Small cEllS coordinAtion for Multi-tenancy and Edge services

Grant Agreement No.671596

Topic: H2020-2014-ICT-14
Advanced 5G Network Infrastructure for the Future Internet
Research and Innovation Action

Deliverable D7.2
Integrated CESC Prototype Validation

Document Number:	H2020-5GPPP-GA No.671596/*WP7/D7.2/30.06.2017*
Contractual Date of Delivery:	*30.06.2017*
Editor:	Alan Whitehead – IP.Access Ltd.
Work-package:	*WP7*
Distribution / Type:	Public (PU) / Report (R)
Version:	1.0
Total Number of Pages:	36
File:	*SESAME_Deliverable 7.2_v1.0_Final*

Abstract

This document describes the validation performed and planned for the SESAME Cloud Enabled Small Cell (CESC) Prototype. It is an interim report that describes the progress to date.

A later report (Deliverable D7.4) will provide an update.

Version History

Version	Date	Comments, Changes, Status	Authors, contributors, reviewers
0.1	12.07.2017	Initial draft	Alan Whitehead (IPA)
0.2	17.07.2017	Added contributions for vWatermark & EPC	Alan Whitehead (IPA)
0.3	19.07.2017	Merged Italtel contribution	ITL
0.4	19.07.2017	Contributions from FLE, Orion & NCSRD	FLE, Orion and NCSRD
0.5	20.07.2017	VOSYS contribution	VOSYS
0.6	24.07.2017	CNET & FLE updates	CNET, FLE
0.7	24.07.2017	Conclusion and corrected a number of typos	Alan Whitehead (IPA)
0.8	25.07.2017	Correction to CNET contribution.	Alan Whitehead (IPA)
0.9	28.07.2017	Final draft	Alan Whitehead (IPA)
0.10	28.07.2017	Clean version with change marks removed	Alan Whitehead (IPA)
0.11	29.07.2017	Inclusion of CNET contribution	CNET
0.12	29.07.2017	Full editorial review and inclusion of contributions	SMNET
1.0	31.07.2017	Full conceptual and editorial review by the project coordinator with contributions in all sections. Submission to the European Commission	OTE

Contributors

First Name	Last Name	Partner	Email
Alan	Whitehead	IPA	Alan.Whitehead@ipaccess.com
Michail-Alexandros	Kourtis	NCSRD	akis.kourtis@iit.demokritos.gr
George	Xilouris	NCSRD	xilouris@iit.demokritos.gr
Ioannis	Giannoulakis	NCSRD	giannoul@iit.demokritos.gr
Bego	Blanco	EHU	begona.blanco@ehu.eus
Ianire	Taboada	EHU	ianire.taboada@ehu.es
Jose Oscar	Fajardo	EHU	joseoscar.fajardo@ehu.eus
Fidel	Liberal	EHU	fidel.liberal@ehu.eus
Elisa	Jimeno	ATOS	elisa.jimeno@atos.net
Javier	García	ATOS	javier.garcial@atos.net
Pouria Sayyad	Khodashenas	i2CAT	pouria.khodashenas@i2cat.net
Athanasios	Chalas	i2CAT	athanasios.chalas@i2cat.net
Emmanouil	Kafetzakis	ORION	mkafetz@orioninnovations.gr
Daniele	Munaretto	ATH	Daniele.munaretto@athonet.com
Antonino	Albanese	ITL	antonino.albanese@italtel.com
Claudio	Meani	ITL	claudio.meani@italtel.com
Pietro	Paglierani	ITL	pietro.paglierani@italtel.com
Marco	Beccari	ITL	marco.beccari@italtel.com
Hui (Julie)	Xiao	FLE	Hui.Xiao@uk.fujitsu.com
Mick	Wilson	FLE	Mick.Wilson@uk.fujitsu.com
Michele	Paolino	VOSYS	m.paolino@virtualopensystems.com
Leonardo	Goratti	CNET	lgoratti@fbk.eu
Tejas	Subramanya	CNET	t.subramanya@fbk.eu
Roberto	Riggio	CNET	rriggio@fbk.eu
Athanassios	Dardamanis	SMNET	adardamanis@smartnet.gr
Ioannis	Chochliouros	OTE	ichochliouros@oteresearh.gr

Glossary

Acronym	Explanation
3GPP	The Third Generation Partnership Project
5G	The Fifth Generation of Mobile Communications
AC	Admission Control
AEC	Analysis and Event Capture
API	Application Programming Interface
AR	Augmented Reality
ARM	Advanced RISC Machine
BF	Broadband Forum
CCTV	Closed Circuit Television
CE	Crowded Event
CESC	Cloud Enabled Small Cell
CLI	Command Line Interface
CP	Control Plane
CPU	Central Processing Unit
CV	Computer Vision
DB	Database
DC	Data Centre
DL	Downlink
DNAT	Destination Network Address Translation
DP	Data Plane
DPI	Deep Packet Inspection
E2e	End-to-End
EMS	Element Management System
eNB	eNodeB
EPC	Evolved Packet Core (network)
EuCNC	European Conference on Networks and Communications
E-UTRAN	Evolved Universal Terrestrial Radio Access Network
fps	frames per second
FTP	File Transfer Protocol
FW	Firewall
GA	Grant Agreement
GB	Giga Bytes
GbE	Gigabit Ethernet
G-PDU	GTP PDU
GPRS	Generalised Packet Radio Service
GPU	Graphics Processing Unit
GTP	GPRS Tunnelling Protocol
GTP-U	GTP User Plane
GW	Gateway
H2020	Horizon 2020
HEVC	High Efficiency Video Coding
HSS	Home Subscriber Server
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
HW	Hardware
I/O, i/o	Input/Output
ICT	Information and Communication Technology
ID, id	Identifier
IMSI	International Mobile Subscriber Identity
initrd	initial RAM disk
IoT	Internet of Things

IP	Internet Protocol
Light DC	Light Data Centre
LGPL	Lesser General Public License
LRU	Least Recently Used
LTE	Long Term Evolution
MAC	Medium Access Control (layer of the protocol stack)
MEC	Mobile Edge Computing
MME	Mobility Management Entity
MNO	Mobile Network Operator
MOCN	Multi-Operator Core Network
MPEG	Moving Pictures Experts Group
MRI	MEC-RAN Information
NAPI	New API
NFV	Network Function Virtualisation
NIC	Network Interface Card
NO	Network Operator
NS	Network Service
OAI	Open Air Interface
OS	Operating System
OVS	Open vSwitch
PBFS	Packet Based per Flow State
PDN	Packet Data Network
PDU	Protocol Data Unit
PFS	Proportional Fair Scheduler
PLMN	Public Land Mobile Network
PNF	Physical Network Function
PoC	Proof of Concept
PoP	Point of Presence
PPP	Public Private Partnership
RAB	Radio Access Bearer
RAM	Random Access Memory
RAN	Radio Access Network
RIA	Research and Innovation Action
RISC	Reduced Instruction Set Computing
RNIS	Radio Network Information Service
RRC	Radio Resource Control (layer of the protocol stack)
RRM	Radio Resource Management
RTT	Round-Trip Time
S1	The 3GPP S1 Interface between an eNodeB and an MME
S1-Flex	A 3GPP S1 feature in which an eNodeB has S1 interfaces to multiple MMEs
SATA	Serial Advanced Technology Attachment
SC	Small Cell
SCNO	Small Cell Network Operator
SCTP	Stream Control Transmission Protocol
SD	Secure Digital
SDN	Software-Defined Networking
SDR	Software-Defined Radio
SecGW	Security GateWay
SGW	Serving GateWay
SIM	Subscriber Identity Module
SNAT	Source-Network Address Translation
TAI	Track Area Identity
TEID	Tunnel Endpoint ID
TEMU	Telecommunications and Multimedia
TOFS	Traffic Offload Service

TR	Technical Report
TRAN	Terrestrial Radio Access Network
TS	Technical Specification
UDP	User Datagram Protocol
UE	User Equipment
UEFI	Unified Extensible Firmware Interface
UL	Uplink
UMTS	Universal Mobile Telecommunications System
URL	Uniform Resource Locator
USB	Universal Serial Bus
USIM	UMTS Subscriber Identity Module
UTRAN	Universal Terrestrial Radio Access Network
vDPI	virtual Deep Packet Inspection
VA	Video Analytics
VIM	Virtual Infrastructure Manager
VM	Virtual Machine
VNF	Virtual Network Function
VNFC	Virtual Network Function Component
VNG	Variable Number of Gradients
VSCNO	Virtual Small Cell Network Operator
vTU	Virtual Transcoding Unit
WP	Work Package
XML	eXtensible Mark-up Language

Table of Contents

List of Figures

List of Tables

1. Introduction

1.1. Deliverable outline

The target of the 5G-PPP EU-funded SESAME project is to design and develop a novel 5G platform based on small cells (SCs), featuring multi-tenancy between network operators and also attach to them edge cloud capabilities to be offered to both the network operators (NOs) and the mobile users. Thus, the "key innovations" proposed by SESAME focus upon the novel concepts of a multi-operator (i.e.: multi-tenancy) enabling framework and also upon providing an edge-*based*, virtualised execution environment.

