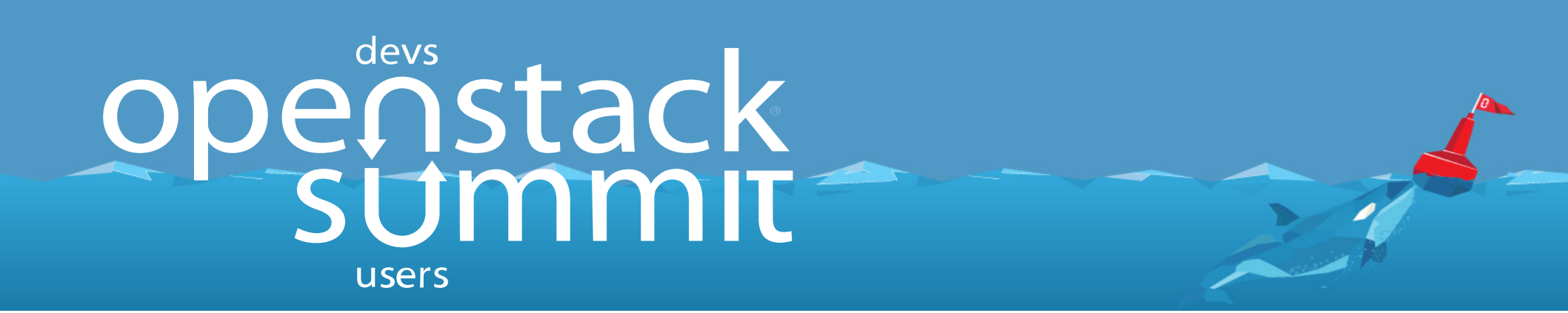


Flexible NFV WAN interconnections with Neutron BGP VPN

Thomas Morin
Orange

OpenStack Summit, May 2018, Vancouver BC

orange™

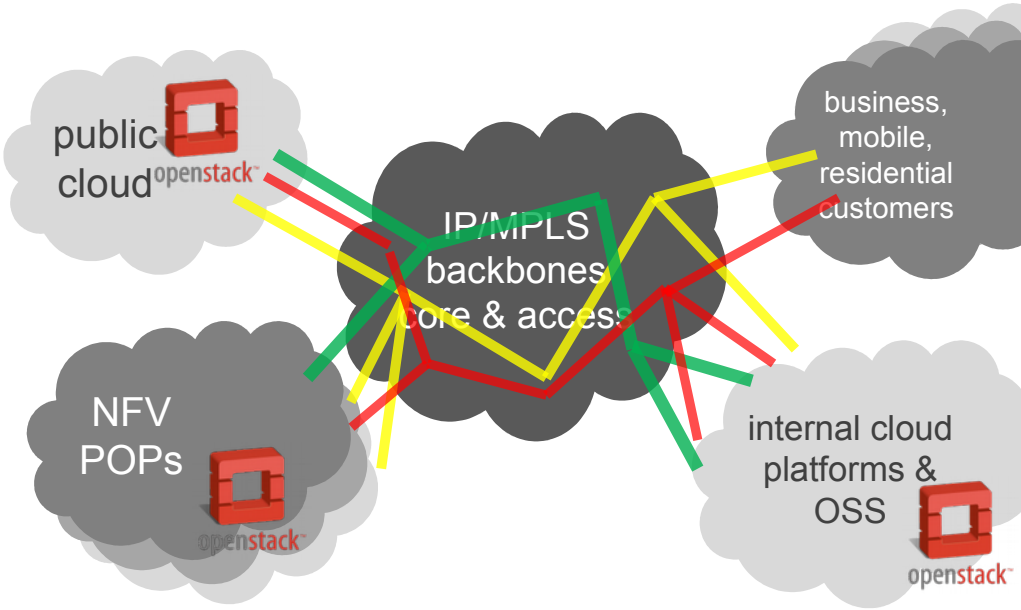
Agenda

- BGP VPNs as a key building block for Telcos
  - 1-slide reminder on BGP VPNs
- Why we like dynamic routing in these contexts
- An example with vEPC
- How do OpenStack and SDN controllers come into the picture ?
- Neutron BGP VPN
  - 3-slides drill down
- Live demo !

