NETWORK POLICY FUNCTION VIRTUALIZATION VIA SDN AND PACKET PROCESSING

Titus Constantin BĂLAN

"Transilvania" University of Braşov

Abstract: *So far affecting mostly the IT world, the virtualization trend is becoming active in the mobile telecommunications network by the Network Function Virtualization (NFV) initiative, in conjunction with the Software Defined Networks (SDN) concept, of moving network applications from dedicated hardware to virtual containers on commercial-off-the-shelf (COTS) hardware. Policy and charging rule functions are basic and very important elements for the operators of 3G and LTE mobile networks, aiming to monetize at maximum their networks. This paper describes a method for an implementation of the policy function based in SDN and its deployment on the telco specialized COTS platform ATCA (Advanced Telecom and Computing Architecture).*

Keywords: *SDN, network function virtualization, policy rules, ATCA*

1. INTRODUCTION

As the innovation cycles continue to accelerate, legacy telecom hardware-based appliances rapidly reach end of life.

Simply having a hard-wired network with boxes dedicated to single functions is not the optimal way to achieve dynamic service offering and the signal for a change in the network world was given by the SDN concept of separating the control and data plane, followed by the Network Function Virtualization, an ETSI (European Telecommunications Standards Institute) initiative. NFV is a network architecture concept that proposes using IT virtualization related technologies to virtualize entire classes of network node functions into building blocks that may be connected or chained together to create communication services.

Because the network design must be more agile and able to respond on-demand to the dynamic needs of the traffic and services running over it, the solution would be having software implementations of the network functions running in the Cloud, on Data Center infrastructure or on dedicated COTS platforms (like ATCA [7]).

Most of the important telecom vendors have started developing telco virtualized products, some focused on the end-to-end solution, like Ericsson and Huawei, others aiming to virtualize specific parts of the network: Cisco, F5, Juniper, Tekelec etc. Separately, carriers are considering the deployment of SDNs in which packet forwarding is separated from control functions, and the latter are provided via a "controller" function that tells simple packet forwarding devices how to direct flows, based on "Northbound" APIs, interconnected to the OSS and BSS (Operational- and Business-Service Support) systems.

The goal of this paper is to demonstrate a method for policy rules implementation at a centralized SDN controller point and the possible virtualization of the policy network function using a standard ATCA platform with dedicated packet processing and computing modules.

2. PCRF NETWORK FUNCTION VIRTUALIZATION

ETSI's Industry Specification Group for Network Functions Virtualization (NFV ISG) has published the first batch of specifications in October 2013, including an Architectural

Framework based on functional blocks [3] and indicated the first NFV use-cases [4]. Special categories in the indicated use-cases are dedicated to the Evolved Packet Core (EPC) transformation towards NFV: "UseCase#5 Virtualization of Mobile Core Network and IMS" and to Radio Access Network part: "UseCase#6 Virtualization of Mobile base station".

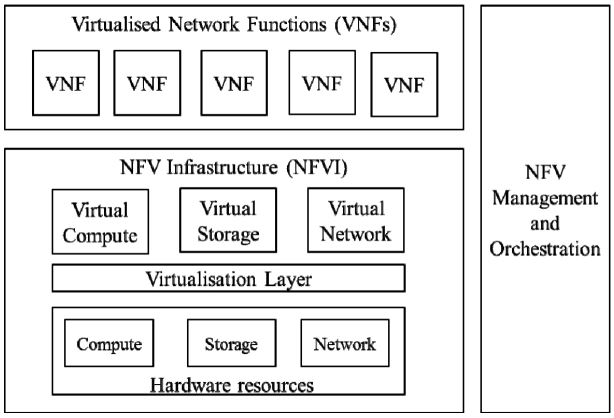

Fig.1 NFV architecture

LTE formally separates the control and bearer planes (it is sometimes called a "SDN-like" architecture), so virtualization of these functions is viewed as attractive [9].

The creation of policy architectures was driven by telecom vendors' and operators' need to find more intelligent ways to allocate resources: differentiated services, real-time control and dynamic offers – from "shared wallet" and roaming control to "happy hour" and toll free – that increase customer loyalty and create new revenue opportunities.

A dedicated policy function was first introduced by 3GPP as part of the IMS specifications. As the interest in policy control increased, 3GPP introduced it as a separate policy platform, which typically includes a policy server and policy enforcement devices (e.g. Deep Packet Inspection, DPI appliances), often also connected to charging systems and subscriber databases, that can be generically named PCRF (Policy and Charging Rules Function).

