as follows;

- Global Database: manages binding information for all UEs in the network, manages Subscriber information (IMSI, MMC, MNC, …).
- Mobility Management: PMIPv6-based Signaling between DMM-GW, SDN-based path optimization for transport network.

Based on this architecture, we make mobility scenario for the UE as follows

- When the UE moves to the new distributed vEPC core,
  - The UE attach to the access network and the new vEPC core
  - When the attach request of UE is forwarded to the new DMM-GW, the new DMM GW send Proxy Binding Update(PBU) message to the Center Cloud
  - MANO receives PBU message, finds the previous DMM-GW by using binding information, and send PBU message to the previous DMM-GW
  - The previous DMM-GW updates its routing table for tunneling to the new DMM-GW, and sends Proxy Binding Ack (PBA) message to the Center Cloud
  - MANO updates binding information for the UE, and sends PBA message to the new DMM-GW
  - After receiving PBA message at the new DMM-GW, PMIPv6 tunnel is established between two DMM-GWs and traffic of UE is forwarded through the tunnel.
- For route optimization, SDN-based transport network can be used
  - At handover, the SDN controller in the Center Cloud can re-configure the transport network for optimizing traffic route by pushing rules to the SDN-enabled network devices
  - In this case, tunneling between two DMM-GWs is not required

3.3.2 Prototyping

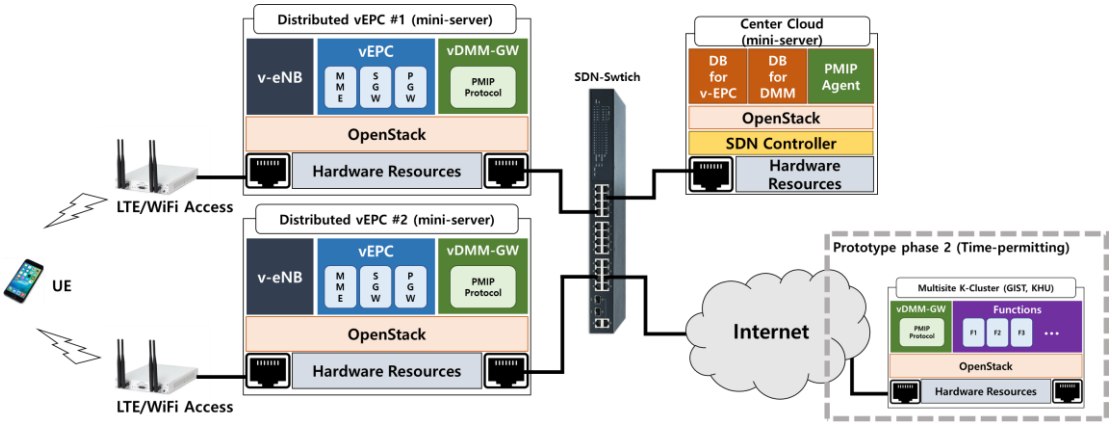

Figure 39 - DMM Prototype

To implement a real testbed, we make a prototype of DMM functional test by using three physical servers and one SDN-enabled switch as illustrated in Figure 39. All physical servers are using OpenStack and EPC SW components are running as several virtual machines. EPC SW components are implemented based on the 3GPP Rel.11 standards. For the DMM-GW, each distributed cloud has one VNF which perform PMIPv6 signaling and packet routing function. In addition, each distributed cloud has LTE/Wi-Fi access system to support attaching UE. LTE access is deployed using development kit. In the center cloud, there are two databases and one PMIPv6 functional component SW are running on each VMs. Database for mobile subscribers is implemented using the MariaDB and this database is connecting to the vEPC components via control plane interface. For supporting PMIPv6 based DMM, we customize open source and implement as a VNF in the center cloud and distributed clouds. Each virtualized DMM component in the distributed cloud is connecting with virtualized DMM component SW in the center cloud. For transport network management, SDN controller is deployed on the center cloud. The SDN controller manages the SDN-enabled switch that all distributed clouds and the internet is connected. For the prototype phase 2, we consider to deploy DMM-GW on the other academic site (GIST, KHU) and interconnect with our Center Cloud for supporting the DMM but it is time-permitting task.

4 Mobile core testbed integration

We described in deliverable D2.1 section 6.1 two main types of physical *interoperability architectures* between the European (EU) and Korean (KR) mobile core networks. These architectures align with the use cases 6 and 7 where two users, residing in different mobile cores, use a latency sensitive application in the former case and a broadband application in the latter. The first interoperability architecture involves fully remote Evolved Packet Cores (EPC) where both control plane and user plane are located in their respective home networks. The second architecture refers to an architecture where the KR site executes a full EPC and either only the user plane or the full stack of the EU EPC.

	Fully remote EPCs	**Co-located EPCs**
Client-server	Dedicated network and one of the EPC	No interworking
P2P	Dedicated network and both EPCs	The two EPC

Table 8 - Comparison of the physical architecture with the application architecture based on the interoperability partners

Next to the physical architecture, we can look at the interoperability from the applications' point of view. The applications in use case 6 and 7 is realizable with a client-server or a peer-to-peer *application architecture* and, depending on the implementation, the number of interoperating partners are different. In case of a client-server implementation, neither of the UEs are communicating directly with the other, but indirectly through an application server (see Figure 40). Consequently, the end-to-end connection consists of two independent parts and, depending on the location of this server, one part spans between an EPC and the dedicated interconnecting network, while the other part only an EPC. Therefore, the number of interworking partners is two for the first part of the connection and there is no need for interworking for the second part.

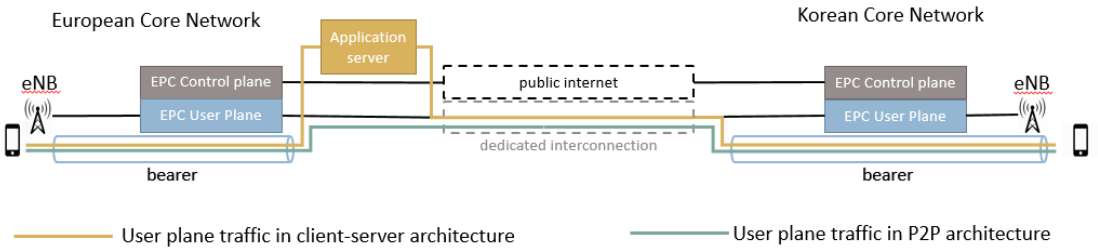

Figure 40 - User plane traffic in client-server and P2P application architecture in case of fully remote EPCs

In Figure 41, we show the co-located setup, where the application server is local for the two EPCs so they can decide about the parameters of the connections independently from any other system. In the other, peer-to-peer (P2P) case, all the 3 components must negotiate with each other the connection's parameters if the EPCs are fully remote (Figure 40). However, only the two EPCs must interwork if the EPCs are co-located as we can see in Figure 41. We summarized the resulting different interoperability scenarios in Table 8.

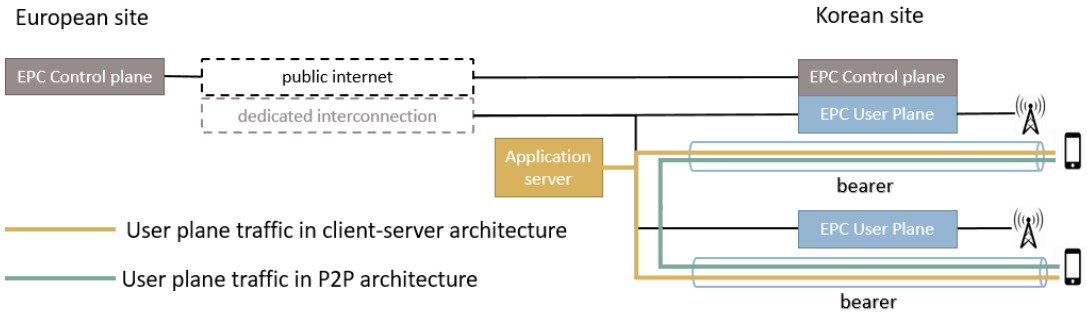

Figure 41 - User plane traffic in client-server and P2P application architecture in case of partly co-located EPCs

4.1 Connection setup within an Evolved Packet Core

In Figure 42 the main functional blocks of an EPC are depicted. The presented structure is aligned to our needs by exposing the Rx interface and allowing any authorized Application Function (AF) to initiate a connection setup in the EPC through the Diameter protocol. The AF can be any type of application that require dynamic policy or charging control over the behaviour of the user plane's network, like latency and bandwidth [8]. The most common use of the AF is the Proxy Call Session Control Function (P-CSCF), that is part of the operator's IP Multimedia Subsystem (IMS) and its main role is to setup multimedia sessions on request of an UE (e.g. voice call setup through the Session Initiation Protocol (SIP) [11]. In our case, however, the AF is the main entity that negotiates the connection parameters between the EU EPC, the KR EPC and the interconnecting network. This process might be service- or provisioning-driven. In the first case the entity triggering the process can be for example the multimedia session mechanism of the IMS system. The provisioning-driven process, which is the one followed by the project, is triggered by management components or by the Network Function Virtualization Orchestrator of the MANO framework.

The network-initiated bearer workflow relying on the AF is depicted in Figure 42. The process introduced as the network-requested secondary PDP context activation procedure in 3GPP Rel-7 is characterized by 8 steps:

1. In the first step, an application on the UE (e.g., IMS voice call) or an external management component is triggering the signaling which is intercepted by the AF.
2. Based on the external signaling information, the AF provides the Policy and Charging Rules Function (PCRF) of the EPC with service-related information over the Rx interface. This includes QoS information (e.g., bit rate, delay, etc.) as well as the traffic parameters (e.g, IP 5-tuple) that allow for identification of the IP flows corresponding to the service.
3. The PCRF may request subscription-related information from the Service Profile Repository (SPR).
4. Based on this session information and operator policies, the PCRF makes Policy and Charging Control (PCC) rules.
5. The PCC rules are sent by the PCRF to the Policy and Charging Enforcement Function (PCEF) to configure them in the P-GW-U (following the "on-path" model).
6. Optionally charging occurs for this enforced PCC rules, by contacting the Online Charging System (OCS)

7. The P-GW-U installs the rules and performs bearer binding, either by establishing a new bearer or by modifying an existing one, to ensure that the traffic for this service receives appropriate QoS.
8. The resulting service session can now be transported over the appropriate bearer by mapping the uplink IP flow to the adequate bearer.