BGP VPNs: a base building block in the NFV toolbox

- Telcos require network isolation
  - mobile backhaul
  - wholesale/B2B offers
  - triple-play

    … these do not run directly over the Internet !
- BGP VPNs: a key building block used by Telcos to address this need for isolation
- And then NFV comes into the picture !



=> need to interconnect these WAN BGP VPNs with the NFV Infrastructure, i.e. OpenStack

Base principles of BGP VPNs (simplified)

- dataplane: **MPLS**, to isolate the traffic of different VPNs on the wire
  - MPLS (in this context): an encapsulation carrying packets of a VPN
  - MPLS “label”: dataplane identifier used for isolation
- control plane: **BGP** routing protocol, to indicate how to reach a destination
  - advertise routes:
    - “10.11.0.0/16 in VPN 888:42 is reachable via router @X using MPLS Label N”
  - VPN “identifiers”: “Route Target” (e.g. 888:42)
    - calling them ‘identifier’ is very simplified, there is much more flexibility
    - only present in the control plane, not on the wire!
- initially for L3VPNs (end of 20th century)
- then extended for L2/Ethernet, in particular E-VPN (a few years ago)
- dataplane later extended to other encapsulations:
  - MPLS/GRE, MPLS/UDP
  - VXLAN for E-VPN

Dynamic routing required, why ? [1/2]
Anycast load-balancing

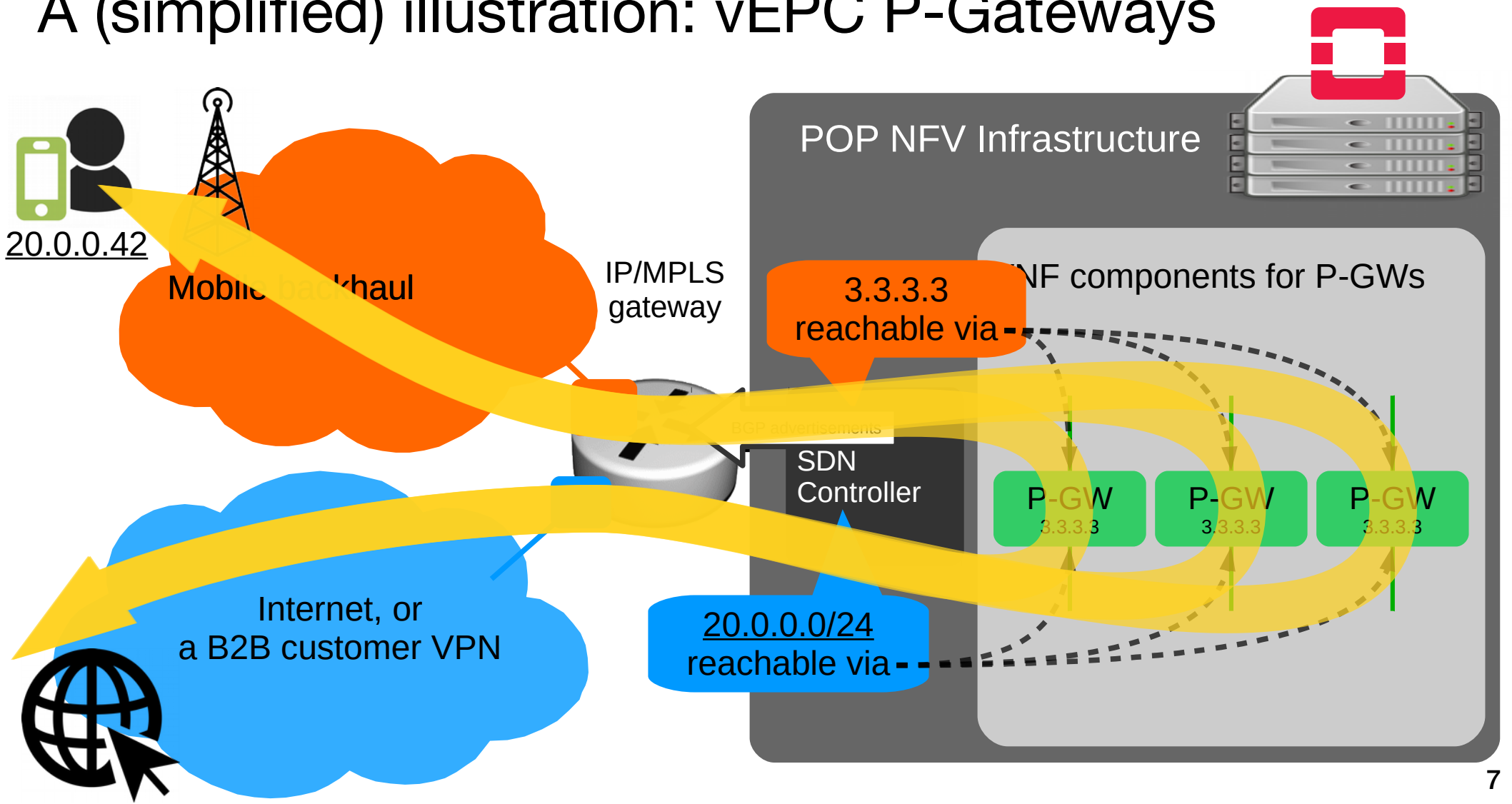
- « Anycast »: multiple hosts (e.g. VMs) sharing a given IP
- Equal Cost Multi-Path (ECMP)
  - ECMP: the kind of packet load balancing done by routers
  - most often done per-flow
    - per packet
    - 5-tuple hashing to always load balance a given flow on the same path
- Scale-up/Scale-down
  - Dynamic routing lets routers dynamically know the (multiple) places where a given service IP is present
    - e.g. the multiple VM ports where a given service IP is defined