The original Deliverable D2.2 [1] presented the overall design and specification of the SESAME system architecture. The subsequent Deliverable D2.3 [2] has been built on this, with a more detailed specification of the CESC components. The Deliverable D3.1 [3] has "crystallised" these essential elements into a design for the Small Cell Prototype and Proof-of-Concept (PoC).

This report is an output of Task 7.3 ("*Test-bed Integration and Prototype Testing*") and forms the respective contractual Deliverable D7.2. It reports on the status of the validation of the CESC prototype *performed to date*. This integration and testing work will continue as part of Work Package 7 and a further report (Deliverable D7.4) will be produced upon the conclusion of this work.

1.2. Definitions of Terms and SESAME concepts

At this point, it is useful to provide definitions of terms and processes which will be used later in this document to describe the SESAME main concepts.

- **Small Cell (SC):** Does not change in the context of SESAME.
- **Execution infrastructure, micro-server:** Specific hardware that is placed close to or inside the Small Cell and provides processing power (and also memory and storage capabilities).
- **CESC (Cloud Enabled Small Cell):** The Small Cell device which includes a micro-server in hardware form.
- **Cluster of CESCs:** A group of CESCs that are collocated, exchange information and are properly coordinated. As a trivial case, one CESC can be called CESC cluster.
- **Light Data Centre (Light DC):** The hardware entity composed by the micro-servers of the CESCs forming a cluster.
- **Evolved Packet Core (EPC):** The 4G core network functionalities to enable the CESC components.

Figure 1-1 below provides an overview of SESAME CESC Cluster from a physical system perspective.

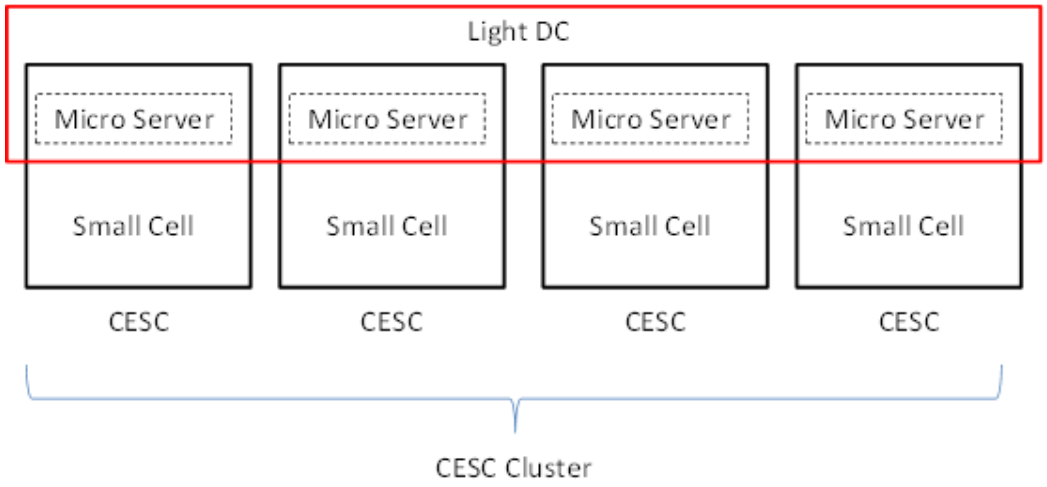

Figure 1-1: Conceptual view of SESAME CESC Cluster components

1.3. Scope

This report covers the following elements of the SESAME architecture implemented on or closely coupled with the Cloud Enabled Small Cell (CESC):

- The CESC and Micro Server hardware,
- The SC-PNF,
- The SC-Common VNF,
- The SC-VNF,
- The EPC,
- The demonstration Service chain VNFs (vDpi, vTU, vFirewall and vVideo Analytics).

Elements specifically excluded from this report are:

- The Small Cell EMS,
- The Virtual Infrastructure Manager (VIM),
- The NFV Orchestrator.

These will be reported on in Deliverable D7.4.

2. Implementation Status

The components described in this section form part of SESAME Light DC and CESC architecture as illustrated by Figure 2-1 below. For a detailed description of their functionality, see the description provided within the deliverables D2.2 [1] and D2.3 [2].

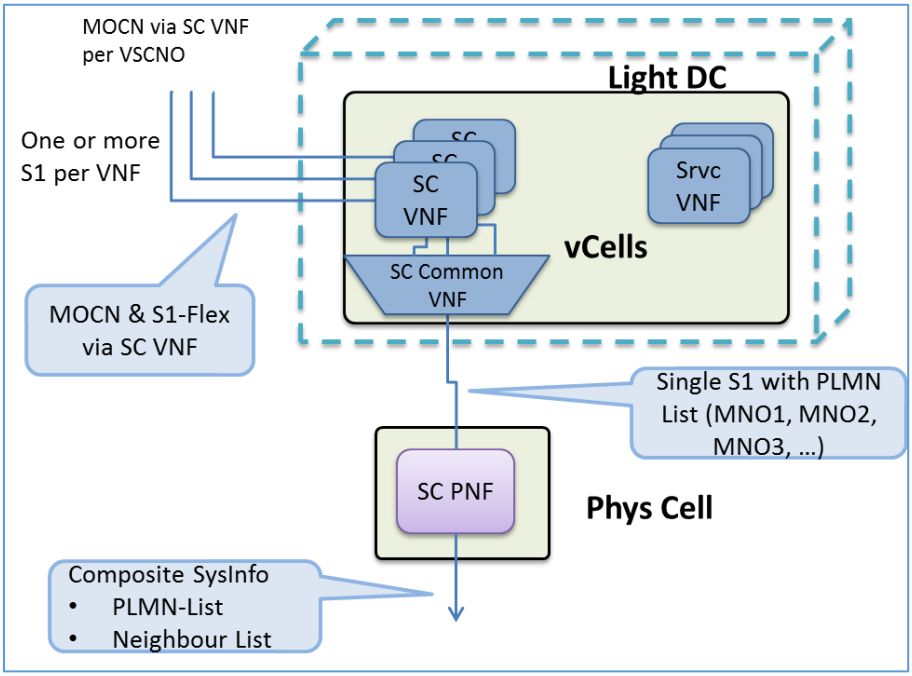

Figure 2-1: Light DC Functional Architecture

2.1. Light DC

The system proposed for the PoC provides the use of four different blocks acting as micro-servers:

1. STM board[1] running a virtualisation layer to host Common SC-VNF and SC-VNF.
2. Raspberry Pi-3[2] board (very low cost micro server node; used as development platform for lightweight virtualisation solutions, especially for the case of virtualisation of small cell-related functions).
3. NXP board[3] running a virtualisation layer to host SC-VNF, Service VNFs (SW only) and storage integrated with OpenStack Kilo[4] and VOSYSwitch.
4. INTEL node (Xeon v3[5]) equipped with a NVIDIA GPU M4000[6] for vTU HW acceleration and storage.

---

1 See: http://www.st.com/en/evaluation-tools.html

2 For more details also see: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

3 See: http://www.nxp.com/products/microcontrollers-and-processors/arm-processors/qoriq-arm-processors/qoriq-ls2085a-rdb-reference-design-board:LS2085A-RDB

4 OpenStack Kilo, the 11th release of the open source software for building public, private, and hybrid clouds has nearly 400 new features to support software development, big data analysis and application infrastructure at scale. For more details see: https://www.openstack.org/software/kilo/

5 See: http://www.intel.com/content/www/us/en/processors/xeon/xeon-processor-e5-family.html

The main characteristics of the micro server are described in Table 1 below.

Micro-server	Architecture	Cores	RAM	Storage	PCI-e[7] Acceleration
NXP LS2085A[8]	ARMv8[9], A57[10]	8	16 GB	500 GB	no
GOMA[11] (FlexPAC)	Xeon E5-2630v3[12]	8	64 GB	4+2 TB	GPU
STM board	ARMv8, A53[13]	4+1	1 GB	xx GB (SATA disk[14])	no
Raspberry pi 3	ARMv8, A53	4	1 GB	xx GB (microSD[15])	no

Table 1: Main characteristics of micro servers potentially usable in the Light DC

Assessment of each of these platforms and their ability to run the required virtual environment is currently in progress (see section 3.1.1).

2.2. SC PNF

There is a single SC PNF connected to each CESC. In the PoC, the Small Cell PNF is provided by the ip.access E40 LTE Access Point [4]. It has been extended to provide per-PLMN versions of a number of performance management counters. These are then post-processed by the EMS into separate per-VSCNO reports. In every other respect, the SC PNF is identical to the standard E40 product.

The SC PNF supports multiple PLMNs (and therefore multiple tenant VSCNOs) via a single S1 connection to the SC-Common VNF.

A version of the SC PNF incorporating the required SESAME functionality has been available for integration and testing since M18. To date, testing has involved establishing an S1 connection to the SC-Common VNF and, *as a reference*, directly to the EPC. This functionality has now been successfully demonstrated.