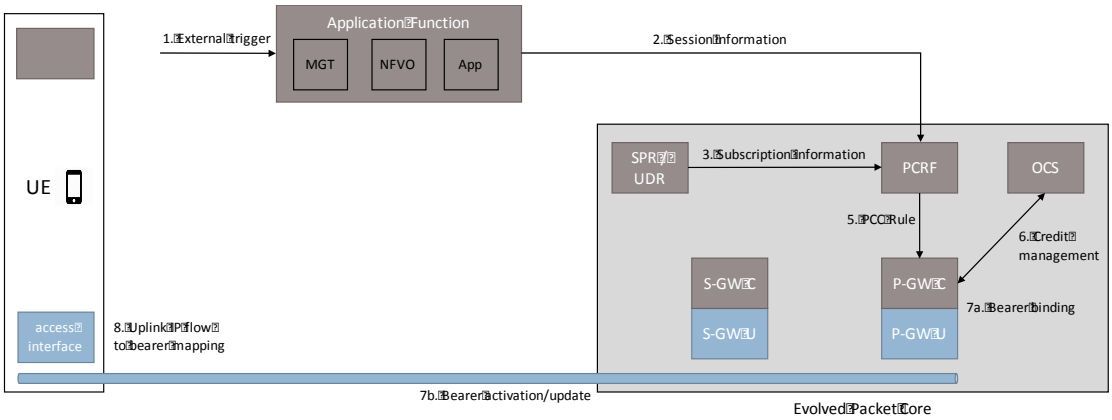

Figure 42 – Network-initiated QoS bearer procedure (3GPP Rel-7)

Note that this process can be static or dynamic. The latter enables to dynamically monitor a range of metrics, and depending on the outcome to re-provision the resulting resources in order to modify requested QoS. Dynamic provisioning functionality enables scale-out and migration functionality as described in Section 4.4. The same workflow could be followed in support of the coordinated setup of dedicated bearers required in support of use case 6 and 7. The resulting process is illustrated in Figure 43. The overall steps are similar as in the generic workflow, however, the AF will now instruct not only both EPCs (EU and KR) to update the associated bearers, but also the management or control system of the interconnecting WAN to trigger the configuration/signaling of the dedicated interconnection.

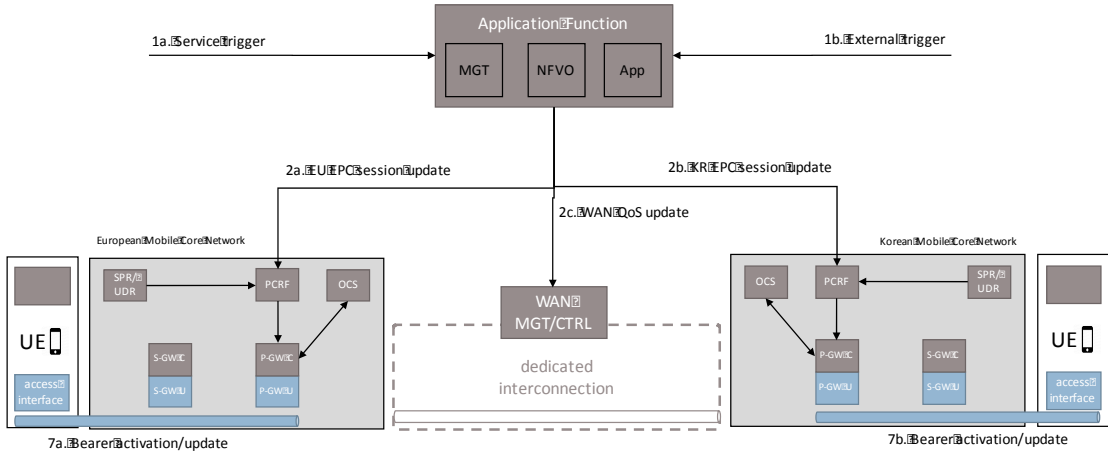

Figure 43 - AF-initiated QoS interoperability in 5G CHAMPION

In the client-server architecture, the AF is part of the application server. Whenever the UE connects to the server and initiate a new session, the server must connect to the PCRF through the Rx server and request the appropriate QoS changes for this new session. Evidently, the UE must be able to connect to the server in the first place, therefore IP connectivity must be in place between the UE and the application server. However, this is not the role of the application server, or in generally of the AF, but of the EPC during the initial attachment process of the UE. We consider this default IP connectivity as given and we suppose that the application server (the AF on Figure 40) is accessible from both the EU and the KR site. This is the reason for indicating the AF on the public Internet.

In the P2P architecture, a similar application server is responsible for the connection setup as before, but it does not take part in the user plane communication. The server's main role in that case is to negotiate QoS parameters between the interworking components and to handle the peer-to-peer sessions by providing the necessary information for the UEs to be able to connect each other. The latter functionality is very similar to the registration functionality of SIP [11]. Every subscriber has a public ID, like a user name, available at the application server and this public ID is bound to the current location of the subscriber's UE (e.g. public IP address). Whenever the subscriber likes to have a P2P connection, it uses the public ID of the other peer to query the actual IP address of that peer from the application server. The proper operation of this schema necessitates the regular update of the current location at the application server by the UE.

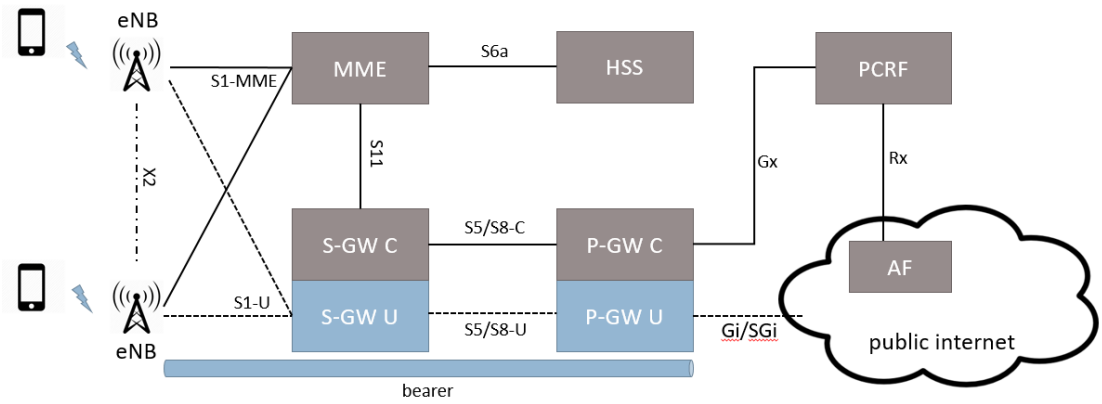

Figure 44 - Functional architecture of an LTE Evolved Packet Core (EPC) with the main interfaces.

4.1.1 Co-located EPCs with client-server application architecture

In this setup, there is no interworking between the systems. The AF is part of the application server and on the request of the UE it initiates a connection setup procedure through the PCRF by sending the bandwidth and latency requirements with a flow description containing the UE's IP. The QoS requirements are based on a priori measurements on the application performance, so there is no need to monitor the system for the initial connection setup. After providing the above information to the PCRF through the Rx interface [9], the PCRF contacts the Subscription Profile Repository (SPR) or uses its predefined configuration to make a Policy Control and Charging (PCC) decision [10]. Based on the outcome, it either contacts

the P-GW on the Gx interface to setup the connection or refuses the request of the AF, i.e. our application server. In case of rejection, the PCRF sends an appropriate message through the Rx interface and the AF can try to allocate a new connection with lower requirements or refuse all together the UE's request.

4.1.2 Fully remote EPCs with client-server application architecture

The symmetry of this setup makes it irrelevant where the application server is located, the only difference is the EPC that takes part in the QoS negotiation with the dedicated network. If the application server is part of the EU site, then the KR EPC has to align its QoS parameters with the dedicated network and the EU site hat to do the same if the server is at the KR site. In either case the same server, described above, should contain the AF block and a control block for the dedicated network in this scenario. The server is aware of its requirements for bandwidth and latency and tries to allocate the necessary resources in one of the EPCs and on the dedicated network. In case of bandwidth requirements (use case 7), the AF is either able to allocate in both systems the necessary capacity or it must refuse the UE's request. To fulfill the latency requirements (use case 6), however, the server is able to distribute the given latency budget between the EPC and the dedicated network, as it can satisfy the requested end-to-end delay by creating a connection with higher latency on the dedicated network on the expense of the EPC bearer or the other way around. The actual implementation should consider the physical limitation and choose the latency ratio between the two systems accordingly (e.g., the latency on the interconnecting network is limited by the physical distance between the EU and KR site). After the initial connection setup, monitoring is required for dynamic reconfiguration. By monitoring the real end-to-end delay on the connection, the server can detect any divergence from the requirements and should modify the bearer in the EPC or the MPLS tunnel on the interconnecting network.

4.1.3 Co-located EPCs with P2P application architecture

This scenario is similar to the above one, but instead of controlling the interconnecting network, the server acts like the AF for both EPCs. The outcome for use case 7 is again binary: the requested bandwidth is either available in both EPCs or not. For use case 6, the server chooses an available latency ratio and monitors the real end-to-end delay for any divergence just like in section 4.1.2. As we have no physical difference between the two sites, (e.g. physical distance as in section 4.1.2), we cannot use a priori latency ratio. However, we can rely on historical measurements of the available resources in the two systems. This again necessitate some kind of a monitoring procedure, but in that case not just for tuning the connection, but for the initial setup too.

4.1.4 Fully remote EPCs with P2P application architecture

In the last scenario, the server incorporates an AF for both EPCs and the interconnection controlling function. In use case 7, the required bandwidth must be available in the KR EPC, in the EU EPC and on the dedicated network or the server must refuse the request, while in use case 6 it should distribute the end-to-end delay between the three systems.

4.2 Quality of Service in the EPC and on the dedicated network

The topic of Quality of Service (QoS) mechanisms on Wide Area Networks (WAN) has a long history and two main most important approaches are resource dedication like the IntServ architecture [14] and packet marking like the DiffServ architecture [13]. IntServ is known to have better QoS guarantees, but all the network element must support a specific protocol to

work properly. On the other hand, the DiffServ architecture is more like a best effort method and does not rely on protocols other than the widely used IP or MPLS [15]. Beside these well-known procedures, there is still the promise of Software Define Networking (SDN) [12]. In theory, one can implement quite robust and efficient routing algorithms with good QoS guarantees, if the underlying programmable network (e.g. network of OpenFlow switches) supports some kind of queue management.