We like to combine these !

Dynamic routing required, why ? [2/2]
Let's avoid router configuration provisioning when we can !

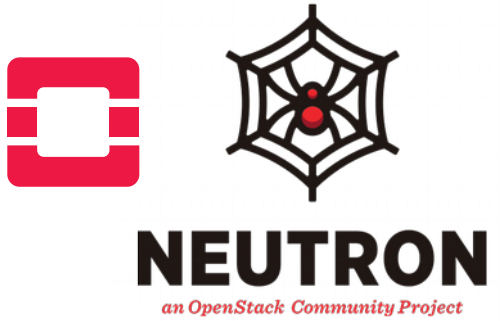
- Typical hurdles if router configuration provisioning is involved:
  - need to configure VRFs, VLANs, static routes
  - router configuration is not managed by the same ops team (sometimes)
  - router automation tooling/standards:
    - not yet easy enough so that we can assume its here
    - even when done : need to interconnect the tools together
- How to avoid that ? => dynamic routing from the SDN controller !
  - have the SDN controller advertise BGP VPN routes
  - only one-shot router configuration: no per-VPN, or per-VM configuration
  - IP/MPLS gateways know about VMs coming & going
    thanks to dynamic route updates

A (simplified) illustration: vEPC P-Gateways



Let's do this with the Openstack Networking API !

- Initial context: some SDN controllers support BGP VPN routing
  - each with their own API
  - no possibility to let tenants manage their BGP VPN connectivity
- Need for an API being :
  - SDN-controller agnostic
  - multi-tenant

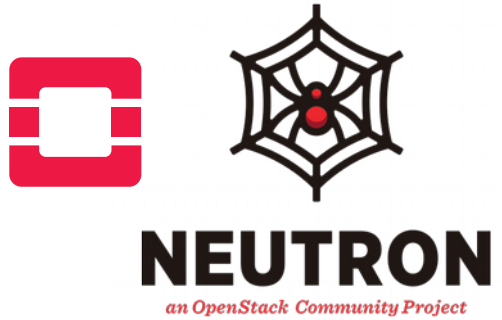


an API to control …

BGP VPN features of ...

