

Revisiting Routing Control Platforms with the Eyes and Muscles of Software-Defined Networking

Christian E. Rothenberg,
Marcelo R. Nascimento,
Marcos R. Salvador
Telecomm. Research and
Development Center (CPqD)
Campinas - SP - Brazil
estev@cpqd.com.br

Carlos N. A. Corrêa,
Sidney C. de Lucena
Federal University of the Rio
de Janeiro State (Unirio)
Rio de Janeiro - RJ - Brazil
carlos.correa@uniriotec.br

Robert Raszuk
NTT MCL
San Mateo - California - USA
rr@nttmcl.com

ABSTRACT

Prior work on centralized Routing Control Platform (RCP) has shown many benefits in flexible routing, enhanced security, and ISP connectivity management tasks. In this paper, we discuss RCPs in the context of OpenFlow/SDN, describing potential use cases and identifying deployment challenges and advantages. We propose a controller-centric hybrid networking model and present the design of the RouteFlow Control Platform (RFCP) along the prototype implementation of an AS-wide abstract BGP routing service.

Categories and Subject Descriptors

C.2.1 [Network Architecture and Design]: Store and forward networks; C.2.2 [Network Protocols]: Routing protocols; C.2.6 [Internetworking]: Routers

Keywords

Routing architecture, OpenFlow, SDN, BGP, virtualization

1. INTRODUCTION

Providing a clean separation of functions is paramount to any system or network architecture. In networking, we have seen multiple flavors of successful split architectures such as SS7, IMS/NGN, GMPLS, PCE, and so on. Recently, in the switching/routing networking realm, this architectural approach is being touted as Software-Defined Networking (SDN) and is based on new abstractions that effectively decouple data-plane forwarding from control and management plane functions. As a means to realize the so-sought clean separation of functions [11], OpenFlow is arising as the protocol (or API) of choice between controller and forwarding elements. New services, feature velocity and cost reduction are expected benefits of SDN.

Examples of different SDN use cases are increasingly popping up, some already in commercial form and a handful

of them in academic/lab environments. Curiously, little attention has been paid to the implications of SDN meeting traditional IP routing and forwarding. IP routing protocols have been studied and investigated to such an extent that nowadays are naturally considered uncool/cold research areas. SDN-driven APIs provide a unique opportunity to reheat cold topics around IP routing (e.g. BGP) by deploying and evaluating in real conditions (past and new) research results.¹ OpenFlow opens a real possibility to the direct modification of current routing² services by users and operators without the costly dependence on vendors. The level of modifications can range from minor additional debugging and show commands (to simplify most tedious operational tasks) to modifying libraries for prefix/interface dampening or even modifying routing itself in order to allow for new service proposals to customers.

Around one decade ago, research on so-called Routing Control Platforms (RCP) [1, 7, 24, 26] pioneered refactoring the IP routing architecture to create a logically centralized control plane separated from forwarding elements to focus on the BGP decision process and the route control needs from a large operator's perspective. In contrast to the complex distribution of BGP configuration across many routers, RCPs allow an individual AS to easily deploy new, customer-facing (and revenue-generating) services [24]. Moreover, such approach also provides a very good perspective on network effectiveness, for instance to perform optimal best path calculation on a per client or group of clients basis [17, 25]. Today, AT&T operates a RCP (aka IRSCP [24]) differentiating its "dynamic connectivity management" offer.

In this paper, we re-examine the concept of BGP-based RCPs (Sec. 2) with the visibility and direct control capabilities (i.e., actual FIB installation, rich matching and instructions) of an OpenFlow-based SDN approach. Building upon our earlier work on RouteFlow [16], we present our envisioned hybrid networking model and describe the new datastore-centric platform design (Sec. 3) that allows the introduction of advanced routing services. Under the lens of OpenFlow/SDN, we rescue use cases from previous work and introduce new applications (Sec. 4). We then describe de-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

HotSDN'12, August 13, 2012, Helsinki, Finland.

Copyright 2012 ACM 978-1-4503-1477-0/12/08 ...\$15.00.

¹Arguably, OpenFlow/SDN is a (1) new technology, that allows (2) getting back to basics, with potential to (3) multi-disciplinize or even (4) extrapolate on IP control (cf. [4]).