ARP) The Allocation and retention priority is a simple number indicating the importance of the given bearer. In congested situations, the EPS drops first the bearers with lower ARP if necessary. Moreover, the EPS refuses bearer creation requests with low ARP if the resources are scarce.
QCI) The QoS Class Identifier is also a positive integer that represents a maximum packet delay and packet loss rate. The meaning of this number is vendor specific a must be configured separately in every network element of the EPS.
GBR) The Guaranteed Bit Rate is the minimum capacity of the bearer stated in bit per seconds. It is an optional value and the bearer is a non-GBR one without it.

Table 9 QoS Parameters coupled with a bearer

Parallel to the evolution of the above mechanisms, the 3rd Generation Partnership Project (3GPP) developed a different concept of QoS in the context of Long-Term Evolution (LTE) mobile communication standards. These standards describe the Evolved Packet System (EPS) with the logical entity of EPS-bearer. Every subscriber in the system has at least one bearer and all the traffic in a bearer gets the same QoS treatment. Consequently, when the subscriber's traffic requires some special QoS treatment, then a new bearer must be created with the required parameters like latency or bandwidth. This concept of bearers is purely logical and realized with different technologies depending on the given network element in the EPS. In the Radio Access Network (RAN), for example, it means dedicated radio resources while in the EPC it is typically a GPRS Tunneling Protocol (GTP) tunnel. The actual steering into the tunnel or the mapping to the radio resource is the task of the UE in the uplink direction and the task of the PGW in the downlink direction. For this purpose, both the UE and the PGW use one or more Traffic Flow Template (TFT). Every template is a list of traffic flow filters and there is a template for every bearer. Naturally, this list, and so the TFT, can be modified in case of a request for a new service. For this update procedure, the Policy and Charging Control (PCC) is responsible. The main entity here is the Policy and Charging Rules Function (PCRF) that makes the PCC decisions about a service request and updates the TFT through the Gx interface (see Figure 42). As we mentioned, every packet gets the same QoS treatment in the same bearer and there is three QoS parameter for every bearer (see Table 9).

4.3 Monitoring

In section 4.1, we described the different interoperability scenarios and the necessary steps to initiate a new connection. We also noted that it is not enough to create a connection based on the available resources, but the monitoring of allocated resources is necessary. By getting real information about, for example the latency or the bandwidth, one can tune the QoS parameters to better align with the application session's requirements.

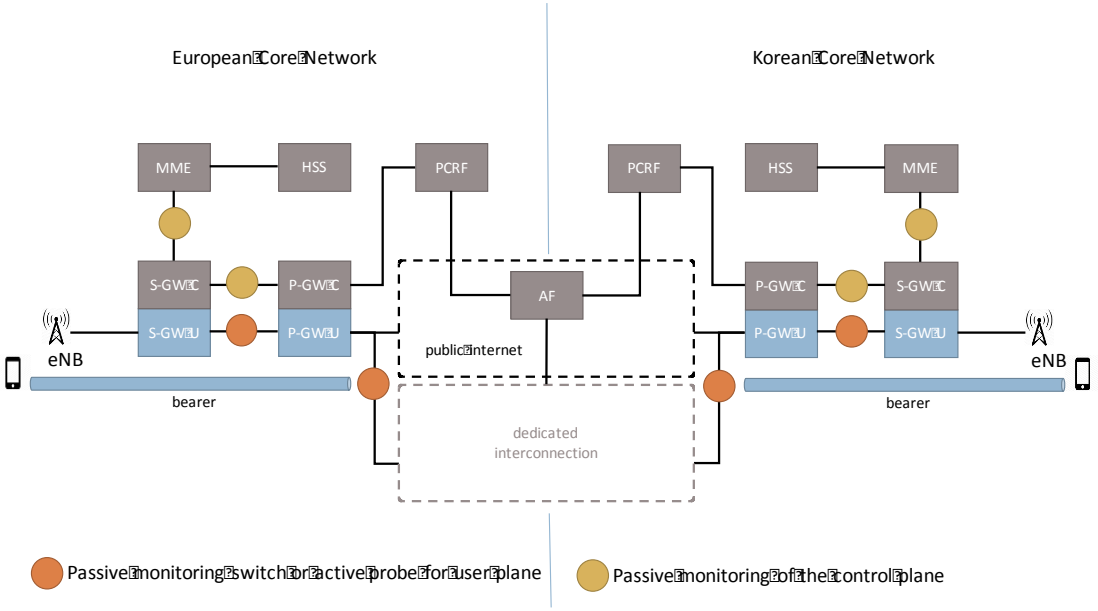

Figure 45 - Traffic monitoring location at the two interworking sites

One of the most trivial metrics to measure is the end-to-end delay and bandwidth on the newly created path. Depending on the chosen application architecture, one has to monitor the links between the UEs and the application server or between the two UEs. In case of client-server architecture, the application server can initiate active measurements or it can passively capture the behaviour of an underlying protocol like the TCP's windows size and RTT. We have the same possibilities with the P2P architecture, except that the UE executes the monitoring application. Such an application is the easiest solution, but it would put an unnecessary load on the UE. Moreover, we cannot infer the causes of any quality degradation by measuring end-to-end metrics. To overcome these difficulties, we can extend the EPCs with monitoring functions or use the existing ones if there are any. One can also install dedicated switches between the Serving and PDN gateways in both EPCs to monitor passively the application flows, see Figure 45. As the traffic is IP-based, we can use OpenFlow switches or even NetFlow [16] supporting ones. Moreover, accurate delay measurements require clock synchronization between the monitoring nodes. The same synchronization is necessary for active measurements, where, instead of switches, probes perform the monitoring. These probes have to be aware of the properties of the flows, in order to inject traffic into the bearers. Therefore, the application server must inform them about the newly created connection.

Besides the monitoring of the user plane traffic, one can capture the control traffic too (green marker in Figure 45). Observing the connection setup messages between the MME and eNodeB, we can derive the time required for the initial attachment or for a handover. The control messages between the gateways provide us with information about the duration of network initiated (i.e. on the request of AF) connection setup. Most likely, these setup times have no or little effect on the overall user experience, but they can inform us about potential slowdowns. In case of a burst in the number of users, for example, the application server should refuse some of the connection request, when it experiences increased setup times in one or both of the EPCs.

Not just the user and control plane traffic is a good indicator of the performance, but we can use also the CPU and memory consumption of the network elements. Monitoring the CPU usage of the MME, we can forecast system slowdowns, just like in the previous paragraph from the control traffic. The high resource usage of the gateways in one of the EPCs indicates failing QoS requirements and one can proactively redistribute the resources between the two EPCs.

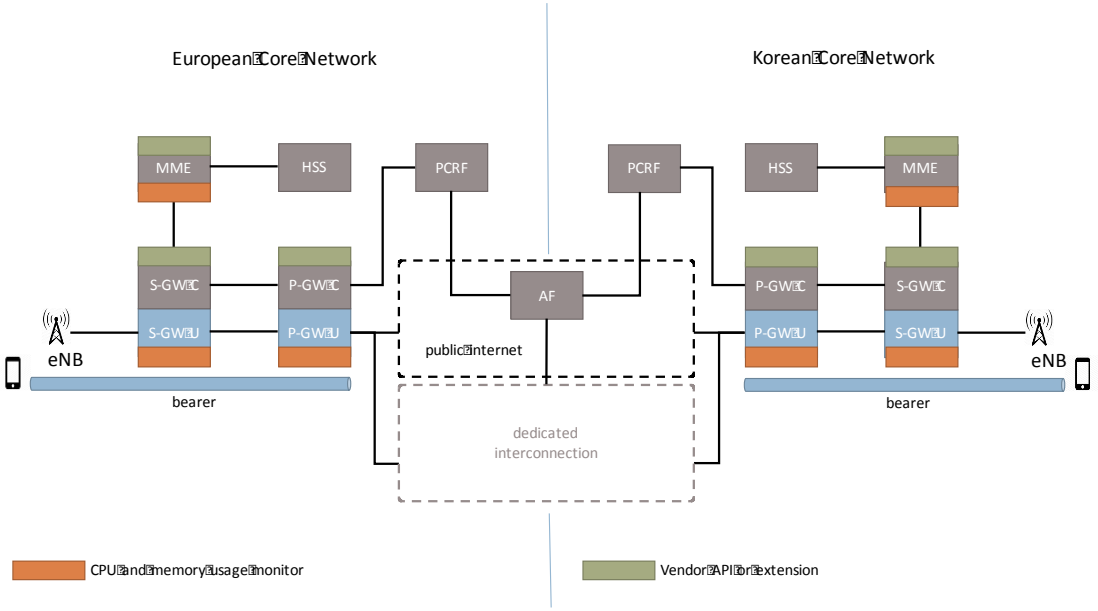

Figure 46 - CPU and memory monitoring, possible vendor API exposures

The monitoring procedures presented so far handle the functional blocks of the EPCs as a black box software. They do not require any domain-specific knowledge about the mobile core inner working, nor do they use any API possible provided by the vendor of the systems, see Figure 46. Such an API could give us information about the number of active connection, the number of bearers or even indicate if some of the QoS requirements are failing. Information from the MME could reveal the physical location of the UE, or at least the cell to which it connects, which can help us to determine the initial latency ratio (see section 4.1.2).

4.4 Dynamic interoperability provisioning

The key benefit of an SDN/NFV-enabled mobile core architecture, is its ability to dynamically adapt required resources to the changing context and environment. As documented in D2.1, in a (partly) co-located scenario, the Network Function Virtualization Infrastructure (NFVI) on which EU and KR vEPCs are deployed are (partly) under the control of the same Network Function Virtualization Orchestrator (NFVO). Figure 47 illustrates the resulting network architecture, where the common NFVO oversees one or more Points of Presence (PoP), each managed by their own Virtual Infrastructure Manager (VIM), as well as the interconnecting WAN network managed by its WAN Infrastructure Manager (WIM).

The static scenario involves that the NFVO receives a network service request to deploy the EU and KR vEPC on their respective PoPs, as well as their interconnection via the WAN network (in the figure they are deployed on 2 separate PoPs, however, certain scenario's might require them to be deployed - partly - on the same PoP). As a result, the NFVO will

instruct the VIMs to instantiate the required Network Function instances, as well as the WIM, to set up the interconnection.