BGP VPN : also with Neutron drivers !

- Beyond an API towards BGPVPN features in SDN controllers, an implementation in Neutron is also important :
  - as a reference driver, for use in the OpenStack CI
  - because you can want to use these features with Neutron ML2 drivers, without adding a heavier SDN controller to your deployment



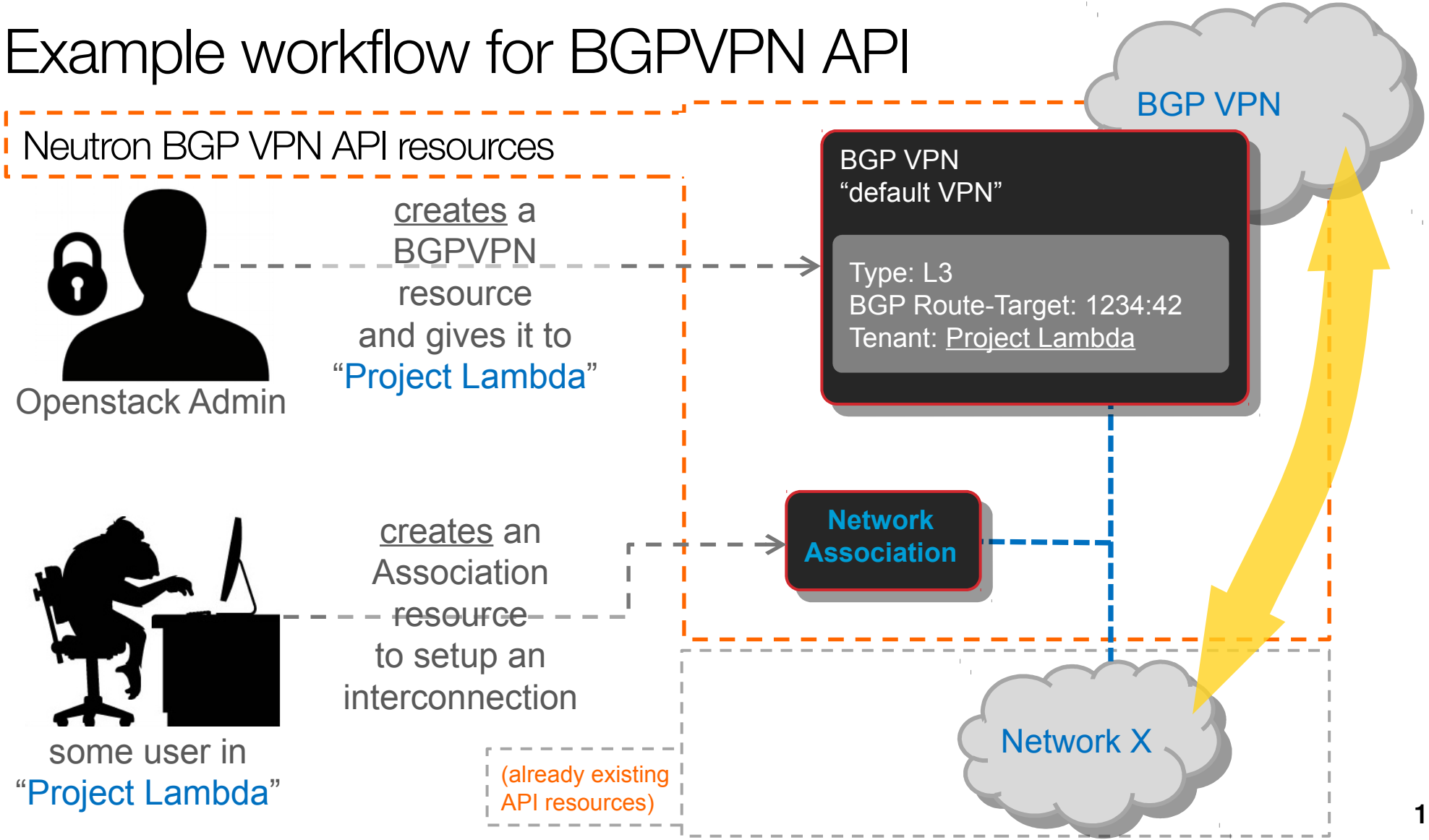
an API to control …

BGP VPN features of ...

