Virtualising and Orchestrating a 5G Evolved Packet Core Network

David Lake, Gerry Foster, Serdar Vural, Yogaratnam Rahulan, Bong-Hwan Oh, Ning Wang, and Rahim Tafazolli
5G Innovation Centre, Institute for Communication Systems, University of Surrey, Guildford, UK,
{d.lake, g.foster, s.vural, y.rahulan, b.oh, n.wang, r.tafazolli}@surrey.ac.uk

Abstract— **In this paper, the design, construction, and testing of a fully-functional virtualised mobile core network is outlined. Lessons learned and recommendations for future improvements are provided. The presented work uses open-source software for virtual network function infrastructure control (OpenStack), network flow programming (OpenDaylight), and network orchestration (OpenBaton) to virtualise a commercial software evolved packet core solution deployed on common off-the-shelf hardware. The findings presented in this paper prove the concept of function virtualisation for mobile networks, and paves the way towards future mobile core network function flexibility as required for 5G networks. The paper provides researchers and network operators with first-hand experience to help build similar virtual mobile network infrastructures, and highlights the challenges to tackle and the issues to address to harness the power of virtualisation in 5G networks.**

Keywords—5G; testbed; next generation networks; network function virtualisation; software defined networking, management and orchestration.

I. INTRODUCTION

Essential for Fifth Generation (5G) multi-access mobile networks, Network function virtualisation [1] and orchestration enable automatic management and orchestration in a flexible manner [2]. To efficiently manage the total cost of ownership of a mobile network, operators need to be able to rapidly deploy core infrastructure on off-the-shelf, multi-vendor, enterprise blades, which run a variety of 5G Virtualised Network Functions (VNF).

The University of Surrey setup the joint industry/academic 5G Innovation Centre (5GIC) in 2015. Campus-wide 5G mobile network infrastructure comprises an ultra-dense outdoor network, indoor coverage of two floors of the 5GIC building and a new core network architecture, designed around 5G use-cases [2], developed by the 3rd Generation Partnership Project (3GPP), the Next Generation Mobile Networks (NGMN) alliance [3], International Telecommunication Union Telecommunications standardisation sector (ITU-T), and the 5G infrastructure Public Private Partnership (5G-PPP).

Current 4G Evolved Packet Core (EPC), using for-purpose hardware elements, is difficult to scale to meet service demands. The 5GIC has developed techniques to virtualise and orchestrate the core of the 5G mobile network. Orchestration providing a single management view of the entire environment has the potential to reduce operational costs compared to today's multiple purpose-specific management; virtualisation of network functions and compute resources allows optimum placement of components compared to the current fixed network layout.

A Flat Distributed Cloud (FDC) architecture is presented; split control/user planes allow traffic to be dynamically directed to separate network functions as demonstrated on the Campus-wide, near real-world LTE network.

This paper details the experience and lessons learned in virtualising and orchestrating the experimental 5G mobile core at 5GIC (Section II), presenting initial test results of the impact of virtualisation. Network core virtualisation is explained in Section III. Section IV presents proof-of-concept performance results; Section V discusses the lessons learnt; key achievements and recommendations in Section VI, Section VII concludes the paper.

II. 5G MOBILE FLAT DISTRIBUTED CLOUD ARCHITECTURE

The Flat Distributed Cloud (FDC) which forms the basis of the RAN and the 5G multi-access mobile network at 5GIC covering 50m^2 per floor indoors and the entire campus. The core comprises a software-based commercial EPC (Fig. 1) with modifications to address the requirements and use cases of 3GPP TR 23.799. Cluster-based, the FDC envisages distribution of core network functions at multiple *network nodes* - a distributed core is a key tenet of 5G [2].

Consisting of multiple clusters per datacentre, FDC (Fig. 1) contains multiple Cluster Member (CM) network nodes and a single Cluster Controller (CC) network node (iCM is the interface between CM and CC). Each cluster connects to the 5G common evolved distributed Home Subscriber Server system (HSS). The FDC connects to the legacy EPC indicated by the Mobility Management Entity (MME).