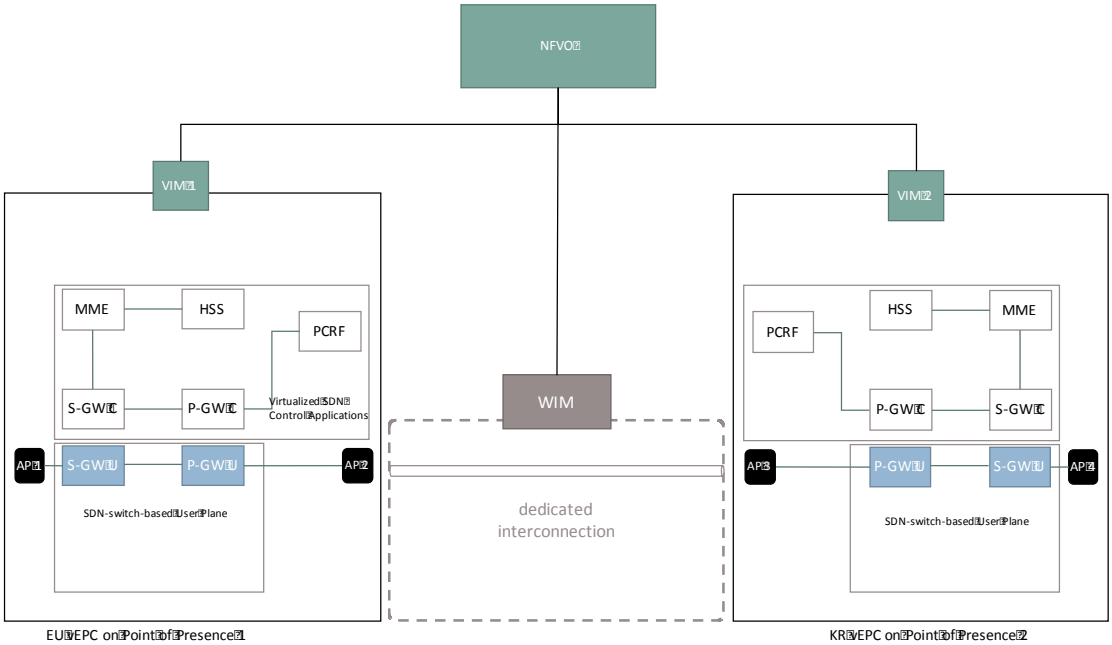

Figure 47 - SDN/NFV-based interoperability architecture

A more advanced and dynamic scenario, which is building further on the connection setup workflow described in Section 4.1, involves the dynamic re-provisioning of the interconnected bearers, as well as of the underlying virtual network function resources to fulfil necessary QoS requirements. This scenario is depicted in Figure 48. In this scenario, the NFVO is not only used for static provisioning of the different parts of the mobile core interoperability setup, but as well as for the dynamic re-provisioning of this network service based on monitoring components (cfr. previous section), as well as other external triggering systems (OSS/BSS, or services). These components might for example trigger the scale-out of the P-GW-U (1a), or the migration of of S-GW (1b) Virtual Network Functions. Note that monitoring components, per se are not necessarily directly interacting with the NFVO, but are usually relying on the interaction of the management functionality of associated VNF (VNFM) or services. As a result, the NFVO will (re-)instruct the corresponding VIMs and WIM(s) to instantiate new VNFs (indicated in dark-blue), and re-wire the associated network connectivity via the WIM(s).

Future work will refine this process, and determine which degree of dynamics will be implemented for the considered project scenario's and associated demonstrations.

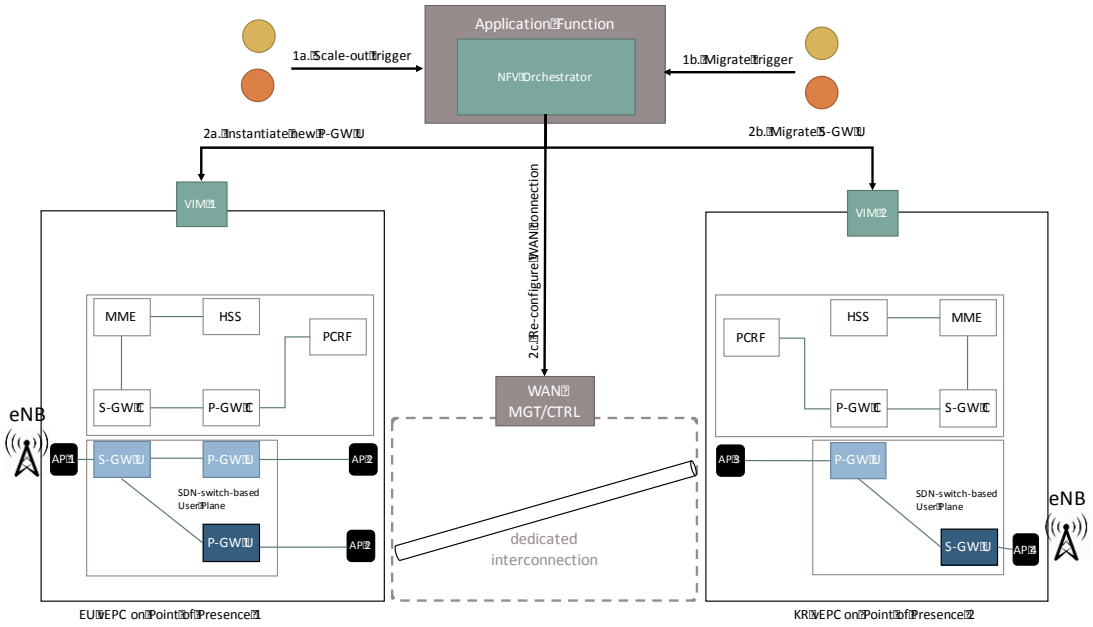

Figure 48 - Dynamic re-provisioning and NW-initiated bearer setup

4.5 Interoperability tests

As documented in D2.1, the interoperability architecture and setup between the EU and KR testbed follows the structure depicted in Figure 49.

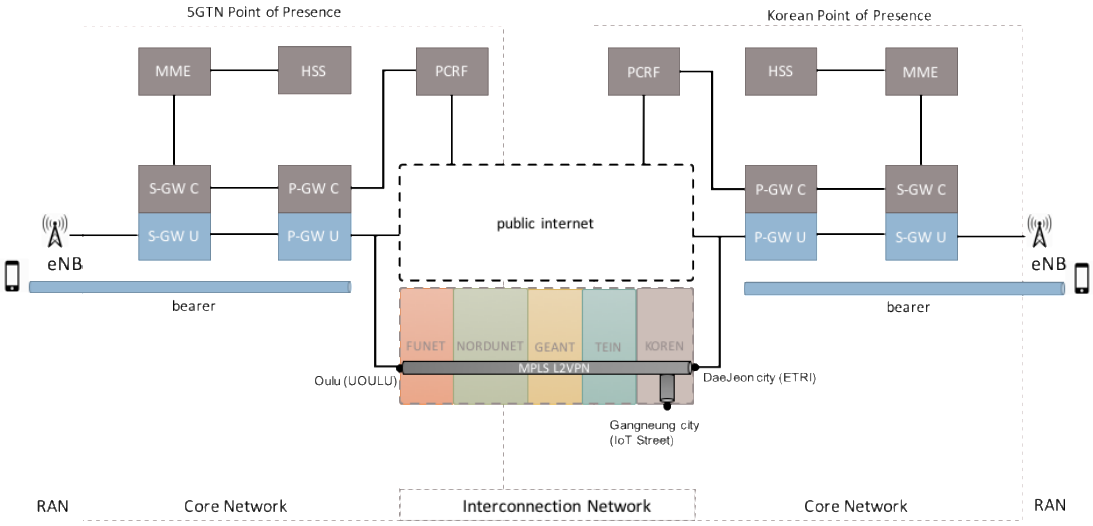

Figure 49 - EU-KR network interoperability architecture

This is refined in Figure 50 below, which shows the TEIN network topology for EU and KR core connectivity. For the reliability of interconnection, we are considering 1 primary and 2 back-up paths. The primary path is KR-Hong Kong- Singapore-London and two back-ups are KR-Hong Kong-Beijing-London and KR-Hong Kong-Singapore-US-EU. All three routes can provide up to 10Gbps.

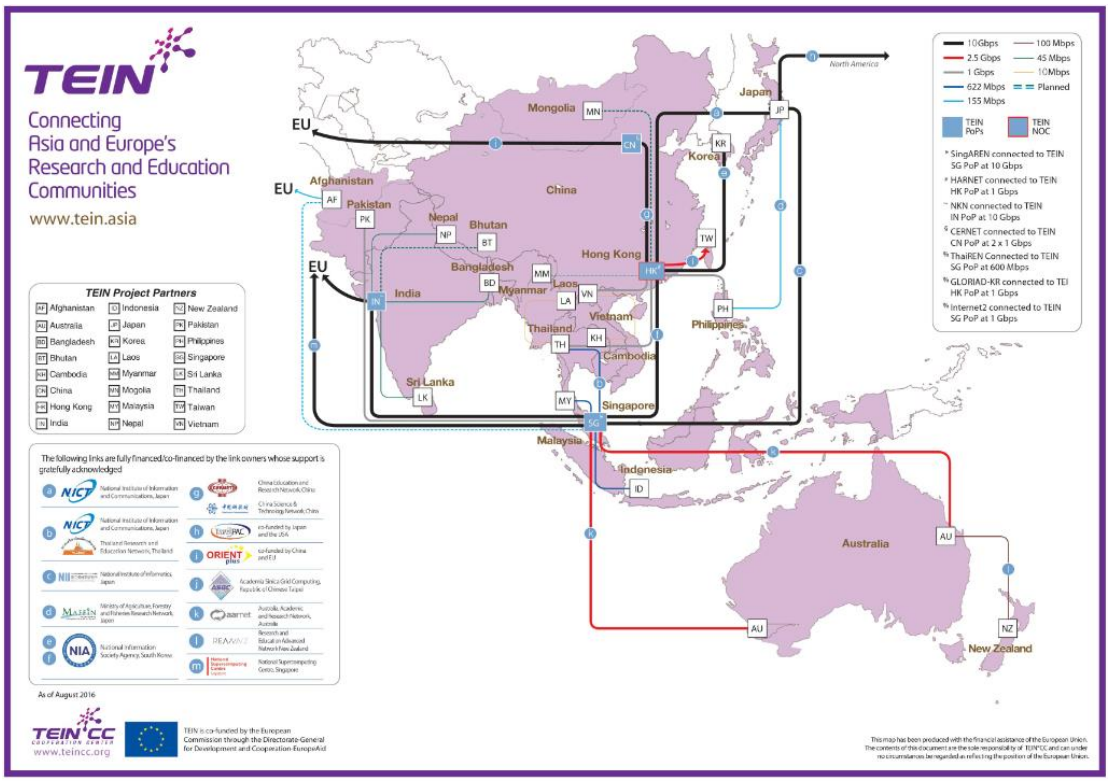

Figure 50 – TEIN network topology

5 Security architecture for the virtualized mobile core network

The security of the core network can be grouped into two, first, security of the core network elements and second, security of the communication channels in the core network. Since SDN and NFV will play a major role in the next generation mobile networks, it is necessary to first provide a basic overview of the SDN and NFV principles. SDN separates the network control from the data forwarding elements. The network control is centralized in high-end servers and programmable APIs are introduced in the forwarding elements to control them from the centralized servers. The centralized control platform has global visibility of all the underlying network forwarding elements and is capable of updating the forwarding elements through APIs whenever needed. The basic architecture of SDN is presented in Figure 51.