---

[6] For more details about the respective equipment see: *https://images.nvidia.com/content/quadro/product-literature/data-sheets/12489_NV_DS_Quadro_M4000_US_NV_FNL_HR.pdf*

[7] For more relevant information also see, *for example*: *https://en.wikipedia.org/wiki/PCI_Express*

[8] For more details see: *http://www.nxp.com/products/no-longer-manufactured/qoriq-layerscape-2085a-and-2045a-multicore-communications-processors:LS2085A*

[9] The ARMv8 architecture introduces 64-bit support to the ARM architecture with a focus on power-efficient implementation, while maintaining compatibility with existing 32-bit software. More related information can be found at: *https://www.arm.com/products/processors/armv8-architecture.php*

[10] For more details see: *https://www.arm.com/products/processors/cortex-a/cortex-a57-processor.php*

[11] For more details see: *https://www.gomaelettronica.it/en/server-and-workstation-portables-high-density-storage-server-intel-i7-i5-i3-series*

[12] For more details see: *http://ark.intel.com/products/83356/Intel-Xeon-Processor-E5-2630-v3-20M-Cache-2_40-GHz*

[13] For more details see: *https://www.arm.com/products/processors/cortex-a/cortex-a53-processor.php*

[14] More related informative details can be found, for example, at: *https://en.wikipedia.org/wiki/Serial_ATA*

[15] Also see: *https://en.wikipedia.org/wiki/Secure_Digital#Microhttps://en.wikipedia.org/wiki/Secure_Digital#Micro*

2.3. EPC

In order to enable the MOCN functionality, three virtual EPCs (vEPCs) are deployed remotely and connected via separate S1-C/U interfaces[16] to each SC VNF in the CESC. Each couple, comprising SC-VNF and corresponding vEPC, are identified by a unique PLMN ID. UEs are equipped with SIMs provisioned in the HSS such that each vEPC has its own HSS with a list of SIMs registered to the same PLMN of the EPC.

End-to-end (E2E) testing has established correct operation of the EPC with the SC-VNF instances.

2.4. SC-Common VNF

Each CESC contains a single instance of the SC-Common VNF. It provides two main functions in the SESAME architecture:

- It provides a multiplexing – de-multiplexing function. In the uplink (UL) it separates the control plane traffic associated with each virtual cell received from the PNF and routes it to the appropriate SC VNF. In the downlink (DL)), it combines the control plane traffic from each SC VNF onto the single S1 connection established by the SC PNF.
- It provides a PNF wide admission control function, rejecting UE attachment or bearer establishment if the *configured* capacity of the PNF is reached or exceeded.

The first version of the SC-Common VNF was provided for test purposes in M22. Currently, the SC-Common VNF is supported on the Ubuntu 14.04 operating system[17] and has been tested in an environment that approximates the CESC using Oracle Virtual Box 5.1.14 [5]. Thus far, testing has focused on establishing end-to-end connectivity and has not included any tests associated with admission control.

End-to-end testing is not yet complete and correct operation under all circumstances[18] has not yet been verified. Porting to the CESC HW has not been attempted.

2.5. SC VNF

A CESC may contain up to five[19] instance of the SC VNF. Each SC VNF instance and the single SC-Common VNF lie on the S1 interface between the SC PNF and the EPC of a particular tenant. As such, the SC VNF provides the tenant specific functions of a virtual cell hosted by the CESC.

In particular, it is responsible for:

- Establishing one or more S1 connections towards the EPC of the VSCNO owning the virtual cell.
- Performing VSCNO specific admission control, capping both the number of UEs that can be served by the virtual cell and the aggregate uplink and downlink data throughput of the virtual cell.
- Optionally passing user plane traffic into the service chain for network edge processing.

---

[16] For further information also see, *inter-alia*: *http://lteworld.org/wiki/long-term-evolution-lte*

[17] See: *http://releases.ubuntu.com/14.04/*

[18] In particular, the actions of provisioning and de-provisioning multiple virtual cell instances (and therefore multiple SC VNFs) on a CESC has not been validated.

[19] One PLMN ID out of the six permitted by MOCN is reserved for use by the SCNO.

The first version of the SC VNF was provided for test purposes in M22. Currently, the SC VNF is supported on the Ubuntu 14.04 operating system and has been tested in an environment that approximates the CESC using Oracle Virtual Box 5.1.14 [5]. Thus far, testing has focused on establishing end-to-end connectivity and has not included any tests associated with admission control.

End-to-end testing is not yet complete and correct operation has not yet been verified. Porting to the CESC HW has not been attempted.

2.6. GTP De-capsulation and Encapsulation

The GTP traffic de-capsulation and encapsulation is part of the caching functionality implementation and, *anyway*, common to any MEC RAN service. For such reason, the following explanation of the development of the GTP traffic de-capsulation/encapsulation function will implicitly rely on the MEC RAN system architecture shown in Figure 2-2.

The GTP de-capsulation/encapsulation function is composed of the following elements:

1. MEC-RAN Information interface (MRI): it is implemented using Tshark[20] (a terminal version of wireshark[21]), by spawning a new process running in parallel with the GTP evaluation functionality. MRI parses only the S1-C communication from the live capture of the SCTP[22] traffic as a dissector filter.
2. Radio Network Information Service (RNIS): it is implemented as an XML database (DB) containing all the S1AP messages exchanged between the small cell and the EPC which are extracted in the above step.
3. Analysis and Event Capture (AEC): the XML parser processes only the relevant S1AP messages from RNIS, particularly Context Management Messages, and it extracts all the necessary information to detect Radio Access Bearer (RAB) setup, bearer tear down and bearer reconfiguration events. The most important information extracted include: International Mobile Subscriber Identity[23] (IMSI), Track Area Identity[24] (TAI), UE IP, SGW IP, SGW Tunnel Endpoint ID (TEID), PLMN ID, eNB IP and eNB TEID. Based on the information extracted, the AEC determines whether the UE attach or detach procedure is successful or not, and it sends event triggers to the Traffic Offload Service (TOFS) to perform GTP de-capsulation and encapsulation.
4. To implement TOFS it was used the Lagopus 0.2.10 software switch[25] that supports OpenFlow 1.3 standard[26] and it is connected to Ryu[27] for the SDN controller part. RYU API were extended to support GTP flow modification control messages to modify the flow table

[20] For further details see: https://www.wireshark.org/docs/man-pages/tshark.html
[21] For further details see: https://www.wireshark.org/
[22] For more details see, *for example*: https://en.wikipedia.org/wiki/Stream_Control_Transmission_Protocol
[23] See, *for example*: https://en.wikipedia.org/wiki/International_mobile_subscriber_identity
[24] For more details see, *for example* http://lteworld.org/forums/lteworld-forum/what-tracking-area-identity
[25] For more details see: https://github.com/lagopus/lagopus/releases/tag/v0.2.10
[26] For further details see: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.3.0.pdf
[27] Ryu is a component-based software-defined networking framework. For more details see: https://osrg.github.io/ryu/

in the Lagopus switch. CLI Lagosh[28] was used to configure, observe and manage the Lagopus.

Workflow of the GTP de-capsulation/encapsulation:

The experimental testbed consists of OAI open source software[29] for the small cell protocol stack [6], the Soekris[30] server to implement the CESC of SESAME, and the Athonet EPC[31]. Commercial LTE smart-phones with programmable SIM cards were used for the mobile user's part. A low-power, low-cost, advanced Linux computer (i.e. the Soekris net6501[32]) was used as a light-weight MEC server that includes Ryu controller, Lagopus software switch and a video caching application (i.e. Squid[33]).

The GTP functionality is crucial in SESAME, since it ensures that the CESC is transparent to both core network (CN) and UE. Indeed, the EPC will simply be connected to the small cell, as well as the UE, and both are unaware that the user plane traffic passed through a service chain.

By default, the Lagopus works as a standard Layer 2 switch[34]. It forwards packets from the small cell to the SGW/MME and *vice versa*. However, when it receives a flow modification request from the Ryu controller, flow rules are installed to divert UE connections to different destinations (e.g. vCache).

The Lagopus matches on uplink packet headers to check if the traffic is GTP type (i.e. UDP port=2152 in the experiments conducted) originated from a UE, and if the TEID is the same as that of the SGW TEID.

In positive case, the Lagopus strips off the outer UDP, outer IP and GTP headers on the matched packets and it forwards the traffic towards a MEC RAN service (vCache in this case) by changing the destination MAC address to that of the next hop in the path.

IP packets are matched to see if the packets arriving from the service chain are originated from a UE (i.e. source IP=UE IP), and if so, the Lagopus performs GTP, outer IP and outer UDP header encapsulation and forwards the traffic towards the EPC by changing the destination MAC address to that of next hop in the path.