²While inter-domain protocols are quite robust in the loop/error handling of misbehaving implementations, for IGPs that may not really be the same.

scribe our prototype (Sec. 5) of a scale-out aggregated BGP router, showing that a RouteFlow Control Platform (RFCP) can be used to implement routing applications by acting as an indirection layer for control protocol messages and RIB-(to-FIB)-to-OpenFlow transformations. Finally, we discuss challenges and identify benefits (Sec. 6) of the proposed architecture and conclude (Sec. 7) with final remarks and lines of future work.

2. BACKGROUND AND CONTEXT

There are three cases of BGP deployment within ASes today: full mesh, confederations, and route reflection, which is the most popular way to distribute BGP routes between routers of the same AS. Avoiding routing pitfalls (e.g., oscillations, loops, path inefficiencies) requires careful topology design, route reflector (RR) placement, and link-metric assignment. The left half of Fig. 1 shows a typical Tier-1 ISP network divided into data-plane core, control-plane core, and the edge, consisting of specializable devices known as Provider Edge (PE) routers, that peer with external ASes and are configured to provide services to Customer Edge (CE) devices.

Traditionally, RRs have been deployed in the data path and carefully placed on the edge to core boundaries. That model has started to evolve taking RRs out of the data paths to deliver applications requiring PE-PE encapsulation (e.g. L3VPN). With edge to edge MPLS or IP encapsulation also used to carry Internet traffic, this model has been gradually extended to other BGP address families, including IPv4/v6 Internet routing.

Very often, control plane RRs are placed in arbitrary core locations of large networks. The major issue of such deployment models relies in the fact that best path selection (e.g. hot potato) is performed by the RRs from their IGP reference point and gets propagated to each RR client in very different locations. Clearly, that such best path is not optimal for a lot of RR clients.

To provide finer-controlled, advanced routing capabilities while overcoming operational expenses, inefficiencies, and complexities of BGP at Tier-1 service and device scales, proposed RCPs [1, 7, 24, 26] argue for more flexible and intelligent route control by implementing a logically centralized routing element that colligates the routing decision process. In essence, RCPs are based on three architectural principles [7]: i) path computation based on a consistent view of network state, ii) controlled interactions between routing protocol layers, and iii) expressive specification of routing policies. The SoftRouter [13] is another noteworthy architectural work that promotes the separation and logical centralization of control plane using the IETF ForCES protocol.

With backwards compatibility in mind and lacking of a standardized mechanism (e.g., ForCES, OpenFlow), previous work on RCPs was constrained to iBGP-based solutions. Similar to RRs, iBGP-speaking RCPs limit the number of routes they can learn. Even when centralizing eBGP control [24], the FIB installation in edge devices follows iBGP-driven IP prefix rules, still distributed and effectively done by the routers themselves.

In this paper, we investigate an OpenFlow/SDN approach to IP routing control which can easily implement eBGP-speaking applications, providing full route control and direct manipulation of the actual FIBs (now OpenFlow tables), augmented with flexible flow match and rich instruc-

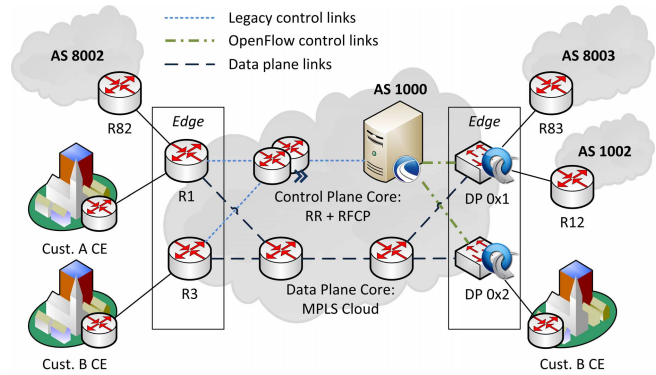


Figure 1: Routing architectures: Current (left) vs. OpenFlow/SDN-based “BGP-free edge” (right).

