High-Fidelity Interdomain Routing Experiments

Brivaldo Junior, UFMS
Ronaldo A. Ferreira, UFMS
Ítalo Cunha, UFMG
Brandon Schlinker, USC
Ethan Katz-Bassett, Columbia

ABSTRACT

The PEERING research platform lets researchers exchange actual BGP routes and traffic with hundreds of networks. To date, several researchers have used PEERING to perform interdomain routing experiments. However, PEERING's original design sends experiment data-plane traffic through a VPN, introducing latency and jitter, hindering the applicability of the platform to performance-sensitive experiments. In this work, we propose extensions to PEERING to allow researchers to run applications inside containers on PEERING routers, mitigating performance degradation on data-plane traffic and empowering new classes of experiments. Our goals include strong isolation between experiments and flexible routing to steer traffic in and out of containers.

CCS CONCEPTS

• **Networks → Network experimentation**;

KEYWORDS

Interdomain routing, experimentation, testbed, BGP

ACM Reference Format:

Brivaldo Junior, Ronaldo A. Ferreira, Ítalo Cunha, Brandon Schlinker, and Ethan Katz-Bassett. 2018. High-Fidelity Interdomain Routing Experiments. In *Proceedings of ACM SIGCOMM Conference Posters and Demos (SIGCOMM Posters and Demos'18).* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3234200.3241324

1 INTRODUCTION

Researchers' lack of access to the Internet interdomain routing ecosystem is one of the major hindrances to interdomain routing research and one possible cause for the slow development in this area. The PEERING research platform operates a network with routers at 12 Points of Presence spread across three continents, with interconnections to hundreds of real networks (including Hurricane Electric, Facebook, and Akamai) [4]. PEERING provides researchers with control of routers, empowering interdomain routing experiments that exchange routes and traffic with the real Internet.

An experiment in PEERING lets researchers connect real or emulated networks of their choice to the Internet. PEERING routers work as border routers for the researchers' networks and allow researchers to actively manipulate routes and exchange traffic with the real networks that connect to PEERING. The experiment connects to each PEERING router through a VPN tunnel. Both control (BGP) and data traffic cross the VPN tunnel, which introduces delays as well as jitter in the traffic, and may reduce available bandwidth or incur packet loss. Traffic performance degradation has a minor impact on control-plane experiments, but it significantly hinders data-plane experiments—e.g., experiments that require precise timestamps, fast round-trip times, or high throughput.

Ideally, PEERING should let researchers control how data-plane traffic is handled at PEERING routers, removing the need to transmit performance-sensitive traffic through the VPN tunnel. This functionality would enable experiments that collect measurements at the routers or that provide services with realistic performance to geodistributed clients.

In this work, we present our design for enabling researchers to run arbitrary code on PEERING routers to handle their experiment's data-plane traffic. Our design overcomes three main challenges.

- *Isolation:* PEERING supports multiple concurrent experiments from different researchers. The design must provide resource and traffic isolation. Our proposal is to run experiment code inside containers and use virtual bridges to isolate traffic from different experiments.
- *Routing flexibility:* The design must let researchers dynamically control which in/outbound traffic should be routed through the container. Our proposal lets researchers use BGP to route fractions of an experiment's traffic into and out of containers and prevents propagation of unsafe routes to the Internet.
- *Management overhead:* PEERING has limited personnel for developing and operating a geo-distributed infrastructure. The design should not add management overhead or significantly increase complexity. We plan to let researchers control their containers remotely via a Web interface and route announcements via BGP.

SIGCOMM Posters and Demos'18, August 20–25, 2018, Budapest, Hungary
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5915-3/18/08...$15.00
https://doi.org/10.1145/3234200.3241324

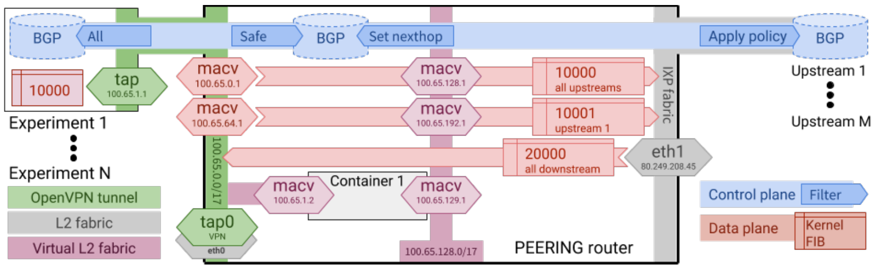

Figure 1: The PEEERING testbed architecture

2 PEERING'S ARCHITECTURE

Figure 1 shows the architecture of a PEERING router and how it interacts with experiments and upstream ASes.

