RAUflow: Building Virtual Private Networks with MPLS and OpenFlow

Santiago Vidal Jorge Rodrigo Amaro Emiliano Viotti
Martín Giachino Eduardo Grampín
{santiago.vidal, jorge.amaro, eviotti, giachino, grampin}@fing.edu.uy

Instituto de Computación (INCO)
Universidad de la República (UdelaR)
Montevideo, Uruguay

Abstract

Control and Data Plane separation is a well established networking paradigm, fuelled by the raising of Software Defined Networking (SDN), which foresee the implementation of complex, valuable network behaviour over commodity hardware. The Academic Network of Uruguay (in spanish *Red Académica Uruguaya - RAU*) comprises several universities, research centres and government institutions; RAU is planning a major upgrade, and SDN is a candidate technology to tackle present and future requirements. To test the feasibility of this approach, we built a *RAU2* network prototype composed of NetFPGA based routers over 10G optical links. Over this infrastructure we developed *RAUflow*, a network control application on top of the Ryu SDN controller, which combines OpenFlow and MPLS to implement network services conforming to RAU evolution requirements. In this paper we analyze the deployment and scalability of VPN services over RAU2 prototype.

1 Introduction

RAU is planning a major upgrade, seeking for a boost on coverage and capacity, focused on flexibility for the deployment of new, evolved network services. RAU is built over typical leased line infrastructure provided by ANTEL, the public telecommunications operator, which also provides inter-domain connectivity with the commercial Internet; RAU is part of RedCLARA[1], responsible for the implementation and management of the network infrastructure that interconnects the National Research and Education Networks (NRENs) of Latin America, and provides connectivity with global academic networks such as GÉANT in Europe and Internet2 in USA. RAU is a multi-tenant infrastructure, shared by universities, research centres and government institutions. Therefore, traffic isolation is a major requirement, which leads to the implementation of Virtual Private Networks (VPNs) and/or slicing.

The rest of the paper is organized as follows: in next section 2 we review some related work, while providing some insights about the RAUflow application in section 3. Scalability is analyzed in section 4, closing with a discussion and some directions for future work.

2 Related work

MPLS is the basic building block for Layer 2 and Layer 3 VPNs; while MPLS *data plane* is quite simple, and open source implementations have been around for more than 15 years[2], the *control plane* has become increasingly complex, and therefore costly. The typical Label Switch Router (LSR), implements both forwarding and control plane in the box, which comprises distributed routing protocols such as OSPF or IS-IS for routing information gathering and dissemination, distributed signaling and label distribution mechanisms such as RSVP-TE and LDP, and also MP-BGP (Multi Protocol BGP) for actual VPN routing implementation, as defined by well established IETF standards.

Spring School on Networks, Santiago, Chile, 21-11-2016, published at http://ceur-ws.org

[1] Online: `http://www.redclara.net/`

[2] Online: `http://sourceforge.net/projects/mpls-linux/`, updated by `https://github.com/i-maravic/MPLS-Linux`

Table 1: Comparison of RAUflow vs. legacy MPLS

	Legacy	RAUflow	Comment
Traffic Classification	Per-node FEC	Global edge classification: Open Flow matching fields	No header processing in core routers (avoid CPU intensive process and added delay). Compatibility between Open Flow and legacy networking
Path Computation	Distributed Algorithm (with TE extensions)	Distributed topology database acquisition, centralized SPF algorithm	Centralized ad-hoc Traffic Engineering algorithms, possible delay penalty
Label distribution and LSP signalling	Distributed signalling (LDP/RSVP-TE)	Centralized, per LSP assignment	No need to implement signalling in network nodes (avoid CPU and memory intensive process)
Maintenance routine	Dynamic trigger / dynamic computation and signalling	Dynamic trigger / centralized computation and label assignment	Centralized re-optimization, possible delay penalty

OpenFlow, proposed by McKeown et al. [1] as an standardized interface to add and remove flow entries in a generic Ethernet switch with an internal flow-table, established a foundation for the implementation of the SDN idea: a vendor-independent protocol which defines syntax and semantics for the programmability of the switch by an external/third party controller. In our previous work [2] we reviewed early efforts to implement MPLS forwarding, which has been incorporated as part of OpenFlow specification since version 1.3.

Regarding VPN implementation, the model proposed by Suzuki et al. [3] follows the IETF L3 VPN specification, implementing the route dissemination in a per-customer, centralized BGP instance running in the SDN controller using the Quagga routing suite; in this proposal, the VPN tunnels are implemented using VLAN tags, but the authors foresee to migrate to MPLS.

