An SDN/NFV Proof-of-Concept Test-Bed for Machine Learning-based Network Management

Wei Jiang^{1,2 *}, Mathias Strufe¹, Michael Gundall¹ and Hans Dieter Schotten^{1,2}

¹German Research Center for Artificial Intelligence (DFKI)
Kaiserslautern, Germany

²Department of Electrical and Computer Engineering
University of Kaiserslautern
Kaiserslautern, Germany

Abstract—Complexity and heterogeneity of the fifth generation (5G) and beyond mobile systems impose a great challenge on current network managing approaches, which are vulnerable, time-consuming and costly. The state-of-the-art research direction in this field is to apply machine learning (ML) techniques to realize intelligent and highly self-organized networking. Unlike the physical layer, theoretical analyses and numerical simulations on the management layer are generally infeasible or not scientifically rigorous enough. Therefore, in this paper, we present a software-defined and virtualized wireless test-bed that is established to evaluate ML-based network management. Based on open-source software and off-the-shelf hardware, this test-bed is easily reproducible, which in turn is hopeful to foster innovative works in this field.

Keywords-5G; Machine Learning; Network Management; NFV; SDN; Tactile Internet; Wireless Test-Bed.

I. INTRODUCTION

A wide variety of revolutionary applications, such as automated driving, virtual and augmented reality, Internet of Things and Tactile Internet [1], impose challenging technical requirements on the fifth generation (5G) mobile system. For example, 5G-based Tactile Internet is required to provide extremely low latency in combination with high availability, reliability and security to satisfy the demand of Industry 4.0. The upcoming 5G system is envisioned to become more flexible, scalable and efficient, whereas the underlying infrastructure is increasingly complicated and heterogeneous. This dilemma inevitably causes a significant challenge from the perspective of network management. Unfortunately, today's manual and semi-automatic managing approaches [2] are already costly, vulnerable, time-consuming and therefore inapplicable to the 5G system.

Network Function Virtualization (NFV) [3] and Software-Defined Networking (SDN) [4] are two cutting-edge networking innovations. By decoupling software from hardware, NFV allows mobile operators to deploy network functions as software instances rather than dedicated appliances. Network virtualization opens many possibilities, such as Network Slicing [5] and Multi-Tenancy [6]. Exploiting the economy of scale in general-purpose high-volume servers, NFV provides a cost-efficient architecture that reduces both equipment costs and power consumption. In SDN, the control functionalities are decoupled from the data plane and logically centralized into a controller. By

abstracting the information of underlying infrastructure and exposing it through the northbound interface, the network is possible to be *programmable*. SDN can simplify network control, improve Quality-of-Service (QoS), and enable flexible and efficient network management. The application of NFV and SDN in the 5G system provides a possibility for exploring advanced network managing approaches.

In 2016, an overwhelming victory of Google AlphaGo [7] versus human champion in the game of Go sparked the passion of exploring Artificial Intelligence (AI). In the field of network management, the research community also started applying Machine Learning (ML) techniques [8]. For example, the EU SELFNET project [9] took advantage of ML techniques to design and implement a Self-Organizing Network (SON) over SDN/NFV-based 5G infrastructure. The ML-based management is able to assist network operators to simplify maintenance and minimize human intervention, which in turn could lower OPEX, improve user experience and shorten time-to-market of new services.

Evaluation, verification and demonstration of the novel approaches are essential steps before the deployment of ML-based network management in practical systems. Unlike the physical layer in wireless systems, theoretical analyses and numerical simulations upon the management layer are generally infeasible or not scientifically rigorous. Therefore, in this paper, we present an SDN/NFV wireless test-bed that is established to evaluate ML-based managing approaches. Taking reproducibility and generalizability into account, this test-bed is deliberately built from open-source software such as OpenStack [10], OpenDaylight [11] and OpenAirInterface [12], off-the-shelf high-volume servers and software-defined radio modules. Proof-of-concept experiments are carried out, where two isolated virtual networks are instantiated and the ML-based managing functionalities of QoS provisioning for video transmission and Distributed Denial-of-Service (DDoS) mitigation are illustrated.

This paper is organized as follows: Section II details the set-up of the wireless test-bed. Section III illustrates the experiments. Finally, Section IV concludes this paper.

