Decoupling BGP Policy from Routing with Programmable Reactive Policy Control

Peter W. Thai
ECE Dept., Drexel University
3141 Chestnut Street
Philadelphia, USA
pwt23@drexel.edu

Jaudelice C. de Oliveira
ECE Dept., Drexel University
3141 Chestnut Street
Philadelphia, USA
jau@coe.drexel.edu

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

Software-Defined Networking, Policy, BGP

1. INTRODUCTION

BGP (Border Gateway Protocol) is the Internet's de facto interdomain routing protocol. In essence, BGP is a path-vector routing protocol that has been incrementally modified to support AS(Autonomous System)-controlled routing policies. These modifications behave unpredictably because many of them, including the route selection process, is not defined in the BGP specifications. The resulting protocol suffers from policy mismanagement that causes routing anomalies, slow convergence times, and security risks. Additionally, the protocol's widespread use and unpredictability makes network administrators wary to incorporate improvements or replace it altogether. This limits new sorely-needed technologies, such as interdomain Quality of Service (QoS), rich traffic engineering (TE) capabilities, and multipath performance advantages. ASes sharing resources are constrained by these limitations of BGP. In order to continue providing new and useful service for the applications it supports (i.e. the Internet), BGP's shortcomings must be addressed.

Software-Defined Networking (SDN) inherently solves several of BGP's issues: it forms a centralized control plane on which routing applications can be predictably deployed and monitored, it allows new features such as multipath support to be efficiently deployed through software, and detecting routing anomalies on a centralized controller is more manageable. By encapsulating routing within centralized software, interdomain policy control can be achieved elsewhere in the network stack. This effectively decouples policy control and routing in a way BGP has struggled to achieve. Decoupled policy control opens innovation for operator-facing policy languages. Distinguishing policy language from the routing mechanisms is now considered good practice and there has been more interest in writing expressive high-level policy languages while abstracting routing to low level protocols.

In this extended abstract, we propose an SDN-based interdomain policy control methodology. Our main focus is on ASes built entirely on SDN architecture, but we will also provide methods for integrating ASes built fully on traditional packet forwarding devices. We seek to support the three basic AS relationships traditionally supported by BGP. They are (*i*) customer-provider: one AS pays another to forward traffic, (*ii*) peer-to-peer: both ASes mutually benefit from forwarding each others' packets, and (*iii*) redundancy: one AS uses another as backup in case of path failure [1].

2. PROPOSAL

The proposed system is composed of three parts: Policy Language, Policy Controller, and AS Bridge.

2.1 Policy Language

The Policy Language is used to code all policies between ASes and captures the business logic in an agreement between ASes. To replace BGP, this language must be expressive enough to define at least the three most common types of AS relationships defined in [1]. Voellmy et al. proposed Procera, a high-level reactive network controller framework for SDNs [3], for facilitating network policies with end-users (e.g. home or university networks). Procera leverages the well-established functional reactive programming (FRP) approach for defining policies that will activate upon certain conditions. Expressive polices can easily be written in a combination of predefined functions and network parameters - called *signal functions* and *signals*, respectively. The output from Procera is a flow constraint function that determines the constraint to be applied to a flow (or a group of flows). We choose Procera because of the relative ease of writing expressive policies, and its reactive nature is better suited for interdomain resource sharing.

2.2 Policy Controller & AS Bridge

As part of an agreement, an AS can share all or just a subset of its available nodes. For SDN-based ASes, this means provisioning a virtual network using some sort of network virtualization software. We choose FlowVisor because it is full-featured and well supported. The AS then exposes this network to its peering AS through our proposed Policy Con-

CoNEXT Student'12, December 10, 2012, Nice, France.

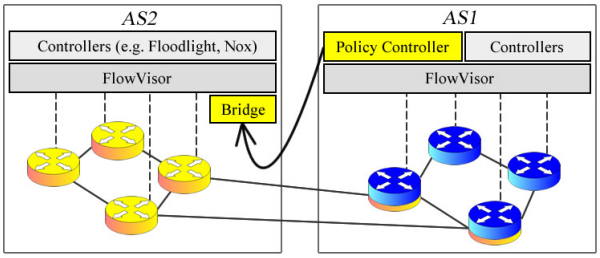

Figure 1: Peering AS_2 can deploy its own controllers over nodes in the provider AS_1 using the Bridge and Policy Controller.

troller, which is a special purpose controller for forwarding packets upstream to any authenticated destination. This includes destinations outside the host AS. We are writing software to sit in an external AS that will interface with this controller and the foreign host's SDN stack. This piece of software is called the AS Bridge and is currently being written as a FlowVisor plugin. To be clear, packets are not forwarded between ASes via the controller; rather, it is assumed that there exist physical links between ASes to form a uniform data plane. The controller simply creates a virtual network between the ASes by pushing rules for routers to cache.