Control plane (CP) traffic of each device is handled by the CC (macro-cell) of each cluster to which the device is associated; user plane (UP) traffic is routed through the CC or the CM (small cells), depending on user profile, e.g. mobility, type of service. In a high-mobility case, UEs are handled by CCs to minimise handover signalling; CMs serve the UP of stationary or low-mobility UEs. Fig. 1 shows UE 1 "UP-anchored" at the CC - UP traffic goes through the CC. UE 2 is UP-anchored at a CM, with CP on the CC of the cluster it resides in. The CC is the context-aware control plane entity.

CC can handle UP and CP functions but CM can perform only UP functions; the distributed architecture re-arranges core

This project is partially funded by the 5G Innovation Centre (5GIC), and EU SoftFIRE project, Grant Number 687860.

network functions into a Control Plane Node (CPN) and a User Plane Node (UPN). Fig. 2 demonstrates these logical node types, which provides 5G Control and User Plane Separation (CUPS)[4] for traffic re-direction and moveable breakout.

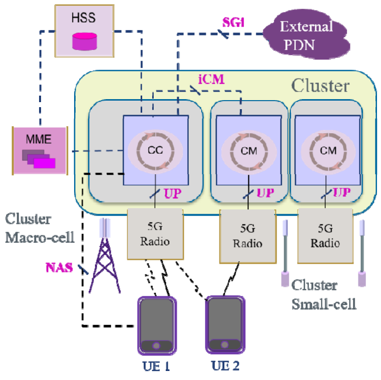

Fig. 1. The Flat Distributed Cloud (FDC) network architecture of the 5G network at 5GIC, University of Surrey.

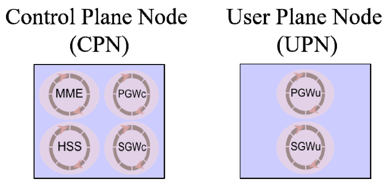

Fig. 2. The two logical nodes of FDC in 5G: Control Plane Node (CPN) and User Plane Node (UPN).

CPN comprises the LTE-A control plane component functions, part of the MME functions (e.g. mobility context of a user within a cluster) and HSS (e.g. active user profile and context management within a cluster) and control plane functions of PGW and SGW, (PGWc and SGWc), whilst the UP functions of the PGW and SGW (PGWu, SGWu), are now part of the UPN. FDC integrates these to a single component, the Packet Processing Entity (PPE), collapsing S1 and eliminating one GTP tunnel. A CC comprises a UPN and a CPN; a CM comprises a single UPN only. CC and CM use initial programming and then context learning (Fig. 3).

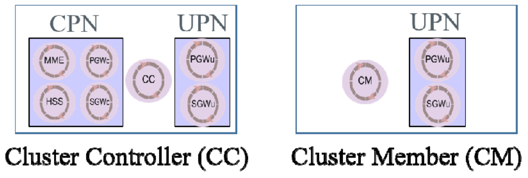

Fig. 3. Cluster Control encapsulates a CPN and a UPN, whereas Cluster Member has only UPN.

The FDC distributes core functions across a number of network nodes, decentralising the core, providing control/user plan separation and moving services closer to the edge.

III. Virtualisation of the Network Core

Each element of the FDC is hosted within a Virtual Machine (VM) instantiated on the physical environment. Ephemeral state of the virtual environment (e.g. a virtual machine which may move between physical hosts) may be hidden from the network allowing operational flexibility required in placement of the network functions.

Section A describes the hardware and the software used; Section B outlines the integration of software components; Section C details the logical creation of the EPC Network Functions; Section D describes network slicing. Finally, Section E explains the use of Software Defined Networking (SDN).

A. Network virtualisation hardware and software

The environment consists of the infrastructure, server hardware, and virtualisation software.

1) Network Function Virtualisation Infrastructrue (NFVI): a platform capable of hosting the 5G functions built to provide "Production/Pre-Production" capabilities: as i) a Live system where limited changes are allowed to allow for a high level of availability (e.g. supporting already deployed technologies), ii) Test system development/validated prior to being moved to the Live system allowing new networking technologies to be tested in a realistic network environment without affecting the existing production networks.Software versions and architectures of the two systems are identical.