The Community Connection (CoCo) project[3] proposes an on-demand SDN-based Layer 3 VPN implementation [4], which follows the same approach: MP-BGP information is exchanged among external "CoCo Agents", while the forwarding is also implemented using VLAN tags. The deployment is targeted for sharing resources among the research community, and therefore is particularly interesting for our use case; the authors include some scalability evaluation which partially inspired our work.

The OPEN proposal [5] centralizes all control plane functionality in the SDN controller, and claims to eliminate the need of distributed protocols to implement MPLS-based services. Our approach is similar, but we maintain the IGP (OSPF) as a basic building block for our solution.

3 RAUflow design and implementation

In light of RAU requirements, and considering our previous experience and related work, we designed

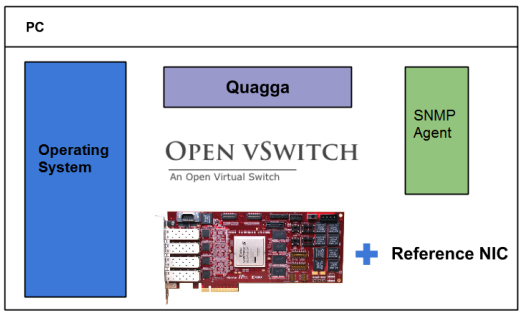

Figure 1: Architecture of the RAUswitch

and built our own open-source switch-router, named *RAUswitch* (Figure 1), made using a standard x86 motherboard with PCIe bus and NetFPGA[4] 10G networking cards, running the Quagga routing suite over Linux OS, therefore supporting legacy routing protocols such as OSPF, which provides dynamic topology discovery and routing information dissemination, which is gathered by the SDN controller to deploy L2 and L3 VPNs[5], as described in [2].

RAUflow is a network control application which permits to define basic services based on MPLS, for instance Layer 2 Virtual Private Wire Service (VPWS), and Layer 3 VPN services. Besides MPLS forwarding (label PUSH, POP and SWAP operations), the implementation of VPN services requires other fundamental functionalities, such as i) traffic classification, ii) a path computation mechanism, iii) MPLS label distribution and Label Switched Path (LSP) signalling mechanisms, and iv) a maintenance routine in response to network dynamics. In addition, MAC ethertype transparency must be supported in L2 services, while IP addressing overlapping and transparent per-customer routing instances must be supported for L3 VPN services.

Table 1 is a summary of the differences between RAUflow and legacy MPLS. Note that RAUflow VPN

[3] Online: `https://blog.surf.nl/en/coco-an-exploration-of-software-defined-networking/`

[4] Online: `http://netfpga.org/`

[5] Updated code and documentation is freely available at: `https://github.com/ProyectoRRAP`

Table 2: Layer 2 VPN setup time in ms.

Path Lenght	Basic	Small	Medium
1	58.3	65.0	83.1
2	N/C	100.8	124.0
4	N/C	105.2	124.1
6	N/C	107.8	131.4
8	N/C	109.6	128.6
10	N/C	N/C	139.8
12	N/C	N/C	133.4

Table 3: Layer 3 VPN setup time in ms.

Path Lenght	Basic	Small	Medium
1	6.3	6.8	19.3
2	N/C	7.8	21.5
4	N/C	9.4	22.3
6	N/C	15.1	23.8
8	N/C	12.0	23.7
10	N/C	N/C	26.4
12	N/C	N/C	27.5

implementation does not use MP-BGP, which is an extra advantage for three reasons: i) complex BGP configuration is avoided, ii) CPU and memory usage are reduced, and iii) BGP messaging churn is completely eliminated.

4 Proof of concept and scalability analysis

We successfully built and performed functional testing of RAUflow over the RAU2 physical testbed, composed by four RAUswitches connected by a full-mesh of 10Gbps optical links, also verifying the maintenance procedure (i.e. re-routing after link failure). The testing scenarios included i) multi-site L3 VPN (one customer), ii) multi-site L3 VPN (two customers with overlapping IP addresses), and iii) point to point L2 Pseudo Wire with VLAN support.

The physical testbed size prevents to run scalability tests, and therefore we ported the complete environment[6] over the well-known Mininet emulator [6].