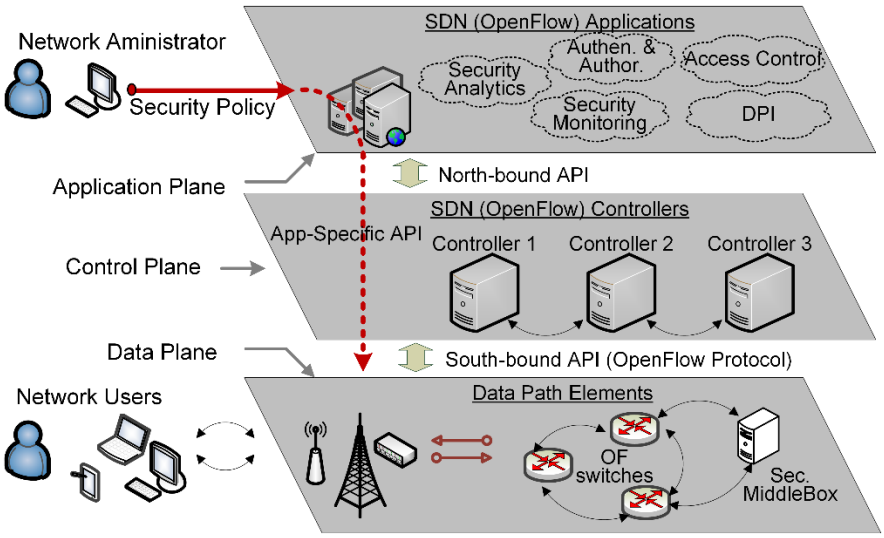

Figure 51 - SDN architecture, presenting security services and their deployment.

In SDNs, controlling the behavior and interworking of different heterogeneous networks is carried out with the logically centralized control architecture having a global view of all the forwarding elements. An operating system maps the entire network to services and applications that are implemented on top of the control plane. Hence, security services will be implemented as security applications using the network stats provided either proactively or reactively by the network control platform. The centralized control, which can be either logically or physically centralized, enables programmability of the network thus will provide fine-grained network security control, remote monitoring, and dynamic security service insertion. For these reasons SDN is considered to be highly important for innovation in network security. Therefore, SDN will play an important role in 5G network security.

5.1 SDN based Virtualized Core Network Architecture

In SDN-based mobile networks or Software Defined Mobile Networks (SDMNs), the legacy mobile network control functions, i.e., MME (Mobility Management Entity), HSS (Home Subscriber Server), PCRF (Policy and Charging Rules Function) and the control planes of S/P-

GW (Serving/Packet Gateway) run on the mobile network cloud as SDN applications and enforce the desired function by means of SDN technology. With this approach, the user plane is composed only by strategically located SDN capable switches and devices. This generic architecture is shown in Figure 52.

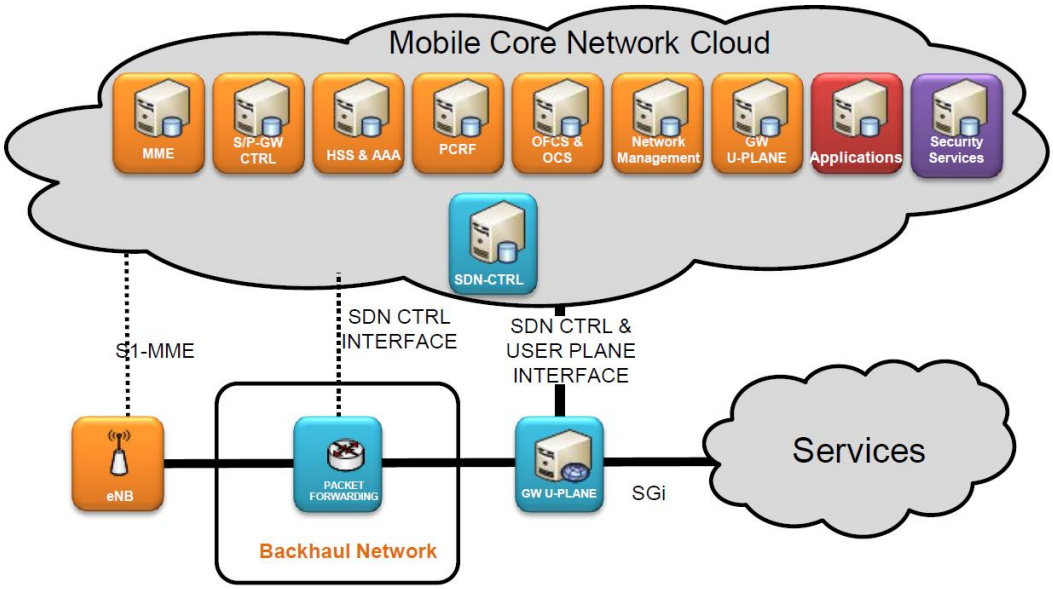

Figure 52 - Software Defined Mobile Networks architecture.

However, this SDMN architecture must be secured. SDN itself has several security limitations, hence, before using the SDN concepts we need to address the limitations of SDN and then secure the core and transport networks of SDMN. For example, centralizing the network control and softwarizing network function open new security challenges. Similarly, the centralized control will be a favorable choice of Denial of Service (DoS) attacks, and exposing the critical APIs to unintended software can render the whole network down. For brevity the main security challenges of SDN are presented in the following Table 10.

SDN Layer	Type of Threat	Threat Description
Application	Lack of authentication & authorization	There are no compelling mechanisms for authentication and authorization of applications, and is more threatening in case of large number of third party applications.
	Fraudulent rules insertion	Malicious applications can generate false flow rules.

	Lack of access control & accountability	A problem for the management plane & for illegal usage of network resources.
Control	DoS, DDoS attack	Due to visible nature of the control plane.
	Unauthorized controller access	No compelling mechanisms to obligate access control for applications.
	Scalability or availability	Centralizing intelligence in one entity will most likely have scalability and availability challenges.
Data Plane	Fraudulent flow rules	Data plane is dumb and hence more susceptible to fraudulent flow rules.
	Flooding attacks	Flow tables of OpenFlow switches can store a finite or limited number of flow rules.
	Controller hijacking or compromise	Data Plane is dependent on the control plane making its security dependent on controller security.
Ctrl-Data Int.	TCP-Level attacks	TLS is vulnerable to TCP-level attacks
	Man-in-the middle attack	Optional use of TLS and complex configuration of TLS

Table 10 – Security challenges in SDN

5.2 Security of SDN based Virtualized Core Network Architecture

5.2.1 The basis of Security in SDN

The current version of SDN, i.e. the OpenFlow operates on traffic flows. A flow can be a number of packets with same characteristics e.g. same TCP connection, or packets with a particular MAC or IP address. Operating on flows has shown to be much more feasible in terms of control and granularity. The basic operation on flows is such that, OpenFlow has three main entities as explained for the concept of SDN. These are 1) OpenFlow applications: SDN application plane, 2) OpenFlow controllers: the SDN control plane and 3) OpenFlow Switches: the SDN data plane. The OpenFlow switches are dumb data path elements that forward packets between ports based on the instructions installed in its flow tables by the controller. The OpenFlow switch has three basic elements. I) a flow table with actions associated with each flow, II) a secure channel to the controller using III) an OpenFlow protocol which provides an open and standard mechanism for the controller to communicate with the switch [17], [18].

When a new flow arrives, the switch checks its flow table for a matching entry. If there is no matching entry, the switch forwards it to the controller. The controller installs a matching entry in the switch flow table. Henceforth, when flows arrive at the switch, the switch checks its flow tables and acts accordingly. The flow tables have basically three types of actions for the packets. First, forward the flow to a given port as enlisted in the matching flow entry in the

table. Second, encapsulate and forward the flow to the controller. Third, drop the flow's packets. This makes security services rather simple in SDN and makes the basis of security in future technologies.

- ***Flow sampling:*** Flow sampling is the selection of packets or packet header fields through various algorithms for analysis. Selected samples can be sent to security applications or systems to analyze the content of the flow and verify security threats or vulnerabilities. Basic analysis targets can be the content of the flow packets or header fields, frequency of particular types of packets, and inter-arrival times of packets with different characteristics. In SDNs, flow sampling can be as easy as changing the output port numbers and counters in the flow tables of the switch. The destination on that port can be a security systems and the counter can show the number of packets to be sent to that destination.

In the following sections, we elaborate how the concepts of SDN can be used to provide robust security for mobile networks.

5.2.2 Data Link Security

The data link security is necessary to ensure that the data flows between the authorized end-points and is not diverted or intercepted while in transit. The previous generations i.e. 3G and 4G did not provide cryptographic integrity to user plane communication. In 5G, it will be a major security concern and will expose private communication not only between users but between devices carrying sensitive information such as data of health care systems and other critical infrastructures. Therefore, new mechanisms are needed to secure the data communication between users and devices. The OpenFlow protocol supports Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS). TLS is used to provide privacy and data integrity for the communication between users. DTLS is used to secure data between communicating applications, mainly UDP traffic. These technologies use symmetric cryptography for data encryption. The TLS protocol is composed of two layers, i.e. the TLS record protocol and the TLS handshake protocol. The Record Protocol guarantees connection privacy and reliability by means of data Encryption. The TLS Handshake protocol authenticates the communicating parties with each other and negotiate the encryption algorithm and cryptographic keys before transmitting the first packet of an application.

Besides the use of TLS and DTLS, virtual networking or network slicing can be used to provide private communication channels for both data and control information as shown in Figure 53.

