Virtualized PE for BGP/MPLS L3-VPN using Open-Source Software

NANOG 74 (October 2018)

Bilal Anwer, Robert Bays, Vijay Gopalakrishnan, Bo Han, Dewi Morgan, Patrick Ruddy, Aman Shaikh, Susheela Vaidya, Chengwei Wang and George Wilkie
Introduction

Objective
• Demonstrate feasibility of creating a BGP/MPLS L3-VPN vPE using open-source software

Motivation
• Use-case for AT&T’s DANOS (Disaggregated Network OS)
• Why L3-VPN vPE from open-source software?
  – L3-VPN
    • Allows creation of multiple layer-3 virtual networks on top of a shared service-provider network
    • Widely used service by enterprises
  – vPE
    • Enabler VNF which acts as the ingress and egress for L3-VPN traffic in the service-provider network
  – Open source software
    • Allows increased agility in providing new features while reducing the cost

Challenges
• Required functional and integration-related extensions to open-source components
Software Components of Open Source vPE

Control-plane
• FRR (5.1-dev, snapshot e8f9540) for OSPF, LDP and Zebra
• GoBGP (version 1.31.1 = version 1.31 + our enhancements)

Data-plane
• AT&T-Vyatta’s (DPDK-based) data-plane

• We also verified feasibility with ...
  – Linux data-plane (kernel 4.14.4-mpls)
  – VPP data-plane (release 1801 + router plug-in with our enhancements which have been up-streamed)
Software Architecture of DANOS Open Source vPE

Control-plane

- GoBGP
- FRR OSPF
- FRR LDP
- FRR Zebra
- AT&T-Vyatta Route Broker
  Reliable & high scale route download

vPE

- FIB
- AT&T-Vyatta DPDK based Data-plane

Linux User Space

- Linux Networking Stack
- Linux Forwarding Table
- NetLink

Linux Kernel Space

- Kernel FIB
- TAP Interface
- NIC

Existing clients of netlink
Verifying Feasibility

- Demonstrated feasibility by concurrently running two video streams
  - Keep video traffic separate despite same IP addresses being used by two customers
  - Each client/server has a static route pointing to its upstream CE
  - Each CE advertises appropriate prefix to the PE
Packet Capture at Core Router during Video Streaming

NANOG 74: Open Source vPE for BGP/MPLS L3-VPN
Configuration Details

Establish LSP (Label Switched Paths) between PEs

- Enable IP and MPLS forwarding
- Configure OSPF and LDP on service provider routers

Enable L3-VPN service

- Configure VRFs
- Configure eBGP sessions between PEs and CE
- Configure iBGP sessions between PEs and route reflector

Note: We used AT&T/DANOS Yang Modules for configuring vPEs where possible, but show equivalent Linux, FRR and GoBGP commands in subsequent slides
Tale of Two Loopbacks

**Configured two loopback addresses on vPEs and core router**

- **Loopback1**
  - Used for IP traffic including control-plane traffic

- **Loopback2**
  - Used for MPLS traffic
    - Hence all traffic from VPN customers
# Enable IPv4 forwarding
$ sudo sysctl -w net.ipv4.ip_forward=1

# Load MPLS modules
$ sudo modprobe mpls_router
$ sudo modprobe mpls_iptunnel

# Enable MPLS forwarding on the interface facing the core router
$ sudo sysctl -w net.mpls.conf.ens4.input=1

# Allocate entries in MPLS label table
$ sudo sysctl -w net.mpls.platform_labels=1048575
vPE East: FRR OSPF and LDP Configurations

OSPF Configuration

```
1 router ospf
2 !
3 ! Loopback1 address as router-id
4 ospf router-id 192.168.1.3
5 !
6 ! Add both loopbacks to OSPF
7 network 192.168.1.3/32 area 0
8 network 192.168.2.3/32 area 0
9 !
10 ! Adjacency to the core router
11 network 10.31.11.0/24 area 0
12 !
```

LDP Configuration

```
1 mpls ldp
2 !
3 ! Use loopback1 as router-id
4 router-id 192.168.1.3
5 address-family ipv4
6 !
7 ! Discover the neighbor on
8 ! specified interface
9 discovery transport-address 192.168.1.3
10 interface ens4
11 !
12 !
```
vPE East: VRF Configuration

# Create VRF blue and bring it up
$ sudo ip link add blue type vrf table 1
$ sudo ip route add table 1 unreachable default metric 4278198272
$ sudo ip link set dev blue up