tion capabilities beyond basic next-hop forwarding. Replacing iBGP with OpenFlow can be seen as an (orthogonal) advancement of phases 2 and 3 in the RCP taxonomy [7], heading towards an effective “BGP-free edge” architectural proposal for intra-domain BGP deployments. While some of the use cases we discuss in this paper might be feasible with a mashup of traditional routing protocol and box-centric configurations, our work differentiates in being truly open and promising simplicity, reduced cost, and vendor-independence. The resulting architecture is however not a revolution, but arguably an evolution of current iBGP RRs to essentially eBGP Route Controllers—a completely different perspective and solution to intra-domain BGP routing problems. The BGP control plane is removed from both core and edge devices and shifted into a higher control layer running on one or more generic computing machines. Only their final product gets distributed via OpenFlow to datapath switches.

3. DESIGN

3.1 Hybrid networking model

There are number of ways one could define the meaning of *hybrid network*, where hybrid is the co-existence of traditional environments of closed vendor’s routers and switches with new OpenFlow-enabled devices. However, for very clear decoupling as well for avoiding any upgrade of existing devices, the proposed hybrid approach refers to the interconnection of both control and data plane of legacy and new network elements.

To assure a smooth migration, our starting point is the traditional network model of PEs interconnected via iBGP through RRs. As shown in the right half of Fig. 1, the new control plane is added in the form of a BGP controller (further referred to as RFCP) that acts as a gateway between existing RRs and OpenFlow controllers programming the datapaths. The scale of such deployment can vary and completely be controlled by network operator.

3.2 RFCP Components

RFCP is an evolution from previous prototype designs [16] to a better layered, distributed system, flexible enough to accommodate different virtualization use cases ($m : n$ mapping of routing engine virtual interfaces to physical OpenFlow-enabled ports) and ease the development of advanced routing-

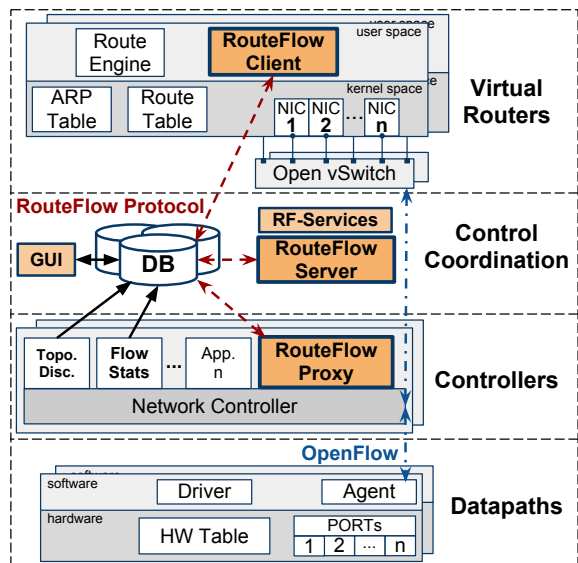


Figure 2: Architecture design of the RFCP.

oriented applications. Anticipating the need for updating (or even replacing) parts of the RFCP architecture while facilitating multi-controller support, the implementation is segregated into the following three components (cf. Fig. 2):

- **RF-Client:** Collects routing and forwarding information generated by the routing engine (e.g. Quagga) of the Linux system,³ where it runs as a user-space daemon. Optionally, to extract additional routing information (e.g. all BGP paths in the RIB-in), it hooks into or peers (e.g. iBGP) with the routing engine(s).⁴
- **RF-Server:** Standalone application responsible for the system’s core logic (e.g., event processing, VM-to-DP mapping, etc.). RFCP Services are implemented as operator-tailored modules which use the knowledge information base to deliver arbitrary, high-level routing logics (e.g., load balancing, preferred exit points, etc.).
- **RF-Proxy:** Simple ‘proxy’ application on top of an OpenFlow controller (e.g., NOX, POX) which serves the RFCP with switch interaction and state collected from topology discovery and monitoring applications.

In line with the best design practices of cloud applications, we rely on a scalable, fault-tolerant datastore that centralizes (i) the RFCP core state (e.g., resource associations), (ii) the network view (logical, physical, and protocol-specific), and (iii) any information base (e.g., traffic histogram/forecasts, flow monitoring, administrative policies) used to develop routing applications. Hence, the datastore embodies so-called Network Information Base (NIB) [22] and Knowledge Information Base (KIB) [5]. In addition, the distributed NoSQL database of choice (e.g., MongoDB, Redis, Cassandra) is used as the pubsub-like message queuing IPC that loosely couples the modules via an extensible JSON-based implementation of the RouteFlow protocol. Without sacrificing performance, the datastore-based IPC eases fault-management, debugging, and monitoring.