[28] For further details see: *http://www.lagopus.org/lagopus-book/en/html/lagosh.html*
[29] See: *http://www.openairinterface.org/*
[30] See: *http://www.soekris.com/*
[31] See: *http://www.athonet.com/athonet/epc/*
[32] For more details see: *http://soekris.com/products/net6501-1.html*
[33] For further information see: *http://www.squid-cache.org/*
[34] For further information see, *for example*: *https://en.wikipedia.org/wiki/Network_switch#Layer_2*

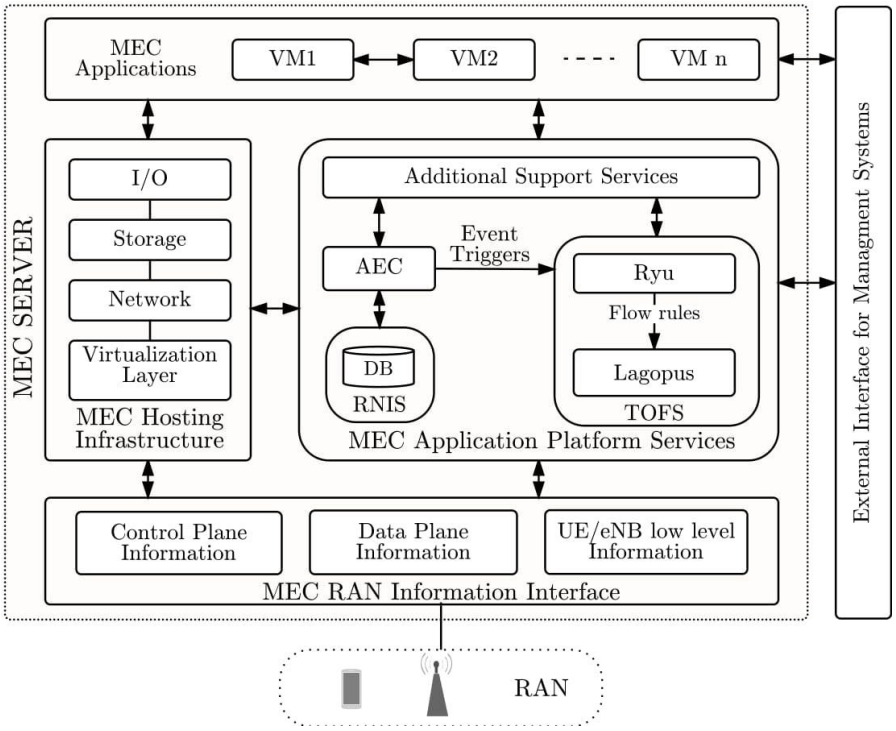

Figure 2-2: MEC RAN system architecture

2.7. Service Chain VNFs

2.7.1.vDpi

The virtual Deep Packet Inspection (vDPI) VNF used comprises one Virtual Network Function Component (VNFC), namely the Traffic Inspection engine and Classification. The VNFC is implemented in a single VM. The proposed DPI solution is based upon the approach to analyse a small number of initial packets from a flow in order to identify the flow type. After the flow identification step no further packets are inspected. The vDPI follows the Packet Based per Flow State (PBFS) in order to track the respective flows. This method uses a table to track each session based on the 5-tuples (source address, destination address, source port, destination port, and the transport protocol) that is maintained for each flow.

The vDPI is a computationally intensive VNF. It implements the filtering and packet matching algorithms in order to support the enhanced traffic inspection capability. The component supports a flow table (exploiting hashing algorithms for fast indexing of flows) and an inspection engine for application recognition. The vDPI VNF is based upon the SESAME network architecture and uses the management/monitoring and datapath interfaces. All the computational and packet processing utilise mostly CPU and memory resources. The VNF requires intensive CPU tasks and a large number in memory I/Os for the traffic analysis.

The vDPI utilises various technologies in order to offer a stable and high performance VNF compliant to the high standards of legacy physical network functions. The implementation for the traffic inspection used for these experiments is based upon the open source nDPI library [7]. The packet capturing mechanism is implemented using various technologies in order to investigate the trade-off between performance and modularity. The packet handling technology used is PF_RING. PF_RING is a set of library drivers and kernel modules, which enable high-throughput, packet capture and sampling. For the needs of the vDPI the PF_RING kernel module

library is used, which is polling the packets through the LINUX NAPI[35]. The packets are copied from the kernel to the PF_RING buffer and then they are analyzed using the nDPI library. The various technologies used generate a great variety of test case scenarios and exhibit a rich VNF test case. The PF_RING case has the capability of keeping the NIC driver, and so the VNFC maintains connectivity with the OpenStack network connected.

Regarding the SESAME specific service chaining under study, the vDPI also functions only with IP packets, thus the need for a GTP decapsulator near the Edge. The data packets exchanged between the EPC and eNB that need to be inspected, are decapsulated from their GTP header and are forwarded in the Light Data Center where the vDPI VNF is hosted in order to be further processed.

2.7.2.vWatermark

This section describes the experimental testbed that was implemented for the needs of the SESAME project demonstrating a real-time video watermarking scenario. The testbed, as Figure 2-3 shows, implements a fully operational LTE mobile network domain and consists of an EPC, a SC, a UE, a NFVI-PoP hosting the VNFs and a video server.

The LTE network is built upon OpenAirInterface (OAI) wireless technology platform. The OAI EPC software runs on a 64 bit x86-*based* computer, while the small cell is implemented by using a B210 Ettus card[36], installed on a similar computer, running the appropriate OAI eNodeB software, which for our case is acting as the Remote SC. The UE is based on a laptop equipped with 4G LTE USB Adapter, which includes a USIM card with the appropriate keys stored in it, so that the UE can be authenticated by the EPC. The two computers (EPC and SC are interconnected over an S1 interface [8], which provides all the control signaling and data transport between EPC and eNodeB.

2.7.2.1. Virtualised Network Service Operation in SESAME environment.

Referring to Figure 2-3, the NFVI-PoP is located between the EPC and the Remote SC. The virtualisation platform of the NFVI-PoP is supported by the OpenStack open source cloud computing platform. The release used was the Liberty candidate[37], which was the latest stable version during the time of the experimental tests. The NFVI-PoP is capable of instantiating NSs and performing also the appropriate network traffic steering, in order to support service chaining (i.e. the forwarding of the traffic seamlessly to each VNF) and finally to the UE. The NS under demonstration exhibits a multimedia and traffic identification scenario. The video server in the testbed hosts the video files to be tested at their original version. In order to stream the video files and add a pre-defined watermark through the network and also analyse all the passing network traffic, a vWatermark, a vDPI and a vGTP are instantiated at the EPC NFVI-PoP. These VNFs form the NS this section describes. The vWatermark implementation is based on the widely used FFMPEG [9] and is instantiated as a VNF in the EPC NFVI- PoP. The vDPI implementation is based upon the widely used protocol identification library nDPI. The implementation of the proposed scenario is split into three VNFs, distributed in the NFVI-PoP.

In order to perform the tests, a mechanism to vary the bandwidth of the backhaul link (and thus its quality) is required. For this reason, an Open vSwitch[38] (OVS) is deployed in the testbed, as shown in Figure 2-3. OVS is a virtual switch, licensed under the open source Apache 2.0 license[39], and is very suitable for virtualisation purposes. OVS is able to perform many network functions,

[35] Further information can be found at: *https://wiki.linuxfoundation.org/networking/napi*
[36] See: *https://kb.ettus.com/B200/B210/B200mini/B205mini*
[37] See: *https://www.openstack.org/software/liberty/*
[38] See: *http://openvswitch.org/*
[39] For more details see: *https://www.apache.org/licenses/LICENSE-2.0*

one of which is the control of the bandwidth of the IP traffic among its ports. An OVS was installed in the Central SC NFVI PoP, controlling the traffic between the vGTP and the rest of the VNFs, and directing each traffic to the appropriate VNF. The deployed OVS was able to receive commands through OpenFlow protocol [10]. So, through OpenFlow commands to the OVS, it was possible to emulate conditions for a multi-service environment, where traffic splitting is required, in order to provide different types of services.

In the typical LTE architecture, the traffic exchanged between the EPC and the SC, either data or control signals, is encapsulated by using the GPRS Transport Protocol (GTP), as previously explained. On the other hand, the vWatermark -as well as the OVS- require pure IP packets, so it is necessary for the traffic to be de-capsulated from its GTP headers and then forward the inner IP packets, which contain the actual video service data, to the upper modules. For the needs of the paper, a vGTP decapsulation and re-encapsulation software has been implemented, running on top of the widely used packet processing library PF_RING [10]. vGTP is running as a VNF, in the Central SC NFVI-PoP forwarding the traffic both directions. It also passes through all the signaling and data traffic between EPC and SC, thus preserving the connectivity between them.

The service chaining demonstrated in this scenario is based upon a three-VNF virtualised network service consisting of three virtualised network functions. More specifically, the vGTP enables the proper network operation of the other two VNFs in the NFVI-PoP, as up to this point the system handles GTP packets. The GTP packets cannot be used by the rest of the VNFs in next steps of the service function chain, if they remain unmodified. At this the vGTP will strip them of their GTP header, so they can be usable from the vDPI and the vWatermark VNFs.

The video traffic is filtered from the rest of the control traffic and the GTP header is removed from the filtered packets, which are then forwarded to the OVS. The bandwidth regulated IP traffic is sent back to the vGTP through OVS, where they are re-encapsulated with the valid GTP header and further on forwarded to their original path, to arrive at the UE. As the decapsulation and re-encapsulation operations can introduce a penalty to the performance of the system, they are performed in a parallel manner to the rest of the process. The signaling and the rest of the data traffic is forwarded by using the zero-copy PF_RING library, and only the video service packets are copied to memory, as they need to be further processed by a GTP agnostic mechanism. The GTP header storage for the re-encapsulation is considered insignificant as the GTP header, merely allocates 8 bytes to the memory.

The functionality provided by the vGTP is vital to the proposed framework, as it handles GTP traffic and delivers it in a valid IP format to the vWatermark and vDPI VNFs to process it. The vGTP enables the integration of the SC architecture and environment into a multimedia-over-IP environment seamlessly, by handling the SC GTP traffic. GTP traffic is used by telco providers to secure network traffic between the EPC and the eNodeB.