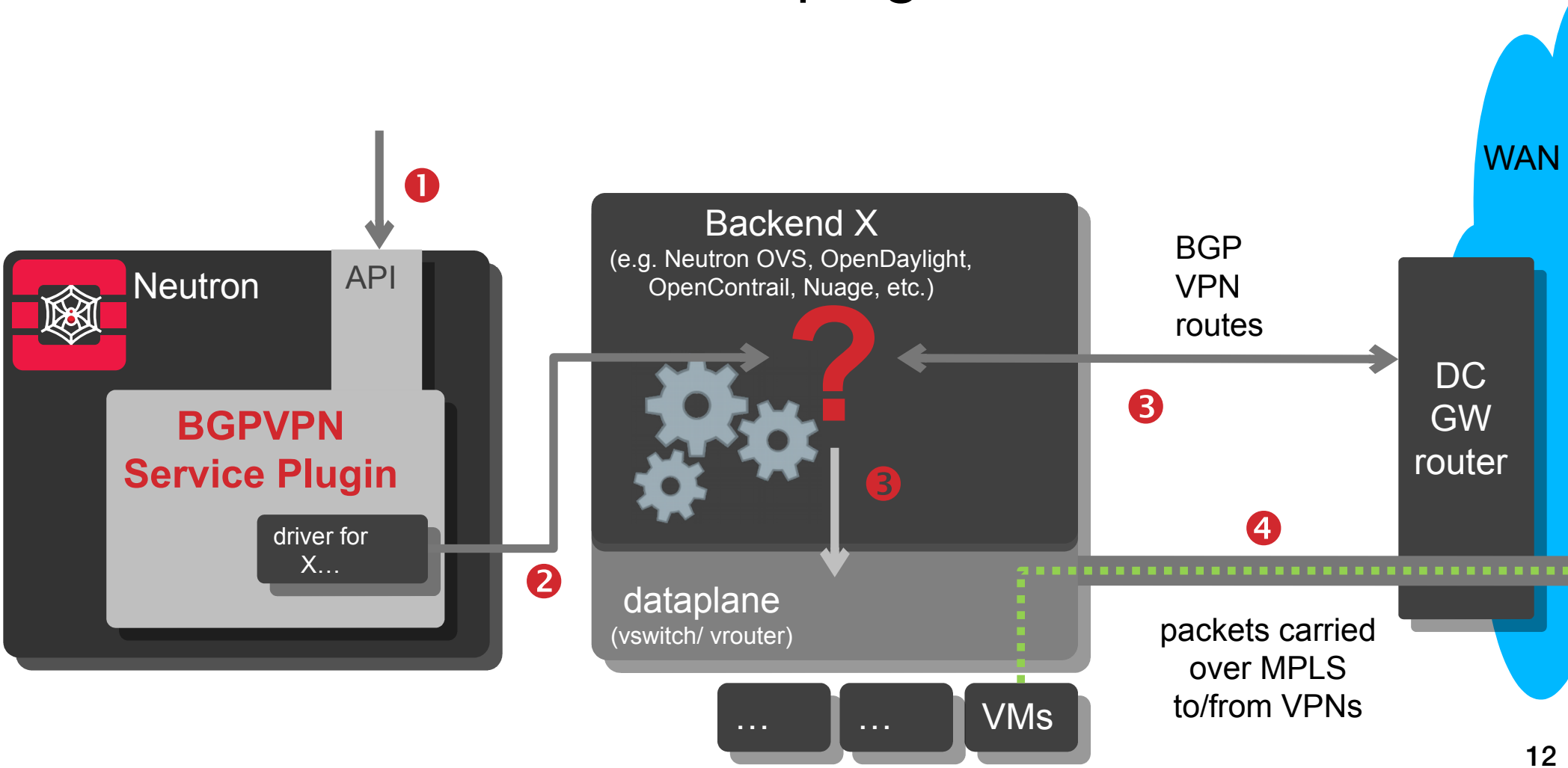
Networking BGPVPN features

- BGPVPN definitions:
  - L2 – EVPN
  - L3 – IP VPN
- Granularity of what is interconnected:
  - a Network
  - a Router
  - a Port
- Fine-grained control of routing (Queens)
  - static prefixes reachable via a Port
    - a.k.a « static routes »
  - routes of another BGPVPN reachable via a Port
    - a.k.a « route leaking »