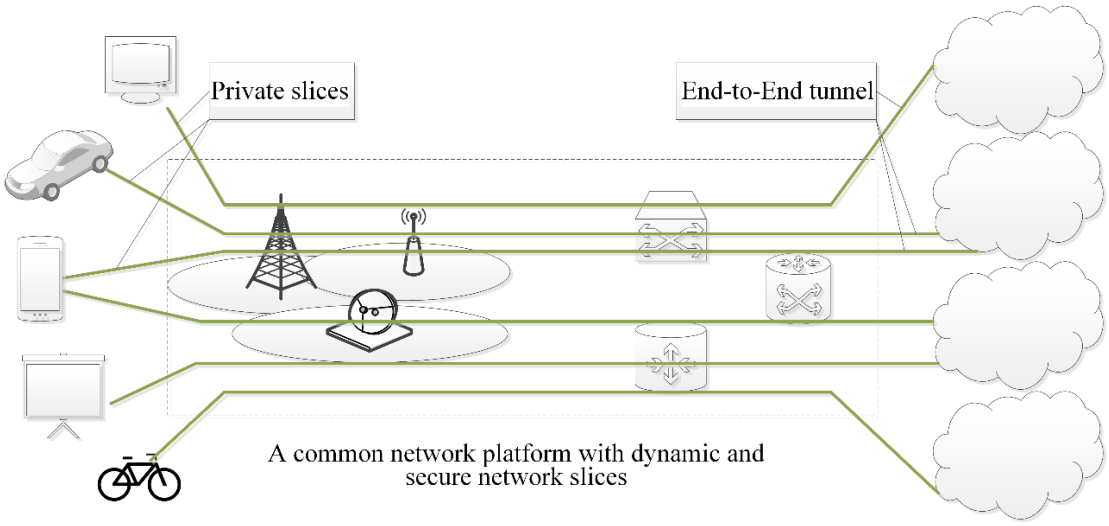

Figure 53 – Secure network slices, data and control channels.

Slicing can also provide isolation-based data integrity and privacy. Slices of individual users can be separated by a networking hypervisor such as the FlowVisor [19]. Traffic isolation can be used to protect one traffic from another or strengthen the confidentiality and integrity of user traffic [20]. Hence, the Open vSwitch platform provides isolation in multi-tenant environments and during mobility across multiple subnets [21]. The OpenFlow Random Host Mutation (OF-RHM) [22] technique is proposed to avoid scanning attacks on end-hosts. Using the moving target defense (MTD) technique, the OF-RHM mutates IP addresses of end-hosts to avoid scanning attacks.

5.2.3 Control Channels Security

Control channels carry the important control information between user and network, and among network entities. To authenticate a user, mutual authentication and key agreement is performed between the user and the network. In LTE, the UE and the network or its entities such as the Mobility Management Entity (MME) perform mutual authentication through the Evolved Packet System (EPS) Authentication and Key Agreement (AKA) known as the EPS AKA. The EPS AKA is secure enough and has no visible vulnerabilities demonstrated so far [23]. When a UE connects to the EPC through non-3GPP access network, the UE is authenticated through the AAA server. For trusted non-3GGP access networks, the UE and AAA server use Extensible Authentication Protocol-AKA (EAP-AKA) or improved EAP-AKA for authentication. For un-trusted non-3GPP access networks, the UE use the evolved packet data gateway (ePDG) IPsec tunnel establishment to connect to the EPC [24]. Such control channels have the following benefits besides being secure;

- The messages are short compared to other authentication protocols
- It requires only one handshake between the UE and serving network, and between the serving and home networks

- The HSS is updated through the serving network, thus is capable to handle many request
- The symmetric-key-based protocol makes the computations required in the authentication center (part of the HSS), and in the USIM (Universal Subscriber Identity Module) very efficient compared to public-key-based mechanisms. However, the advantages of the use of public-key based authentication and key agreement schemes could include that the home network does not need to be contacted for each authentication.

It is expected that in 5G there will be multiple control elements in a network that will require security of the control channels among those control points. For example, the concepts of SDN will be used for the benefits described in the previous sections. In the case of SDNs, multiple controllers will be used for higher availability and scalability. Therefore the control channels among the controllers must also be secured. Similarly, the control channel between SDN controller and SDN switches must also be secured. OpenFlow variant of SDN uses TLS in which identification certificates are properly checked in either direction and allows encrypting the control channel in order to secure it and prevent it from eavesdropping. Furthermore, multiple control channels (associations) between switches and controllers are suggested to avoid the chances of services outages due to connection failures. The latest OpenFlow specifications support multiple connections between switches and controllers to improve network resilience in case of link failures. Therefore, fast link restoration mechanisms, and backup entries with different priorities in the OpenFlow switches have been proposed and demonstrated in [25]. The backup links are computed by the controller and the traffic is switched to the backup link upon failure of the existing link. Similarly, flow entry migration techniques are proposed in [26] to reinstate a flow within 36ms. This mechanism fulfills the carrier grade recovery requirement of 50ms. Furthermore, HIP [27] based secure control channels between the switches and the controllers are also [28].

Moreover, IPsec is the most commonly used security protocol to secure the communication channels in current telecommunication networks such as 4G-LTE [29]. Thus, novel IPsec based communication architecture were designed to secure control and data channels of 5G SDMNs [29], [32]. Proposed architecture use distributed Security Gateways (SecGWs) to secure the controller and IPsec Encapsulating Security Payload (ESP) Bounded-End-to-End-Tunnel (BEET) mode tunnels to secure the control and data channels communication. Moreover, Identity-Based Cryptography (IBC) protocol based security mechanism is also proposed to secure the inter-controller and control channel traffic in a general multi-controller SDN networks [33].

5.2.4 Security of the control plane

The SDMN controller will provide the necessary services to the core network functions by working as intermediary in the architecture shown in Figure 52. The network control functions of the core elements e.g. MME, S/P-GWs etc will reside in a centralized cloud in the form of SDN applications that will leverage the NFV technology to be instantiated in different hardware or even different network perimeters for higher scalability and availability. Hence the main security concern in such architectures will be the SDN controller since the SDN controller can become a potential bottleneck for the overall network. Therefore, there are many proposals and approaches for securing the control plane. The Security-Enhanced (SE) Floodlight controller [34] is an extended and secure version of the original floodlight controller [35]. Securing the SDN control layer, the SE-Floodlight controller provides mechanisms for privilege separation by adding a secure programmable north-bound API to the controller and

operates as a mediator between the application and data planes. It verifies flow rules generated by applications and attempts to resolve flow rules conflicts between applications.

To mitigate the risks of controller failure due to scalability, or the chances of DoS attacks due to its centralized role, controller resilience strategies have been proposed. The strategies include controller resilience through redundancy, maximizing its storage and processing capabilities, and distributing controller functionalities among multiple control points in the network. The OpenFlow variant of SDN supports wildcard rules so that the controller sends an aggregate of client requests to server replicas. By default, microflow requests are handled by the controller that can create potential scalability challenges, increase the chances of failures due to DoS attacks. Normally reactive controllers are used that act on a flow request when it arrives at the controller. Proactive controllers would install the flow rules in advance, thus, minimizing the flow request queue in the controller. Similarly, various load balancing techniques are suggested that would balance the load among multiple controllers in a network.

5.2.5 Key performance indicators

In present mobile networks, IPsec tunneling and security gateways are widely used to secure backhaul communication. We have worked on a novel communication architecture based on Host Identity Protocol (HIP) to secure both control and data channels in SDN-based mobile networks (SDMNs). We aimed at analyzing the added security features as well as the performance penalty on both control and data channels inherent to the proposed architecture (shown in Figure 54). These performance penalties are considered in terms of throughput, jitter and latency. The key performance indicators in our performance analysis are :

- The performance penalty of security on TCP Throughput
- The performance penalty of security on UDP Throughput
- The latency introduced
- The performance penalty of security on Jitter

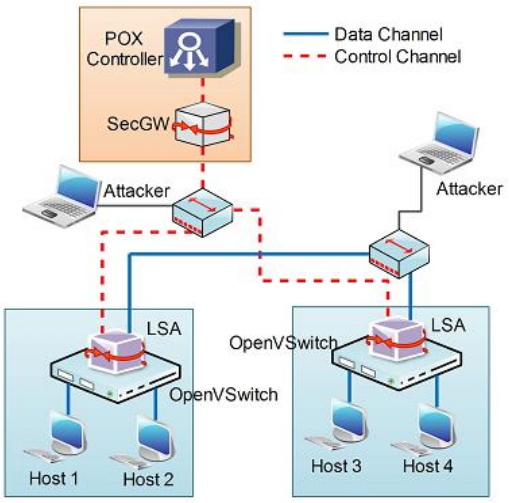

Figure 54 – Testbed Testbed for IPsec tunneling architecture for SDMN communication Channels

5.2.5.1 Performance Analysis of Control Channel

In the first set of experiments, we analyse the performance penalty of security on SDMN control channel due to the proposed architecture.

5.2.5.1.1 Connection Establishment Delay

In the first experiment, we measure the connection establishment delay between OVS1 and the POX controller under different scenarios. Here, we try to send a ping request from Host1 to Host2 and measure the connection establishment delay.

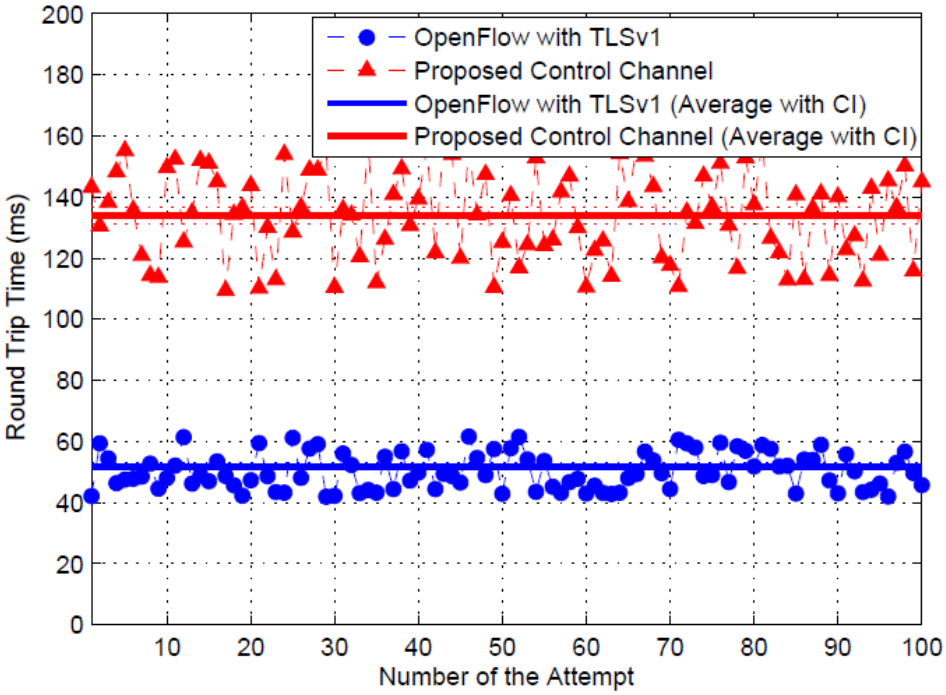

Figure 55 – The connection establishment delay

Experiment results (Figure 55) reveal that the proposed secure architecture significantly increases (136\%) the tunnel establishment delay. HIP tunnel establishment between LSA and SecGW adds extra delay to the tunnel establishment. However, the impact of this delay can be minimized by keeping the established HIP tunnels for a long period. It is possible to maintain established HIP tunnels for long periods (i.e. 15 mins).