The second VNF, the vDPI is located after the OVS, at the EPC NFVI-PoP. The main objective of this VNF is to analyse the incoming traffic and provide advanced traffic statistics and protocol identification. The third VNF, the vWatermark receives the incoming video signal and creates an overlay of a pre-defined watermark to it. Both VNFs can be deployed in a lightweight virtual machine, requiring at least 512 MB of memory and one virtual CPU. However, if the workload is increased significantly, it would require a larger amount of resources, to maintain a satisfactory quality of operation.

2.7.2.2. Multi-tenancy Support in SESAME testbed

The experimental testbed in SESAME is built in order to support multi-tenancy operation. Multi-tenancy in SESAME translates into different mobile network operators providing their services over a common infrastructure. The shared utilization of resources and infrastructure triggers various issues about isolation and security. In this section, the multi-tenancy aspect is narrowed down to the implementation details regarding the SESAME testbed. Based on the described architecture in Figure 2-3, in a multi-tenant scenario the traffic follows the same pattern although this time, each tenant flow has a different Public Land Mobile Network (PLMN) ID. The vGTP in this case before stripping the GTP header stores the flow information, correlating the PLMN ID to the corresponding packets.

In order to maintain isolation in the virtualisation plain of the SESAME testbed, each tenant has its own network service, with independent VNFs, totally isolated from other tenants. As the described testbed is implemented over OpenStack, the VMs hosting the networks services, have each one their own operating system and in extension their own kernel. Whereas, in the case of containers the shared kernel can prove to be a security breach for the system.

Bear in mind that the current report presented a part of final SESAME demo which has been used for the validation purposes. SESAME integration and testbed delivery effort will continue in the framework of WP7 and further details about the actual integration steps and future tests will be reported on the upcoming WP7 deliverables.

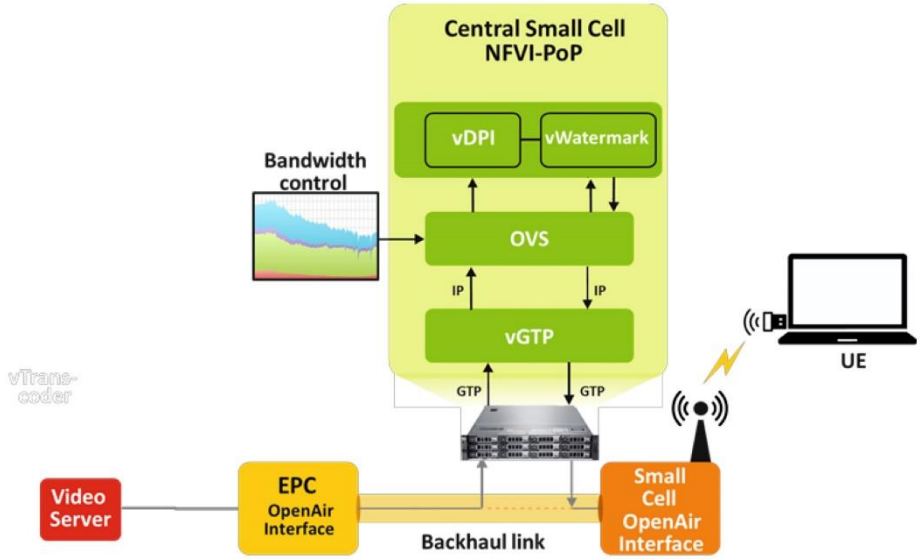

Figure 2-3: An overview of the experimental testbed

2.7.3.vTU

The Video Transcoding Unit (vTU) VNF provides some basic functionality, which can be used to build video content sharing services for users. In particular, the vTU provides video and audio transcoding functions, together with local storage capabilities of pre-recorded audio/video files.

The services provided by the vTU can be accessed through a web-service-*based* interface, available from any browser. Through the vTU, users can originate or receive live video streams. In particular, they can stream an originated video content to the vTU, or upload it as a pre-recorded file. Other users can receive this live content, or download it, at a later time. If the originated video is shared among users in real-time, the vTU concurrently saves it to a distributed storage system, for successive content sharing. In the vTU, a common storage area is also dedicated to system managers that can upload contents to be offered to all the users. Such functionalities can be used to build enhanced video services, well-suited for the highly debated Crowded Event (CE) use cases. In this scenario, a high number of users concentrates in a small area for a short time, typically ranging from few hours to a week. Well-known examples of CEs are sporting matches or concerts at a stadium, congresses or exhibitions hosted by dedicated venues, international events spread over a university campus or even an entire city.

A distinctive feature of vTU is the possibility not only to run on general purpose CPUs but also to exploit the Hardware acceleration provided by a GPU, to improve the compute performance of

video codecs. To this end, two different architectural approaches can be used. The first one, also known as "cooperative CPU- GPU" makes use of a GPU to offload the most compute-intensive functions of the video codec (usually, the Motion Estimation block), while the main algorithm is kept running on the CPU. The second approach, conversely, uses full HW implementation of video codecs. Today, various HW versions of the most popular encoding schemes, such as H.264[40], HEVC[41], VP8[42] and VP9[43] are available. The fully HW approach can provide higher compute performance than cooperative CPU-GPU algorithms. The HW approach very often lacks the flexibility in service management needed by operators, thus the cooperative approach is still preferred in many real-life implementations.

The vTU can adopt both GPU-accelerated approaches. In fact, it can use the NVidia NVENC[44] encoder for the H.264 and H.265 encoding scheme. Also, the CPU-GPU cooperative approach described in [11], [12] can be used for the Google Open Source VP8 encoder[45].

Thus, in SESAME three versions of the vTU VNF have been developed:

- SW-only vTU, for ARM[46] processors;
- SW-only vTU, for x.86[47] processors;
- HW-accelerated vTU, for x.86 processors with NVidia Graphics Processing Unit[48] (GPU).

Based on the needed performance, the power consumption budget and the expected costs, an operator can choose the most appropriate platform for the vTU services to be deployed.

2.7.4.vFirewall

vFirewall (vFW) is a VNF designed to protect computer networks from unwanted traffic, or to regulate specific applications. This device can be active and block unwanted traffic. This is the case, *for instance*, of firewalls and content filters. In the context of SESAME, we have suggested a vFW to filter the traffic for a specific network service, targeting a specific user, in the cases it looks suspicious, or a network regulation is required by the system. For instance, a high throughput for some applications is identified for several users in the Edge of the network, by revising the rules in the firewall we can block this traffic.

The different components of the vFW architecture interact in the following way:

1. Data packets are first of all filtered by the firewall (ingress interface) before being forwarded to the service (egress interface);
2. Filtered data packets are analyzed by the vDPI for further inspection (internal interface). The vDPI will monitor and analyze all the services passing through the network;
3. If suspicious, or unwanted traffic is detected at this stage, an alarm is generated and the firewall is instructed to revise its rules (internal interface);
4. If no attack is detected, no further action is required.

In addition to this, there is one extra interface, which is the in charge of the vFW lifecycle management, monitoring of the status of the vFW and sending the related information to the monitoring server.

---

[40] For more details see, *for example*: https://en.wikipedia.org/wiki/H.264/MPEG-4_AVC
[41] For more details see, *for example*: https://en.wikipedia.org/wiki/High_Efficiency_Video_Coding
[42] For more details see, *for example*: https://en.wikipedia.org/wiki/VP8
[43] For more details see, *for example*: https://en.wikipedia.org/wiki/VP9
[44] For more details see, *inter-alia*: https://en.wikipedia.org/wiki/Nvidia_NVENC
[45] http://www.streamingmedia.com/Articles/News/Online-Video-News/Google-Open-Sources-VP8--67265.aspx
[46] ARM is the industry's leading supplier of micro-processors technology, offering the widest range of micro-processors cores to address the performance, power and cost requirements for almost all application markets. More information can be found at: https://www.arm.com/products/processors
[47] x86 is a family of backward compatible instruction set architectures based on the Intel 8086 CPU and its Intel 8088 variant. More relevant information can be found, *inter-alia*, at: https://en.wikipedia.org/wiki/X86
[48] See: http://www.nvidia.com/object/gpu.html

As performance is one of the main issues when deploying software versions of security appliances, the implementation of vFW was based upon the widely used OpenVSwitch (OVS). OVS offers stability along with performance and can be easily integrated in a SDN-enabled environment, which is the case of SESAME. Various modifications were made in the network setup of OpenStack in order to accommodate the SESAME scenarios. The first one was the already mentioned GTP header manipulation, as vFW also requires IP packets to function properly. The second modification was on the Neutron port topology[49], where the vFW ports have to be directly attached to the SESAME provider network in order to operate properly.

2.7.5.vVideo Analytics

The two real-time video analytics (VA) based VNFs are examples of applications that require low end-to-end service latency and consume a large amount of real-time video streaming data to perform desired video analytics tasks. The implementations of the two VA VNFs are in Python[50], which are made as realistic as possible to enable two VA-*based* services, namely augmented reality (AR) service and real-time remote control of IoT devices (Smart IoT) services.

In terms of workflow, the two VA VNFs have some commonalities in that they both take the real-time video data streamed from video cameras, such as fixed CCTV cameras or mobile cameras, as the input data and video analytics is applied to the video inputs in real time to carry out meaningful analyses and output the derived analysis results or processed video data to the relevant receiving devices. However, the difference is in the case of AR service, the output data is the processed video data with additional information added to the input video data, whereas in the case of Smart IoT service, the output data is the derived video analysis results or instructions for autonomous IoT device control.

