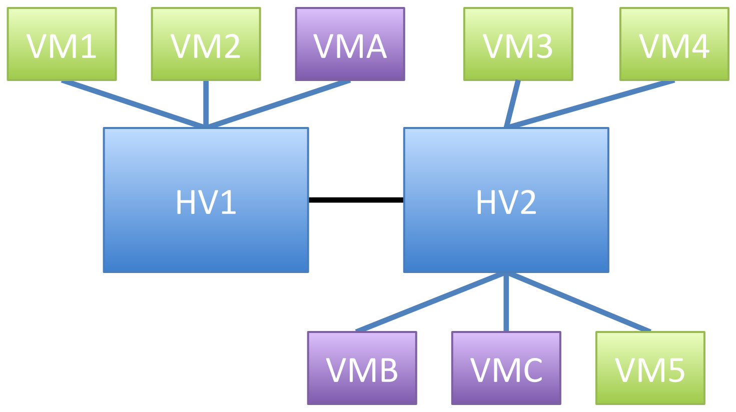
OVN:
Open Virtual Network for Open vSwitch

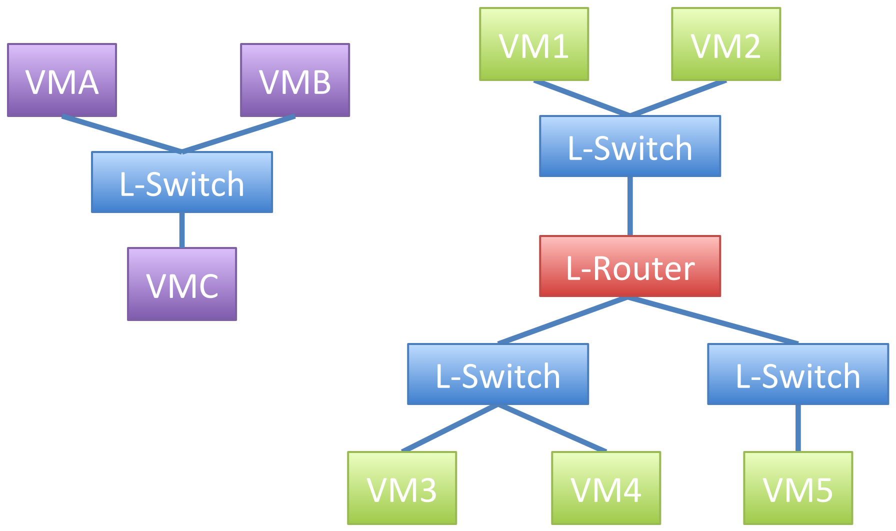
Russell Bryant (@russellbryant)
Kyle Mestery (@mestery)
Justin Pettit (@Justin_D_Pettit)

Virtual Networking Overview

Provides a logical network abstraction on top of a physical network



Physical



Logical

What is OVN?

- Open source virtual networking for Open vSwitch (OVS)
- Provides L2/L3 virtual networking
  - Logical switches and routers
  - Security groups
  - L2/L3/L4 ACLs
  - Multiple tunnel overlays (Geneve, STT, and VXLAN)
  - TOR-based and software-based logical-physical gateways
- Work on same platforms as OVS
  - Linux (KVM and Xen)
  - Containers
  - DPDK
  - Hyper-V
- Integration with OpenStack and other CMSs

The Particulars

- Developed by the same community as Open vSwitch
- Vendor-neutral
- Architecture and implementation have all occurred on public mailing lists
- Developed under the Apache license

Goals

- Production-quality
- Straight-forward design
- Scale to thousands of hypervisors (each with many VMs and containers)
- Improved performance and stability over existing plugin

Why OVN is different

- Will not require any additional agents for functionality for simplified deployment and debugging
- Security groups using new in-kernel conntrack integration
  - More secure and faster than other methods
  - “Taking Security Groups to Ludicrous Speed with Open vSwitch” at 9:50 on Thursday
- DPDK-based and hardware-accelerated gateways
  - Leverages new OVS DPDK port
  - Works with switches from Arista, Brocade, Cumulus, Dell, HP, Juniper, and Lenovo