The AS running the AS bridge will transparently view the provisioned foreign network as its own, and local AS controllers can route AS-agnostic data between both ASes without the need for implementing local policy control because the policy is enforced by the Policy Controller. When a policy is breached by a peering AS, the Policy Controller can apply flow constraint functions to peering flows. The constrained path will be detected by the peering controllers and adjust its routing accordingly. Of course, the AS Bridge must authenticate with the Policy Controller.

3. POLICY USE-CASE

In this section, we concretely show how ASes can use our method to establish and enforce relationships. In Figure 1, AS_1 would like to allow AS_2 access to specific nodes in its network under policies that both ASes have agreed upon. Both ASes are SDN-based. AS_1 will first provision a virtual network using FlowVisor, in which all of its traffic will be regulated by P_1 (the Policy Controller of AS_1). Specifically, P_1 will cache the appropriate routes in SDN routers to sandbox the AS_2 data plane to permitted nodes. The AS Bridge in AS_2, B_1, securely retrieves a list of these exposed nodes from P_1. B_1 then combines the nodes from AS_1 and AS_2 into a single resource pool and transparently exposes them to the AS_2 SDN stack. AS_2 can now deploy a single routing algorithm on nodes owned by itself and AS_1.

To form a peering relationship, AS_2 also exposes a provisioned network.

3.1 Controlling Inbound Traffic

ASes today have difficulty controlling the amount of traffic they receive from neighbors and manipulate route advertisements with trial-and-error. Our method allows an AS to enforce an inbound traffic cap with policy.

Algorithm 1 is an example of a policy written in Procera to limit the input rate of a flow at a certain edge router. When a packet arrives, lines 2-3 load a log of global network usage statistics (*world*) and a table of caps to enforce upon those network parameters (*useTable* and *capTable*, respectively). Lines 4-6 look up the specific usage profile and policy caps for a certain MAC address, and store them in *use* and *cap*. Line 8 enforces the cap and will return either an *allow* or *deny* function, depending on the device's usage. Denying traffic will reduce the traffic to a certain threshold at a specific link, and it is up to the peering AS's controller to start routing through a different path. It is also possible to group all flows from a certain AS and impose a global inbound bandwidth limit.

Algorithm 1 Procera Bandwidth Cap Policy

```
proc world → do
    useTable ← useTableSF ← world
    capTable ← capTableSF ← world
  let policy req =
    let src = srcEthAddr req
      use = lookupUse src useTable
      cap = lookupCap src capTable
     in if use < cap then allow else deny
return → policy
```

3.2 Iterative Deployment

Given BGP's ubiquity and that today's ASes are largely based on traditional packet forwarding devices, deploying the proposed system requires a network operator to transparently introduce this solution alongside existing networks. For a network composed of purely traditional devices, a neighboring SDN-based network can still exchange paths with the BGP protocol. For an AS composed of both SDN and traditional devices, peering with an SDN neighbor can be accomplished by deploying SDN routers at its edge and using their controller to exchange information with the peering SDN AS; however, traditional packet forwarding devices will continue to use iBGP for intradomain routing with the only difference being the network's SDN edge controller will be intercepting and manipulating iBGP advertisements. This method is proposed in [2], in which route reflectors (RR) within the AS are replaced with SDN switches to aggregate iBGP messages and provide additional services to the network. Using these techniques, SDN controlled interdomain routing can slowly decouple routing policy from the routing algorithm.

4. REFERENCES

[1] M. Caesar and J. Rexford. BGP routing policies in ISP networks. *IEEE Network*, 19(6):5–11, Nov.-Dec. 2005.

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

[3] A. Voellmy, H. Kim, and N. Feamster. Procera: a language for high-level reactive network control. In *Proceedings of the first workshop on hot topics in software defined networks*, HotSDN '12, pages 43–48, Helsinki, Finland, 2012.