³Typically, a lightweight virtual container like LXC.

⁴We are also ready to collect information from commercial BGP implementations as soon as they export RIB with SDK-like APIs (with read-only access at the initial phase).

Altogether, the RFCP design tries to follow principles that allow architectural evolvability [9]: layers of indirection, system modularity, and interface extensibility.

3.3 Protocols and Abstractions

The RouteFlow protocol glues together the different modules with a simple command/response syntax northbound to the RF-Clients and a subset of the OpenFlow messages southbound to the RF-Proxy application. As one approach to OpenFlow polyglotism, the RouteFlow protocol layer abstracts most of the details of the differences from controller implementing OpenFlow versions 1.X and the slightly higher-level RouteFlow APIs.

Besides being the API to the datapaths, OpenFlow is used as the vehicle to deliver control plane messages from/to the interfaces of the VMs running the routing engine and the RF-Client. Programming the virtual switches (OVS in Fig. 2), we can select a mode of operation where routing protocol messages are sent ‘down’ to the physical devices or are kept ‘up’ in the virtual network plane, which can be a reproduction of the discovered physical connectivity or a simplified/arbitrary version (e.g. single router abstraction [12]) of the HW resources. In the former, messages follow the physical path, so no additional fault-detection mechanism are required at the cost of extra propagation delays. In the latter, signaling is kept in the virtual domain, benefiting from low latencies and contributing to better scalability. This option provides a more natural support for virtual networks but requires extra programmability and fault-detection extensions to update VMs’ logical associations due to physical link changes.

All in all, the resulting software artifact is capable of instructing the SDN data plane with regard to IP forwarding and addresses the following premises:

- Ability to define desired routing criteria (i.e. business policies) in an abstraction level which does not require individual configuration of multiple routing processes;
- An unified, programmer-friendly database that conveys knowledge and network information, allowing for provisioning and traffic-engineered route control services.
- Scalable virtual control plane elements, which provide strong automation and flexibility for in-memory representation of intended control scenarios.

4. USE CASES

Pioneering work on RCP has shown the benefits in more flexible routing [26], security [19] or connectivity management [24]. Not requiring changes to routers or protocols, previous use cases (e.g. planned maintenance dry-outs [24]) could be easily supported. For the sake of novelty, we will pay special attention to new OpenFlow-enabled scenarios, previously unfeasible or harder to implement in multi-vendor environments.

Engineered path selection. Since the RFCP knows all available paths, it is able to offer advanced load-balancing and cost/performance path selection [6] [15] [26] per application or customer using OpenFlow flexible matching to push traffic into traffic-engineered paths or VRFs. Moreover, OpenFlow allows for direct preparation for optical switching in the core as well edge to edge. Recent research [21] has shown that network-wide visibility and control enable joint traffic engineering and failure recovery by balancing load efficiently.

Optimal best path reflection. RRs may not yield the same route selection result as that of a full iBGP mesh, thwarting the ability to achieve hot potato routing in the perspective of the RRs clients –as the choice of exit point is the closest to the RR [17]. With full route visibility and OpenFlow control, the “optimal” route selection and FIB ‘download’ can be done from each datapath’s perspective, with the flexibility to calculate customized BGP best paths for each ingress point and also per customer in case of L3VPNs.

Path protection with prefix independent convergence. RFCP’s complete view of route state allows for novel IP route protection schemes like per-edge computation of primary and secondary paths for each prefix, considering shared risk group inferences [5]. OpenFlow v1.1 group- and multi-tables allow for an implementation of hierarchical FIB organization [8] to quickly recover from edge failures. Network-wide, loop-free alternate next hops can be pre-calculated and maintained in the group tables, removing the burden of one flow-mod per prefix updates, and allowing a fast local repair –provided an OAM extension is available, for instance, triggered by a Bidirectional Forwarding Detection (BFD) state change. The common design principle is moving away from dynamic route computation by decoupling failure recovery from path computation [2].