# Add interface to vCE blue-east to VRF blue
$ sudo ip link set dev ens6 master blue

# Allow BGP to listen on port 179 over the VRF-bound interface
$ sudo sysctl –w net.ipv4.tcp_l3mdev_accept=1
$ sudo sysctl –w net.ipv4.udp_l3mdev_accept=1
vCE Blue East: FRR BGP Configuration

```
router bgp 65101

bgp disable-ebgp-connected-route-check

! Use IP address of the interface to
! vPE-east as router-id
bgp router-id 10.31.4.10

! Network with the video server
network 10.31.0.0/24

! Loopback address of the video server
network 172.16.0.0/32

! Peering session with vPE-east
neighbor 192.168.1.3 remote-as 65001
neighbor 192.168.1.3 ebgp-multihop
```

Allows configuration of eBGP session to vPE East loopback
vPE East: GoBGP Configuration

```
[global.config]
as = 65001
    router-id = "192.168.1.3"

[global.apply-policy.config]
    export-policy-list = [ "set-next-hop-self" ]

[zebra.config]
    enabled = true
    url = "unix:/var/run/frr/zserv.api"
    version = 5
    mpls-label-range-size = 100

# iBGP client-session to core router.
# AFI/SAFI supported: VPNv4 unicast.
[neighbors.config]
    neighbor-address = "192.168.1.2"
[neighbors.transport.config]
    local-address = "192.168.1.3"
[neighbors.afi-safis.config]
    afi-safi-name = "13vpn-ipv4-unicast"
```

Use of policy for setting next-hop-self requires policy

Communicate to Linux kernel and Vyatta data-plane via FRR Zebra

iBGP session with Route-Reflector
vPE East: Defining “Set Next-Hop Self” policy

```
# Neighbor set.
[[defined-sets.neighbor-sets]]
neighbor-set-name = "core-route-reflectors"
neighbor-info-list = [ "192.168.1.2" ]

# Policy definition for next-hop-self
[[policy-definitions]]
name = "set-next-hop-self"
[[policy-definitions.statements]]
[policy-definitions.statements.conditions.match-neighor-set]
neighbor-set = "core-route-reflectors"

[policy-definitions.statements.actions.bgp-actions]
set-next-hop = "192.168.2.3"
```

- Match on the route reflector as neighbor
- Set loopback2 of this PE as the next-hop
vPE East: Adding VRF and eBGP Neighbor via GoBGP CLI

Commands for adding blue VRF and eBGP session to vCE east-blue

$ gobgp add vrf blue id 11 rd 100:1 rt both 100:1
$ gobgp nei add 10.31.4.10 as 65101 vrf blue

Use of ifIndex value assigned by Linux as VRF id

$ ip link show blue
11: blue: <NOARP,MASTER,UP,LOWER_UP> mtu 65536 qdisc noqueue state UP
   family inet6
   mtu 65536 qdisc noqueue state UP
   link/ether ae:07:ef:a3:f3:f7 brd ff:ff:ff:ff:ff:ff

Use of ifIndex value assigned by Linux as VRF id
GoBGP Configuration for Router Reflector

```plaintext
[global.config]
    as = 65001
    router-id = "192.168.1.2"

# iBGP session to vpe-east router.
# AFI/SAFI supported: VPNv4 unicast.
[[neighbors]]
    [neighbors.config]
        neighbor-address = "192.168.1.1"
    [neighbors.transport.config]
        local-address = "192.168.1.2"
    [neighbors.route-reflector.config]
        route-reflector-client = true
        route-reflector-cluster-id = "192.168.1.2"
    [[neighbors.afi-safis]]
        [neighbors.afi-safis.config]
            afi-safi-name = "l3vpn-ipv4-unicast"
```

No need to communicate with FRR Zebra since VPNv4 routes are not installed in forwarding table.

---

NANOG 74: Open Source vPE for BGP/MPLS L3-VPN
Implementation: L3-VPN Support in GoBGP

Key building blocks

- Internet routing with BGP
  - Message handling, route computation, and policies
- Partition of routing table into global and VRF
  - Assign BGP sessions to appropriate partition
- VPNv(4|6) BGP address family
  - IP prefix, Route Distinguisher (RD) and MPLS label
- Route targets (RTs)
  - To associate routes with VRF(s)

Interaction with “outside world”

- Allow configuration of VRF(s)
  - Associate an eBGP session with CE to a VRF
- Handle a route received from a CE
- Handle a route received from RR (or remote PEs)
- Communicate with Zebra

Existing support was adequate

Needed some enhancements
GoBGP: Handling a Route Received from an Adjacent CE

- Install the route in VRF forwarding table via Zebra
  - Allows vPE to send traffic coming from other PEs to the CE
- Attach VRF label to the route before sending to RR