Experiments connect to PEERING routers through a VPN (green). The PEERING router's control plane (blue) forwards all BGP updates from all upstreams to each experiment using the BGP add-paths extension [5], and relays *safe* BGP updates from experiments to upstreams.

In the data plane (red), the PEERING router maintains a FIB with the best routes selected from all routes announced by upstreams (10000), a FIB with the routes from each upstream (e.g., 10001 for upstream 1), and a FIB with the routes from all experiments (20000). PEERING associates one virtual `macvlan` interface with each upstream FIB. PEERING's control plane then rewrites the next-hop of BGP updates received from upstream x with the IP address associated with x's `macvlan` interface. An experiment routes a packet via a particular route by forwarding the packet via the `macvlan` interface associated with that route. This way, experiments can transparently choose which upstream to route towards by using the corresponding next-hop.

3 DESIGN FOR CONTAINER SUPPORT

We extend both control and data planes to add support for running experiment containers on PEERING routers. Each container has a downstream `macvlan` interface connected to the same network as the experiments's VPN interface, and an upstream `macvlan` interface connected to a virtual bridge inside the PEERING router. To route outbound traffic through its container, an experiment simply uses its container's downstream interface's IP address as the next-hop instead of an upstream `macvlan`'s IP address.

An experiment can steer inbound traffic from an upstream into its container by rewriting the next-hop of its BGP updates with the IP address of its container's upstream interface (e.g., 100.65.129.1 in the figure). This will cause inbound traffic, routed by table 20000, to be forwarded to the container's upstream interface instead of the experiment's VPN tunnel. An experiment can advertise prefixes at any granularity (e.g., /28s) to the PEERING router to control which traffic is steered into the container. The remainder of traffic is forwarded directly through the VPN. (These updates are subject to PEERING's policy of never exporting prefixes more specific than /24 to upstreams and can be combined with the standardized *no-export* BGP community [1].)

Traffic isolation. We provide traffic isolation between experiments by using `macvlan` interfaces connected to a virtual bridge configured in `vepa` mode, which prevents communication between `macvlans`. A container from one experiment has no visibility of traffic from other experiments' containers and cannot directly—i.e., via layer two—send them traffic.

Routing policy. An experiment can choose which upstream the container uses to egress outbound traffic by using the corresponding `macvlan`'s IP address. As an example, experiments can establish an iBGP session with its container to control routing.

Limiting container resources. We provide Linux containers with CPU and memory quotas, and constrain experiment containers to a small set of CPU cores.

Protection. Allowing researchers to run arbitrary code on the containers can expose PEERING to malware and bring down the whole router because of a security exploit. To harden the platform for experiments requiring additional protection, we are considering Intel Clear Containers, which run on a separate VM with a lean modified kernel [2].

4 USE CASES

Supporting containers on PEERING routers opens possibilities for new research that are impossible today. Below we list example experiments using the new functionality.

Performance-sensitive experiments. An experiment can run a Web server on multiple routers to quantify the performance gains of different anycast configurations (and configuration changes) on page load times, similar to a CDN.

Real services. A researcher may want to run a service on a PEERING router, such as a honeypot for security research. This way, the honeypot avoids the extra delay introduced by the VPN tunnel, which could make the attacker suspicious of the interaction with the "victim" and stop the attack.

Middleboxes. A researcher may want to subject part of the experiment traffic to a middlebox running on a PEERING router, e.g., to filter unwanted traffic before the VPN tunnel, possibly using deep packet inspection, or perform application-dependent routing as in SDX [3].

REFERENCES

[1] R. Chandra, P. Traina, and T. Li. 1996. RFC1997: BGP Communities Attribute. (1996). https://tools.ietf.org/html/rfc1997.

[2] Clear Linux Project. 2018. Intel Clear Containers. (2018). https://clearlinux.org/containers.

[3] A. Gupta, L. Vanbever, M. Shahbaz, S. P. Donovan, B. Schlinker, N. Feamster, J. Rexford, S. Shenker, R. Clark, and E. Katz-Bassett. 2014. SDX: A Software Defined Internet Exchange. In *Proc. ACM SIGCOMM*.

[4] B. Schlinker, K. Zarifis, Í. Cunha, N. Feamster, and E. Katz-Bassett. 2014. PEERING: An AS for Us. In *Proc. ACM HotNets*.

[5] D. Walton, A. Retana, E. Chen, and J. Scudder. 2016. RFC7911: Advertisement of Multiple Paths in BGP. (2016). https://tools.ietf.org/html/rfc7911.