Broadband Network Gateway implementation using a programmable data plane processor

Juan Sebastian Mejia Vallejo , Christian Esteve Rothenberg

Departamento de Engenharia de Computação e Automação Industrial (DCA)
Faculdade de Engenharia Elétrica e de Computação (FEEC)
Universidade Estadual de Campinas (Unicamp)
Caixa Postal 6101, 13083-970 – Campinas, SP, Brasil

{jmejia,chesteve}@dca.fee.unicamp.br

Abstract – The broadband network gateway (BNG) play a crucial role in today's networks, it hands all access network traffic (e.g DSL traffic), provisioning of such policies like rate limiter, tunneling and complex network management mechanisms that an Internet service providers (ISPs) needs to implement his services. These network devices are expensive, proprietary, limited and slow device upgrading, become a point of failure to deploy new features, add functionalities and correct issues on the network without disrupting the normal service operation.
In this article, we propose a software-based virtualized BNG device running on inexpensive commodity hardware using high-level language for programming protocol-independent packet processors over a Multi-Architecture Compiler System in order to remove some of these barriers to innovation.

Keywords – BNG, P4 language, Virtualization,Open Data plane.

1. Introduction

The advent of new services like Video-on-Demand (VoD), video conference, Virtual Private Network (VPN) and cloud-based new services, has increased the demand for access to broadband services [1]. In addition, many new access technologies such as xDSL, optical access and wireless technologies such as WiMAX and LTE require rapid deployment of services and devices guaranteeing performance in Internet Services provider (ISP) networks.
In an Internet service provider (ISP) network the Broadband Network Gateway (BNG) has the function of managing all access network traffic (e.g, DSL traffic) and other critical functions like to allow access and authentication for thousands of subscribers, monitoring, establishing sessions and tunnels, and controls the user line rate. The fact that all sessions tunnels (e.g.., PPPoE, GRE) are terminated at the BNG means that is aggregated in a single point causing poor performance and quality of service. Therefore is no surprise that this device becomes expensive and hardware proprietary boxes, often the operator pay for some functionality that won't be used and the hardware boxes upgrade require a long wait until the next version available.
In recent years, in order to resolve this problem to turn this rigid hardware device into a software-based network and reduce time to market of new services and those functionalities have been decomposed and dynamically instantiated at different points of the network. This concept is following the trend of Software-defined networks (SDN) and Network function virtualization (NFV) that turn some network functionalities into virtualized software processing running on a server (e.g., off-the-rack x86 servers), switches or even cloud computing infrastructure [2].
Our approach to create an fully open and programmable BNG data plane is using a high-level language for programming protocol-independent packet processors (P4), this is a domain-specific language (DSL) with a number of functions optimized around network data forwarding.
Such a DSL can support customizing the forwarding behavior of the switch and may also be ported to other hardware or software switches that support the same language. In this paper, we present an BNG architecture and implementation, it is built on top of the MACSAD Switch target and P4 language that can provide dynamic and flexibility to the service provider to optimize the traffic on the network.
We discuss the architecture and protocols to deploy and concepts around the BNG virtualized router.
The rest of this paper is structured as follows: Section II provides the background. Section III Introduces the proposed design , Section IV Evaluation and Section V concludes.

2. BACKGROUND

2.1. Broadband Access Networks

The first generation network based on centralizes BRAS routers was driven for the customer demand for High-speed Internet (HSI) the second generation Ethernet-based Broadband Network Gateway (BNG) routers was driven by subscriber demand for

a linear TV service (content broadcast at specific times, e.g., Netflix) delivered in conjunction with voice and HSI services [3].
The Customer Premise Equipment (CPE) represents the triple play communications devices: Telephone (Voice), PC (Internet), Set-top box (TV) however all these devices are connected to a Home Gateway (HG) that brings the interface with the network through any access technology like Digital Subscriber Line (DSL).
One or more HG can be connected to a single DSLAM that sends the traffic to the BNG device and it route in a core IP network and the edge routers to provide connectivity to the Internet (See Figure 1).

The network operator provides connectivity, au-

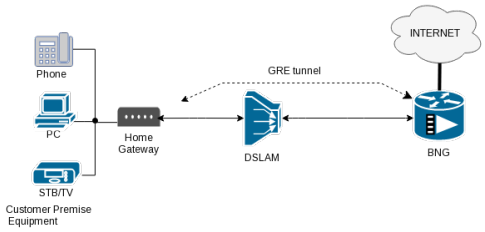


Figure 1. Access Network Provider Model.

thentication, applications and service network policies to his users, therefore, these procedures involve the premise of session establishment using access communication protocols which are managed in the BNG, the most common protocols to establish session are:

- The PPP over Ethernet (PPPoE): Use the point-to-point (PPP) protocol.
- The IP over Ethernet (IPoE): Use IP protocol that runs between CPE and BNG.
- Generic Routing Encapsulation (GRE V2), encapsulation protocol brings virtual Point-to-Point connections through IP network.