2) Infratructure Nodes: three off-the-shelf x86-based servers running CentOS 7 + latest patches; Compute, Admin, and Resource nodes. i) Compute Node: 90 CPUs/252GB RAM (Live), 32 CPUs/32 GB RAM (Test). Virtual machines are instantiated on these machines. In the FDC architecture, the Compute Node is where the CC and CM elements and additional virtual functions (e.g. HSS, MME) reside. Multiple Compute Nodes logically appear as a single set of resources. ii) Resource Node: 16 CPUs/32 GB RAM (Live and Test) for file-store for the source images used to instantiate virtual machines (via an Apache HTTP), and network flow controller; iii) Admin Node: 16 CPUs/32 GB RAM (Live and Test). infrastructure controller.

Each NFVI uses two logically separated networks: (1) "Admin" network (Eth0 interface on the Compute node) connects Compute, Resource, and Admin nodes and management network. (2) "Demo" network (Eth2 interface on the Compute node) provides VM to radio access components connectivity.

3) NFVI hosting software: The 5GIC testbed is virtualised using off-the-shelf open-source software: OpenStack [6] (version Liberty) infrastructure controller deployed using the DevStack with modified configuration/installation scripts; OpenDaylight [5] (version Beryllium) Software-defined networking (SDN) controller integrated over Neutron interface with OpenStack for Layer 3 routing; OpenBaton [7] (version 3.0) open-source network function orchestrator compliant to European Telecommunications Standard Institute (ETSI) Management and Organisation (MANO) 1.0.0.

B. *Building a network virtualisation environment*

Additional development in combining different pieces of software to become one complete system that controls, programs, and manages the virtualised environment is required:

1) Integration of OpenStack and OpenDaylight: VM traffic follows forwarding rules installed at an Open Virtual Switch (OVS) instance associated with Compute Eth2. Programming of OVS/Eth2 is subjugated to OpenDaylight which is under control of OpenStack via Neutron API removing a single-point of failure and a potential traffic bottleneck (Section E).

2) Enabling external reachability of virtual machines: VMs instantiate via Openstack's "Horizon" GUI are allocated internal IP addresses which have outbound-only NAT access. Floating, inbound-IP addresses require a mapping to be built across OVS which are setup by OpenStack/OpenDaylight but maintained in OVS.

C. *Creation of virtualised 5G mobile core network services*

Four elements are required to create a virtualised network function:

1) Virtual Network Function (VNF): Network functions and components re-packaged for the NFVI ("softwarisastion"). 5G FDC comprises Cluster Controller, Cluster Member, and Packet Processing Entity VNFs.

2) Virtual Network Function Descriptor: describes construction of a VNF from a systems level; installation and configuration scripts are used to install and configure the VNF on the VM.

3) Network Service Descriptor: provides specific network-level configuration for the operation of the VNF. It describes how different VNFs interact and the dependencies between them. FDC requires: (1) EPC_CPN: Common Control Plane Node of the Evolved Packet Core, (2) EPC_UPN_CC: User Plane Node of a Cluster Controller in an Evolved Packet Core, and (3) EPC_UPN_CM: User Plane Node of a Cluster Member in an Evolved Packet Core. Additional scripting instantiates the VNF from the VNFD and NSD.

4) Configuration of VNFs for IP reachability

Based on a proprietary softwarised EPC, the VNFs are unaware of the routable, floating IP address allocated to it; this address with a /32 mask is configured as an interface on the VM and scripting developed to make the VNF "believe" it has the external address.

D. *Dynamic and APN-based network slicing*

Described by an NSD (Section I.A.1) multiple slices, controlled by OpenBaton and selected by APN, are supported: i) EPC_CPN - virtualised control plane node in a cluster; ii) EPC_UPN_CC - user plane of a cluster controller; iii) EPC-UPN_CM - user plane of a cluster member.