In the emulated environment, we deployed three representative topologies: i) a basic 4-node full mesh topology, emulating the physical testbed, ii) a small 11-node topology and iii) a medium 45-node topology, both taken from "The Internet Topology Zoo"[7]. Over these topologies, we first re-run the relevant functional tests, and then we explored the setup time for a new VPN. The results are shown in Tables 2 and 3. In a second series of tests, we configure new VPN services cummulatively, measuring both the RAM consumed and the setup time. The results are shown in Figures 2 and 3.

Presumably, the main factors which explain these results are the time consumed by the controller to compute the shortest paths, and the time taken by the configuration of every node in the chosen paths; further and finer grain experiments are needed to confirm such intuition, for instance measuring the breakdown of the controller time across the above mentioned phases.

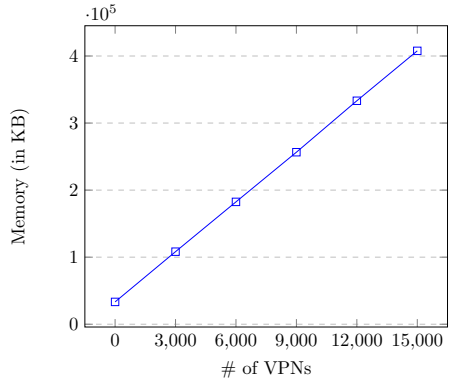

Figure 2: Memory consumption per cummulative VPNs

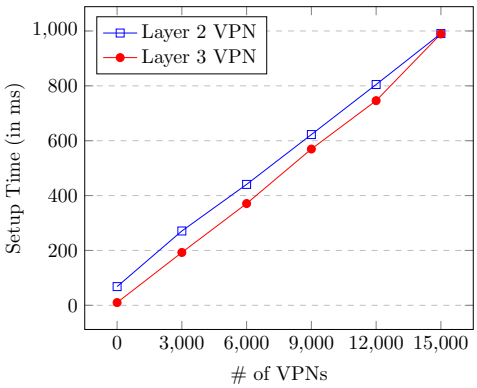

Figure 3: Setup time per cummulative VPNs

5 Conclusion and Future Work

We built a testbed using commodity hardware which suports MPLS-based network services using a centralized, SDN-based control plane. We completely ported the testbed to an emulated environment, which permits to test scalability and facilitates the development of new features. Our preliminar scalability assessment reveals an acceptable behaviour, supporting tens of thousands of services whit linear growth of setup time and consumption of controller resources.

A fair quantitative comparison against legacy MPLS is difficult to execute, since open source MPLS implementations, which can be emulated under the same environment as RAUflow, are far less efficient than industrial ones. Meanwhile, further scalability

[6] Updated code and documentation is freely available at: `https://github.com/santiagovidal/P2015_44`

[7] Online: `http://www.topology-zoo.org/`

testing is being acomplished, while we explore new services and features, emphasizing that SDN adoption would definitively be hybrid and progressive, calling for the emergence of new technical and business models to facilitate the transition, as stated by Vissicchio et al. [7].

References

[1] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner. Openflow: Enabling innovation in campus networks. *SIGCOMM Comput. Commun. Rev.*, 38(2):69–74, Mar. 2008.

[2] E. Grampín, M. Giachino, J. R. Amaro, and E. Viotti. RAU2 testbed: A network prototype for evolved service experimentation. In *LANOMS 2015*, pages 107–108, Oct 2015.

[3] K. H. SUZUKI Kazuya. An OpenFlow Controller for Reducing Operational Cost of IP-VPNs. *NEC Technical Journal*, 8(2):49–52, April 2014.

[4] R. van der Pol, B. Gijsen, P. Zuraniewski, D. F. C. Romão, and M. Kaat. Assessment of SDN technology for an easy-to-use VPN service. *Future Generation Computer Systems*, 56:295 – 302, 2016.

[5] S. Das, A. Sharafat, G. Parulkar, and N. McKeown. MPLS with a simple open control plane. In *Optical Fiber Communication Conference and Exposition (OFC/NFOEC), 2011*, pages 1–3, March 2011.

[6] B. Lantz, B. Heller, and N. McKeown. A network in a laptop: Rapid prototyping for software-defined networks. In *Proc. of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, Hotnets-IX, pages 19:1–19:6, New York, NY, USA, 2010. ACM.

[7] S. Vissicchio, L. Vanbever, and O. Bonaventure. Opportunities and research challenges of hybrid software defined networks. *SIGCOMM Comput. Commun. Rev.*, 44(2):70–75, Apr. 2014.