Why OVN is Important to OpenStack

Why OVN is Important to OpenStack

- Neutron’s default backend is a custom virtual networking control plane

- Long term, we feel Neutron is best served letting a separate project implement the virtual network control plane

Why OVN is Important to OpenStack

- Migration from OVS backend to OVN is very natural for Neutron

- Just taking advantage of increasing functionality in OVS, which is already in use

OpenStack Neutron Platform

- Neutron evolving to be a platform
  - First step: Plugin decomposition
  - Second step: Bringing the plugin and driver backends under the Neutron tent
  - Third step: Open Source backends mature
- OVN fits into this Neutron Platform model

Neutron Integration with OVN

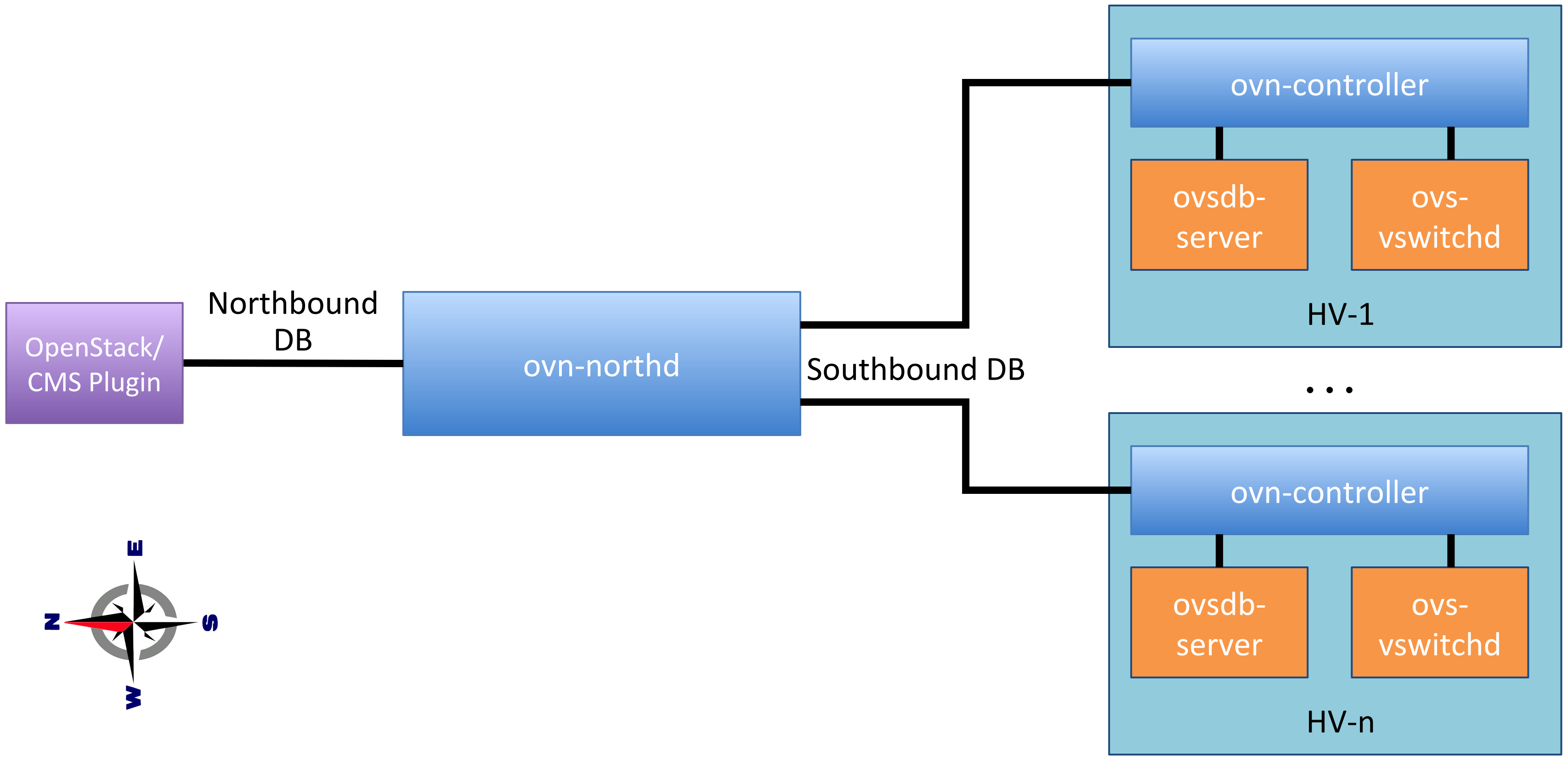
- ML2 driver for OVN
    - replaces OVS ML2 driver and Neutron’s OVS agent