E. *Software Defined Networking (SDN)*

Used to subvert OVS traffic and program switches between the RAN and the core, the 5G network has integrated NFV and SDN [8], to provide a fully virtualised and programmable testbed infrastructure.

IV. Performance Evaluation

This section details the performance of the virtualised mobile packet core, as compared to a non-virtualised one.

A. *Testbed segments*

The 5GIC testbed has been segmented into four segments: *Segment 1*: (Hardware EPC) Reference LTE core network implementation on off-the-shelf EPC hardware. *Segment 2*: (Software 5G FDC) LTE core network implementation using off-the-shelf core network software deployed on off-the-shelf server blades - a "softwarised" EPC. *Segment 3*: (Virtualised 5G FDC – Live, see Section III.A.1) LTE core network implementation using off-the-shelf core network software deployed on virtual machines as per the 5G FDC. VMs are instantiated using the OpenBaton/OpenStack. *Segment 4*: (Virtualised EPC - Test) Similar to Segment 3, the Test System runs here.

Segments 1 and 2 are used as the benchmarks comparison with segments 3 and 4. The performance analysis of softwarisation alone (Segment 1 vs Segment 2) of EPC did not demonstrate significant and consistent differences. The performance of Segment 2 vs Segment 3 is of more interest here given the focus of the study on the impact of virtualisation on an EPC.

B. *Radio Access Network*

The virtualised core running on server equipment is connected to the radio access network (RAN) deployed in the University of Surrey campus (outdoor) as well as the 5GIC building (indoor).

C. *Set of Key Performance Indicators (KPI) and Results*

To assess comparative improvement or degradation due to virtualisation, a set of KPIs are selected and results shown in Table I. Averages of repetitive measurements to cancel out long term radio effects were used. Measurements were taken using Huawei Honor, Huawei P7, and Nexus 6P mobiles. Each measurement set was performed on both Segment 2 and Segment 3, using the RantCELL mobile application. All intranet measurements were taken using servers that reside in the cellular network. All Internet measurements were taken by communicating with the same Internet server target in each measurement. The broadband test application is Ookla^{TM}.

Average values are obtained across femto (FDD LTE-A) and pico (pico-RRU) cells to ensure a valid scope[1].

[1] The LTE-A(TDD) RAN has been setup for asymmetric maximum downlink capacity, whereas for FDD the uplink and downlink allocations are static and symmetric.

D. *KPI Analysis*

Results demonstrate that virtualisation of a software mobile core results in consistent degradation in downlink throughput of about ~7-9% and degradation in time delay of ~ 7%. Impact on the Attach procedure[2] is low: ~2.3% average increase in time delay. User devices will have a similar experience when attaching to the network when served by a virtual core.

TABLE I. IMPLEMENTATION PERFORMANCE KPIs

KPI	Target Network	Nominal Value	Mean (%)	Max (%)
Attach Duration (ms) (from IDLE mode)	Flat distributed cloud	500 to 2,000	2.29	5
Round-trip time (ms)	Intranet	20-30	6.7	17.5
Round-trip time (ms)	Internet	20-40	2.22	5.8
Uplink Throughput (Mbit/s)	Intranet	10 (TDD) 20 (FDD)	-1.21	-2.68
Downlink Throughput (Mbit/s)	Intranet	100 (TDD) 60 (FDD)	-7.31	-14.88
Uplink Throughput (Mbit/s)	Internet	10 (TDD) 20 (FDD)	-2.22	-5.8
Downlink Throughput (Mbit/s)	Internet	100 (TDD) 60 (FDD)	-8.66	-10.76
File download 55MB (s)	Intranet	10	5.25	21.9

The main benefit observed from a virtualised 5G FDC is that the necessary 5G-FDC functions can be deployed in less than 10 minutes on average per package (deployed from the orchestrator). This is a major improvement over traditional core network deployment latency, even for softwarised cores. The impacts of virtualisation are measured as low for most key performance parameters, except downlink throughput.

V. LESSONS LEARNED

Valuable insight into the near real-world performance of a virtualised mobile core has been gained and are discussed in this section.