II. THE TEST-BED SETUP

This section first describes hardware components of the test-bed, where several general-purpose high-volume servers, software-defined radio modules and OpenFlow-enabled switches are adopted to provide an extensible and flexible hardware platform. With the aid of OpenStack and

OpenDaylight, the physical resources are virtualized so as to instantiate virtual networks on demand and build an SDN/NFV-based architecture. In combination with the radio modules, OpenAirInterface, an open-source software that provides a full set of 3GPP LTE/LTE-A network functions, is integrated into the test-bed to build a realistic wireless scenario.

A. Hardware Components

This test-bed consists mainly of four high-volume servers, two mini-computers, two software-defined radio modules, and three OpenFlow-enabled switches.

- **Servers:**

Four generic-purpose servers act as computing resource of the Cloud Computing platform under the control of an OpenStack controller, which is responsible for instantiating Virtual Machines (VMs). Among them, a server equipped with a special graphical card is dedicated to run ML algorithms in a parallel-computing way with its powerful Graphical Processing Units (GPUs).

- **Mini-Computers:**

In addition to the Cloud Computing supported by the servers, two mini-computers are applied to implement Multi-access Edge Computing (MEC). The computers are deployed close to the radio modules and run the Virtual Network Function (VNF) of LTE eNodeB in order to build a reconfigurable MEC environment.

- **Radio modules:**

To support a realistic radio link, a software-defined radio module, Universal Software Radio Peripheral (USRP), is adopted in this test-bed. It is a fully integrated, single-board platform with continuous frequency coverage from 70MHz to 6GHz enabling experimentation with a wide range of radio signals from FM, TV broadcast, and Wi-Fi to cellular. It combines wideband ADC/DAC providing up to 56MHz of real-time bandwidth and a reprogrammable FPGA, which facilitates the user prototype their applications promptly.

- **Switches:**

Switches are utilized to interconnect all equipment and act also as a gateway to Internet. The test-bed adopts OpenFlow-enabled switches to support SDN functionalities. The SDN controller has a complete view of the overall network. Through the OpenFlow protocol, the controller can add, update and delete entries in flow tables of switches, opening a possibility on advanced network management.

The equipment including their brief specifications and Quantities (QTY) used in this test-bed are listed in Table I. The current set-up is only a basic set of equipment to demonstrate ML-based management over a small-scale SDN/NFV-based network. The test-bed is extensible with the help of open-source software Metal-as-a-Service (MaaS) [13]. While MaaS is not a mandatory requirement for a small-scale test-bed, it is a powerful tool with all its automation, machine configuration utilities and zero touch Operating System (OS) deployment to manage the physical

machines and therefore essential if we want a subsequent extension later on.

Table I. Hardware Configurations

Equipment	QTY	Specification
Server	3	Intel Xeon E5, 2x32GB DDR3
GPU Server	1	Intel Xeon E5, 32GB DDR3, Titan X GPU
Mini PC	2	Intel i5, 16GB DDR3
Switch	3	HP Aruba 2920, 1Gbits/s, OpenFlow 1.3
USRP	2	Ettus B210, RF 70MHz-6GHz, 2x2 MIMO, USB3.0, Xilinx Spartan 6 FPGA

Figure 1. Schematic of hardware configuration in the test-bed.

Fig. 1 illustrates the schematic of interconnected hardware in the test-bed. The Cloud Computing part consists of four servers and three switches. One server acts as the Management (Mgt.) node, where MaaS and the OpenStack controller are installed. The network function of virtualized Evolved Packet Core (EPC) is instantiated there and therefore it serves also as the gateway to the Internet. Two generic-purpose servers, connected via two physical network interfaces with the Management node, are used as computing nodes to deploy VMs. The network functions, such as video and web services, are deployed in these computing nodes. The GPU server is as well directly connected to the Management node. With its build-in TitanX graphic card embedding about 3000 CUDA cores, this server is suitable for ML algorithms using its parallel-computing capability. As shown in Fig. 1, the MEC part consists of two mini-computers. Each mini-computer acts as a virtual eNodeB by running the software-based LTE implementation on the top of a real-time Linux kernel. The virtual eNodeB connects to EPC core network on the Management node via a switch and also connects to the USRP radio module via its USB3.0 port.

