SDX: A Software Defined Internet Exchange

Nick Feamster[†], Jennifer Rexford[⋆], Scott Shenker[‡], Dave Levin[◇], Russ Clark[†], Josh Bailey[*]
[*] Google, [†] Georgia Tech, [◇] University of Maryland, [⋆] Princeton University, [‡] UC Berkeley

Today's interdomain routing protocol, BGP, is difficult to manage, troubleshoot, and secure. It makes even simple network management tasks—including implementing business contracts, balancing traffic to relieve congestion, and tuning network paths to achieve good performance and high reliability—both challenging and unpredictable. The research literature is rife with proposals and protocol modifications to improve BGPs security, convergence properties, flexibility for defining contractual relationships, and traffic engineering capabilities. Unfortunately, none of these new protocols or modifications have seen any appreciable level of deployment, as they require substantial modifications to BGPs control plane, where wholesale upgrades and changes have proven difficult.

Software defined networking (SDN) promises to make network protocols evolvable and flexible; that promise has certainly been fulfilled in data center networks. We posit that this new level of evolvability and flexibility may also herald a new day for interdomain routing by allowing BGPs control plane to evolve independently from the underlying switch and router hardware and bringing software control and logic to interdomain routing. We believe that the programmability that SDN offers can mitigate three problems in interdomain routing: security and accountability; pricing and contracts; and traffic management.

Although global adoption of SDN faces the same problems as global changes in BGP, we think that exchange points may be a place where incremental deployment of SDN is both possible and can bring new features to interdomain routing. We first explore how a single software defined Internet exchange (SDX)—with one controller and one SDN-capable switch—can catalyze solutions to problems in these three areas. We then explore how expanding the SDX architecture to incorporate multiple exchange points can enable a broader range of approaches to these problems.

SDX can enable the following applications that are simply not possible in todays routing infrastructure:

- *Domain-based or application-specific peering.* ISPs may wish to establish limited, special-case peering arrangements with one another depending on the traffics application type or destination. For example, two ISPs may decide to establish a settlement-free peering relationship for video streaming, but not for other types of traffic; OpenFlows support for customizing forwarding policies based on a variety of header fields can enable this type of peering relationship, which is simply not possible in BGP today, where peering policies are dictated by IP prefix alone. Or, they may wish to establish peering for traffic that is destined only to a particular DNS domain name or service. In each of these cases, auxiliary measurements can help associate specific traffic flows to a peering agreement and others to a transit relationship.
- *Remote control peering.* To ensure good performance, content and service providers may wish to have greater control over the complete path between the content or service and the client. For example, a streaming video provider may wish to redirect traffic over better-provisioned paths, or possibly even to direct this traffic over paths with a pre-determined (or at least predictable) quality of service. An SDX controller could allow a content provider to remotely affect certain parts of a downstream path, and perhaps even charge a content provider for this additional level of control.
- *Enforceable interdomain routing policies.* Most exchange-point fabrics do not enforce policy, meaning that the traffic that flows through an exchange may not necessarily satisfy operators' high-level policy. Participants can accidentally or maliciously put traffic on the exchange fabric, which may result in misdirected traffic, routing announcements, or even specially crafted BGP packets that can exploit known router vulnerabilities. An SDX controller could potentially improve reliability and security of an exchange point by ensuring that forwarding table entries are only installed if they satisfy the exchanges high-level security and routing policy.

Additionally, by providing a network controller with direct control over the forwarding tables of switches in an exchange point, SDX potentially makes the following tasks easier to coordinate and implement with high-level software control (as opposed to low-level scripts and indirect mechanisms):

- *Time-of-day routing.* ISPs frequently experience traffic fluctuations as a result of diurnal cycles. Today, operators must implement time-of-day policies with scripts that log into individual routers and indirectly change configuration (*e.g.*, manipulating a route map to adjust local preference settings). An SDX could implement such a policy directly, by simply updating forwarding table entries at the appropriate time, based on a previously specified policy. Although this capability is technically possible in the context of todays interdomain routing protocols, SDN control can potentially make this type of function easier, by virtue of the controllers ability to directly control forwarding tables and existing work on SDN controllers that "natively" support time-of-day policies (*e.g.*, Procera [2]).
- *Dynamic traffic engineering for peering policy compliance.* Some peering arrangements have requirements for maintaining traffic volume ratios across peering points. Today, network operators must monitor traffic volumes carefully to ensure that the traffic that they send over different peering links does not violate these ratios, and rebalance traffic flows if traffic ratios become imbalanced. An SDX controller can incorporate information about traffic volumes to rebalance traffic

flows online, as opposed to being forced to manually react to traffic monitoring systems that are not tightly integrated with any control feedback loop.