A. *Network Function Virtualisation*

Issues arose during the design and construction of both the NFVI and VNFs and the instantiation of VNFs:

1) Internal and External Address Mapping and the Subversion of OVS Prior to OpenStack *Liberty:* VMs were allocated an IP address in a private subnet which only exists in the OpenStack addressing space requiring traffic to/from the public network to be VXLAN encapsulated and sent via a Network Node – this node was a performance bottleneck and a single point of failure. In OpenStack Liberty, this functionality has been partially distributed to virtual router functions running on the Compute node; Source NAT (SNAT) is still centred on the network node in OpenStack's Neutron interface so that software NAT from an external IP to an OpenStack internal IP (and vice versa) is performed at the Admin node requiring user plane traffic to be routed from the Compute node to the Admin node. This adversely affects user plane throughput. OpenDaylight is used to subvert the operation of the NAT function to the OpenDaylight SDN controller; local OVS now performs routing and NAT functions. OVS routes need to be programmed at the Compute node to ensure user-plane traffic is forwarded to the correct port using OpenDaylight.

2) External IP addresses unknown by virtual machines: This subversion of OVS leaves a problem in that the VMs have no knowledge of their external IP address. To work-around this issue, a /32 loopback address assigned the external IP address is created and the VNF bound to it. However this requires complex scripting to a) determine and assign the address, b) configure the VNF c) cope with changes to the address.

3) IP Routing: OpenDaylight interacts via Neutron to OpenStack to perform routing on OVS, but it does not have the ability to generate an ARP. The MAC address of the default gateway, the virtaul router redundancy (VRRP) address and any changes must be manually configured. Additionally, this solution cannot handle multiple next-hope routers.

4) OVS performance issues: The NAT mapping places OVS in the path of all user-plane traffic adding extra latencey. The exact cause is being investigated, as is the potetial impact on OVS performance due to associated issues (e.g. loading of the compute server). It must be determined whether OVS is a bottleneck or whether sufficient isolation exists between OVS and other sotware. [9] discuss methods by which OVS performance may be improved; however this assumes the VNF is the terminating application, not a forwarding path. New optimisation solutions effective across every data-forwarding element are required.

5) Machine Isolation: OpenStack allows defined amounts of compute, memory, and storage resources to be assigned to each virtual machine, but it also allows over-provisioning of the system. Core networks rely on maintaining service to agreed levels; further work is required to determine whether sufficient isolation exists between parallel running.

6) Address Allocation/Determination: MANO today focuses on instantiation of VNFs from associated VNFD and NSD; modification of the running configuration is complex. New solutions need to be found for a deeper-level of integration between the OpenStack Nova and MANO. At the infrastructure controller level (e.g. OpenStack), there is a need to support deterministic management of VNF's IP addresses.

7) Stability/Deployment Concerns: Achieving compatibility between versions of OpenStack, OpenDaylight and OpenBaton is complex and finding a stable-working environment difficult with integration experience limited with outdated hampered by invalid documentation and poor support.

[2] Attach procedure: from NAS IDLE Mode to CONNECTED Mode.

Ascertaining the correct configuration for the Neutron Layer 3 connection to OpenDaylight was only successful through the use of reverse-engineering with tools such as Wireshark. Devstack deployment allows one-time provisioning but does not always successfully restart the instances after reboot. Configuration of floating IPs and initiation of the virtual router requires specific configuration file ("local.conf") and command line configuration.

8) Versioning: The open-source community appears to be divergent on base operating system builds for both OpenStack and OpenDaylight. Some versions of OVS suffer from poor performance. A default set of installation options need to be provided to implementers so that successful production-grade solutions can be realised.

B. Flow programming at SDN switches:

Traffic between the BBUs and FDC is directed by switches under control of OpenDaylight which requires the following network support: (i) an ARP responder flow for each subnet connected to a switch's ports so that a switch can act as a default gateway, (ii) flood for default flows, (iii) modification of destination and source MAC addresses of IP frames. Not all switches support ARP generation; code needs to be developed for the controller to handle ARP generation/response [10]. SDN capabilities are limited by differing support for OpenFlow across hardware switches. SDN requires further penetration into the switching hardware equipment before it can easily be adapted by network infrastructures [11].