Data plane security. OpenFlow/SDN can help, firstly, to identify attacks, e.g. interposing an OpenFlow switch just to monitor the flows and try to recognize abnormal patterns. SDN can distribute firewall and IPS/IDS functionality along the path, potentially on a customer basis, instead of either forcing all traffic to pass through or be detoured to a specialized box. Once detected, a selective DDoS blackholing strategy (akin [24]) can be deployed by inserting higher priority flow entries with drop action only on the inports under attack. OpenFlow fine granular rules (*inport,src,dst,tp – port*) allow to blackhole traffic without taking out of service the target IPs (as with today’s common filtering and BGP-based mechanisms). When supported, RFCP can request upstream ASes to perform inbound filtering by triggering BGP-based flow filtering [14]. Similar filtering capabilities can be employed to duplicate traffic for legal intercept purposes.

Secure inter-domain routing. BGP central processing empowers a single AS by greatly simplifying the enforcement of BGP origin validation and benefits route selection of those trustable ASes participating in address registries, PKI, DNSSEC, or new security techniques beyond today’s concepts. For instance, Morpheus [19] proposes algorithms (e.g. history-based anomaly detection) to achieve ‘pretty good’ BGP security without requiring massive S*BGP adoption [10]. An RFCP-like approach eliminates duplicate processing plus the non-trivial SW and HW upgrades of AS border routers. Deployment gets easier due to plenty CPU and crypto assistance on the controller, which centralizes key management and reduces the signing overheads to once per neighbor rather than per peering location.

Simplifying customer multi-homing and IPv6 migration. Enterprise networks lack of customer-friendly, effective multi-homing solutions, especially in the absence of PI addresses or dual-stack IPv4/v6. RFCP could be used as an upstream affinity mechanism, only sending packets sourced from ISP1 networks via ISP1 (and 2 via 2) by flow matching based on source IP subnet. Effectively, it deliv-

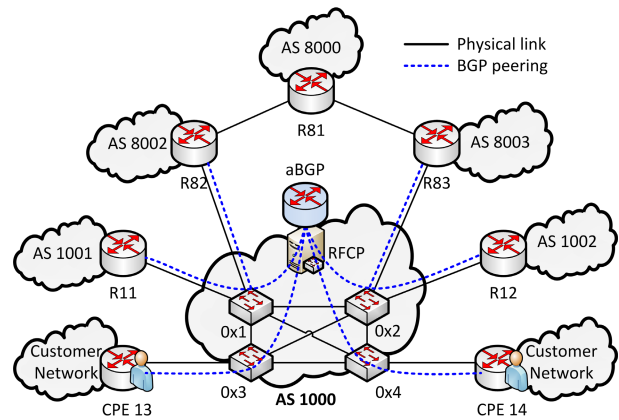


Figure 3: Proof of concept prototype

ers a VRF per WAN link with source address-based VRF selection for egress traffic. The OpenFlow API allows to update flows based on performance- and/or cost [15]. Simultaneously, edge migration to IPv6 could be accelerated and cheaply rolled out with OpenFlow match plus encapsulate actions, potentially enabling new services that leverage IPv6’s strengths and mitigate control plane weaknesses.

5. PROTOTYPE IMPLEMENTATION

Just like RRs act on the routes learned, RFCP-Services play the role of a new processing layer in route computation. While the computation of both RRs and clients are essentially the same (just adjusted to wider topology views), RFCP-Services are written as (C++) RF-Server modules, which can virtually implement any routing logic. RF-Proxy has been implemented for the NOX and POX OpenFlow controllers (with Floodlight and Trema support planned). The RF-Client component works with Quagga and XORP as the Linux-based routing engines running in LXC containers. MongoDB is the NoSQL database of choice due to its user-friendly JSON orientation and the scalability options. It serves both as the backend for the main state storage and as the IPC between the RFCP components.

The proof of concept prototype uses a configuration grammar expressive enough to entail (i) a selection of parameterized RFCP-Services to be applied in the control plane, and (ii) the characteristics of the data plane topology. The dataplane domain can be described by datapath (DP) IDs and links, which can be separated into “trunk” (DP-to-DP) and “edge” (DP-to-device) categories. A descriptive grammar written in lex/yacc was derived from a high-level model encompassing descriptive data of interest.

