# DTunnels: GEC 4 Demo

*Yogesh Mundada, Vytautas Valancius, Nick Feamster*
*Georgia Tech*

## Overview

This demonstration shows the interconnection of the BGP session multiplexer ("BGP Mux") with two upstream "providers", as well as the integration of the BGP Mux with an OpenVZ-based virtual network. The demo has three components:

1. *Topology Creation Service.* Framework and scripts for creating tunnels based on XML specification.
2. *BGP Upstream Connectivity.* Integration of upstream connectivity from Georgia Tech (AS 2637) and PSG Net (AS 3130), with functional BGP sessions.
3. *DTunnel integration.* Connection of BGP routing sessions into virtual containers on an Emulab topology.

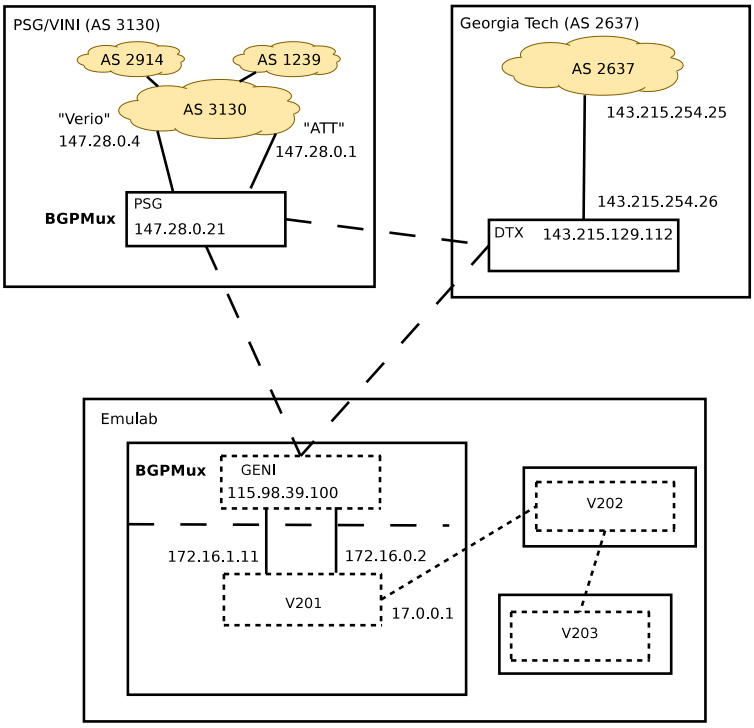


Figure 1: Demo Setup

## Demonstration Details

In this section, we describe a few details of the overall demonstration setup, as well as various next steps for integration.

**Topology creation service.** The topology creation service creates a virtual network using an XML specification file. The demonstration shows an XML specification file for the topology shown in Figure . The specification file contains two parts. The first part of the file specifies which virtual nodes should be created (including the type of virtual node); the second part of the specification shows the virtual links between these nodes (including the type of virtual link between the respective virtual nodes). The topology creation service reads this XML specification and sets up the corresponding topology on the testbed (in this case, Emulab

**BGP connectivity** We presently have upstream BGP connectivity via two upstream providers. First, We have upstream connectivity through Georgia Tech (AS 2637), via which we have direct connectivity to NLR and Internet2, as well as the commodity Internet. Second, we have upstream connectivity through PSGNet (AS 3130), which has upstream connectivity via Verio (AS 2914) and Sprint (AS 1239).

- *PSGNet.* The BGP Mux that resides in AS 3130 currently resides on a VINI node, `vini1.psg.vini-veritas.net`. It connects to two upstream routers on the same subnet in PSGNet and offers two independent *BGP views* via external BGP sessions: an upstream feed to AS 2914 and an upstream feed to AS 1239. A virtual network, such as the one shown at the bottom of Figure , can connect to one or both of these upstream feeds via the BGP Mux. This demonstration sends the AS 1239 BGP feed via a machine at Georgia Tech, which is an artifact (each BGP view requires a separate IP address, and we do not have permissions to set up IP aliases on the VINI node as PSGNet).
- *Georgia Tech.* We have established a direct layer two connection via a VLAN to the Georgia Tech border router. This box, `dtx.gtnoise.net`, currently runs a regular BGP-speaking router but will ultimately run a BGP Mux. Hence, the current setup selects the best route between AS 1239 and AS 2637 and sends it downstream to the experiment at Emulab. Once we configure this machine as a BGP Mux, the downstream experiment will learn routes via all three BGP connections.

Downstream eBGP sessions are established via eBGP multihop (layer 3); in the future, we will establish these sessions via EGRE tunnels (a layer 2 overlay). To do so, we will first need to install an EGRE-enabled kernel on both `dtx` and `psg`.

**Virtual network topology** The Emulab testbed hosts a three-node network, which receives upstream connectivity via the nodes `psg` and `dtx`, as mentioned above. The node labeled "GENI" terminates these BGP sessions in the root context. The virtual environment, `VZ 201`, has two eBGP session to this GENI node, to achieve the equivalent of two upstream BGP feeds. Note that the routes that VZ 201 receives are *exactly the same* as they would be if this experiment had been directly connected to AS 3130 and AS 2647 as upstreams. This three-node network is also running OSPF, and iBGP sessions to distribute the default-free routing table to other nodes in the network.

# Next Steps

We have demonstrated an initial integration of the DTunnel framework and topology specification with the BGP Mux and respective upstream connectivity. Several issues remain:

1. **Moving from eBGP multihop to EGRE tunneling.** The connectivity between the various sites providing upstream connectivity and the testbed nodes themselves currently have reachability via eBGP multihop sessions. An immediate next step is to establish these connections via the EGRE tunneling mechanism that we have patched into the OpenVZ kernel.
2. **Data plane connectivity.** The current setup only shows control-plane function (i.e., the exchange of routing information between various nodes); our testbed does not yet forward data packets from the virtual network to the global Internet. One step in enabling this function is announcing a globally visible IP prefix. The other step is setting up the layer two tunnels.
3. **Integrating topology creation service with BGP connectivity.** The topology creation service currently creates a virtual network topology based on the XML specification; this specification does not yet let a user specify the upstream connectivity that the topology should have. Ultimately, the user should be able to specify which upstream ASes the virtual network should receive upstream connectivity from and which testbed nodes should have those BGP sessions. We are in the process of automating this setup.
4. **Distribution over ProtoGENI nodes.** The current virtual network setup runs on nodes in the Emulab testbed. As part of the component manager integration, we will move the virtual network setup to the ProtoGENI nodes. This will require coordination with the Emulab group, to enable our topology creation service to run on the ProtoGENI nodes.