VI. Achievements and Recommendations

The three major achievements of this work are:

1) Virtualised end-to-end cellular 5G core network system: The authors consider that this is one of the first demonstrations of a fully virtualised 5G cellular core network operating end-to-end between a mobile user equipment and the Internet.

2) Rapid orchestrated cellular core deployment: The facets of OpenStack, the ability to scale, snapshot, and move mobile core functions, is applied to the FDC allowing a greater degree of flexibility compared to hardware-realised core networks, decreasing deployment time from weeks to around 5-10 minutes.

3) Rapid orchestrated slicing deployment: 5G network slicing, the ability to direct traffic to segmented forwarding planes is able to be constructed within the FDC in less than 10 minutes per slice.

In terms of recommendations in network virtualisation to the of benefit to mobile networks:

Standardisation in controller/orchestrator protocol.
Mobility anchor-point issues need further investigating.
Support, maintenance and service-levels in the Free Open Source Software (FOSS) model need to be revisited to support Service Providers.
Further identification and remediation of the points of degradation due to virtualisation is required.
Research into optimisation of SDN for OVS for mobile UP and improvements in OpenFlow programmability.
Orchestration must build/manage end-to-end services based on service telemetry (loading) and be contextually-aware.

VII. Conclusion

This paper presents one of the first realisations of a fully-virtualised 5G mobile core network; the core network functions are software instances running on common off-the-shelf server blades. This core is based on a clustered and distributed cloud network architecture, having the key tenets of a 5G mobile networks; control and user plane separation, collapse of GTP tunnels, and distribution of core network functionalities. Building the NFV infrastructure proved to require substantial effort: the FOSS model for NFV, SDN and orchestration components providing insufficient support and documentation for the commercial telecoms market with a number of technical issues. Compared to current hardware implementations, setting up new core functions can be achieved in less than 10 mins. Collective effort to address and improve some of the technical challenges is required to speed adoption of this promising technology.

Acknowledgement

The authors would like to acknowledge the support of the University of Surrey's 5G Innovation Centre (5GIC) (http://www.surrey.ac.uk/5gic) members for this work.

References

[1] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, "Network Function Virtualization: State-of-the-art and Research Challenges", IEEE Communications Surveys and Tutorials, vol 18, no 1, pp 236-262, 2016.

[2] 3GPP, "TR 23.799 v1.0.2, Study on architecture for next generation system", September 2016.

[3] M. Iwamura, "NGMN view on 5G architecture", IEEE VTC, May 2015.

[4] 3GPP, "TR 23.714 v0.4.0, Study on control and user plane separation of EPC nodes", April 2016.

[5] Linux Foundation, "OpenDaylight: open source software-defined networking platform", https://www.opendaylight.org/, accessed 1 December 2016.

[6] OpenStack Project, "OpenStack open source software for creating private and public clouds", http://www.openstack.org/, accessed 1 December 2016.

[7] Fraunhofer FOKUS, "OpenBaton: an open source reference implementation of the ETSI network function virtualization MANO specification", http://openbaton.github.io/index.html, accessed 1 December 2016.

[8] X An, W. Kiess, J. Varga, J. Prade, H-J. Morper, and K. Hoffmann "SDN-based vs software-only EPC gateways: A cost analysis", IEEE NetSoft, June 2016.

[9] P.Veitch, T. Long, and P. Hutchison, ""NFV Performance Optimization for Virtualized Customer Premises Equipment", https://software.intel.com/en-us/articles/nfv-performance-optimization-for-vcpe, accessed 6 January 2017.

[10] R. di Lallo, G. Lospoto, M. Rimondini, and G. Di Battista, "How to handle ARP in software-defined network", IEEE NetSoft June 2016.

[11] R. di Lallo, M. Gradillo, G. Lospoto, C. Pisa, and M. Rimondini, "On the practical applicability of SDN research", IEEE/IFIP Network Operations and Management Symposium, April 2016.