B. Software Components

On the top of the hardware infrastructure, we install the open-source software, such as OpenStack, OpenDaylight, and OpenAirInterface, to build an SDN/NFV wireless

platform. The main software components are explained as follows:

- **OpenStack:**

OpenStack is a cloud computing platform, mostly deployed as Infrastructure-as-a-Service (IaaS). In ETSI NFV and OPNFV, OpenStack is selected as the fundamental tool to implement Virtualized Infrastructure Manager (VIM). VIM controls the assignment of virtual computing, storage and networking resources to support VNFs. It mainly consists of several components: **Nova** is the primary computing controller that is used for deploying and managing computer resources and other instances to handle computing tasks. **Neutron** provides the networking capability, which manages the virtual network configuration and IP addresses. It provides flat networks, such as Virtual Local Area Networks (VLANs) and Dynamic Host Configuration Protocol (DHCP). **Horizon** is the dashboard that facilitates the administrator and user access to the provision and deployment of cloud-based resources. **Keystone** provides the identity service. It is the central point of authentication. **Glance** is the image service. Glance provides discovery, registration and delivery services for disk and server image, which can be used as templates when deploying new VM instances. **Ceilometer** is the telemetry service that directly outputs metering and monitoring data.

More components such as storage or orchestration services are provided in OpenStack, but for our test-bed the aforementioned components are sufficient. These services are installed in the Management node, while two servers got configured as dedicated OpenStack computing nodes.

- **OpenDaylight:**

OpenDaylight is an open-source SDN platform that uses open protocols to provide centralized and programmable network control. It implements the SDN controller in Java and is kept within its own Java Virtual Machine (JVM), which means that it is flexibly deployed on hardware and operating systems that support Java. In the SDN architecture, the controller has a centralized view of the overall network and can directly interact with forwarding devices via the southbound interface. OpenFlow, which was developed by Open Networking Foundation (ONF), is the first and most well-known standard that defines the communication protocol in the southbound interface. With OpenFlow, entries in internal flow tables of switches and routers can be added, updated and removed by the controller in order to make the network more agile to real-time traffic fluctuations. OpenDaylight supports both OpenFlow and legacy protocols such as NetConf and OSPF so as that the OpenDaylight controller is compatible to either OpenFlow-enabled forwarding devices or traditional switches. The OpenDaylight controller abstracts the network information and exposes it to an open and standard Application Programming Interface (API) that is called the northbound interface. Through the northbound API, the network become programmable and a wider variety of automation and orchestration applications can be developed and deployed.

- **OpenAirInterface:**

OpenAirInterface is created by EURECOM towards an open experimentation and prototyping wireless platform. The

platform offers an open-source software implementation of 3GPP LTE standard spanning the full protocol stack both in E-UTRAN and EPC. It can be applied to build and customize an LTE eNodeB, a user equipment and a core network on a computer. With the aid of a USRP radio module, OpenAirInterface eNodeB can connect to a commercial UE via a radio link. It provides a rich development environment with a range of built-in tools such as highly realistic emulation modes, soft monitoring and debugging tools, protocol analyzer, performance profiler, and configurable logging system for all layers and channels. In the test-bed, the EPC functionalities consisting of a virtual Mobile Management Entity (MME), Serving/PDN Gateway (S/P-GW) and Home Subscriber Server (HSS) implementation are deployed on the Management node, while the virtual eNodeB is installed in mini-computer in order to close to the radio part.

Figure 2. The framework of ML-based Network Management over SDN/NFV-enabled infrastructure.

C. ML-based Management Framework

SDN and NFV were initially developed as independent networking paradigms by ONF and ETSI, respectively. But they have shown strong synergy and combining them in one networking architecture may lead to great value. For example, an SDN controller can be implemented as pure software running on servers like other VNFs. The centralized control and management applications (i.e., SDN Apps), such as mobile load balancing and Quality-of-Experience (QoE) provisioning, can be realized as VNFs. SDN can facilitate the deployment of NFV by providing a flexibility to program the underlying networks.