- Uses Neutron L3 and DHCP agents, but just until OVN support is ready

Designed to Scale

- Configuration coordinated through databases
- Local controller converts logical flow state into physical flow state
- Desired state clearly separated from run-time state
- Grouping techniques reduce Cartesian Product issues

OVN Architecture



The OVN Databases

- ovn-northbound
  - OpenStack/CMS integration point
  - High-level, desired state
    - Logical ports -> logical switches -> logical routers
- ovn-southbound
  - Run-time state
    - Location of logical ports
    - Location of physical endpoints
    - Logical pipeline generated based on configured and run-time state

The Daemons

- ovn-northd
  - Converts from the high-level northbound DB to the run-time southbound DB
  - Generates logical flows based on high-level configuration
- ovn-controller
  - Registers chassis and VIFs to southbound DB
  - Converts logical flows into physical flows (ie, VIF UUIDs to OpenFlow ports)
  - Pushes physical configuration to local OVS instance through OVSDB and OpenFlow

An Example

Logical_Switch

Name	Ports
LS1	LP1,LP2

Logical_Port

Name	MAC
LP1	AA
LP2	BB

Chassis (ovn-controller)

Name	Encap	IP
HV1	Geneve	10.0.0.10
HV2	Geneve	10.0.0.11

Bindings (ovn-controller)

Name	Chassis
LP1	HV1

Pipeline (ovn-northd)

Datapath	Match	Action
LS1	eth.dst = AA	LP1
LS1	eth.dst = BB	LP2
LS1	eth.dst = <broadcast>	LP1,LP2

LP2 Arrives on HV2

Logical_Switch

Name	Ports
LS1	LP1,LP2

Logical_Port

Name	MAC
LP1	AA
LP2	BB

Chassis (ovn-controller)

Name	Encap	IP
HV1	Geneve	10.0.0.10
HV2	Geneve	10.0.0.11

Bindings (ovn-controller)

Name	Chassis
LP1	HV1
LP2	HV2

Pipeline (ovn-northd)

Datapath	Match	Action
LS1	eth.dst = AA	LP1
LS1	eth.dst = BB	LP2
LS1	eth.dst = <broadcast>	LP1,LP2

