A framework for flexible interdomain routing in transit ISPs

Trung Truong, Ian Welch and Bryan Ng
School of Engineering and
Computer Science
Victoria University of Wellington
Wellington, New Zealand
Email: {huutrung.truong,ian.welch, bryan.ng}@ecs.vuw.ac.nz

***Abstract*—ISPs face difficulties in optimizing interdomain routing due to the single path routing constraint of BGP. A single best path makes it difficult to optimize routing and is unable to satisfy various application requirements of customers. In this paper, we present a new routing architecture for ISP edge network which breaks this constraint and allows the operator to flexibly assign interdomain paths to customers and to use these paths for forwarding. We leverage Software Defined Networking (SDN) to implement this new design while also maintaining compatibility with existing networks. Our preliminary results show that the system is capable of flexible routing and can be integrated to existing infrastructure.**

1. Introduction

The Internet is composed of a large number of networks called Autonomous Systems (ASes), each of which is owned and operated independently by different organizations. Internet Service Providers (ISPs) operate transit ASes to provide Internet transit service to their customers. Transit ASes relay traffic from source to destination, and so play a crucial role in maintaining the connectivity and performance of the Internet.

Border Gateway Protocol (BGP) is a de facto protocol for routing between ASes [1]. However, BGP was designed to select and distribute a single "best path" for each destination prefix [2]. This unipath approach fails to exploit the path diversity exhibited in the Internet for improving end-to-end routing performance [3]. Furthermore, BGP paths are often suboptimal in terms of round-trip time, loss rate and bandwidth and there usually exists an alternative path with better performance [4]. In BGP routing, transit providers can autonomously select best routes according to its local policy so that some routing requirements, for instance traffic engineering (TE) goals are fulfilled.

From a customer's perspective, the provider's notion of "best route" does not hold in all circumstances. It is common that stub ASes select best routes based on performance metrics such as delay or packet loss rate [5], [6]. Requirements for best routes also vary between customers and applications. For example, remote surgery requires a highly reliable route and on-demand video needs a high bandwidth one. Thus, one customer may have different route requirements from others. It is apparent that best routes selected by the provider cannot fully satisfy diverse routing requirements.

The current routing architecture makes it difficult for ISPs to satisfy the diverse requirements of their customers because of the network's inability to use multiple paths simultaneously. Moreover, the one-route-fit-all restriction does not provide the transit providers with enough flexibility to achieve multiple policy objectives and to optimize routing [7]. As illustrated in Figure 1, the transit AS1 learns four paths to the destination prefix *1.0.0.0/24*. However, standard BGP limits the two border routers $R3$ and $R4$ to the selection of a single best path, e.g. via $P1$ and $P2$. Similarly, two routers $R1$ and $R2$ can only choose between a path selected by $R3$ or $R4$ for forwarding. Customer $C1$ and $C2$ are forced to use the same path for their traffic to the destination prefix.

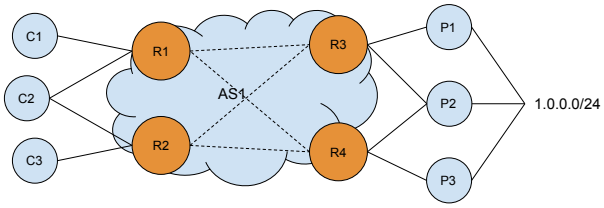

Figure 1: Routing in ISP networks

Several studies have investigated multipath routing for interdomain routing to allow flexibility and better performance [8], [9]. However, the deployment of multipath routing across the Internet is challenging due to its scale and the lack of incentives. Instead of looking at an Internet-wide solution, this paper focuses on a deployable design for enabling routing flexibility only between transit ASes and their customers. As an ISP and its customers already have business contract, the provider has an incentive to deploy such a solution that benefits itself and the customers. Given a critical role

of transit ASes, a non-disruptive solution is necessary for successful deployment.

In this paper, we present a routing architecture for ISPs which utilizes SDN principles and capabilities for routing flexibility, compatibility and deployability. SDN is a emerging networking paradigm where the control plane and data plane are decoupled. The control plane decides how traffic is routed and data plane performs the actual forwarding. [10], [11]. This decoupling enables innovation and simplifies network management. In SDN, network devices are simple packet forwarders as their intelligence is now logically centralized in software-based controllers. Network programmability is enabled through APIs provided by the controller.