Taking advantage of SDN and NFV techniques, network programmability is available through API. The function of ML-based network management can be regarded as an external SON Application (APP) on the top of software-defined and virtualized infrastructure, as shown in Fig. 2. In order to consider “legacy” components (non-NFV and non-SDN), we kept the physical NFs in the leftmost side of the architecture, meaning that we may have Physical Network Functions (PNFs), which do not apply the NFV and SDN models. The same way, one may have VNFs, as shown in

rightmost side of the architecture, with no SDN capabilities. In the middle, all components are SDN-aware, meaning that they are split into two layers. In the bottom layer (user-data plane), one may have physical or virtualized Network Elements (NE), which can be Physical Network Elements (PNEs) or Virtual Network Elements (VNEs), respectively. In this case, the names are chosen from the SDN world, since they describe the roles they are performing more clearly. Finally, for the control layer, there is an SDN controller implemented by OpenDaylight in this test-bed.

Once a network problem, such as DDoS attack, is detected, a ML-management control loop is triggered. The SON APP analyzes reported network states, diagnoses the root cause and decides a tactical action. As soon as the SDN controller or VIM received an action request, it coordinates physical and virtual resources to enforce this action for mitigating this network problem. As shown in Fig.2, the input and output of the SON APPs are called *state* and *action*, respectively, which are defined as follows:

- **State:** A set of network-related information, such as alarms, events or metrics, which can be evaluated to indicate the characteristics of a network problem.
- **Action:** It is an implementable countermeasure to tackle the reported network problem taking into account available physical and virtual resources.

III. EXPERIMENTATIONS

In order to further shed light on the test-bed and illustrate the process of experimentation, two experiments, i.e., End-to-End (E2E) QoE provisioning for video users and DDoS mitigation, are carried out. Two isolated virtual networks that are dedicated to video service and cyber security, respectively, are instantiated upon the physical infrastructure. NFV sensors extract and report a list of metrics indicating the network status, relies on which the ML-based system detects network anomalies, makes tactical decisions and invokes required actions to solve network problems.

As illustrated in Fig. 3, the test-bed can be logically divided into three layers: the physical infrastructure, virtual networks and management. On the top of the physical infrastructure, two isolated virtual networks are instantiated. The left side of this figure shows the virtual network for delivering video to mobile users. Several virtual switches (vSwitch) are instantiated with the aid of Open vSwitch. Open vSwitch is a software implementation of a virtual switch running within a VM hypervisor, designed to enable effective network automation through programmatic extensions. Four virtual UEs are instantiated as the mobile video user, where open-source tool SeleniumHQ is installed. SeleniumHQ is a web browser automation tool, used to generate random Internet traffic. To provide a flow of 4K video streaming over Hyper Text Transfer Protocol (HTTP), a streaming server is deployed in the internal network by means of running a server tool Apache2 HTTP [16], as well as a cross-platform multimedia player called VLC [17] in a server mode for evaluating the quality of streaming video. In addition, on the left most part you can see OpenAirInterface-based radio access network including the physical UE, i.e.,

Apple iPad3 used in our experiment, a virtual LTE base station consisting of the baseband unit running in a mini-computer and the RF unit embedded in a USRP module, and the virtual EPC core network. The physical UE accesses the external video service provider (e.g., YouTube), while the virtual UEs access both the internal streaming server and YouTube server. The internal network is connected to the Internet by a virtual router (vRoute), which is also realized with the aid of Open vSwitch. On this virtual router an NFV sensor called Zabbix [17] is also installed to gather network metrics such as the incoming and outgoing traffic, CPU utilization and memory usage of the virtual machine carrying the virtual router. Important to note is, that all Secure Shell (SSH) remote access and sensor data traffic are transmitted via dedicated paths that are physically isolated with other networks, so as to avoid the conflict with the users' traffic. All collected data is forwarded to ML-based management applications running in the GPU server.

As shown in Fig. 3, all YouTube flows targeting both iPad and virtual UEs pass through the vRouter. Once the traffic load on the vRouter is beyond its maximal capability, traffic congestion occur, leading to increased delay and packet loss rate, as well as deteriorated QoE perceived by video users. The mission of the SON APP for E2E QoE is to apply suitable ML algorithms to detect or predict traffic anomalies, and then make a decision for actions that can mitigate or prevent traffic congestion. In this case, for example, the tactical action may be re-routing the video flow. As shown in Fig. 3, the VIM instantiates a new virtual router by orchestrating virtual resources and then, the SDN controller reconfigures the network by means of updating the flow tables of the involved switches to offload the video flow towards the virtual UEs. YouTube video flow towards the virtual UEs are rerouted to the newly instantiated vRouter, as indicated by the red dashed line in Fig. 3.