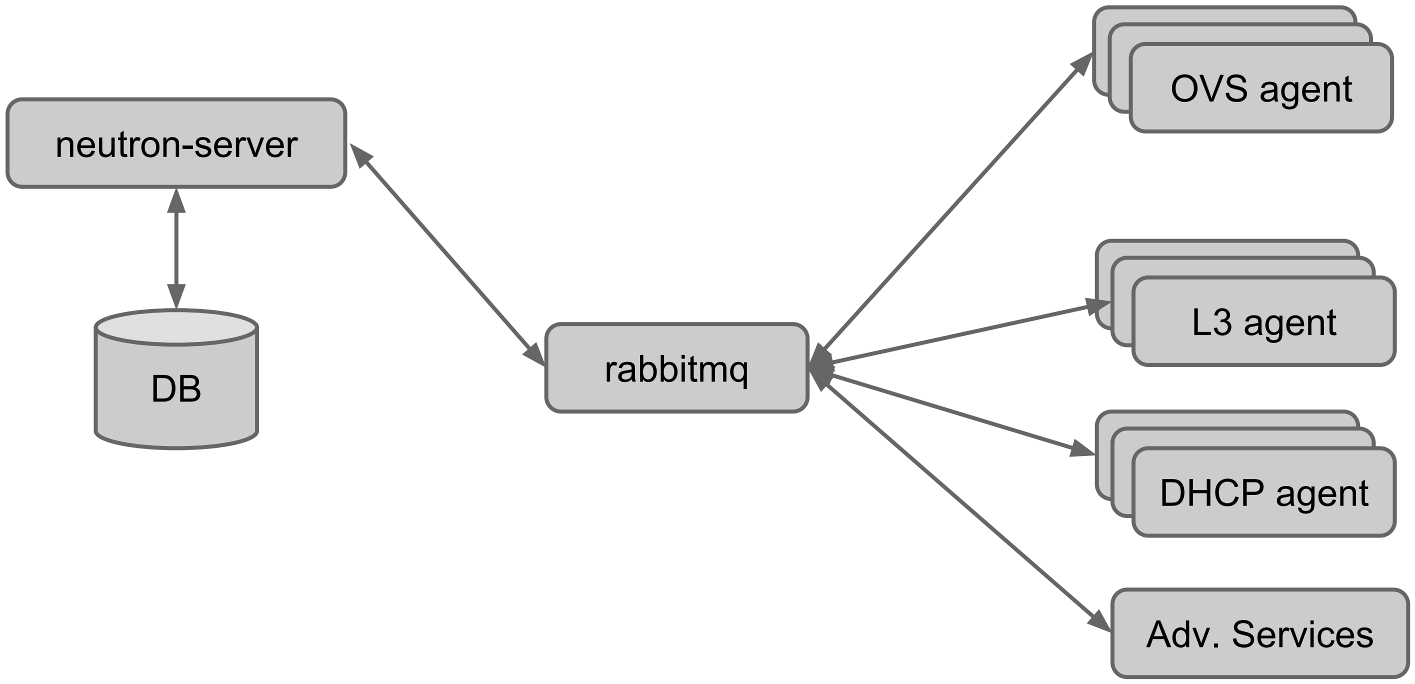
Resources

- Architecture described in detail in ovn-architecture (5)
- Configuration is through a number of databases
  - OVN Northbound – Interface between CMS and OVN (ovn-nb (5))
  - OVN Southbound – Holds the configuration and state of the logical and physical components (ovn-sb (5))
- Available in the "ovn" branch of the main OVS repo:
  - https://github.com/openvswitch/ovs/tree/ovn