The new routing framework removes the constraint of BGP which limits a border router from having single route per prefix and enables finer-grained flows. This allows the assignment of traffic from different neighbors entering the same ingress router to different edge links of the same egress router. SDN is utilized for designing a compatible and incrementally deployable routing architecture which provides greater flexibility in interdomain routing. Our evaluation shows that the system can be easily integrated into existing networks. The system can be configured using policies expressed using the Routing Policy Specification[1] (RPSL). With the capability to flexibly assign and switch any learned path between neighboring AS, the operator may now express policies that satisfy various routing requirements.

2. Flexible routing architecture

Our main goal is to design a scalable, flexible and incrementally deployable multipath interdomain routing system. The system must be compatible with the current interdomain routing protocol and services. Thus, we leverage existing technologies and commercially available hardware. We also aim at a simple and familiar interface between the operator and the system, which preserves the wealth of operational knowledge and expertise embedded in BGP. It must support incremental deployment so that a transit AS can adopt the solution without cooperating with and relying on others.

Our design follows the edge-core separation concept in which the core network is in charge of intradomain routing while the edge network is completely responsible for the interdomain task. This focus on the edge only will make the deployment easier and less disruptive. In addition, this design allows the core and edge network to evolve independently. For example, the core forwarding problem can be addressed using novel techniques such as in [12]. Interaction between the edge and core is purely made in data plane thus improving overall performance and scalability.

The high level architecture is illustrated in Fig. 2. The system is composed of a centralized route controller (RC) and multiple provider edge (PE) routers of which each is controlled by a local controller (called Forward Controller or FC). The RC is responsible for processing BGP updates and computing routes. It learns external routes from neighbors via eBGP. The operator configure the RC with routing policies specified in RPSL. A PE can be composed of multiple OpenFlow switches. FCs are responsible for establishing forwarding paths and mapping traffic to paths. This hierarchical architecture enables the data plane to scale without modifications to the routing control. For example, a PE can be upgraded to support more customers and to accommodate the growth of traffic as well as the forwarding table. This design also allows techniques for scalability such as FIB compression [13], [14] to be implemented without affecting the rest of the system design.

We believe the key to enabling routing flexibility is that the system should be able to compute routes to a destination prefix differently for each neighbor, regardless of their attachment point. As example in Fig. 1, the operator should be able to assign $C1$ and $C2$ routes via two providers $P1$ and $P2$ respectively. This ability requires the system to compute and maintain multiple paths per destination prefix. The following sections describe how this can be achieved by exploiting the SDN architectural separation between control and data plane.

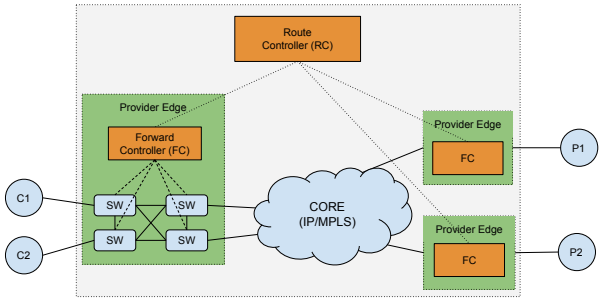

Figure 2: System architecture

2.1. Control Plane

The RC controller's main functions are to process routing updates and compute routes. To achieve routing flexibility, we introduce two mechanisms: multiple routing tables and a preference system for route computation. The controller maintains a separate routing table for each neighbor and is configured by a routing policy specified in RPSL. Two RPSL commands, import and export, dictate whether a route can be used for forwarding and whether it can be exported to a neighboring AS. RC maintains a route ranking for each neighbor and the best ranked one will be selected. By using RPSL, the operator can rank routes based on various attributes allowing the selection of routes to the same destination prefix with different properties.