To verify the platform’s ability to supplement forwarding decisions through the RFCP, we implemented an actual routing service called “aBGP” (or “Aggregated BGP”) that uses a topology description where all edge links are mapped to interfaces of a single virtualized routing engine. The proof of concept case study is depicted in Fig. 3, where every AS 1000 edge switch is mapped to a single VM running a single BGP process that terminates all peering sessions. This single router abstraction (cf. [12]) is opposed to the maintenance of four routers’ configurations.

When adding flows for a destination IP prefix d into a DP n , which is at the best exit point to d according to the best

route selection, aBGP installs in n a prefix-length-based prioritized flow entry matching d with port-forwarding plus L2 re-writing actions. In every other DP, the installed flow actions are forwarding to DP n over their respective paths (e.g. IP/LSP tunnel). Hot potato can then be implemented by inspecting the RIB for multiple path entries and changing the IP prefix match action to inter-AS port-out interfaces. Experimental results show correct BGP operations and forwarding through AS 1000.

We have demoed RouteFlow with commercial OpenFlow devices from Pronto, NEC and IBM, and verified BGP and OSPF interoperability with Juniper MX and Cisco devices. A pilot instance of RouteFlow is used to control experimental traffic at Indiana University.⁵ The experimental deployment is researching whether very different user interfaces can be built with SDN backends, and how to make campus network administration simpler to implement, more robust and consistent, and easier to manage by means of OpenFlow/SDN automation and abstractions.

6. DISCUSSION

We now discuss known challenges of the state of the art. Once resolved, which we expect shortly, the proposed architecture promises many new benefits.

6.1 Known Unknowns

Centralized BGP. Given the advances in (distributed) computational power and considering performance results of previous work [24, 26], RFCP-like systems shall be able to scale to maintain 10,000s of eBGP sessions, perform routing decisions for 100s of PEs, and process updates and store BGP routes from CEs. Further scalability can be achieved by implementing recent ideas [3] on RR BGP distribution based on chunks of address spaces rather than some fraction of speakers.

OpenFlow processing in datapath. While CPU power in the controller platform is abundant, cheap, and can be arbitrary scaled, CPUs of network devices limit the supported rate of OpenFlow operations. Publicly discussed numbers are in the order of few 100s of flow-mod/sec, and decrease with resource competing protocol tasks like flow-stat processing. We believe these numbers are due to unoptimized implementations. Nevertheless, they shall alert that large FIB updates could be very costly and thus the importance of PIC [8] implementations and pro-active backup flow installations, as discussed earlier in Section 4.

OpenFlow table size. Many commercial OpenFlow devices re-use existing TCAM enough for only 1000s of flow entries. Considering that we need at least one entry per destination subnet,⁶ current HW configurations would be not enough for ISP production environments. Again, we believe this is a transient limitation and expect new OpenFlow optimized devices with larger flow capacity, potentially exposing existing L3 and L2 forwarding engines as tables in OpenFlow v1.X pipeline. On the positive side, the RFCP is in an excellent position to eliminate redundant state in the RIB and FIB by executing overlapping prefix suppres-

sion [18]. Alternatively, SW flow offloading decision engines could be used to save the precious flow space in HW [20].

High availability. Based on previously distributed implementations of RCPs [24, 26] and OpenFlow controllers [22], we are confident that the datastore-centric RFCP design has enough foundations to be fault-tolerant. Helpful techniques include RR-like route distribution among RFCP instances or maintaining multiple eBGP sessions with each CE. A failure of the BGP controller application would result in BGP session drops to CEs. To avoid service disruption, BGP route engines are not only physically distributed within each cluster, but moreover and completely transparent to the CE routers, we advocate the introduction of multiple RFCP clusters into the architecture by means of a newly defined BGP SHIM function in the OF edge switch. The BGP SHIM intercepts CE sessions and mirrors them (without need for parsing update messages) to multiple BGP RC clusters. That allows for very robust yet simple design which is arguably more robust than traditional PE local processing and RR advertisements.