Figure 2-4: Results of the Video Analytics VNF for a 2 operators scenario

In terms of the main features of the services, both of the VA VNFs are based on the continuous tracking of a predefined object of interest; that is, for the AR service, the additional annotations regarding the object of interest are added to each video frame based on the continuous object tracking, and for the Smart IoT service, the monitoring camera or cameras of an object of interest are remotely controlled in order to continuously track the object of interest. The object tracking function is a common function used by both VA VNFs which consists of the following functional components: object detection, object recognition, object tagging and object location identification. The implementations of these functional components are in Python, mainly based on the usage of Open Source Computer Vision (OpenCV) libraries [13] and Caffe libraries [14] depending on the type of object being tracked and the real-life scenarios.

To ensure the VA-*based* services are provisioned to the relevant end devices with unnoticeable delays compared to the real-life situations, the required end-to-end service latencies need to be

[49] For more details see: *http://www.innervoice.in/blogs/2015/07/05/ports-in-openstack-neutron/*
[50] For further information see: *https://www.python.org/*

within 100 ms [15], which is the time between when a video frame is sent from a streaming camera to the mobile edge VA VNF and when the processed video frame or the derived instruction corresponding to the sent video frame arrives at a receiving UE. Additionally, the video frame processing rate at the VA VNFs, i.e. the number of frames being processed per second, should be higher than -or equal to- the steaming camera's frame rate, in order to avoid that the end-to-end latency of each frame increases with the number of the frames being processed. To be specific, if the streaming camera's frame rate is 30 fps, then the processing of each video frame at the VA VNF needs to be completed within 33 ms.

In order to meet the stringent requirements on end-to-end service latency and video data processing time, the resources provided to the VA VNF need to be dynamically adjusted to cope with the time varying traffic load and the type of required VA tasks. However, using one single video stream to track one moving object in a real-life setting requires a virtual machine with 8 GB of memory and 2 virtual CPUs, and to ensure smooth video streaming experience, the required transmission data rate between the sending/receiving UE and the VNF server needs to be 8 Mbps, *on average*.

For the needs of SESAME, the Video Analytics software was properly modified in order to address the needs of the SESAME infrastructure. First of all, additional streaming functions are added to the Video Analytics application in order to accept live streaming of stored video files so that to process them. To that extent, the output of the application was pipelined and integrated into a video streamer flow, so as to have a unified 5G application for video analytics totally compliant and compatible to the SESAME architecture. Of course, in a similar manner as previously described the GTP decapsulator and re-encapsulator was used, taking into account that the VA VNF operates in a similar manner to the vWatermarking, but of course with additional processing. As it can be seen in Figure 2-4 in an example SESAME service of VA and vWatermark, the same video streamed to 2 different users each own using a different operator's network, can be analyzed and annotated in real-time, and watermarked based on the corresponding operator in real-time. This experimental result clearly demonstrates the SESAME capabilities in a 5G NFV enabled multi-tenancy environment.

2.7.6.vCache

This section provides the description of the implementation work that was done to develop the vCache function in the SESAME system. The description provided herein below shall rely on the SESAME deliverable [16] and [17] for a detailed description of the functionality itself. It is anyway worth to emphasize that vCache is just one of the possible MEC RAN applications that are developed within SESAME. In addition, the full functionality was developed in the SESAME research platform in which specific advanced functionalities are currently experimented. The research platform relies on the open source software OAI to implement the protocol stack of the small cell.

The Video Caching is based on Squid application, and it is primarily designed to run on Unix systems to provide high performance web proxy caching service to mobile users in the SESAME network. In particular, the caching functionality supports HTTP, HTTPS and FTP protocols transparently caching the most frequently data requested by the users. Squid adopts the Least Recently Used (LRU) cache replacement policy to replace old cached content that has not been accessed for a sufficiently long time. Squid also keeps track of the object obsolescence (how much the content has aged since it was fetched for the first time) and it requests for the updated/refreshed content based on the "refresh_pattern" rules specified in the configuration file. To complement this, as already described in [16], an analytical work was carried out showing a method to cache contents based on their popularity and cache hit rate.

Workflow of the functionality:

When a web request arrives, the vCache first checks the cache to verify whether the content is already present or not. The cache key is the entire URL including the query string. If the content is present and the cache entry has not expired, the content is served directly from the cache. If, on the other hand, the requested content is not in the cache or the cache entry has expired, Squid makes a request to the origin server to retrieve the content. When Squid receives the response from the origin server, it stores the content in the cache based on the HTTP headers of the response. Anyway, some contents cannot be cached as reported below:

- Responses with *Cache-Control: Private*
- Responses with *Cache-Control: No-Cache*
- Responses with *Cache-Control: No-Store*

The response with the following HTTP status codes can be cached: 200, 203 Non-Authoritative Information, 300 Multiple Choices, 301 Moved Permanently, 410 Gone.

Squid provides a web interface along with an access to real-time logs for analysing the cache hit: cache miss ratio and also the number of bytes that are cached. Relying on Figure 2-4and Figure 2-5, two different modes are feasible in order to reach the end-user requesting the content. In both cases, the MEC RAN application will be involved but the traffic shall follow different paths depending upon whether the requested content has to be fetched from the web or it is already present in the cache. The two modes are briefly explained below.

- *Pass-through mode*: If the user requested content is not available in the cache, Squid can modify the data traffic and passes it back to the original Packet Data Network (PDN) connection (i.e. radio access bearer).

- *End-point mode*: If the requested content from the user is already cached, the data traffic is terminated by Squid itself.

Currently, a development work is on-going on implementing the caching functionality as a virtualised function using dockers. Dockers are indeed lightweight VNFs if compared to typical virtual machines such that the image can be deployed, managed and scaled by the orchestrator. Given the flexibility of SESAME and the numerous activities carried out by the project, the use of dockers is done in the research prototype already mentioned above.

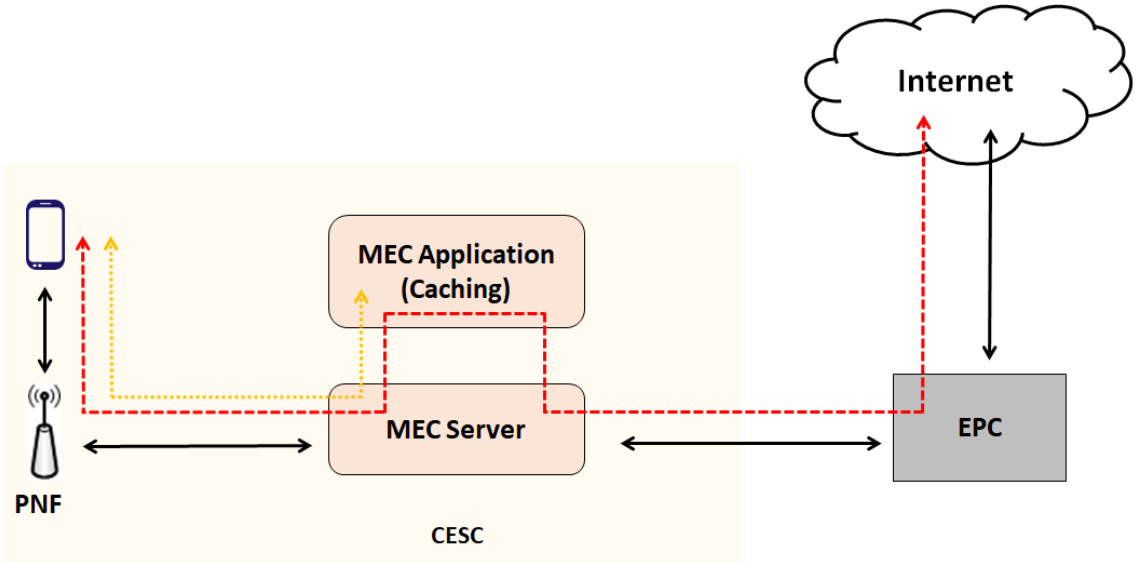

Figure 2-5: Different modes for the traffic going through the caching functionality

The vCache was first tested without virtualisation to perform validation of the whole functionality implemented. This section is used to describe also the functional tests that were conducted during the implementation of caching, and the necessary solutions that were adopted to circumvent problems.

The caching functionality must be configured as a transparent proxy (i.e. UE browser should not be configured with proxy IP address manually) with proper IP table rules for redirecting traffic to the caching MEC RAN application (i.e., Squid in this case) listening port and appropriate network address translation. If there is a cache miss (i.e., the requested content is not present in the cache), since Squid acts as a proxy, it changes the UE IP address to its own IP address. The IP address change would result in an unknown entity at the EPC side due to security reasons. Hence, a proper source-network address translation (SNAT) rule was added to the IP tables of Squid to keep the UE IP unchanged. Similarly, the response behaviour has to be taken care of with an appropriate destination network address translation (DNAT).

Specific test activities were carried out on a x86 platform running Linux operating system (Ubuntu distribution 14.04) for three different test cases:

- Web page caching round-trip time (RTT) measurement: for instance, the RTT experienced at a UE side to request the cached content of Wikipedia homepage takes an RTT of 60 ms, against the 130 ms without caching, hence significantly reducing the delay (less than half).

- Video caching from a local web server: it implies the verification that the full content can be cached in Squid in order to validate the health status of the caching functionality.