In our implementation, the packet sent or received by the HW are encapsulated with GRE headers, creating a point-to-point link with the BNG, but is just one of the options to packets encapsulation.
Since the BNG centralizes all the functions simplify the management functions like:

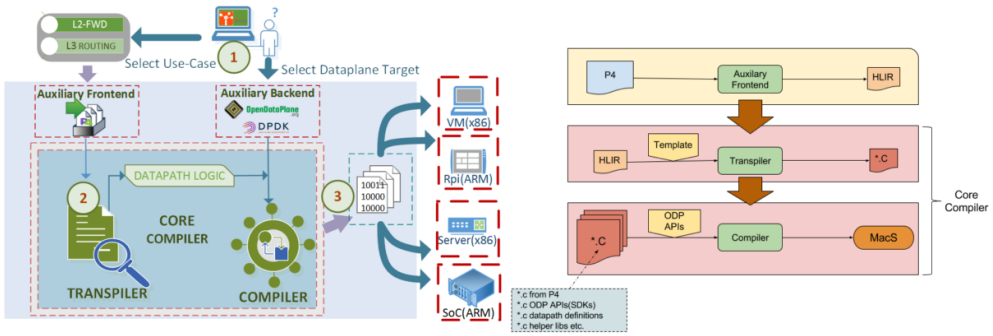
- Session management and header cap/decapsulation.
- Interface to Authorization, Authentication and accounting services.
- ARP proxy to manage the requests from the network interface on the BNG side.
- Network Address translation to route the packets towards the operator's core Network.
- Interface to assignment queues and line rate to subscribers.

On the other hand, all these tasks make the structure of network rigid and become difficult to support the protocols and architectures in the current ISP. In [4] we found some similar approach to virtualize a Broadband Remote Access Server (BRAS) based on Click OS, a tiny Xen virtual machine designed specifically for network processing it can achieve line rate of 10Gbps and it composes of netmap and VALE as the packet I/O framework . Our architecture approach goes with the same trend of network function virtualization using Macsad framework that compiles P4 code to bring more flexibility to the data plane and adding support for Hardware Abstraction Layer (HAL).
In the next session, we will describe our architecture design to be rapidly reconfigured using a "programming protocol-independent packet processors" (P4) and MACSAD to generate the datapath Logic codes.

2.2. Programming Protocol-Independent Packet Processors (P4)

P4 is a high-level language for programming protocol-independent packet processors, that define how the pipeline of a network forwarding device should process the packets using the abstract forwarding model see figure 3. P4 define the header structures and use the parser to extracts the header fields. The pipeline is defined through a series of match-action tables, that execute one or more actions like packet forwarding, drop and so on. This tables can be changed and accessed at "runtime" through a controller software in order to add, remove and modify table entries and finally, deparser writes the header fields back before sending the packets to the output port. The three main advantages of P4 are:

1. Reconfigurability in the field: Programmers should be able to change the way how the switch process the packets once that was deployed.
2. Protocol independence: Capability to deploy any protocol in a switch.
3. Target independence: To describe packet-processing functionality independent of the hardware where it has been deployed [6].



(a) Architecture & Use Case Workflow (b) 3-Tier Compilation Process

Figure 2. Macsad architecture. Source:[5]

Figure 3. P4 Abstract Forwarding Model. Source: Adapted from [6].

2.3. Multi-Architecture Compiler System for Abstract Dataplanes (MACSAD)

This tool brings an environment to compile and deploy switch for L2/L3 applications automatically generating the datapath code for heterogeneous targets over 10Gbps network interfaces setup [5].
The MACSAD architecture is designed around the following three modules: (see the figure 2a)

- Auxiliary Frontend: The P4 code is the MACSAD input that will be converted to an IR representation using the p4-hlir framework that supports different Domain Specific Language (DSL) to generate a High-Level Intermediate Representation (HLIR) to be used in the Macsad core compiler.
- Auxiliary Backend: this module provides a common SDK for the Compiler incorporating the ODP APIs [7] auto-generating support for different platforms with ODP supporting various control protocols like Switch Abstraction Interface (SAI), OpenFlow (OF), and other useful events for buffer like queueing, scheduling, etc.
- Core Compiler: Composed of a Transpiler and a Compiler submodule (see the figure 2b), transforms the IR generated by the frontend into the target imaged in association with the auxiliary backend.
The transpiler takes the HLIR input and auto-generates the Datapath Logic codes. Some important decision is taken in this sub-module like to optimize the 'Dead Code Elimination' by identifying reachability in a graph, decides the type of look-up mechanism to be used, and size and type of tables to be created by. The Macsad compiler creates a set of libraries in 'C' codes for the desired target (x86, x86+DPDK, ARM-SoC).
The BNG P4 code will be added to the system as an input to create the BNG router. In addition Macsad tool brings generation of high-level ODP APIs to deliver platform abstraction with high performance and hardware-acceleration options.