6.2 The Promises

There is a number of advantages to the proposed IP split (hybrid) architecture, some applicable in general and inherent from OpenFlow/SDN, and some BGP-specific:

Simplified edge architecture. Relaxed requirements for every edge device to be fully capable of handling any new service or extension. Eliminated the need to process, store and maintain effectively the same set of control plane data and perform the same tasks across large number of edge platforms.

Lower cost and increased edge speed. Leveraging commodity switches and remote open-source routing software⁷ decouples requirements for HW upgrades enforced by vendors due to end-of-life of particular OS. Increase in edge speed by closely following (with smaller CAPEX) the switching silicon latest technology curve, possibly optimized for flow switching.

Power of innovation leads to differentiation followed by new revenues. The ability to innovate in the network either by internal dev/ops teams or by vendor-independent third party products allows to differentiate the operator’s services portfolio –without the need to convince vendors for support and practically sharing the innovations with other operators. Differentiation via uniquely customized network services allows much faster revenue opportunities for service providers.

BGP security, stability, monitoring, and policy management. As argued earlier, new ideas around BGP security become viable and cost-effective when executed in high CPU systems. Control plane stability (reduction of well-known BGP wave effect) can be increased by elimination of intra-domain BGP path oscillations [23]. BGP monitoring and reporting interfaces can be easily implemented since there is no need to collect all BGP “raw” feeds (aka original intention of BMP) from all border routers. APIs from the RFCP datastore enable view of entire BGP in an AS. Centralization of BGP policy management is a major gain in OPEX reduction and configuration error avoidance.

⁵Topology and an open-accessible UI are available here: <http://routeflow.incentre.iu.edu/>

⁶Conservative estimate assuming v.1.1 multiple tables that reduce cross-product state and enable efficient VRF.

⁷Customers may prefer solutions from major vendors instead of open-source alternatives. Such VM appliances allow for RIB events APIs that can feed the RFCP.

7. CONCLUSION AND WORK AHEAD

In this paper, we discussed how centralized BGP-speaking routing engines coupled with OpenFlow-based installation of IP-oriented flow rules leads to a new degree of control and enables a number of applications that are a real challenge (if not impossible) to realize in classic multi-vendor networks.

The proposed RFCP allows for an incrementally deployable strategy to roll-out OpenFlow-enabled devices following a “hybrid” controller-centric inter-networking approach. We reckon that a more “clean” and efficient replacement of legacy routing control protocols shall be pursued, leveraging the value of existing network configurations while addressing the potential impedance of admin incumbents.

We are devoting efforts on the proof of concept implementation of new BGP-based applications and the deployment of pilots for the under-served mid-market (SMEs and regional ISPs) for which we believe SDN technology may be very appealing. Already underway, enhancements to the platform include a converged dashboard GUI, coupled with historic data and a configuration repository available in the MongoDB. New RFCP services include load-balancing in multi-homed networks and feeding an ALTO server prototype. Ongoing investigations around OSPF and IS-IS promise advantages to ease configuration management and unlock protocol optimizations (e.g. IP Fast Reroute/LFA). Our roadmap includes LDP support to define MPLS paths, in addition to OAM considerations and OpenFlow v1.X advancements, such as IPv6 and overall extensibility.

8. ACKNOWLEDGMENTS

This work is partially supported by the GIGA Project (FINEP/FUNTEL). Carlos N. A. Corrêa was supported by Unimed Federação RJ. The authors are thankful to all contributors of the RouteFlow project. For their valuable comments and suggestions, the authors are grateful to Maurício Magalhães, Jennifer Rexford, Brandon Heller, and the anonymous reviewers.