1. https://tools.ietf.org/html/rfc2622

TABLE 1: Example of routing table

Neighbor	Destination prefix	Path
C1	1.0.0.0/24	P1
C1	2.0.0.0/24	P2
C1	3.0.0.0/24	P3
C2	1.0.0.0/24	P2
C2	2.0.0.0/24	P3
C2	3.0.0.0/24	P3

We utilize RPSL as the interface between the operator and the controller. In RPSL the policy is evaluated in the decreasing order from left to right. We utilize this for the expression of routing preference for each BGP peering session. For example, two policies below express AS2 preference for route to prefix 1.0.0.0/24 via AS4 to AS5 if available, otherwise default routing is used. Meanwhile AS3 prefers routes which avoid AS5 for the same prefix. Using RPSL provides operators with simple and flexible interface for optimizing routing. Coarse- and fine-grained policy e.g. per router, AS or group of ASes, can also be developed via RPSL. The ISP can assign a particular route to a prefix or a range of prefixes to a specific router or a neighbor.

export: to AS2 announce (1.0.0.0/24 AND AS4) OR (1.0.0.0/24 AND AS5) OR ANY
export: to AS3 announce (1.0.0.0/24 AND ^AS5) OR ANY

We implement the selection algorithm as follows to select routes based on the configuration policy. First, a routing update from a peer is filtered by the import policy and imported to the local routing table. Then, for each prefix carried in the update, every peer will filter and rank the route according to its export policy. The standard BGP selection algorithm is applied to tie-breaking when two routes have the same rank. The selected best path is then advertised to the peer and installed to PEs. The output of this process is multiple route tables, each for a peer, as illustrated in Table 1.

The control plane relies on a centralized architecture in order to enable the controller to have complete visibility of externally learned routes to enable flexible routing policies. The route controller establishes BGP session to routers in neighboring ASes.

2.2. Data Plane

The main functions of the data plane are to establish forwarding entries across PEs and to perform actual packet delivery. Two operations are required in the data plane: classification and mapping. The classification operation classifies incoming packets to determine the assigned forwarding path, while the mapping operation places these packets onto path for actual delivery.

An end-to-end forwarding path can be located locally in an ingress PE while some others have two ends located in an ingress and egress PE. Thus, tunnels between pairs of PEs across the core network are required. For example, if C_1 is assigned a path P_1 for a destination prefix p then the ingress PE R_1 must direct traffic from C_1 to p to the egress PE R_2 which in turn forwards the traffic out via P_1. We assume these tunnels already exist and the tunnel management is accomplished by the core network with the involvement of the route controller. The design is independent from the tunnelling technique. Thus, techniques such as GRE or MPLS can be used.

Each PE is composed of multiple physical switches and a local controller. FC exposes APIs to the Route Controller for modifications of multipath forwarding. FC controller handles network events happening within the PE such as next-hop resolution.

Maintenance of multiple paths per prefix significantly increases overheads in both control and data plane. However, the data plane is concerned the most due to limited resources compared with the control plane which runs on cheap commodity hardware. The traditional mechanisms such as Equal Cost Multipath (ECMP) do not support the design requirement, as their static hashing for path selection do not support mapping of incoming traffic to arbitrary paths.

One approach to implement the forwarding plane is to use Access Control Lists (ACL). An ACL entry can be defined to match a particular type of traffic and a destination prefix to determine the path for the traffic. The entry's actions insert the pathID into packet header and forward packets to the next hop. This approach results in a very large forwarding table, thus it is not scalable. Specifically, we will need $C * P$ forwarding entries, where C is the number of neighbors, P is the destination prefixes. A typical large ISP has thousands of neighbors and the current number of prefixes in the Internet is about 600K (and is increasing) making this approach less scalable.

We can implement this by assuming that the operator may only need to customize routing to hundreds of popular prefixes in order to improve routing performance. In reality, a small number of prefixes account for the majority of interdomain traffic in the Internet. Thus, many prefixes will share the same forwarding path.

