Demo: The EmPOWER Mobile Network Operating System

Roberto Riggio
CREATE-NET
Trento, Italy
rriggio@create-net.org

ABSTRACT

In this demo we present the EmPOWER Mobile Network Operating System. EmPOWER supports both WiFi and LTE networks and can interface with state–the–art SDN controllers using an intent–based interface. A Python–based SDK allows network programmers to write and deploy advanced control tasks as Network Apps. The demo will illustrate three Network Apps: mobility management, multicast, and VNF chaining/migration. We will also stage a *speed tutorial* allowing attendees to write and test a simple Network App using the EmPOWER SDK.

Categories and Subject Descriptors

C.2.3 [**Computer Communication Networks**]: Network Operations

Keywords

Software–Defined Networking, 5G, LTE, WiFi.

1. INTRODUCTION

5G mobile networks are expected to deliver significant improvements in terms of both raw performances (e.g., bandwidth) and total cost of ownership (acquisition and operating costs). Software–defined networking and network function virtualization are two of the most important technological enablers for achieving such goals.

Nevertheless, unlike in the wired domain, a clear 5G control plane solution has yet to emerge. The technological challenges in this respect are significant. In fact, the 5G Radio Access Network (RAN) is expected to be highly heterogeneous combining cellular and non–cellular technologies. WiFi in particular will play a key role as traffic offloading solution. Moreover, since resource allocation in the RAN affects the status of the backhaul and viceversa, a 5G Mobile Network Operating System must account for both the wired and the wireless segments of the mobile network.

In this demo we showcase the potential of a unified 5G Mobile Network Operating System. The demo builds upon 5G–EmPOWER[1], an experimental platform converging SDN and NFV into a single framework. EmPOWER supports both WiFi and LTE RANs and can interface with state–the–art SDN controllers. The entire stack including the WiFi data–path implementation, the reference controller/agents, and a Python–based SDK has been released under an APACHE 2.0 License for academic use.

The demo will illustrate three Network Apps: mobility management, multicast, and VNF chaining/migration. We will also demonstrate how to implement in a few lines of code a Network App capable of gathering the global network view of an heterogeneous WiFi/LTE network.

WiNTECH'16 October 03-07 2016, New York City, NY, USA
© 2016 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-4252-0/16/10.
DOI: http://dx.doi.org/10.1145/2980159.2980173

2. THE EMPOWER PLATFORM

The 5G–EmPOWER [1] architecture, sketched in Fig. 1, consists of three layers: infrastructure, control, and service. At the infrastructure layer we have the data–plane elements. In particular we name: Wireless Termination Points (WTP) the Wi–Fi Access Points, Virtual Base Stations (VBS) the LTE eNodeBs, and Click Packet Processors (CPP) the nodes combining computational and forwarding capabilities. At the control layer we have the EmPOWER Runtime and the wired backhaul controller. The EmPOWER Runtime is in charge of translating the high–level control policies specified at the service layer (e.g. Radio Resource Management and Traffic Engineering) into commands for the data–plane elements. Network Apps run on top of the EmPOWER Runtime in their slice of resources and exploit the Runtime Northbound Interface (*Scylla*) in order to implement control tasks. The EmPOWER Runtime interacts with the wired backhaul controller using an intent–based interface (*Tropea*). OpenFlow is used as southbound interface between the backhaul controller and the backhaul Transport Nodes. Finally, at the service layer we have the network services which essentially consists in a set of VNFs and the associated control logic, i.e. the Network Apps.

3. SOFTWARE DEVELOPMENT KIT

A Python–based SDK is made available to developers for implementing and deploying Network Apps on top of the EmPOWER Runtime. Network Apps are essentially Python modules loaded by the EmPOWER Runtime at bootstrap or dynamically using a REST or a CLI Interface.

Network Apps are required to implement a `launch` method that is invoked when the App is loaded by the Runtime. The `launch` method must return an instance of the base class

[1]http://empower.create-net.org/.

