Boosting the BGP convergence in SDXes with SWIFT

Philipp Mao
ETH Zürich
pmao@ethz.ch

Rüdiger Birkner
ETH Zürich
rbirkner@ethz.ch

Thomas Holterbach
ETH Zürich
thomahol@ethz.ch

Laurent Vanbever
ETH Zürich
lvanbever@ethz.ch

ABSTRACT

BGP, the only inter-domain routing protocol used today, often converges slowly upon outages. While fast-reroute solutions exist, they can only protect from local outages, not remote ones (*e.g.*, a failure in a transit network). To address this problem, we proposed SWIFT, a fast-reroute framework enabling BGP routers to *locally* restore connectivity upon *remote* outages by combining fast inference mechanisms in the control-plane with fast data plane updates. While SWIFT is deployable on a per-router basis, we show in this demonstration that we can deploy SWIFT in Software-Defined Internet Exchange Points (SDXes) with a simple software update. We show that "SWIFTing" an SDX is highly beneficial as it enables to converge the entire fabric within few seconds instead of the tens of seconds required by the original software.

CCS CONCEPTS

• **Networks** → **Network performance analysis**; **Network measurement**; **Network reliability**;

KEYWORDS

BGP; Convergence; Fast Reroute; Internet exchange point (IXP)

ACM Reference format:
Philipp Mao, Rüdiger Birkner, Thomas Holterbach, and Laurent Vanbever. 2017. Boosting the BGP convergence in SDXes with SWIFT. In *Proceedings of SIGCOMM Posters and Demos '17, Los Angeles, CA, USA, August 22–24, 2017,* 2 pages.
https://doi.org/10.1145/3123878.3123919

1 INTRODUCTION

Many applications we rely on day-to-day require always-on connectivity. Nowadays, this need for connectivity often clashes with the slowness of BGP, "the glue that holds the Internet together". Among others, it can take *dozens of seconds* for BGP to restore connectivity upon outages [4]. Given the commercial importance of avoiding downtimes, many mechanisms have been proposed to address slow BGP convergence. Yet, all of them either had a limited scope (*e.g.,* they focus on local outages) or required an impracticable Internet-wide deployment. Our measurements show that BGP convergence is still a pressing and unsolved problem [3].

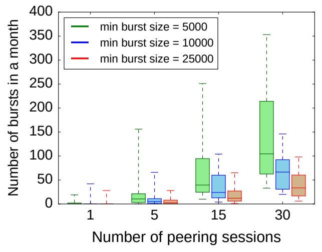

Figure 1: An SDX needs to handle large bursts of BGP withdrawals on a daily basis.

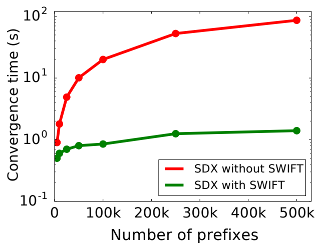

Figure 2: SWIFT greatly speeds up the BGP convergence in SDXes.

We proposed SWIFT [3]. SWIFT is a BGP fast reroute framework working upon remote (and local) outages which can be deployed on an arbitrary set of nodes. SWIFT addresses two fundamental problems: *(i)* slow signaling in the control plane in which bursts of BGP messages are propagated router by router; and *(ii)* slow data plane updates in which many forwarding entries need to be updated to redirect the traffic to the new path. SWIFT speeds up the control plane by quickly localizing the outage and predicting the overall extent of a remote failure after receiving few BGP messages. It also speeds up the data plane update by using a 2-stage forwarding table to reroute groups of prefixes at once by matching on pro-actively computed data plane tags.

In this demonstration, we show that the deployment of SWIFT is not limited to single routers and that its architecture naturally fits in Software-Defined Exchange Points (SDXes) as well [1, 2]. With just a simple software update, all the SDX participants can benefit from protection against remote outages without requiring any changes on their side. Hence, with minimal changes, SWIFT can have a maximal impact as few other locations besides Internet eXchange Points interconnect so many networks in a single point.

2 DEPLOYING SWIFT AT AN SDX

SDX platforms converge slowly. The SDX controller essentially acts as BGP Route Server (RS) receiving, processing and propagating BGP messages between potentially hundreds of participants. Their processing speed is therefore of paramount importance. As large

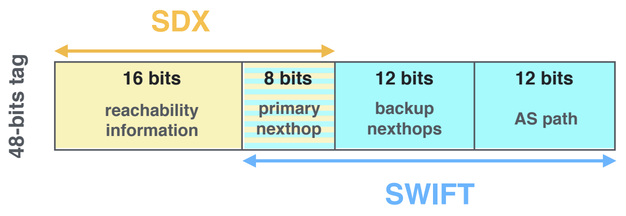

Figure 3: Embedding SWIFT within the SDX platform requires the two to share the 48-bits tag. Yet, we found this does not significantly impact their performance.

outages occur in the Internet, these RSes can be hit with huge bursts of BGP route updates that they have to process. These bursts are frequent. To illustrate this, Fig. 1 shows the number of bursts of BGP withdrawals detected in November 2016 for groups of BGP sessions of different sizes and computed randomly from the pool of 213 BGP sessions we used in [3]. In the median case and with 30 peering sessions, we detected, in a month, 104 (resp. 33) bursts of at least 5k (resp. 25k) withdrawals.