In the field of ML, the classification technique is an instance of supervised learning. It is applied to identify which class a new observation belongs on the basis of a training dataset. An example would be assigning an incoming email into 'SPAM' or 'non-SPAM' classes in terms of the observed metrics of the email (source IP address, text length, title content, flow statistics, etc.). In this test-bed, the classifier is applied to identify whether QoE degradation due to traffic congestion appears based on the collected data. As an important performance metric, the predictive accuracies of different classifiers [18], i.e., Decision Tree (DT), Linear Discriminant (LD), Support Vector Machine (SVM), and K Nearest Neighbor (KNN), in terms of different sets of selected metrics are tested.

From the practical perspective, some metrics are relevant to the target network problem, whereas others are irrelevant or redundant [19]. Packet Loss Rate (PLR) is recognized by most of ML algorithms as a relevant metric. As illustrated in Table II, we can see that a high accuracy of up to 97.3% can be achieved by monitoring only PLR, which is a really encouraging result. Testing another relevant metric, namely the incoming traffic of EPC (*EPC_Traffic_In*), the accuracies are improved, especially an improvement from 91.7% to 99.3% in the case of KNN. Relying on two metrics

together, the classifiers can achieve their optimal accuracies. As shown in Table II, by monitoring the PLR and the

memory utilization of the virtual machine carrying the VNF of EPC denoted by *EPC_Mem_Util*, the classifiers of DT

Figure 3. Proof-of-concept experiments of video E2E QoE provisioning and DDoS mitigation over two instantiated virtual networks.

and KNN achieve an accuracy of 100%. In contrast, the metric *delay* is less relevant, which can provide an accuracy at most 78.3%.

Table II Classification Accuracy

	DT	LD	SVM	KNN
PLR	0.973	0.944	0.973	0.917
Delay	0.783	0.637	0.634	0.741
PLR + EPC_Mem_Util	1.0	0.857	0.886	1.0
EPC_Traffic_In	0.989	0.967	0.976	0.993

To demonstrate the flexibility of SDN/NFV-based infrastructure and further justify the effectiveness of ML-based management, another virtual network showing the mitigation of DDoS attack is instantiated. This network consists of four virtual UEs, two virtual routers, a web server and a Command and Control (C&C) server. On all four user machines, SeleniumHQ is used to generate automated user traffic. Two users are infected that acts as so-called Zombies that periodically contact the C&C server requesting new attack commands. As shown in Fig. 3, there are two kinds of traffic flows, i.e., the regular traffic to the web server and the suspicious traffic to the C&C server. The C&C server is basically a LAMP server as frontend with a MySQL DB, administrating the Zombies as backend. The web server is an ordinary Apache2 HTTP server hosting a simple website. On the VMs of routers that are realized with OpenVSwitch, the Zabbix agents are installed to monitor the network status.

Fig. 4 shows the typical DDoS traffic measurement extracted from the experiment. The C&C packets in the traffic (80 Bytes every 20 seconds) to and from the infected UE is almost invisible in the first half of the upper figure, but exactly what our algorithm is going to detect among the normal user traffic. Necessary to start countermeasures and avoid the actual DDOS attack that here start in the second half of the graph. You can see the immediate effect of this attack in the lower graph, which shows the traffic generated by the other users consuming the website from the web server. It stops immediately since the Zombies cause the web server application threads to await the end of boundless posts in order to process them and therefore causes the exhaustion of the web server resources. From this set-up we are now analyzing the flow metrics (source IP, destination IP, packet size, and communication frequency). This allow us to identify in a first step suspicious periodic traffic. In a second step, only this suspicious connection gets analyzed via the Deep Packet Inspection (DPI) tool. In our test-bed, SNORT, an open-source network-based intrusion detection system that has the ability to perform real-time traffic analysis and packet logging on IP networks, are installed as a DPI tool. The VIM instantiates a VM to deploy an instance of DPI, as well as instantiates a virtual switch to establish a sub-network in preparation for the deployment of DPI and

HoneyNet. The SDN controller modifies the flow tables of OpenFlow switches so that suspicious packets are diverted to the DPI machine afterwards. If the connection gets identified after the packet analysis of DPI as a real Botnet communication, a proper action can be started. For example, a HoneyNet can be instantiated and deployed by the VIM and then the SDN controller updates the flow tables in the OpenFlow switches to redirect that the attack traffic to the HoneyNet.