As previously mentioned, there is a current on-going effort to unveil the complete vCache functionality in the SESAME system whereby dockers virtualisation technology. The choice of dockers adheres with the principle of a lightweight implementation of virtualised functions if compared to a standard virtual machine.

3. Test Plans

3.1. Functional Tests

3.1.1.Light DC

The testbed environment described in Figure 3-1 was used to integrate and verify several aspects of the Light DC development:

1. Hypervisor, Hardware acceleration layer and Accelerated Virtual networking (VOSYSwitch) integration on the NXP platform. In particular, VOSYSwitch GRE tunnelling support has been developed to integrate the switch in the SESAME testbed.
2. OpenStack integration (the Kilo Release was selected): computing and networking managed by OpenStack controller in case of Hybrid node (Intel/ARM). This has process has required an important amount of effort for the integration of the ARMv8 platform with Kilo.
3. VNFs deployment in compute node based on:
   a. Intel, CPU-only;
   b. Intel, CPU+GPU;
   c. ARM, CPU-only.
4. vTU (Video Transcoding Unit) performance characterization in case of video transcoding service for CESCs.

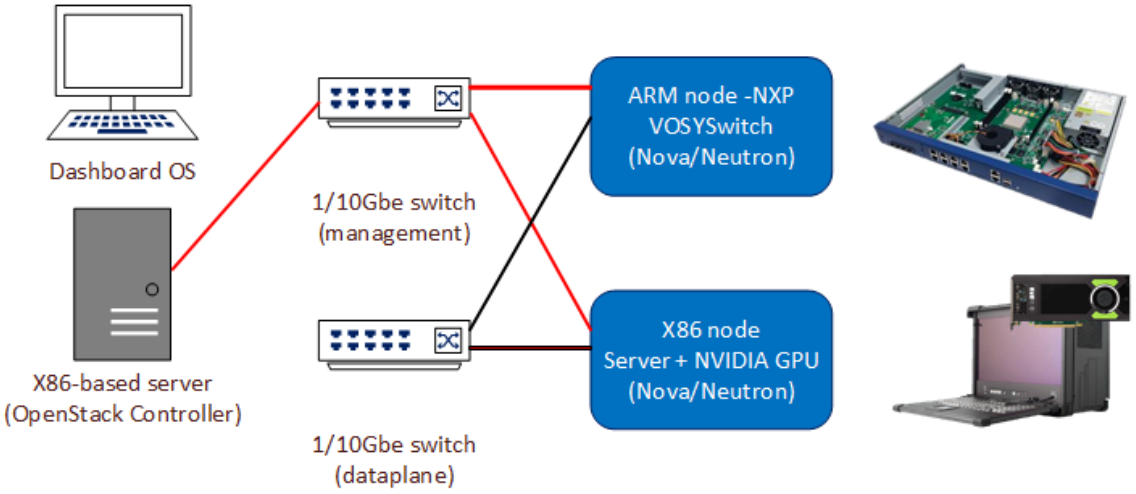

Figure 3-1: Light DC architecture for running functional tests

3.1.2.SC PNF

Unit testing of the SC PNF falls into two broad areas: MOCN support and Performance Management reporting.

3.1.2.1. MOCN Testing

The Multi Operator Core Network (MOCN) support required of the PNF by the SESAME architecture dictates that the following functions be verified:

- That the PNF transmits each PLMN configured in its TR196 [18] *Device.Services.FAPService.{i}.CellConfig.LTE.EPC.PLMNList* parameter in System Information Block 1 (SIB1).
- That the PNF supports up to six such PLMNs.
- That when the *CellReservedForOperatorUse* element is set to TRUE, this is reflected in SIB1 and that normal UEs are then unable to gain admission to the cell.
- That a UE's *Selected PLMN* is carried correctly in the S1 INITIAL UE MESSAGE transmitted to the SC-Common VNF.
- That changes in the list of supported PLMNs are correctly reflected in S1 ENB CONFIGURATION UPDATE messages transmitted to the SC-Common VNF.

Each of these aspects has been verified and is working as planned.

3.1.2.2. PM Testing

The performance management reporting capability of the SC PNF has been extended to provide per-PLMN (and thus per VSCNO) versions of the following counters:

- GTP-U Packets Received per PLMN without Sequence Number,
- GTP-U Packets Received per PLMN with Sequence Number,
- GTP-U Packets Transmitted per PLMN,
- GTP-U Octets Received per PLMN,
- GTP-U Octets Transmitted per PLMN,
- RRC Connection Establishment Success per PLMN,
- Maximum RRC Connection Set-up Time per PLMN,
- Mean RRC Connection Set-up Time per PLMN,
- Maximum RRC Connections per PLMN,
- Mean RRC Connections per PLMN,
- RRC Connection Re-Establishment Attempts per PLMN,
- Failed RRC Connection Re-Establishments per PLMN per Re-Establishment Cause,
- RRC Connection Re-Establishment Success per PLMN.

Validation of these counters has not yet been completed. As a minimum, in order to underpin the functionality required by the KPIs supported by the PoC, the following counters will be verified:

- GTP-U Octets Received per PLMN,
- GTP-U Octets Transmitted per PLMN.

3.1.3.EPC

In addition to the functional tests listed above on the S1 interface to the radio components (SC-VNFs), we will check the integrity of the registration procedures (connection success with PLMN ID), data connectivity and traffic, and session counters.

Basic session S1 establishment testing has been completed but more complicated sequences such as UE call establishment, handover and release remain to be tested.

3.1.4.SC-Common VNF

In addition to correct functionality with respect to the routing of S1 traffic with the appropriate SC-VNFs, testing of the SC-Common VNF will include the following test cases:

- That the configured maximum number of UEs allowed on the PNF is honoured and that a UE attempting to make a call that would exceed the configured limit is rejected.
- That the configured maximum uplink and downlink throughputs are honoured and that an attempt to establish a GBR bearer that would exceed the configured limit is rejected.
- That a change to the configuration of the SC-Common VNF has more-or-less immediate effect and subsequent tests demonstrate the new limit.

To date, testing has focused on end-to-end connectivity and the above capacity related tests have not been performed.

3.1.5.SC VNF

In addition to correct functionality with respect to the routing of S1 traffic to the appropriate EPC, testing of the SC VNF will include the following test cases:

- That the configured maximum uplink and downlink throughputs are honoured and that an attempt transmit more traffic than allowed by the provisioned network slice results in the dropping of packets by the SC VNG.
- That a change to the configuration of the SC VNF has more-or-less immediate effect and subsequent tests demonstrate the new limit.
- That service chaining can be enabled and disabled by configuration management.
- That, when enabled by configuration management, user plane traffic is passed into the service chain by de-encapsulating GTP-U packets and presenting the IP packet payload to the SDN on the configured Ethernet interface.

3.1.6.Service Chain VNFs

3.1.6.1. vDpi

The test plan for vDpi has not yet been finalised. It will be produced prior to Deliverable D7.4.

3.1.6.2. vTU

Test on x86 platform

The x86 version of the vTU runs on a x86_64 multicore server with support to virtualisation (Intel® Virtualisation Technology, Intel® VT) and which provides the PCI pass-through[51] feature for guaranteeing GPU acceleration on the PCIe bus.

The operating system chosen for integrating the NVIDIA CUDA toolkit (release 7.5[52]), needed for managing the GPU M4000, is Linux (kernel version 3.10, CentOS 7[53] distribution).

[51] For further information see: *https://en.wikipedia.org/wiki/X86_virtualization*

The vTU image developed for x86 is in the "qcow2" format[54], OpenStack compatible. It was tested and validated on the testbed (see Section 3.1.1) in two different scenarios:

- CPU-only.
- Use of HW acceleration via GPU.

Test on ARMv8 platform (NXP)

The vTU qcow2 ARM image needs the UEFI[55] boot loader in order to be launched by KVM, but UEFI boot is not managed by the OpenStack Kilo release. To overcome this issue three different images were built:

1. Kernel image[56];
2. Initrd image[57];
3. A raw image with Linux file system.

The kernel image is based on *AltArch* CentOS 7.3[58] for ARM64 (kernel 4.5.0). This choice would imply the porting of audio and video codecs inside the distribution, as they are not available natively. At the end, the chosen solution makes use of a container that includes the whole FFmpeg[59] package (needed by the vTU).

3.1.6.3. vFirewall

The test plan for vFirewall has not yet been finalised. It will be produced prior to Deliverable D7.4.

3.1.6.4. vVideo Analytics

No specific test plan for vVideo Analytics is believed to be necessary above and beyond the testing already performed and described in Section 2.7.5.

3.1.6.5. vCache

The vCache was first tested without virtualization to perform validation of the whole functionality implemented. This section is used to describe also the functional tests that were conducted during the implementation of caching, and the necessary solutions that were adopted to circumvent problems.