As we show in the demonstration (§3), the current SDX platform converges slowly upon these bursts. It takes up to 90 s to converge after a large burst. In contrast, we show that a SWIFTED SDX converges within a few seconds, independently of the number of prefixes impacted.

Integrating SWIFT and SDX. SWIFT is a natural fit for the SDX as they both rely on a 2-stage forwarding table. For both projects, the first stage groups packets using a tag; the second stage then forwards packets according to their tag's values. The main difference is how SWIFT and SDX group packets. SWIFT groups packets based on the common Internet resources they are using (e.g. AS-PATH) whereas SDX groups packets based on their forwarding equivalence class (which depends on the SDX policies). Both systems also rely on the 48-bits destination MAC address as a tag which is provisioned using BGP and ARP.

As both the SDX and SWIFT use the destination MAC as data plane tag, integrating the two projects require them to share this tag space. While this means that each project has fewer bits to work with, we show that we can still maintain SWIFT convergence properties and the ability of SDX to express many policies.

Sharing the 48-bits tag between SDX and SWIFT. As both SWIFT and SDX rely on the same data plane tag, the partitioning of the 48 bits between them has an impact on the performance of SWIFT and determines the number of participants the SDX may support. The operator is free to partition the tag according to her requirements. Fig. 3 shows how we partition the 48 bits between SWIFT and SDX in this demo. 16 bits are reserved for the SDX, and 24 for SWIFT. 8 bits are used to encode the primary next-hop, which is used by both the SDX and SWIFT. The current partitioning supports up to 256 participants, which is sufficient for most IXPs. We configured SWIFT to only reroute traffic for outages happening in the first four ASes on the AS path, own AS excluded (*i.e.,* on the first three remote AS links). With this configuration, SWIFT computes three backup next-hops for each prefix, one for each of the first three remote AS links on the AS path. Hence, 12/3 = 4 bits

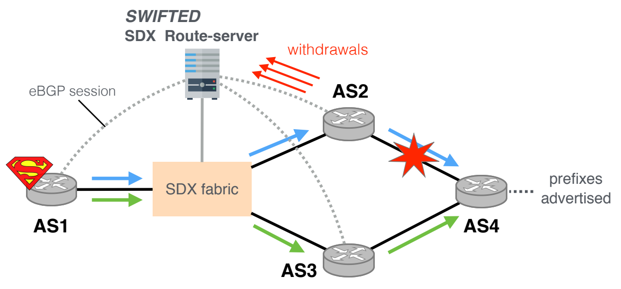

Figure 4: Our Quagga-based demonstration in which we showcase SWIFT running in the SDX controller. Blue (resp. green) arrows indicate the path used before (resp. after) we simulate a failure on (AS2, AS4) link.

are available to encode each backup next-hop, which makes a total of $2^4 = 16$ possible backup next-hops for each participant.

3 DEMONSTRATION

Setup. We built the network depicted in Fig. 4 with Mininet [5] and used Quagga for the routing software. Each router is in a different AS, and AS1, AS2 and AS3 are connected to the SDX. We modified the iSDX [1] implementation to support SWIFT[1]. We configured AS4 to advertise 5k, 10k, 25k, 50k, 100k, 250k and 500k prefixes, and made sure the primary path between AS1 and AS4 traverses AS2 and the backup path traverses AS3.

Experiment. We simulated a link failure on the link between AS2 and AS4, and measured the time AS1 takes to reroute the traffic for all the prefixes on the backup path. The failure generated a burst of BGP withdrawals that SWIFT processed in the SDX controller to trigger the fast reroute. We repeated this experiment with different number of prefixes.

Results. Fig. 2 shows the convergence time of AS1. The convergence time without SWIFT increases linearly with the number of prefixes, and can be close to 90 s for 500k prefixes. When SWIFT is deployed, the convergence time is nearly constant and always within 1.4 s, as only few data plane rule updates are required to converge, irrespective of the number of prefixes affected by the outage. In practice, we expect the difference to be even higher as bursts take time to arrive and hardware-based routers update their forwarding table more slowly than software-based router [3].

REFERENCES

[1] Arpit Gupta, Robert MacDavid, Rüdiger Birkner, Marco Canini, Nick Feamster, Jennifer Rexford, and Laurent Vanbever. An industrial-scale software defined internet exchange point. In *USENIX NSDI 2016.*
[2] Arpit Gupta, Laurent Vanbever, Muhammad Shahbaz, Sean Donovan, Brandon Schlinker, Nick Feamster, Jennifer Rexford, Scott Shenker, Russ Clark, and Ethan Katz-Bassett. SDX: A Software Defined Internet eXchange. In *SIGCOMM 2014.*
[3] Thomas Holterbach, Stefano Vissicchio, Alberto Dainotti, and Laurent Vanbever. Predictive Fast Reroute Upon Remote BGP Outages. In *SIGCOMM 2017.*
[4] Craig Labovitz, Abha Ahuja, Abhijit Bose, and Farnam Jahanian. 2000. Delayed Internet routing convergence. *ACM SIGCOMM CCR* (2000).
[5] Bob Lantz, Brandon Heller, and Nick McKeown. A Network in a Laptop: Rapid Prototyping for Software-defined Networks. In *ACM SIGCOMM Hotnets 2010.*

[1]Source code available at https://github.com/nsg-ethz/iSDX