Figure 4. The DDos traffic measurements.

IV. CONCLUSIONS

In this paper, we presented an SDN/NFV wireless test-bed with an aim to evaluate and demonstrate machine learning-based network management. This test-bed exhibits excellent reproducibility since it was built by open-source software and off-the-shelf hardware. OpenStack and OpenDaylight were applied to implement the Virtual Infrastructure Manager and SDN controller, respectively, on the top of a physical infrastructure mainly consisting of several high-volume servers and OpenFlow-enabled switches. To support a realistic radio scenario, OpenAirInterface that provides a full set of LTE network functions was integrated. Two proof-of-concept experiments, i.e., E2E QoE provisioning for video users and DDoS mitigation, were carried out in two instantiated virtual networks. Experimental results verified the effectiveness of ML-based network management, as well as the network flexibility enabled by the application of SDN and NFV. In the next step, more advanced ML algorithms

and other representative network scenarios will be implemented and experimented in the test-bed.

ACKNOWLEDGMENT

This work was supported by the German Federal Ministry of Education and Research (BMBF) under the TACNET4.0 project with grant number KIS15GTI007.

REFERENCES

- [1] G. Fettweis *et al.*, "The Tactile Internet," ITU-T Technology Watch Report, August 2014
- [2] "Top ten pain points of operating networks," Aviat Network, 2011.
- [3] R. Mijumbi *et al.*, "Network function virtualization: State-of-the-art and research challenges," *IEEE Commun. Surveys*, vol. 18, no. 1, pp. 236–262, 2016.
- [4] B. A. A. Nunes *et al.*, "A survey of software-defined networking: Past, present, and future of programmable networks," *IEEE Commun. Surveys*, vol. 16, no. 3, pp. 1617–1634, 2014.
- [5] X. Zhou *et al.*, "Network slicing as a service: enabling enterprises' own software-defined cellular networks," *IEEE Commun. Mag.*, vol. 54, no. 7, pp. 146–153, Jul. 2016.
- [6] K. Samdanis *et al.*, "From network sharing to multi-tenancy: The 5G network slice broker," *IEEE Commun. Mag.*, vol. 54, no. 7, pp. 32–39, Jul. 2016.
- [7] D. Silver *et al.*, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, pp. 484–489, Jan. 2016.
- [8] W. Jiang, M. Strufe, and H. D. Schotten, "Experimental results for artificial intelligence-based self-organized 5G networks," in *Proc. IEEE Int. Symp. On Personal, Indoor and Mobile Commun. (PIMRC 2017)*, Montreal, Canada, Oct. 2017.
- [9] EU H2020 5G-PPP SELFNET project. [Online]. Available: <https://selfnet-5g.eu/>
- [10] OpenStack [Online]. Available: <https://www.openstack.org/>
- [11] OpenDaylight [Online]. Available: <https://www.opendaylight.org/>
- [12] N. Nikaein *et al.*, "OpenAirInterface: A flexible platform for 5G research," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 5, 2014, pp. 33–38.
- [13] MaaS: Metal as a service. [Online]. Available: <https://maas.io/>
- [14] SeleniumHQ. [Online]. Available: <https://www.seleniumhq.org/>
- [15] Apache. [Online]. Available: <https://httpd.apache.org/>
- [16] VLC. Video LAN Organization. [Online]. Available: <https://www.videolan.org/vlc/index.de.html>
- [17] Zabbix. [Online]. Available: <https://www.zabbix.com>
- [18] W. Jiang, M. Strufe, and H. D. Schotten, "Autonomic network management for software-define and virtualized 5G systems," in *Proc. European Wireless*, Dresden, Germany, May 2017.
- [19] W. Jiang, M. Strufe, and H. D. Schotten, "Intelligent network management for 5G systems: the SELFNET approach," in *Proc. European Conf. on Net. and Commun. (EuCNC 2017)*, Oulu, Finland Jun. 2017.