5.2.5.1.2 Flow Table *Update Delay*

In the second experiment, we measure the delay to update flow tables for new packet flow in the steady state of operation. In the steady state of operation, HIP tunnels between LSAs and SecGW are already established and operational. Here, we ping from Host1 to Host2 and measure Round Trip Time (RTT).

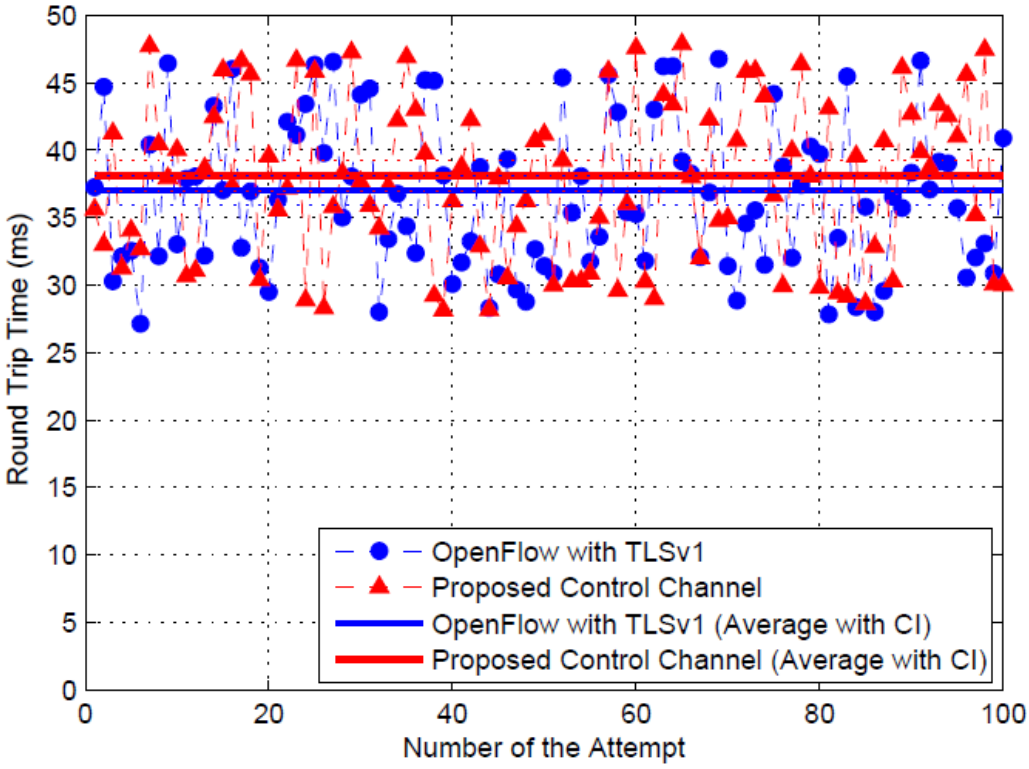

Figure 56 – Flow Table Update Delay

Experiment results (Figure 56**Erreur ! Source du renvoi introuvable.**) reveals that the performance penalty of the proposed secure architecture is less significant in the steady state of operation. The extra IPsec encryption increases the flow update delay only by 2%. However, this delay can be further minimized by using IPsec accelerators. IPsec acceleration is possible by using external accelerators and/or using new AES (Advanced Encryption Standard) instruction sets for processors.

5.2.5.2 Performance Analysis of Data Channel

In the second set of experiments, we measure the TCP and UDP throughput performance of data channel under different scenarios.

5.2.5.2.1 Impact on TCP Throughput

In third experiment, we establish a TCP connection between Host1 and Host3 to measure TCP throughput performance of data channel by using IPERF tool.

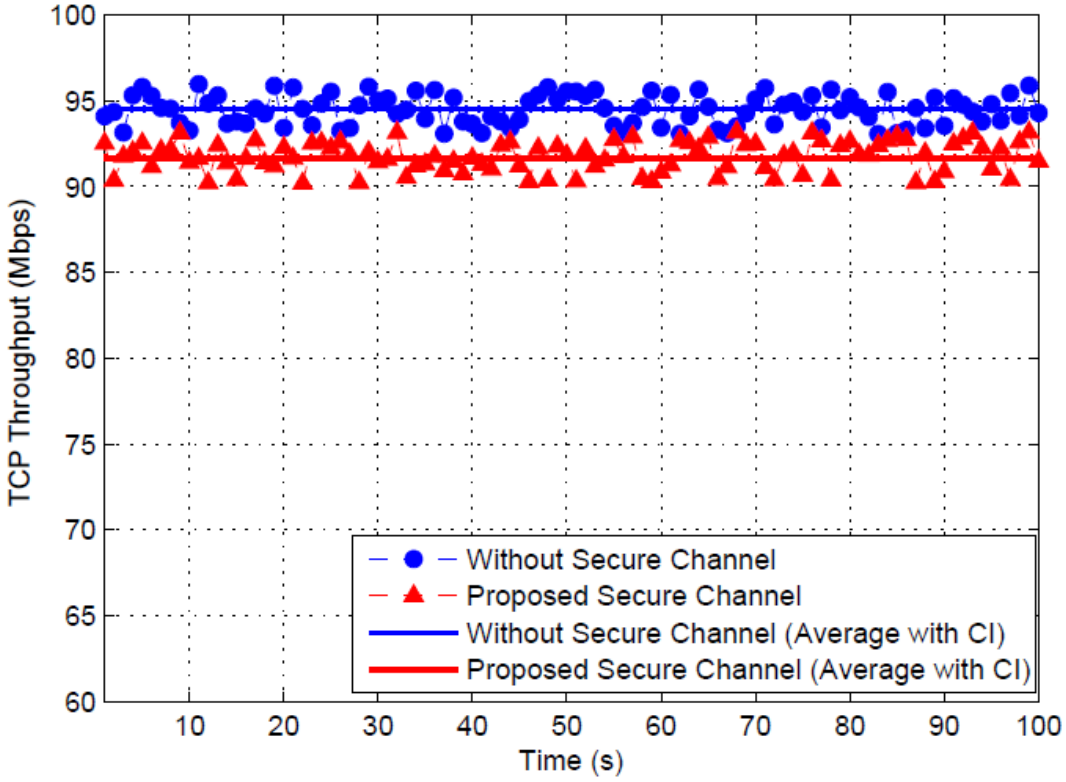

Figure 57 – Performance penalty on TCP thrroughput

Experiment results (Figure 57) reveal that the proposed secure architecture decreases TCP throughput only by 2.3% than the non-secure data channel. The extra layer of encryption decreases TCP Throughput.

5.2.5.2.2 Impact on UDP Throughput

In fourth experiment, we establish a UDP connection between Host1 and Host3, to measure UDP throughput performance of data channel.

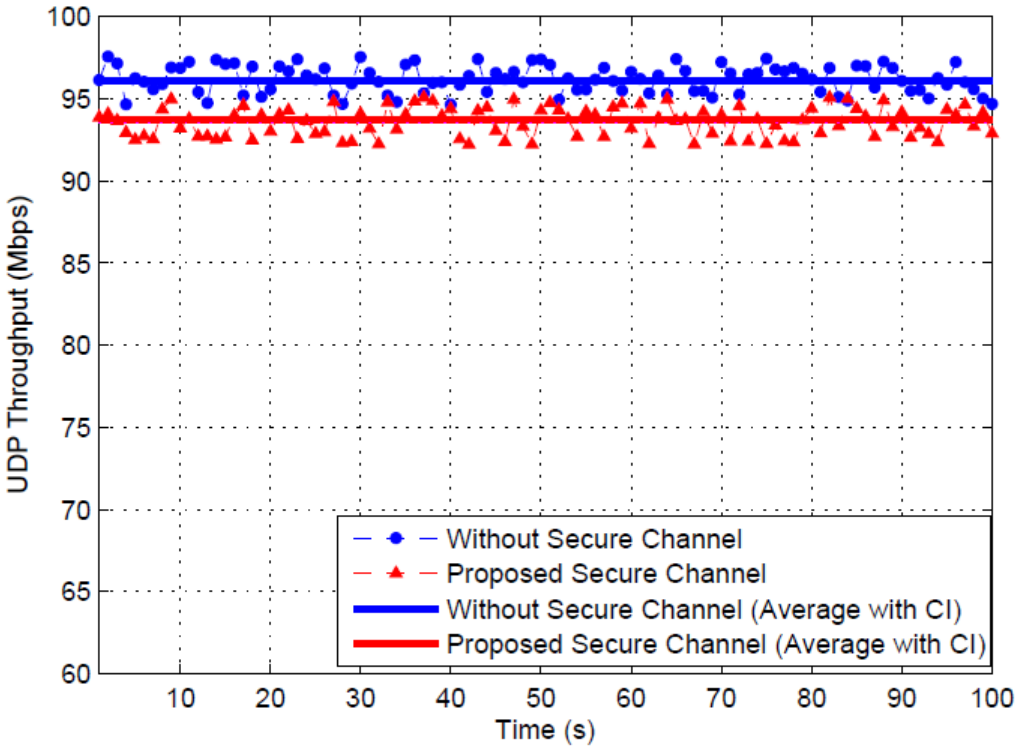

Figure 58 – Performance penalty on UDP throughput

Experiment results (Figure 58) reveal that the proposed secure architecture decreases UDP throughput only by 2.2% than the non-secure data channel. The extra layer of encryption decreases UDP Throughput.

Moreover, the performance penalty of security on throughput is around 2% for both UDP and TCP sessions in compared with non-secure scenario. Thus, we can conclude that the performance penalty of security on throughput is independent of the transport layer protocol.

5.2.5.3 Impact on JItter

In fifth experiment, the jitter performance of a UDP session between Host1 and Host3 is measured by using the IPERF tool.