We leverage this observation and the multi-table capability of SDN to reduce the memory footprint in the data plane. We describe the implementation as follows. First, the classification uses destination MAC address to distinguish traffic and is performed on a separate forwarding table for classification. Traffic from different neighbors which traverses the same path will use the same MAC address. This reduces the number of classification rules to N which is the total number of paths. Second, we leverage the metadata capability in OpenFlow switch for path identification. A unique metadata ID is inserted to classified packets. Packets are then matched against the metadata and destination address in a subsequent table. We denote Q as the

number of prefixes which the operator need routing flexibility. Then upper bound number of forwarding entries in this table will be $Q*N+P$. The total number of entries will be $N+Q*N+P$. We assume that limiting Q to a thousand of prefixes and N to less than 100 would be enough. This approach significantly reduces the table sizes and the complexity is independent of the number of customers, making it scalable.

The forwarding table is constructed using multi-table capability supported by various SDN hardware switches. Incoming packets are classified by the *classification table* which determines which forwarding path the packets will take. A metadata representing the path is added to the packet. In the *multipath table*, the nexthop will be determined by matching against the PID and destination address. The actual forwarding operation which modifies packet TTL, changes source and destination MAC addresses, and send packets out of a port, will be conducted in the *forwarder table*. Using multiple table reduces the number of forwarding entries and enhances the management.

3. Related Work

Feamster et al. introduce a concept of a routing control platform (RCP) in which a logical centralized controller computes interdomain routes on behalf of all routers within an AS [15]. RCP obtains BGP routes via iBGP connections with internal routers and IGP topology via an intradomain protocol. A design and implementation of a prototype is presented [16]. Although RCP demonstrates the feasibility of the concept, it does not provide operators with the flexibility such as supporting multipath routing. Work in [17] extends the RCP concept with a design of Intelligent Service Control Point (IRSCP). IRSCP allows operator to explicitly select egress router for a destination prefix by removing IGP factor from selection process. However, in IRSCP ingress routers can use only a single path to a destination prefix. RCP implementation on SDN-enabled networks has been proposed [18], [19]. In [18] a framework that integrates existing routing protocols such as BGP, OSPF with OpenFlow data plane called RouteFlow is presented. RouteFlow demonstrates the feasibility and compatibility of using SDN switches, but no functionality is realized. A similar work, SDN-IP [19] integrates BGP with OF networks.

Chiu et al. [20] present a new edge router architecture following SDN and NFV principles called EdgePlex. A PE in EdgePlex is composed of servers and switches. The servers host virtual machines (VM), each of which represents a single customer endpoint. The switches connect the servers and the customers' network. EdgePlex gives operators great flexibility so customers' routing and forwarding can be controlled independently. This design differs from ours in the sense that it considers architecture of edge router individually. Moreover, our design forwarding is done in hardware, thus physical line rate can be achievable and we realize a multipath routing architecture.

Wang et al. discuss the possibility and benefits of having different paths for different neighbor ASes, such as solving policy conflict or route oscillation in [21] and present a new RCP architecture which allows multiple path computation and incorporates external measurements into the routing decision called Morpheus [7]. In Morpheus, routing updates are tagged by multiple classifiers before going through mapping functions which translate the tags into numerical values. Sum score of each path is computed and path with the best score is chosen. Multiple decision processes can be used to realize multipath routing. Each classifier is associated with a path characteristic such as latency, security, or business relationship and a weight representing preference of a classifier over another. This allows operators to trade-off path requirements. Morpheus does not propose design of multipath forwarding in the data plane. Their route selection approach can be well-integrated to our design.

4. Implementation and Evaluation

We describe in this section the implementation of the RC controller and the PE datapath. We also present the results of a preliminary evaluation of our prototype.

4.1. Implementation

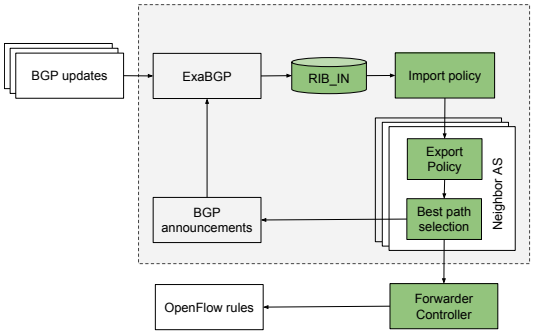

Figure 3: Route Controller implementation.