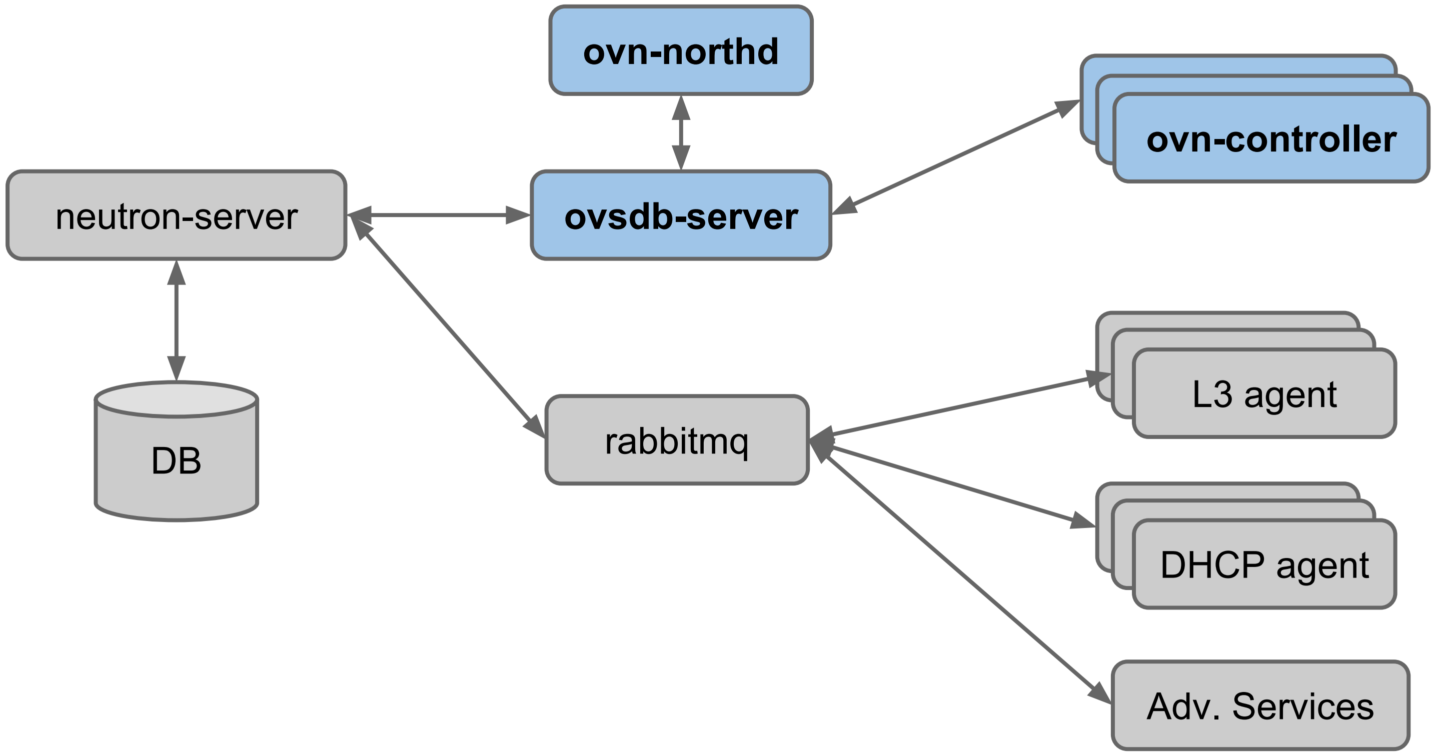
Status – The EZ Bake Milestone

- From start of coding to first ping: 6 weeks
- Needs more testing, obviously
- Haven't tried any scale testing
- Features listed on first page should be ready by end of the year
- Expect rapid progress!