9. REFERENCES

- [1] M. Caesar, D. Caldwell, N. Feamster, J. Rexford, A. Shaikh, and J. van der Merwe. Design and implementation of a routing control platform. In *NSDI'05*, 2005.
- [2] M. Caesar, M. Casado, T. Koponen, J. Rexford, and S. Shenker. Dynamic route recomputation considered harmful. *SIGCOMM CCR*, 40:66–71, April 2010.
- [3] R. Chen, A. Shaikh, J. Wang, and P. Francis. Address-based route reflection. In *CoNEXT '11*, 2011.
- [4] J. Crowcroft. Reheating cold topics (in networking). *SIGCOMM CCR*, 39:48–49, March 2009.
- [5] D. Saucez et al. Low-level design specification of the machine learning engine. FP7 ECODE Deliverable D2.3, Dec 2011.
- [6] N. Duffield, K. Gopalan, M. R. Hines, A. Shaikh, and J. E. Van Der Merwe. Measurement informed route selection. In *PAM'07*, 2007.
- [7] N. Feamster, H. Balakrishnan, J. Rexford, A. Shaikh, and J. van der Merwe. The case for separating routing from routers. In *FDNA '04*, 2004.
- [8] C. Filsfils and et al. BGP Prefix Independent Convergence (PIC) Technical Report. Technical report, Cisco, 2011.
- [9] A. Ghodsi, S. Shenker, T. Koponen, A. Singla, B. Raghavan, and J. Wilcox. Intelligent design enables architectural evolution. In *HotNets '11*, 2011.
- [10] P. Gill, M. Schapira, and S. Goldberg. Let the market drive deployment: a strategy for transitioning to BGP security. In *SIGCOMM '11*, 2011.
- [11] A. Greenberg, G. Hjalmtysson, D. A. Maltz, A. Myers, J. Rexford, G. Xie, H. Yan, J. Zhan, and H. Zhang. A clean slate 4D approach to network control and management. *SIGCOMM CCR*, 35(5):41–54, 2005.
- [12] E. Keller and J. Rexford. The ‘Platform as a Service’ model for networking. In *INM/WREN 10*, Apr. 2010.
- [13] T. V. Lakshman and et al. The SoftRouter architecture. In *HotNets-III*, 2004.
- [14] P. Marques, N. Sheth, R. Raszuk, B. Greene, J. Mauch, and D. McPherson. Dissemination of Flow Specification Rules. RFC 5575, Aug. 2009.
- [15] M. Motiwala, A. Dhamdhere, N. Feamster, and A. Lakhina. Towards a cost model for network traffic. *SIGCOMM Comput. Commun. Rev.*, 42(1):54–60.
- [16] M. R. Nascimento, C. E. Rothenberg, M. R. Salvador, C. N. A. Corrêa, S. C. de Lucena, and M. F. Magalhães. Virtual Routers as a Service: the RouteFlow approach leveraging Software-Defined Networks. In *CFI '11*, 2011.
- [17] R. Raszuk, C. Cassar, E. Aman, and B. Decraene. BGP Optimal Route Reflection (BGP-ORR). I-D draft-ietf-idr-bgp-optimal-route-reflection-01, September 2011.
- [18] R. Raszuk, A. Lo, L. Zhang, and X. Xu. Simple Virtual Aggregation (S-VA). I-D draft-ietf-grow-simple-va-04.txt, September 2011.
- [19] J. Rexford and J. Feigenbaum. Incrementally-Deployable Security for Interdomain Routing. In *CATCH '09*, 2009.
- [20] N. Sarrar, S. Uhlig, A. Feldmann, R. Sherwood, and X. Huang. Leveraging Zipf’s law for traffic offloading. *SIGCOMM CCR*, 42(1):16–22.
- [21] M. Suchara, D. Xu, R. Doverspike, D. Johnson, and J. Rexford. Network architecture for joint failure recovery and traffic engineering. In *SIGMETRICS '11*, 2011.
- [22] T. Koponen and et al. Onix: A Distributed Control Platform for Large-scale Production Networks. In *OSDI '10*, Oct 2010.
- [23] R. Teixeira, A. Shaikh, T. Griffin, and J. Rexford. Dynamics of hot-potato routing in IP networks. *SIGMETRICS '04/Performance '04*, June 2004.
- [24] J. Van der Merwe and et al. Dynamic connectivity management with an intelligent route service control point. In *INM '06*, 2006.
- [25] I. Varlashkin and R. Raszuk. Carrying next-hop cost information in BGP. I-D draft-ietf-idr-bgp-nh-cost-00, January 2012.
- [26] Y. Wang, I. Avramopoulos, and J. Rexford. Design for configurability: rethinking interdomain routing policies from the ground up. *IEEE J.Sel. A. Commun.*, 27:336–348, April 2009.