Policy entities [1] are used to help operators to dynamically control the way that users and applications consume data network (IP and Internet) resources.

Policy decisions can be based on a wide variety of triggers, including a customer's data volume usage, service tier, location, application, URL, time of day, congestion level and so on [6].

The all-IP "connected anywhere" concept evolution has introduced different types of users and profiles for the mobile network, including M2M or industrial communication, with different QoS, security and traffic bandwidth profiles: in some cases the user would be highly mobile and consume a lot of bandwidth, and thus require a particular EPC, traffic management and security configuration; whereas another type of service may be more static and predictable, and thus require a different workflow configuration (see Fig. 2, as proposed in [2]).

The policies became more complex and different for each case so, similar to the service creation for Intelligent Networks based on service building blocks (SBBs), the policies are created and adapted based on specific "workflows", which could be easily implemented in software, at SDN controller level, as we will detail in the next paragraph.

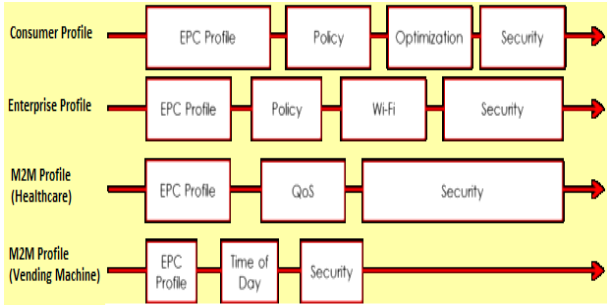

Fig. 2 Different scenarios ("workflows") for policy function building blocks

In each scenario, a service definition is created for each user type or application, and a workflow is configured accordingly.

3. POLICY RULES IMPLEMENTATION USING SDN

The Software Defined Networking (SDN) is a new architectural concept that aims to decouple the network control and forwarding functions [8].

This separation enables the network control layer to be programmable, which is an important advantage for real-time traffic discrimination in case of elastic PCRF implementations. Furthermore, SDN uses the OpenFlow protocol that was specially designed to include packet processing techniques like *match* and *modify actions* for the flow entries.

The OpenFlow wire protocol is used for establishing a control session, defining a message structure for exchanging flow modifications (flowmods) and collecting statistics, and defining the fundamental structure of a switch (ports and tables).

In an OpenFlow flow entry, the entire packet header (at least the layer 2 and layer 3 fields) are available for match and modify actions, so basically the DPI is natively considered as part of SDN. That is why implementing the policy function using SDN is maybe one of the most straight-forward use-cases for software defined networking.

Below is an example of a packet filtering function implemented in Phyton that was tested using the POX controller and the Mininet SDN environment:

```
    def addRule(self, address, toAll=False,
dl_type=0x800, nw_proto=1):
policyAddRule = of.ofp_flow_mod()

if dl_type == 0x800:

ip_addr = IPAddr(address)

rule_group = (ip_addr, dl_type, nw_proto)

policyAddRule.match.dl_type = dl_type

policyAddRule.match.nw_dst = ip_addr

policyAddRule.match.nw_proto = nw_proto

policyAddRule.priority = 65535
```

Mininet creates a realistic virtual network, running real kernel, switch and application code, on a single machine (VM – Virtual Machine, Cloud or native), in seconds, with a single command.

Mininet allows simulation of SDN networks including controllers, switches and hosts.

Open vSwitch (OVS), which comes preinstalled on the Mininet VM, and this multilayer open switch implementation also offers support for OpenFlow protocol.

Mininet was used for testing and simulation of the policies implementation in a virtual network [5]. POX is a software platform that acts as SDN Controller, written in Python 2.7 that is communicating with elements that understand the Openflow 1.0 protocol [10].