3. Broadband Network Gateway Design

MACSAD is able to generate from a BNG P4 code the 'C' libraries and support the software router in addition to API HAL and other Software Development Kits (SDKs).
The both packets that are arriving from the access network and leaving to the core network through the Network interface card (NIC) are interpreted by a Hardware Abstraction Libraries (HAL) API it is a module Linux subsystem (see figure 4a) block that lets abstract the functionality of the NIC and connects the physical network interface card to the virtualized BNG core.
When the L2 packet coming to the BNG core takes care of handles ARP queries and replies. In addition, for L3 packet it takes care of packet forwarding onto subscribers first doing the tunneling point-to-point mechanism tunnel Encap/Decap, save ID in order to establish the user session, Network Address Translation (NAT) and identify the users to forward the packets to different rates. From there, packets go once through a Traffic Manager and then to the access network. Traffic Manager let us control some events in order to support different requirements like packet scheduling, rate limiter to improve QoS, high throughput, and low latency.

4. Evaluation

We propose two methods to evaluate our prototype: The first one focused in asses the functionality of

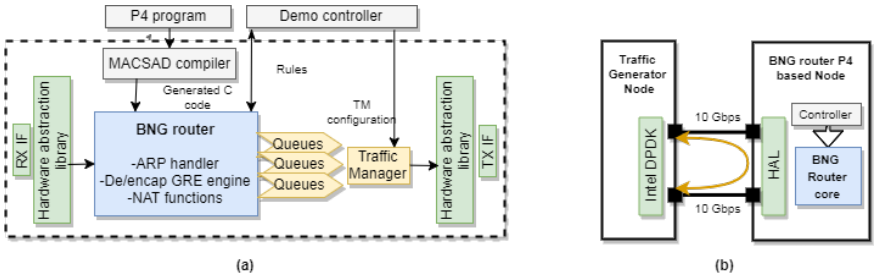


Figure 4. (a) BNG software architecture. (b) Evaluation setup.

the pipeline and controller, the second one is using MACSAD in a Linux server to run, compile and assess router performance in a real environment.

4.1. Functional Evaluation

We assess the functionality of the pipeline and control plane of the BNG router using a packet generator (e.g Scapy) to transmit packets through the router and a network protocol analyzer (e.g Wireshark) in a development environment to verify the sent and received packets along the virtual interfaces. The basic BNG functions are: Encap/Decap GRE header, NAT table functions, L3 forwarding, and controller functions like send entry tables, receive and send packets for/to the router.

4.2. Performance Evaluation

In order to validate the BNG performance in a real network, we compile on the Macsad framework in a common Linux server with an x86 processor, where the Macsad compiled datapath is connected to the Network Function Performance analyzer (NFPA) [8] via two 10G links to generates test traffic (see figure 4b). The NFPA allows send different traffic configurations and evaluates the throughput for the best and worst case.

5. Conclusions and Future Work

In this article, we presented a brief description about the BNG and programing independent-target processor P4 language; a powerful and promising language to program and change the way switches work and we presented an architecture of BNG router over the Macsad platform to rapid and flexible data plane prototyping, Then, we made our first approach to develop a BNG architecture in a virtualized environment.

As a future work, we need to create a proof-of-concept of this model and validate, starting inserting GRE and ARP packets to establish user session, to limit bandwidth using the controller and configuring in a real scenario using specific HAL modules like DPDK interfaces in order to test the maximum throughput.

ACKNOWLEDGMENT

This work was funded by Ericsson Telecomunicações.

References

[1] The Organization for Economic Co-operation and Development (OECD). Improving Networks and Services Through Convergence discussion paper. *2016 Ministerial Meetign*, 2016.

[2] B. Han, V. Gopalakrishnan, L. Ji, and S. Lee. Network function virtualization: Challenges and opportunities for innovations. *IEEE Communications Magazine*, 53(2):90–97, Feb 2015.

[3] Alcatel-Lucent. Evolution of the Broadband Network Gateway. 2010.

[4] B Roberto, D Thomas, Felipe H, Mohamed A, Joao M, Saverio N, and Hans-Joerg K. Rethinking Access Networks with High Performance Virtual Software BRASes. *EWSDN*, 2013.

[5] P.G. Patra, C.E. Rothenberg, and G. Pongracz. MACSAD: High performance dataplane applications on the move. *IEEE International Conference on High Performance Switching and Routing, HPSR*, 2017-June, 2017.

[6] Bosshart P, Daly D, Izzard M, N McKeown, J Rexford, C Schlesinger, D Talayco, A Vahdat, G Varghese, and David W. Programming Protocol-Independent Packet Processors. 2013.

[7] Opendataplane. [Online]. Available: http://www.opendataplane.org.

[8] L. Csikor, M. Szalay, B. Sonkoly, and L. Toka. Nfpa: Network function performance analyzer. 2015.