The RC controller implementation is based on ExaBGP which is a BGP server written in Python. The processing pipeline of the RC is shown in Fig. 3. RC receives BGP advertisements from neighbor. It then constructs a path for each destination prefix carried in the update. These paths are verified by the import policy configured for the neighbor which either filters or accepts them and applies modifications to the path attributes as specified. An accepted path will be passed to other neighbors' export policy which determines if the path can be used for that neighbor and ranks them. The best path selector will select the best ranking path for each neighbor. The resulting multiple best paths are

then advertised to affected neighbors and sent to the ingress and egress PE where they are transformed into OpenFlow rules and installed into the datapath.

PE's datapath and FC controllers are implemented based on Faucet [2] (V1.2), an open-source controller for L2/L3 switching and routing. We implement the forwarding logic by extending Faucet routing functionality. Faucet's forwarding pipeline is constructed using multiple tables as shown in Fig. 4. IP forwarding entries are stored in two FIB tables: *IPV4_FIB* and *IPV6_FIB* table for IPv4 and IPv6 prefixes, respectively. In order to support multiple forwarding entries per prefix, we extend the *FIBs* tables to match on *metadata* and the destination IP address. Traffic classifier is implemented by adding rules that match MAC addresses associated with the virtual IP addresses in the *ETH_SRC* table. The *ETH_SRC* table keeps track of learned hosts and determines whether traffic will be handled by L2 or L3 routing. For tunnelling traffic between PEs, a *tunnel* table is added. In the new pipeline, the FIB tables determine whether packets will be forwarded straight to output port or are handled by the *tunnel* table. The *tunnel* table encapsulates packets in MPLS headers which carry path identifier and tunnelling information used by the core network to forward the packets to the correct egress PE.

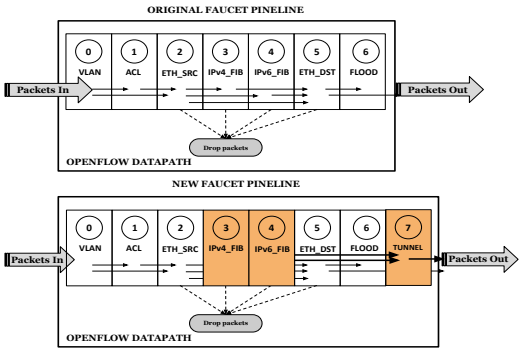

Figure 4: Forwarding pipeline of original Faucet and multipath routing-enabled Faucet

4.2. Evaluation

We built a simple ISP topology as shown in Fig. 5 in order to verify the operation of the system. The testbed is setup using GNS3 [3] - a network emulation tool and Docker containers. Neighbor ASes are represented by Docker containers running Quagga, naming R1 to R7. Two containers used as PE routers are connected via a Cisco router acting as the core network. The Cisco router is configured to tunnel MPLS traffic between the two PEs. The PE containers run Openvswitch as datapath and Faucet as the Forward Controller. The

2. https://faucetsdn.github.io/
3. http://www.gns3.com

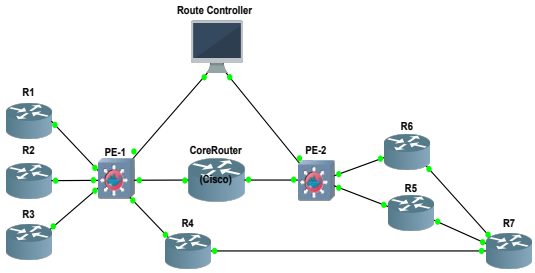

Figure 5: Experimental topology

experiments were run on a single laptop with the configuration of 1.6Ghz Intel Core i5-4200U CPU and 8GB RAM. The RC learns three paths via R4 (local path), R5, and R6 (remote paths) toward prefix 12.0.0.0/24 attached to R7. The Route Controller is configured so that R1, R2 and R3 prefer path via R5, R6 and R4, respectively. We use ping and tcpdump to verify if traffic takes the proposed path. In another experiment, we verify the ability of the system to switch between paths. We run script which send commands directly to PE Faucet-1 causing it to switch R1's traffic between R5 and R4 path back and forth, while R1 and R3 are sending UDP traffic. Fig. 6 shows the bandwidth utilization captured on links R4-R7 and R5-R7.