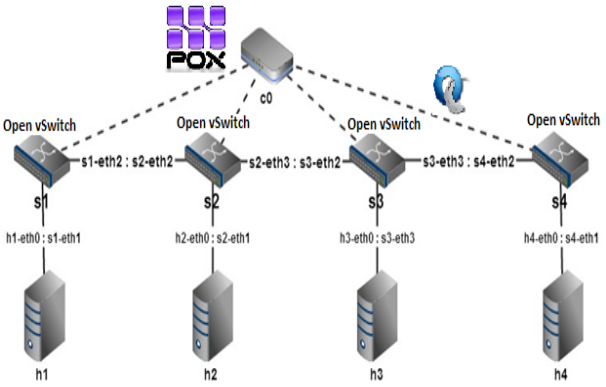

Fig. 3 The Mininet test environment based on POX SDN Controller and virtual switches

The implemented application does packet filtering based on implemented rules, but is exemplificative for the packet inspection capability of SDN. The main algorithm for stream discrimination implies the packet arrival, comparing information from the header with the policy rules described at controller level and applying the actions to the packet (modifications, forwarding or packet dropping). When a network element is connected to the network the event **ConnectionUp** is triggered, and the **start_policy(event)** from the **launch()** function creates a Policy object for every network connected element.The function **addRule(self, address, toAll=False, dl_type=0x800, nw_proto=1)** accepts 4 arguments, an address (IP or MAC), dl_type (dl_type=0x800 – type IP packet, dl_type= 0x806 – type ethernet) or protocol number (6 = TCP) and, if the argument toALL is True, than the rule is added on all policy elements in the network. An advantage of a SDN distributed policy function is that different rules could be applied on each equipment that understands the Openflow protocol by sending a message **ofp_flow_mod()** for flow modification (the Flow-table is updated).

4. ARCHITECTURE FOR POLICY FUNCTION IMPLEMENTATION USING ATCA

ATCA (Advanced Telecom and Computing Architecture [7]) represents a high-tech modern platform with a standard shelf and a back-plane ("mother-board") compliant with the regulations of PIGMG (PCI Industrial Computers Manufactures Group).

The modular ATCA platform can be configured to act as any element of the 3G or LTE core network, while there are already developed very compact implementations of the "LTE in a box"[12] concept, where all core network elements are implemented on the same ATCA chassis.

In our case, the available configuration, having the dedicated packet processing board and the computing board, is extremely suitable for a policy function implementation that includes, but not be limited, to the functions of a 3GPP LTE PCRF implementation.

These blades are particularly the following (see Figure 4): the FlexCore ATCA-FM40 – Ethernet switching boards, ATCA-XE80 computing boards, ATCA-PP50 – packet processing boards that will be responsible for flow discrimination and policy check.

The two packet processing blades (PP50) have each with two packet processors RMI-Netlogic-Broadcom – type XLR732 – that have a „super-scalar "architecture.

Each processor is multi-core MIPS64 ("without Interlocked Pipeline Stages") and each core can run 4 threads (than a total of 8 x 4 = 32 virtual cores/processor).

Fig. 4 The experimental configuration – front view of the installed ATCA 40G platform

All applications run on Linux virtual machines, while the data fast-path is operated by the RMI-OS. Operating Systems (OS) aren't recommended to run on a per thread basis, so they run on a per core basis, so the existing cores are split between Linux virtual machines, that run on N cores (N>1) and RMI-OS, running on 8-N cores, on top of an XLR "bootloader".

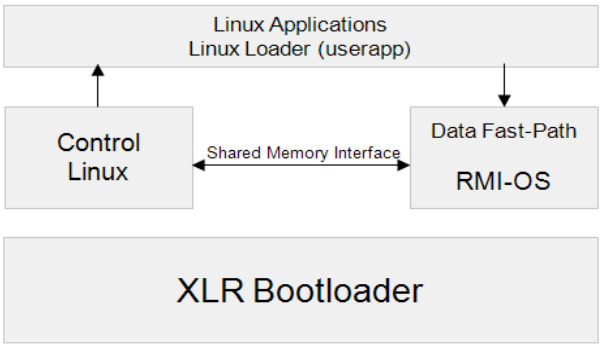

Fig. 5 Software configuration of the PP modules

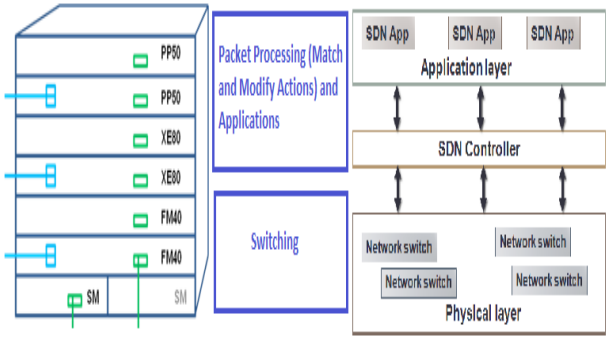

Fig. 6 Proposal of SDN functions implementation on the ATCA experimental configuration

Similar to the SDN logic where the data path is separated, the PP80 board functions are divided: in the Telecom use-cases, Linux could run control software (e.g. call control, bandwidth allocation, load balancing) while RMI-OS should manage packets analysis and discrimination (e.g. packet I/O via 10Gb "fast-path" interfaces).