- *Route preference based on external inputs.* A variety of measurement systems have been developed to compute reputation based on various detection heuristics (*e.g.*, Hostexploit's AS-based reputation, PHASs hijack alerts). For example, several systems have been developed to assign reputation scores to BGP routes based on characteristics of the route advertisements. Today, operators must respond manually to these alerts; an SDX could simply affix reputation scores to routes and let each AS at the exchange determine how to incorporate reputation into route preference.

Architecture. We envision two instantiations of the SDX architecture: (1) a single-site deployment, where an SDN controller acts as a "smart route server" on behalf of the participating ASes at the exchange; and (2) a multi-site deployment, where SDN controllers across multiple exchange points coordinate to enable more sophisticated wide-area policies. Some of the examples above can be realized with a single-site SDN deployment: for example, a single SDX controller could check routes advertised at a single exchange point against a registry or RPKI infrastructure as input to an ASs route selection process at that exchange point. Other applications—such as enforcing traffic ratios across multiple peering points or implementing more sophisticated peering policies (*e.g.*, allowing ASes to bid on routes for transit at different exit points)—become more interesting when SDX controllers at multiple physical exchange points can communicate with one another.

The single-node SDX architecture bears some resemblance to a conventional route server, but its design makes a few significant departures from a route server. First, the SDX controller allows each AS to apply a custom route selection process to select one or more best routes to each Internet destination. This feature contrasts with existing interdomain routing practices, whereby each AS must apply the conventional BGP route selection process to select a single best route to each destination. Second, the SDX controller can directly affect forwarding by updating switch-table entries, as opposed to indirectly affecting route control via BGP policy mechanisms (*e.g.*, local preference). An SDX control architecture involving multiple sites will entail the design of an inter-exchange protocol to exchange various information; designing a multi-site control framework could be realized through Google's Cardigan architecture [3], an SDN control fabric that makes a multi-site SDN switch deployment appear as a single logical switch.

The direct control that an SDX controller has over forwarding tables at the exchange point makes it possible to both increase flexibility (by allowing more flexible software control to dictate behavior) and predictability (by allowing direct manipulation of forwarding tables, rather than indirect manipulation via obscure mechanisms such as BGP local preference). Multihoming also becomes more straightforward, since the SDX controller can install multiple routes for the same IP prefix on behalf of a particular AS. Troubleshooting and inference become simpler, as well: rather than relying on complex inference through simulation or static configuration analysis, operators can directly determine how traffic will behave.

Challenges. Realizing the SDX control architecture poses several challenges. The first challenge is developing appropriate isolation mechanisms, so that the route selection mechanisms of each AS do not interfere with one another. The second challenge is dealing with the realities of incremental deployment, which give rise to heterogeneity: specifically, any particular AS is likely to have conventional BGP peering relationships and peering relationships at SDXes. A third challenge involves determining the programmatic interface than an SDX controller should expose to users of the exchange point, for various classes of users (i.e., ISPs, content providers), and how to resolve potential conflicts that may arise in route selection and control when multiple parties can control route selection at a single exchange point.

Initial deployments. We have created an initial SDX deployment working with ColoAtl [1] in Atlanta, Georgia. This deployment includes OpenFlow capable switches and an open-source Floodlight controller. This facility provides commercial peering services with network operators and content providers and is providing us with a realistic platform for the evaluating SDX solutions that we have described above. Additionally, as of December 2012, Google's Cardigan SDN-controlled exchange fabric provides the appearance of a single logical switch that currently connects REANNZ and the Wellington Internet Exchange (WIX) to other upstream ASes; this deployment provides another potential testbed for these ideas.

References

[1] ColoAtl: A JT Communications Company. `http://www.coloatl.com/`, 2013.

[2] H. Kim and N. Feamster. Improiving Network Management with Software Defined Networking. *IEEE Network Magazine*, 43(2), Feb. 2013.

[3] Scott Whyte. Project CARDIGAN: An SDN-Controlled Exchange Fabric. `http://www.nanog.org/meetings/nanog57/presentations/Wednesday/wed.lightning3.whyte.sdn.controlled.exchange.fabric.pdf`, 2012.