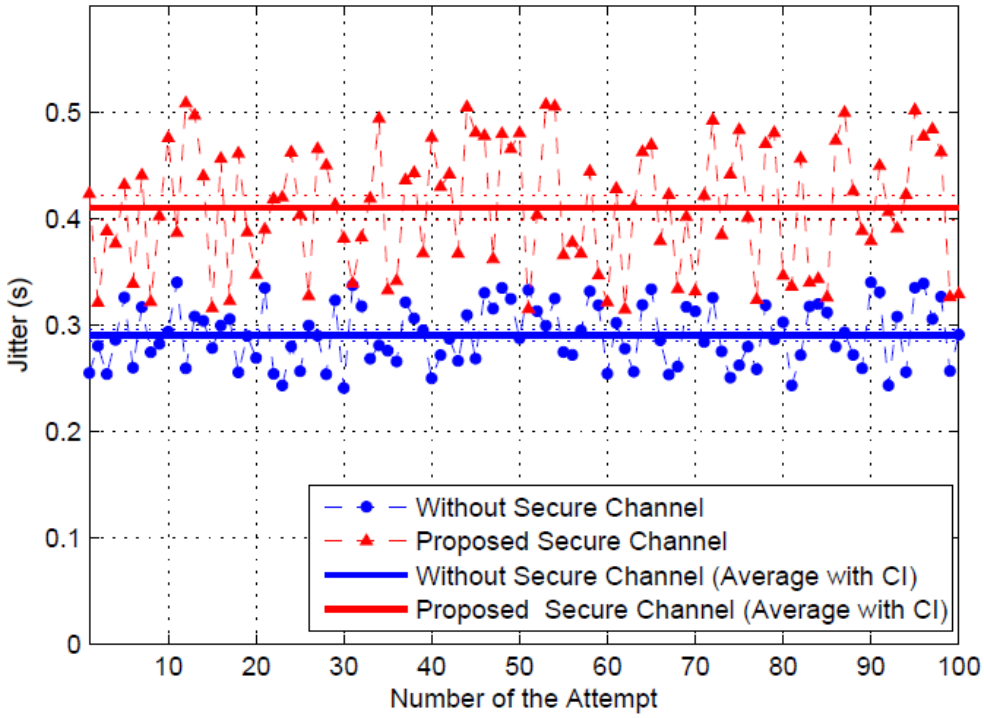

Figure 59 – Performance penalty on Jitter

Experiment results (Figure 59) reveal that the performance penalty of secured architecture is 41% compared with the non-secure data channel. However, the jitter is still well below 500 µs (VoIP required a jitter below 4 ms) and the impact of jitter for real-time application such as VoIP, video streaming is less significant in a short range network.

5.2.5.4 Results

The expected results from the validation are focused on resolving the following issues:

- Mitigate the IP based attacks on control channel
- Mitigate the IP based attacks on data channel
- Better utilization of IPsec tunnels in SDMNs
- Improve the visibility of IPsec based communication channels
- Enable the centralized controlling feature for IPsec based communication channels
- Design of collaborative network security and traffic forwarding policies with fair tradeoff between network security mechanisms, network latency and overhead.

6 Conclusion

This deliverable provides the outcomes of the first phase of WP4 work in 5G CHAMPION. The focus of this work has been on the setup of the individual EU and KR testbeds using hardware and cloud stack (control) infrastructure to create scalable and elastic resource pools for the vEPC enabling high-availability and low latency. Section 2 and 3 focus on the deployment work that has been done in order set up the individual 5GTN and KR testbeds, while Section 4 provides first guidelines on integrating and both EPCs, and testing the physical network interoperability of the testbeds.

Attention has been paid to the security architecture for virtualized mobile backhaul network, which has been documented in Section 5, including a set of KPIs and validation ideas.

Future work in WP4 consists of the further implementation of involved EPC network functions, as well as monitoring- and MANO functionality enabling the validation and demonstration of the interoperability scenario's defined in WP2. Implementation outcomes will be documented in D4.2, and the outcome of the interoperability scenario's will be reported in D4.3.

References

[1] Gudipati, Aditya, et al. "SoftRAN: Software defined radio access network." Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking. ACM, 2013.
[2] Jin, Xin, et al. "Cellsdn: Software-defined cellular core networks." Open Networking Summit SDN Event (2013).
[3] Jin, Xin, et al. "SoftCell: Taking control of cellular core networks." arXiv preprint arXiv:1305.3568 (2013).
[4] Li, Li Erran, Z. Morley Mao, and Jennifer Rexford. "Toward software-defined cellular networks." Software Defined Networking (EWSDN), 2012 European Workshop on. IEEE, 2012.
[5] Costa-Requena, Jose, et al. "SDN and NFV integration in generalized mobile network architecture." Networks and Communications (EuCNC), 2015 European Conference on. IEEE, 2015.
[6] http://tools.ietf.org/html/draft-nadeau-sdn-problem-statement-00 (SDN Problem Statement)
[7] Madhusanka Liyanage, Andrei Gurtov, Mika Ylianttila, Software Defined Mobile Networks (SDMN) : Beyond LTE Network Architecture , Wiley, 2015.
[8] Policy and charging control architecture (3GPP TS 23.203 V 13.10.0, Dec 2016).
[9] Policy and charging control over Rx reference point (3GPP TS 29.214 V 13.0.0, March 2017)
[10] Policy and charging control signaling flows and Quality of Service (QoS) parameter mapping (3GPP TS 29.213 V 13.8.0, Dec 2016)
[11] IETF RFC 3261, SIP: Session Initiation Protocol, https://tools.ietf.org/html/rfc3261
[12] Karakus, M., & Durresi, A. (2017). Quality of Service (QoS) in Software Defined Networking (SDN): A survey. Journal of Network and Computer Applications, 80(May 2016), 200–218. https://doi.org/10.1016/j.jnca.2016.12.019
[13] IETF RFC 2475, An Architecture for Differentiated Services, https://tools.ietf.org/html/rfc2475
[14] IETF RFC 1633, Integrated Services in the Internet Architecture: an Overview, https://tools.ietf.org/html/rfc1633
[15] IETF RFC 5462, Multiprotocol Label Switching (MPLS) Label Stack Entry: "EXP" Field Renamed to "Traffic Class" Field, https://tools.ietf.org/html/rfc5462
[16] Lee, M., Duffield, N., & Kompella, R. R. (2012). Opportunistic flow-level latency estimation using consistent netflow. IEEE/ACM Transactions on Networking, 20(1), 139-152. doi:10.1109/TNET.2011.2157975
[17] E. Al-Shaer and S. Al-Haj, "FlowChecker: Configuration analysis and verification of federated openflow infrastructures," in Proc. 3rd ACM Workshop SafeConfig, 2010, pp. 37– 44.
[18] McKeown, Nick, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. "OpenFlow: enabling innovation in campus networks." ACM SIGCOMM Computer Communication Review 38, no. 2 (2008): 69-74.
[19] R. Sherwood et al., "Flowvisor: A network virtualization layer," OpenFlow Switch Consortium, Tech. Rep. OPENFLOW-TR-2009-1, Stanford Univ., Stanford, CA, USA, 2009.
[20] S. Gutz, A. Story, C. Schlesinger, and N. Foster, "Splendid isolation: A slice abstraction for software-defined networks," in Proc. 1st Workshop HotSDN, 2012, pp. 79–84.
[21] B. Pfaff, J. Pettit, K. Amidon, M. Casado, T. Koponen, and S. Shenker, "Extending networking into the virtualization layer," in Proc. Hotnets, 2009, pp. 1–6.
[22] J. H. Jafarian, E. Al-Shaer, and Q. Duan, "OpenFlow random host mutation: transparent moving target defense using software defined networking," in Proc. 1st Workshop Hot Topics Softw. Defined Netw., 2012, pp. 127–132.
[23] P. Schneider and G. Horn, "Towards 5G Security," 2015 IEEE Trustcom/BigDataSE/ISPA, Helsinki, 2015, pp. 1165-1170. doi: 10.1109/Trustcom.2015.499
[24] J. Cao, M. Ma, H. Li, Y. Zhang and Z. Luo, "A Survey on Security Aspects for LTE and LTE-A Networks," in IEEE Communications Surveys & Tutorials, vol. 16, no. 1, pp. 283-302, First Quarter 2014. doi: 10.1109/SURV.2013.041513.00174
[25] A. Sgambelluri, A. Giorgetti, F. Cugini, F. Paolucci, and P. Castoldi, "Effective flow protection in OpenFlow rings," in Proc. OFC/NFOEC, Mar. 2013, pp. 1–3.
[26] J. Li, J. Hyun, J.-H. Yoo, S. Baik, and J.-K. Hong, "Scalable failover method for data center networks using OpenFlow," in Proc. IEEE NOMS, May 2014, pp. 1–6.

[27] P. Nikander, A. Gurtov, and T. Henderson, "Host Identity Protocol (HIP): Connectivity, mobility, multi-homing, security, and privacy over IPv4 and IPv6 networks," IEEE Commun. Surveys Tuts., vol. 12, no. 2, pp. 186–204, 2nd Quart. 2010.

[28] S. Namal, I. Ahmad, A. Gurtov and M. Ylianttila, "Enabling Secure Mobility with OpenFlow," 2013 IEEE SDN for Future Networks and Services (SDN4FNS), Trento, 2013, pp. 1-5. doi: 10.1109/SDN4FNS.2013.6702540

[29] M. Liyanage, A. B. Abro, M. Ylianttila, A. Gurtov, "Opportunities and Challenges of Software-Defined Mobile Networks in Network Security Perspective", published in IEEE Security and Privacy, August 2016.

[30] A.N. Bikos , N. Sklavos , "LTE/SAE security issues on 4G wireless networks", Secur. Privacy IEEE 11 (2) (2013) 55–62 .

[31] M. Liyanage, A. Braeken, A. D. Jurcut, M. Ylianttila, A. Gurtov, "Secure Communication Channel Architecture for Software Defined Mobile Networks", to be appear in Elsevier Journal on Computer Networks (COMNET), 2017.

[32] M. Liyanage, M. Ylianttila, A. Gurtov, "Securing the Control Channel of Software-Defined Mobile Networks" , in Proc. of IEEE 15th International Symposium on World of Wireless, Mobile and Multimedia Networks (WoWMoM), Sydney, Australia. June 2014.

[33] Lam, Jun-Huy, et al. "Securing distributed SDN with IBC." 2015 Seventh International Conference on Ubiquitous and Future Networks. IEEE, 2015

[34] Security-enhanced floodlight," SDx Central, Sunnyvale, CA, USA. [Online]. Available: http://www.sdncentral.com/education/towardsecure- sdn-control-layer/2013/10/

[35] B. Switch, "Developing floodlight modules. Floodlight OpenFlow controller," 2012. [Online]. Available: http://www.projectfloodlight.org/ floodlight/