The caching functionality must be configured as a transparent proxy (i.e. UE browser should not be configured with proxy IP address manually) with proper IP table rules for redirecting traffic to

---

[52] For more details see: *https://developer.nvidia.com/cuda-75-downloads-archive*
[53] See: *https://www.centos.org/*
[54] For more details see: *http://git.qemu.org/?p=qemu.git;a=blob;f=docs/specs/qcow2.txt*
[55] See: *https://en.wikipedia.org/wiki/Unified_Extensible_Firmware_Interface*
[56] For more details also see, *inter-alia*: *https://en.wikipedia.org/wiki/Kernel_(image_processing)*
[57] For more details also see, *among others*: *https://en.wikipedia.org/wiki/Initial_ramdisk*
[58] Download from: *http://mirror.centos.org/altarch/7/isos/aarch64/*
[59] For more details see: *https://www.ffmpeg.org/*

the caching MEC RAN application (i.e., Squid in this case) listening port and appropriate network address translation. If there is a cache miss (i.e., the requested content is not present in the cache), since Squid acts as a proxy, it changes the UE IP address to its own IP address. The IP address change would result in an unknown entity at the EPC side due to security reasons. Hence, a proper source-network address translation (SNAT) rule was added to the IP tables of Squid to keep the UE IP unchanged. Similarly, the response behaviour has to be taken care of with an appropriate destination network address translation (DNAT).

Specific test activities were carried out on a x86 platform running Linux operating system (Ubuntu distribution 14.04) for three different test cases:

- Web page caching round-trip time (RTT) measurement: for instance, the RTT experienced at a UE side to request the cached content of Wikipedia homepage takes an RTT of 60 ms, against the 130 ms without caching, hence significantly reducing the delay (less than half).
- Video caching from a local web server: it implies the verification that the full content can be cached in Squid in order to validate the health status of the caching functionality.

As previously mentioned, there is a current on-going effort to unveil the complete vCache functionality in the SESAME system whereby dockers virtualisation technology. The choice of dockers adheres with the principle of a lightweight implementation of virtualised functions if compared to a standard virtual machine.

3.1.7.End-to-End System

As a minimum, the following will be performed to ensure correct end-to-end operation of the system:

- That two or more virtual cells (with different PLMNs) can be hosted on a CESC and that the traffic for each virtual cell may be routed to different EPCs.
- That UEs with SIMs belonging to each of the PLMNs hosted by the CESC can access it and that their control plane and user plane traffic is routed to the correct EPC.
- That UEs with a PLMN **not hosted** by the CESC are unable to access it.
- That one or more pairs of UEs, where both UEs in a pair share the same PLMN hosted by the CESC, can make calls to each other.

3.2. Performance Tests

3.2.1.SC PNF

The SC PNF as provided by the ip.access E40 LTE Access Point is largely unchanged with respect to the commercial product upon which it is based. Performance characterisation of this product has been undertaken by ip.access with the following objectives:

- To establish the limits of the PNF hardware and, therefore, set expectations for the maximum throughput that might be achieved by the end-to-end system for a given mix of users,
- To establish a performance target for the end-to-end system. Ideally, the CESC should be able to exploit the maximum capacity of the PNF.

Broadly speaking, the throughput of the PNF is at its maximum when serving a single connected UE and falls off at a rate of approximately 10% for each doubling of connected UEs. Thus, *for example*, a PNF with two UEs connected exhibits a throughput approximately 10% lower than a PNF with a single UE. A PNF with four UEs connected exhibits a throughput that is approximately

10% lower than one with two connected UEs. For the ip.access E40 used in the SESAME PoC, the maximum, single user uplink and downlink throughput when operating with a 20MHz bandwidth is 25 Mbps and 125 Mbps, *respectively*.

3.2.1.SC-Common VNF

3.2.1.1. Control Plane Load

The load presented by the SC-Common VNF should be relatively light, comprising of the associated S1 signalling traffic for a maximum of five[60] SC VNFs supporting a total of 16 users[61]. The CPU load of the SC-Common VNF will be measured and extrapolated to determine the approximate maximum load. It is anticipated that, even at maximum load, this load will be relatively light allowing the CPU allocation of SC-Common VNF to be set at a modest level, thus allowing a greater share of resources to other VNFs.

3.2.1.2. User Plane Load

The SC-Common VNF is bypassed by user plane traffic, which flows between the PNF and each SC VNF. Thus, there is no associated load.

3.2.2.SC VNF

3.2.2.1. Control Plane Load

The control plane load presented by an SC VNF instance will typically not exceed the load presented by the SC-Common VNF and, depending upon the number of virtual cells hosted by the CESC and their relative network slice sizes, may be considerably less. Performance tests will confirm that this is the case.

3.2.2.2. User Plane Load

User plane traffic will provide the most significant contribution to the load of the SC VNF. It will be at its maximum when there is: (a) a single virtual cell provisioned on a CESC with an unconstrained network slice, *and*; (b) when the virtual cell has a single active UE using all of the available bandwidth. The load presented by the SC VNF will be measured under the following circumstances:

- Single active UE, Service Chaining Disabled, Downlink saturated,
- Single active UE, Service Chaining Disabled, Uplink saturated,
- Single active UE, Service Chaining Disabled, Uplink and Downlink saturated,
- Single active UE, Service Chaining Enabled, Downlink saturated,
- Single active UE, Service Chaining Enabled, Uplink saturated,
- Single active UE, Service Chaining Enabled, Uplink and Downlink saturated.

These measurements will enable to maximum load presented by an SC VNF and the ability of the CESC to support it to be assessed.

---

[60] In the current SESAME architecture, one PLMN is reserved for the SCNO. Thus, with a total of six PLMNs provided by MOCN, only five are available to VSCNOs.

[61] This is the maximum number of active users supported by the E40 hardware and is representative of most small cells, which typically support between 8 and 32 users.

4. Conclusions

This report demonstrates that the integration and testing of the Cloud Enabled Small Cell (CESC) Prototype and its component parts is progressing largely according to plan. Whilst some end-to-end testing has been completed, to date, the majority of the testing has been unit-oriented with each component being tested largely in isolation. In addition, many of the components have yet to be tested on the Light DC platform itself.

This testing will continue during the course of Work Package 7 with greater and greater degree of end-to-end integration and with an associated focus on end-to-end functionality.

5. References

[1] Deliverable D2.2: "*Overall System Architecture and Interfaces*", H2020 SESAME Project, 2016 (March). Available at: *http://www.sesame-h2020-5g-ppp.eu/Deliverables.aspx*

[2] Deliverable D2.3: "*Specification of the CESC Components – First Iteration*", H2020 SESAME project, 2016 (March).

[3] Deliverable D3.1: "*CESC Prototype design specifications and initial studies on Self-X and virtualization aspects*," H2020 SESAME project, 2016 (June). Available at: *http://www.sesame-h2020-5g-ppp.eu/Deliverables.aspx*

[4] *E40 nanoLTE, 4G Enterprise Access Point Data Sheet*, Available at: *http://www.ipaccess.com/uploads/wysiwyg_editor/files/2017/E40-Datasheet-v1.0.pdf* ip.access Ltd., 2017.

[5] Oracle Corporation, "*Virtual Box 5.1*" [Online]. Available at: *https://www.virtualbox.org/*.

[6] Open Air Interface. Available at: *http://www.openairinterface.org/*.

[7] http://www.ntop.org, "nDPI, Open and Extensible LGPLv3 Deep Packet Inspection Library," [Online]. Available AT: *http://www.ntop.org/products/ndpi/*.

[8] The 3rd Generation Partnership Project (3GPP) (2017, January): 3GPP TS 36.413 V9.5.0: *Technical Specification Group Radio Access Network; Evolved Universal Terrestrial Radio Access Network (E-UTRAN); S1 Application Protocol (S1AP) Specification*". 3GPP.

[9] "FFmpeg [Online] Available at: *https://www.ffmpeg.org/," 2017.*

[10] McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G/. Peterson, L., Rexford, J., Shenker, S., and Turner, J. (2008, March): Openflow: Enabling innovation in campus networks", *SIGCOMM Comput. Commun,* **38**(2), pp.69-74.

[11] Comi, P., Secondo-Crosta, P. Beccari, M., et *al.* (2016): "Hardware-accelerated high-resolution video coding in Virtual Network Functions". In *Proceedings of the European Conference on Networks and Communications 2016 (EuCNC-2016),* pp.32-36, Athens, Greece, *June 27-30, 2016*.

[12] Paglierani, P., Grossi, G., Pedersini, F., and Petrini, A. (2016): "GPU-based VP8 encoding: Performance in native and virtualized environments". In *Proceedings of the International Conference on Telecommunications and Multimedia 2016 (TEMU-2016)*, pp.1-5, Heraklion, Greece, *July 25-27, 2016*.

[13] OpenCV.org, "*OpenCV API Reference*" [Online]. Available at: *http://docs.opencv.org/3.0-beta/modules/refman.html* .

[14] BerkeleyVision.org, [Online]. Available at: *http://caffe.berkeleyvision.org/*.

[15] Stackoverflow.com, "*What is the shortest perceivable application response delay?*" [Online]. Available at: *https://stackoverflow.com/questions/536300/what-is-the-shortest-perceivable-application-response-delay* .

[16] Betzler, A. (editor) (2017, June): "*Framework of a Distributed Network Management System Capable to Host and Run Self-X Functionalities*", Deliverable D3.3 of 5G-PPP SESAME Project.

[17] Whitehead, A. (editor) (2017, June): "*CESC Small Cell Prototype and PoC*", Deliverable D3.4 of 5G-PPP SESAME Project.

[18] Broadband Forum (BF) (2011, November): TR-196: Femto Access Point Service Data Modell, Issue 2". [Online]. Available at: *https://www.broadband-forum.org/technical/download/TR-196.pdf* .