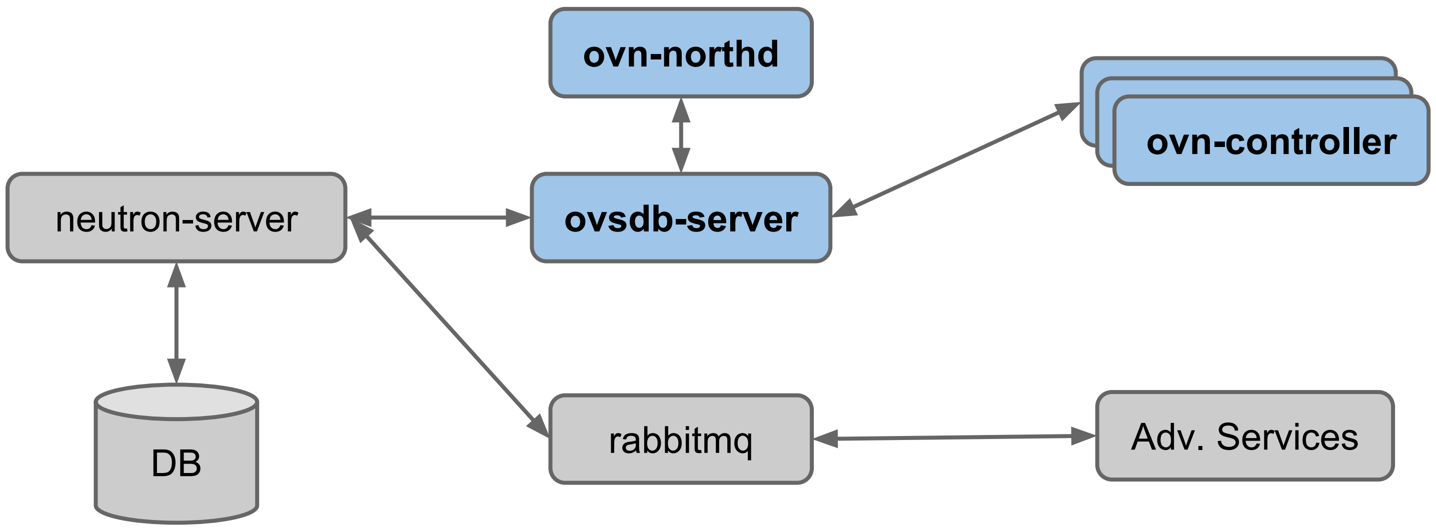
Neutron with built-in solution



Neutron with OVN (so far)



Neutron with OVN (later this year)



Trying out OVN

Test #1 - ovs-sandbox

```
$ git clone http://github.com/openvswitch/ovs.git
$ cd ovs
$ git checkout -b ovn origin/ovn
$ ./boot.sh && ./configure && make
$ make sandbox SANDBOXFLAGS=”--ovn”
```

Test #1 - ovs-sandbox

```
$ ovn-nbctl lswitch-add sw0
$ ovn-nbctl lport-add sw0 sw0-port1
$ ovn-nbctl lport-add sw0 sw0-port2
$ ovn-nbctl lport-set-macs sw0-port1 00:00:00:00:00:01
$ ovn-nbctl lport-set-macs sw0-port2 00:00:00:00:00:02
$ ovs-vsctl add-port br-int lport1 -- \
  set Interface lport1 external_ids:iface-id=sw0-port1
$ ovs-vsctl add-port br-int lport2 -- \
  set Interface lport2 external_ids:iface-id=sw0-port2
```

Test #1 - ovs-sandbox

```
# Trace OpenFlow flows for a packet from port 1 to 2
$ ovs-appctl ofproto/trace br-int \
  in_port=1,dl_src=00:00:00:00:00:01,\
  dl_dst=00:00:00:00:00:02 -generate
```

Test #2 - Multi-node DevStack

```
$ git clone http://git.openstack.org/openstack-dev/devstack.git
$ git clone http://git.openstack.org/stackforge/networking-ovn.git
$ cd devstack
… Get local.conf from networking-ovn/devstack/
… local.conf.sample or computenode-local.conf.sample
$ ./stack.sh
```

More cool stuff that works

- Can be used to create overlay networks for containers across many hosts

- If OVN backs Neutron, containers in VMs can be hooked up to virtual networks managed by Neutron

What’s Next for Core OVN

- Security groups using in-kernel conntrack
- ovn-controller that translates to “vtep” schema to enable physical gateways
- OVS-DPDK gateway that uses “vtep” schema
- L3 routing and native IP management
- New test framework that allows local build-time testing with tunnels and arbitrary topologies
- Merge “ovn” into OVS master branch

OVN Neutron Integration Future

- L3 service plugin
- security groups
- get tempest CI job passing
- create multi-node CI job

Longer Term

- DPDK datapath
  - Move beyond the capabilities of the “vtep” schema to support fail-over, scale-out, and more stateful services
  - Will become a reference for building OVS DPDK applications
- Architecture will allow innovation in the logical network space
  - New approaches to networking and security

How you can help

- Try it! Test it! Write Code!
- Report bugs and try it at scale
- Core OVN is being developed on ovs-dev mailing list:
  - http://openvswitch.org/pipermail/dev/
  - #openvswitch on Freenode
- Neutron plugin for OVN is being developed here:
  - http://git.openstack.org/stackforge/networking-ovn.git
  - openstack-dev mailing list
  - #openstack-neutron-ovn on Freenode

Thank you!

Russell Bryant (@russellbryant)
Kyle Mestery (@mestery)
Justin Pettit (@Justin_D_Pettit)