Our enhancement; pull request 1572 accepted into GoBGP upstream repository

IPv4 BGP route for prefix p

VPNv4 BGP route for prefix p with blue RD and blue label

GoBGP pull request 1587
GoBGP: Handling a Route Received from an RR or a PE

- Prevent the route from being installed in global FIB
- Import the route into appropriate VRF based on route target
- Install the route with label in Linux VRF forwarding table via Zebra
- Send the route to adjacent CE(s) belonging to the VRF

Our enhancement to GoBGP

We fixed this

IPv4 BGP route for prefix p
IPv4 BGP route for prefix p with blue label
VPNv4 BGP route for prefix p with blue RD and blue label
Summary

Demonstrated feasibility of creating an L3-VPN vPE using Open Source Software

• Control-plane:
  – GoBGP, FRR (OSPF, LDP and Zebra)

• Data-plane:
  – AT&T-Vyatta DPDK based data-plane
  – Also verified feasibility with VPP and Linux data-planes

Required us to make some enhancements to GoBGP 1.31

• Proper installation of routes into FIB
• Assign MPLS labels to VPNv4 routes
• Modifications available on Github at:
  https://github.com/amanshaik75/gobgp/tree/zapi_version_5

DANOS URL: https://www.danosproject.org/
Acknowledgements

**AT&T**
- Bill Benson, Ramana Chinnapa, Kenneth Duell, Jennifer Yates

**Cumulus Networks**
- David Ahern (for explaining how Linux VRFs work)

**FRR**
- Donald Sharp, Renato Westphal, Russ White, https://github.com/paulzlabn

**GoBGP**
- Iwase Yusuke

**VPP**
- Michael Borokhovich, Pierre Pfister, Jeff Shaw
Backup
Open-Source Software across the Feasibility Test-bed

<table>
<thead>
<tr>
<th>Network Function</th>
<th>VNF OS</th>
<th>Control-plane</th>
<th>Data-plane</th>
</tr>
</thead>
<tbody>
<tr>
<td>vCE</td>
<td>Ubuntu 16.04.2 LTS Linux Kernel 4.4.0-64 generic</td>
<td>FRR 5.1-dev BGP and Zebra</td>
<td>Linux</td>
</tr>
<tr>
<td>vPE</td>
<td>Debian 4.14.62-0 Vyatta1+9.1 Linux Kernel 4.14.0-trunk-vyatta-amd64... (DANOS)</td>
<td>GoBGP 1.31.1 FRR 5.1-dev OSPF, LDP and Zebra (snapshot e8f9540)</td>
<td>AT&amp;T-Vyatta DPDK</td>
</tr>
<tr>
<td>Core router</td>
<td>Ubuntu 16.04.3 LTS Linux kernel 4.14.4-mpls (custom configuration)</td>
<td>GoBGP 1.31.1 FRR 5.1-dev OSPF, LDP and Zebra (snapshot e8f9540)</td>
<td>Linux</td>
</tr>
</tbody>
</table>

**Control-plane**

- GoBGP 1.31.1 = version 1.31 + our enhancements
- FRR 5.1-dev = snapshot e8f9540

**When Linux is used as data-plane on vPE**

- vPE OS: Ubuntu 16.04.3 LTS, Linux kernel 4.14.4-mpls

**When VPP is used as data-plane on vPE**

- vPE data-plane: VPP release 1801 + our enhancements to router plug-in
- OS: same as when Linux is used as data-plane
GoBGP: Configuration of VRF and Associated CE Sessions

- Import matching VPNv4 routes into VRF
  - Routes received from RRs and/or other PEs
  - **Outstanding issue; we use a workaround**

- Obtain unique label for VRF from Zebra
  - Zebra acts as a central agent for label assignment
    - Prevents label collision between different protocols like BGP and LDP
  - **Pull request 1587 to GoBGP repository**

- Install an MPLS route for the label in Linux default forwarding table
  - Allows vPE to handle traffic from other PEs
  - **We enhanced GoBGP code**

# Example of MPLS route installed
# in Linux kernel by GoBGP
$ ip -f mpls route
144 dev blue proto bgp
GoBGP: Interacting with Zebra

GoBGP by default uses API version 4 for interaction with Zebra

- API version 4 does not have all features to support L3-VPN
  - Example: lack of support for multi-level recursive next-hop lookup
- Required us to upgrade to Zebra API version 5

Added partial support for API version 5 in GoBGP

- Support for parts required for L3-VPN, not everything