  - control of BGP local preference
    - e.g. active/backup
- Drivers for...
  - Neutron ML2
    - OVS
    - linuxbridge
  - OpenDaylight
  - Tungsten Fabric / Contrail
  - Nuage Networks
- And also...
  - Heat bindings
  - Horizon GUI
  - Tempest suite

Example workflow for BGPVPN API

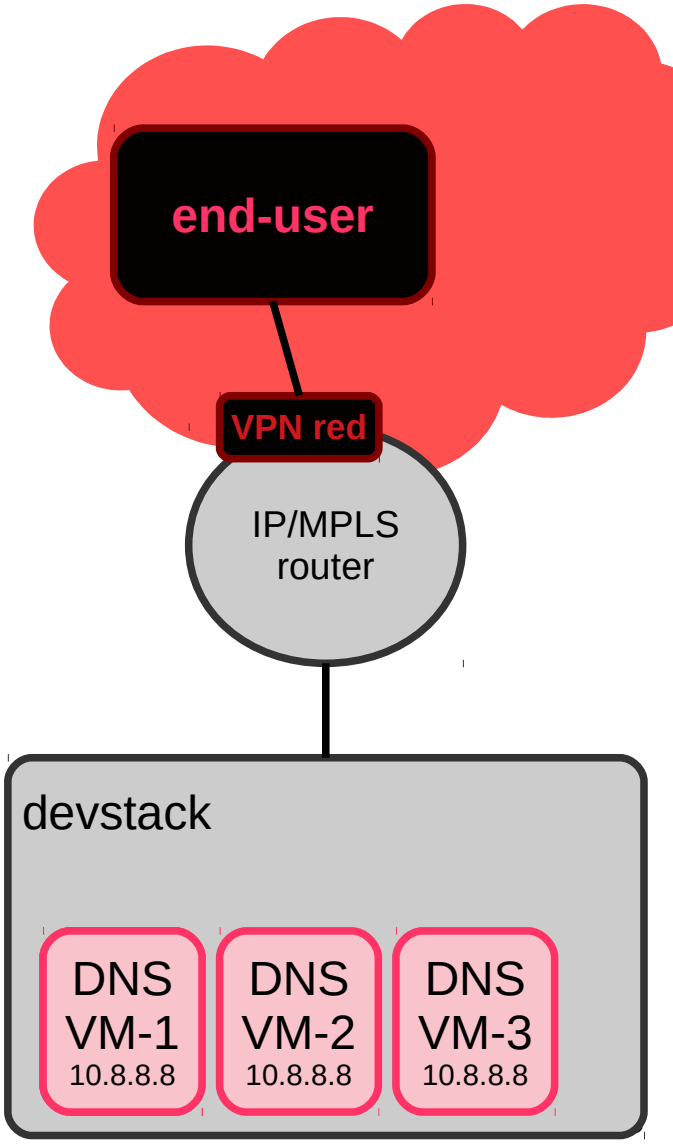


Neutron BGPVPN service plugin architecture



Let's do a demo !