The 8-N RMI-OS cores can communicate with the N Linux cores by a shared-memory interface. A more detailed description of the testing environment and the detailed commands used for synchronous operation of Linux virtual machines and RMI-OS was presented in [11].

In our case, the packet processing functions,

programmed at SDN controller level, could wrap the low-level functions of the packet processing board from ATCA and should communicate with databases and policy servers in order to take decisions.

Acting as the brains of the system, the policy controller tells the SDN controller how to route a specific flow, working closely with other elements in the service chain such as DPI, video optimization and firewalls.

CONCLUSIONS

Network Function Virtualization, though not yet standardized, is being already deployed by vendors that have launched different telco appliances running in virtualized environment.

This is the effect of the "as-a-Service" and Cloud influence that has a boom in IT area. Control-plane applications, such as policy servers, subscriber databases and IMS, will be among the first virtualized 4G core functions to be implemented in commercial mobile networks. These applications are already server-based and are relatively straightforward to port to a virtualized environment.

Because the concept of SDN was build based on flow-tables, including *match* and *modify action* triggers, so based on a native traffic discrimination, up to the layer 7 of OSI, the decision to implement policy enforcement functions using Software Defined Networks is ideal. At higher level, we demonstrated the possibility to implement traffic filtering based on a POX SDN Controller implementation, using Python programming language for the SDN controller (POX), and Mininet as SDN test environment. Policies were implemented at controller lever. At lower architectural level, the COTS hardware to implement the PCRF logic can be the standard ATCA platform, bringing the advantage of high processing power for DPI (the used ATCA platform has a dedicated board) and fast connections due the incorporated switching fabric. We were able to test packet processing elements at XLR processor level, using RMI-OS implementation and their communication with applications that reside in Linux virtual machines.

Being part of the same modular chassis, the packet processing, computing and switching blades have a great advantage, as the main components of an SDN network are clearly delimited: SDN controller functionality and flow analyze is done via packet processing, the computing blades take care of applications, while the data-forwarding plane is the role of the switching elements.

BIBLIOGRAPHY

1. Amdocs Whitepaper, *The Power of Policy- Enriching the data experience with advanced policy control* (2012)
2. Brown Gabriel, Heavy Reading, WhitePaper *Control Plane Elasticity & Virtualization in the 4G Core*, March 2013
3. ETSI NFV ISG (Industry Specification Group) – *Architectural Framework* http://www.etsi.org/deliver/etsi_gs/NFV/001_099/002/01.01.01_60/gs_NFV002v010101p.pdf
4. ETSI NFV ISG (Industry Specification Group) – *Usecases* http://www.etsi.org/deliver/etsi_gs/NFV/001_099/001/01.01.01_60/gs_NFV001v010101p.pdf
5. Feamster Nick, *Online Course, Software Defined Network*, https://www.coursera.org/course/sdn
6. Finnie Graham Heavy Reading, WhitePaper, *Policy Control & SDN: A Perfect Match?*, August 2013
7. Kuhlmann R. - *ATCA: Advanced Telecom Computing Architecture - Die Plattform der Zukunft für Telekommunikations-systeme* - Franzis Verlag GmbH, 2007 – ISBN 9-7837-7234-1298
8. Nadeau T., Gray K., *SDN – Software Defined Networks*, O'Reilly, 2013, ISBN: 978-1-449-34230-2
9. Natarajan, S. and Wale, K. (2009): *Implementing ATCA in LTE network serving gateway nodes*, Radisys Whitepaper

10. POX online documentation, https://openflow.stanford.edu/display/ONL/POX+Wiki

11. Sandu F., Costache C., Balan T.C. Balica A.N, *Packet Processing on an ATCA 40G Platform*, MACRO 2013- International Conference on Recent Achievements in Mechatronics, Automation, Computer Science and Robotics, Tg. Mures, 2013

12. Solucomp, *"LTE in a Box" solution based on ATCA*, available web at: http://www.solucomp.com/LTE_in_a_box.html