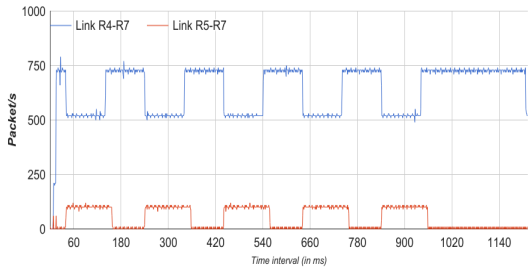

Figure 6: Traffic patterns on link R4-R7 (top) and R5-R7 (bottom). The experiment switches R1's path between R4 and R5 10 times causing traffic to shift between two links.

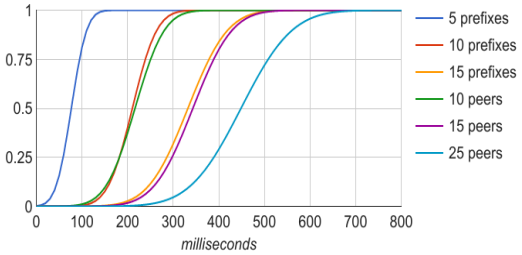

Figure 7: Traffic patterns on link R4-R7 (top) and R5-R7 (bottom). The experiment switches R1's path between R4 and R5 10 times causing traffic to shift between two links.

In another experiment, we investigate the processing performance of the Route Controller. We use a BGP update generator to generate announcements at the rate of 1000 updates/second. Each update carries 5 prefixes from a random neighbor AS. As shown in Fig. 7, the system takes an average of 76ms to compute the best path for the configuration of 5 neighbors and 5 prefixes per update. This figure increases to 200ms and 300ms when doubling and tripling the number of prefixes respectively. This requires a significant amount of time to handle routing updates. We are investigating the factors leading to this issue. Probably this is due to a lack of multi-threading in the implementation as we observed that the CPU utilization was at low level.

5. Conclusion

In this work, we leverage SDN to design and implement a routing architecture for ISP networks which offer much higher routing flexibility than the current system. The system can be deployed using commercially available hardware and is compatible to existing networks. Our evaluation shows the possibility to integrate the system into the ISP network and demonstrates its capability for flexible routing.

In future work we will investigate thoroughly the performance issue of the Route Controller. We also aim to design an extensive experiment to study other performance aspects such as impact on traffic in terms of delay, jitter and packet loss. In addition, we will also look forward to investigate the efficacy of the system through applications such as traffic engineering and customizable routing.

References

[1] Y. Rekhter, E. T. Li, and E. S. Hares, "A Border Gateway Protocol," *RFC 1105*, pp. 1–17, 1989.

[2] M. Yannuzzi, X. Masip-Bruin, and O. Bonaventure, "Open issues in interdomain routing: a survey," *IEEE Network*, vol. 19, no. 6, pp. 49–56, nov 2005.

[3] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson, "Measuring ISP Topologies With Rocketfuel," *IEEE/ACM Transactions on Networking*, vol. 12, no. 1, pp. 2–16, 2004.

[4] S. Savage, A. Collins, E. Hoffman, J. Snell, T. Anderson, S. Savage, A. Collins, E. Hoffman, J. Snell, and T. Anderson, "The end-to-end effects of Internet path selection," in *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication - SIGCOMM '99*, vol. 29, no. 4. New York, New York, USA: ACM Press, 1999, pp. 289–299.

[5] H. Wang, H. Xie, L. Qiu, A. Silberschatz, and Y. R. Yang, "Optimal ISP subscription for internet multihoming: Algorithm design and implication analysis," in *Proceedings - IEEE INFOCOM*, vol. 4, 2005, pp. 2360–2371.

[6] A. Akella, B. Maggs, S. Seshan, and A. Shaikh, "On the performance benefits of multihoming route control," *IEEE/ACM Transactions on Networking*, vol. 16, no. 1, pp. 91–104, feb 2008.