- What will we do ?
  - a set of anycast DNS servers dynamically made reachable in a BGPVPN
  - dynamically scale up / down
    - zero touch on the gateway IP/MPLS router
- How ?
  - Under the hood:
    - IP/MPLS router running in a VM
    - simulated end-user (VM) connected in the VPN
    - Openstack : a devstack VM
      - Neutron OVS reference driver for BGP VPN (a.k.a bagpipe)
      - OpenVSwtich >= 2.8 (for MPLS/GRE support)
  - In the OpenStack tenant :
    - a BGPVPN (L3/IPVPN)
      - one-shot creation by the admin
      - match VPN instance on router
    - simple DNS server VMs
      - each configured to give a different DNS answer to make load balancing easily observed
    - each VM Port is associated with the BGPVPN, with a route set for the DNS servers anycast IP (10.8.8.8)



(demo script)

- bgpvpn list
- port create port1 --network private
- server create dns-vm1 --port port1 --user-data cloudinit-dns1.sh --flavor cirros256 --image cirros
- port set port1 --allowed-address ip-address=10.8.8.8
- bgpvpn port association create vpn-red port1 --prefix-route prefix=10.8.8.8/32
- # dig @10.8.8.8 vancouver.demo
- port create port2 --network private
- server create dns-vm2 --port port2 --user-data cloudinit-dns2.sh --flavor cirros256 --image cirros
- port set port2 --allowed-address ip-address=10.8.8.8
- bgpvpn port association create vpn-red port2 --prefix-route prefix=10.8.8.8/32
- # dig @10.8.8.8 vancouver.demo # multiple times to see the effect of load balancing
- port create port3 --network private
- server create dns-vm3 --port port3 --user-data cloudinit-dns3.sh --flavor cirros256 --image cirros
- port set port3 --allowed-address ip-address=10.8.8.8
- bgpvpn port association create vpn-red port3 --prefix-route prefix=10.8.8.8/32
- # dig @10.8.8.8 vancouver.demo
- server delete dns-vm1
- # dig @10.8.8.8 vancouver.demo
- server delete dns-vm3
- # dig @10.8.8.8 vancouver.demo

cloudinit-dns.sh:

```
#!/bin/sh
ip addr add 10.8.8.8/32 dev lo
echo vancouver.demo 1.1.1.1 > /etc/dnsd.conf
dnsd
```

What's next... ?

- Rocky
  - Implement support for Router Association advertise_extra_routes attribute
- On the radar
  - possible API evolutions...
    - BGPaaS
    - trigger to enable BFD healthcheck
    - control of BGP Communities
    - P2P/VPWS
    - self-service BGPVPN
  - driver for networking-ovn ?
  - driver for dragonflow ?
- Related
  - « Neutron-Neutron Interconnections »
    how to let cloud users get private, on-demand interconnections without the overhead of IPSec

Wrap up

- Neutron BGPVPN API extension provides key features to let us do NFV interconnects in flexible ways
- Many other use cases as well
  - multi-DC / inter-DC
  - cloud / business interconnects
- This is opensource: your contributions are welcome ! (and needed!)
  - #openstack-net-bgpvpn (irc.freenode.net)

NEUTRON

an OpenStack Community Project

Useful pointers...

- Related talks during this summit:
  - « Integration of Multiple OpenStack Clouds with a Core MPLS Network »
  - « Using Neutron BGP VPN for edge networking »
- Related work in progress
  - « Neutron-Neutron Interconnections »
    https://specs.openstack.org/openstack/neutron-specs/specs/rocky/neutron-inter.html
- Docs
  - API: https://developer.openstack.org/api-ref/network/v2/#bgp-mpls-vpn-interconnection
  - Service plugin and drivers: https://docs.openstack.org/networking-bgpvpn/latest
- Release notes
  - https://docs.openstack.org/releasenotes/networking-bgpvpn

Hint: these links are clickable in the PDF at