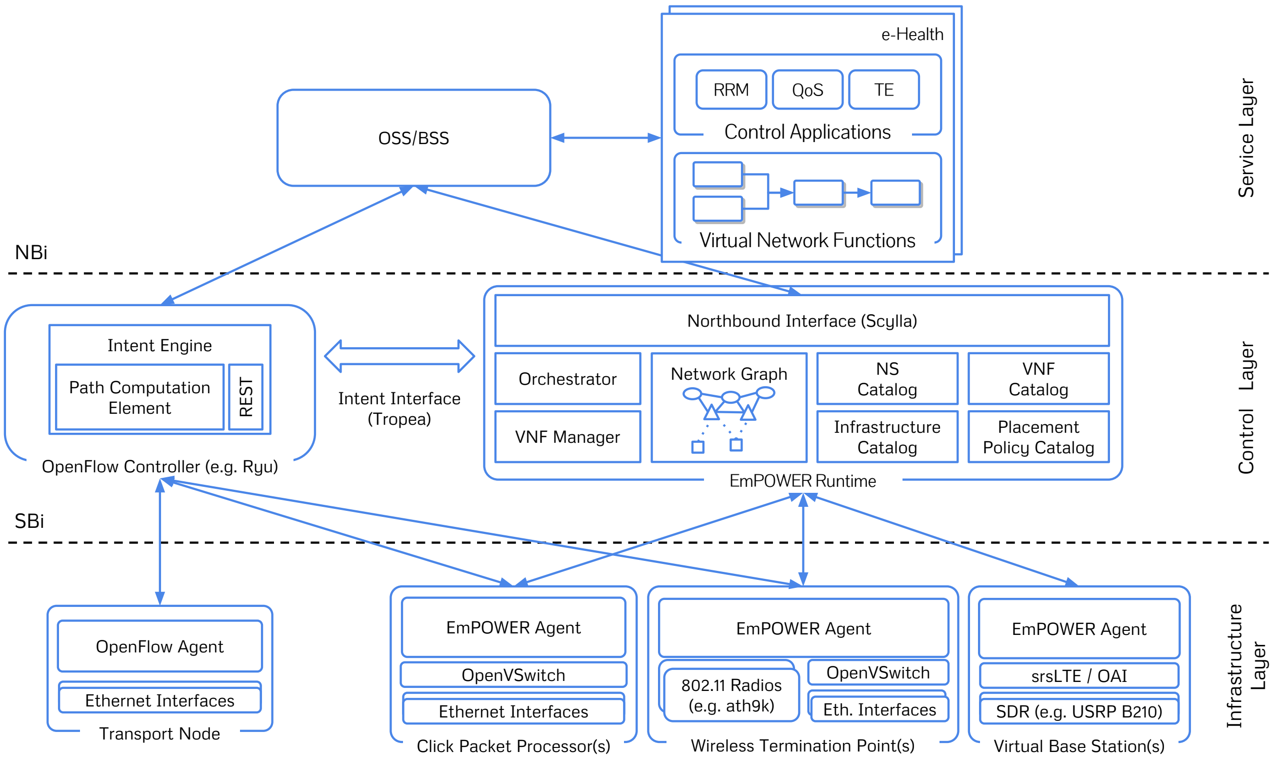

Figure 1: The 5G–EmPOWER Architecture.

`EmpowerApp` and must be provided with the mandatory initialization parameter *tenant_id* which specifies in which Network Slice the App must be loaded. Multiple instances of the same Network App can be deployed in different slices. Similarly, a slice can run multiple Network Apps. The EmPOWER Runtime ensures that a Network App is only presented a view of the network corresponding to its slice.

A simple *Hello World* App is reported in Listing 1. The App uses the `lvapjoin` primitive to register a callback method to be invoked when a new client joins the network, e.g. when a Wi–Fi station successfully associates to the slice's SSID.

```
from empower.core.app import EmpowerApp

class HelloWorld(EmpowerApp):
  def __init__(self, **kwargs):
    EmpowerApp.__init__(self, **kwargs)
    self.lvapjoin(callback=self.lvap_join)

  def lvap_join(self, lvap):
    self.log("New client %", lvap)

def launch(tenant_id):
    return HelloWorld(tenant_id=tenant_id)
```

Listing 1: A simple Hello World Network App

4. DEMO

During the demo we will stage a *speed tutorial* allowing attendees to write a simple Network App (like the one in Listing 1) and to test it using their own mobile devices. We will also demonstrate the following Network Apps:

Mobility Management. This demo will show how to seamlessly handover Wi–Fi stations in such a way to minimize the number of active WTPs. The demo leverages on the Network Graph and the Mobility Management APIs.

Multicasting. This demo will show how to implement a multicast rate adaptation mechanism for Wi–Fi networks. The implemented approach will allow to utilize high MCSes for multicast transmissions while preserving the reliability of the communication. The demo leverages on the TX Statistics API and the TX Policy APIs.

VNF Migration and Scaling. This demo will illustrate how to deploy Click–based VNFs and to steer a precise portion of the flow space across them in a particular order. The demo will also illustrate VNF Migration and Scaling. The demo leverages on the Virtual Port API.

Network Graph. This demo will illustrate how to use the Network Graph API to tap into the state of an heterogeneous Wi–Fi and LTE network.

Acknowledgment

This work is supported partially by the European H2020 Research and Innovation project COHERENT (GA No. H2020-ICT-671639) and partially by the EIT Digital Future Network Solutions Action Line, project grant No. 16242-A1605.

5. REFERENCES

[1] R. Riggio, M. Marina, J. Schulz-Zander, S. Kuklinski, and T. Rasheed, "Programming abstractions for software-defined wireless networks," *Network and Service Management, IEEE Transactions on*, vol. 12, no. 2, pp. 146–162, June 2015.