[7] Y. Wang, I. Avramopoulos, and J. Rexford, "Design for configurability: Rethinking interdomain routing policies from the ground up," *IEEE Journal on Selected Areas in Communications*, vol. 27, no. 3, pp. 336–348, 2009.

[8] J. Domzał, Z. Duliński, M. Kantor, J. Rząsa, R. Stankiewicz, K. Wajda, and R. Wójcik, "A survey on methods to provide interdomain multipath transmissions," *Computer Networks*, vol. 77, pp. 18–41, 2015.

[9] J. He and J. Rexford, "Toward internet-wide multipath routing," *IEEE Network*, vol. 22, no. 2, pp. 16–21, mar 2008.

[10] D. Kreutz and F. Ramos, "Software-Defined Networking: A Comprehensive Survey," *arXiv preprint arXiv: ...*, p. 49, 2014.

[11] Y. Jarraya, T. Madi, and M. Debbabi, "A Survey and a Layered Taxonomy of Software-Defined Networking," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, pp. 1955–1980, jan 2014.

[12] K. Agarwal, C. Dixon, E. Rozner, and J. Carter, "Shadow MACs: Scalable Label-switching for Commodity Ethernet," in *Proceedings of the third workshop on Hot topics in software defined networking - HotSDN '14*. New York, New York, USA: ACM Press, aug 2014, pp. 157–162.

[13] W. Braun and M. Menth, "Wildcard Compression of Inter-Domain Routing Tables for OpenFlow-Based Software-Defined Networking," in *2014 Third European Workshop on Software Defined Networks*. IEEE, sep 2014, pp. 25–30.

[14] X. Zhao, Y. Liu, L. Wang, and B. Zhang, "On the Aggregatability of Router Forwarding Tables," in *2010 Proceedings IEEE INFOCOM*. IEEE, mar 2010, pp. 1–9.

[15] N. Feamster, H. Balakrishnan, J. Rexford, A. Shaikh, and J. van der Merwe, "The case for separating routing from routers," in *Proceedings of the ACM SIGCOMM workshop on Future directions in network architecture - FDNA '04*. New York, New York, USA: ACM Press, aug 2004, p. 5. [Online]. Available: http://dl.acm.org/citation.cfm?id=1016707.1016709

[16] M. Caesar, D. Caldwell, N. Feamster, J. Rexford, A. Shaikh, and J. van der Merwe, "Design and implementation of a routing control platform," *2nd conference on Symposium on Networked Systems Design & Implementation*, pp. 15–28, may 2005.

[17] J. Van der Merwe, H. Nguyen, M. Nguyen, A. Ramarajan, S. Saad, M. Satterlee, T. Spencer, D. Toll, S. Zelingher, A. Cepleanu, K. D'Souza, B. Freeman, A. Greenberg, D. Knight, R. McMillan, D. Moloney, and J. Mulligan, "Dynamic connectivity management with an intelligent route service control point," *Proceedings of the 2006 SIGCOMM workshop on Internet network management - INM '06*, pp. 29–34, 2006.

[18] C. E. Rothenberg, M. R. Nascimento, M. R. Salvador, C. N. A. Corrêa, S. Cunha de Lucena, and R. Raszuk, "Revisiting routing control platforms with the eyes and muscles of software-defined networking," *Proceedings of the first workshop on Hot topics in software defined networks - HotSDN '12*, p. 13, 2012.

[19] P. Lin, J. Hart, U. Krishnaswamy, T. Murakami, M. Kobayashi, A. Al-Shabibi, K.-C. Wang, and J. Bi, "Seamless interworking of SDN and IP," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4, pp. 475–475–476–476, sep 2013.

[20] A. Chiu, V. Gopalakrishnan, B. Han, M. Kablan, O. Spatscheck, C. Wang, and Y. Xu, "EdgePlex: Decomposing the Provider Edge for Flexibilty and Reliability," *Proceedings of the 1st ...*, pp. 15:1–15:6, 2015.

[21] Y. Wang, M. Schapira, and J. Rexford, "Neighbor-specific BGP: more flexible routing policies while improving global stability," *ACM SIGMETRICS/Performance'09*, no. 1, pp. 217–228, 2009.