OVN: Open Virtual Network for Open vSwitch

Ben Pfaff (@Ben_Pfaff)
Justin Pettit (@Justin_D_Pettit)
Virtual Networking Overview

Provides a logical network abstraction on top of a physical network
What is OVN?

• Open source virtual networking for Open vSwitch (OVS)
• Provides L2/L3 virtual networking
  ✓ Logical switches
  ✓ L2/L3/L4 ACLs (no connection tracking yet)
    – Logical routers
    – Security groups
  ✓ 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
    – Other CMSes
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
Container Integration

Containers nested inside VMs can be in logical networks too!
OpenStack Integration with OVN

- OVN has its own Neutron driver
  - Use instead of OVS ML2 driver and agent
- Goal: Reliability and good integration with OVS
  - Existing OVS plugin has poor reputation
- Goal: Avoid needing Neutron-specific agents on hypervisors
  - Currently, Neutron L3 and DHCP agents are used
  - OVN will supplant these over time.
- Long term goal (?): Supplant existing OVS driver in deployments
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

OpenStack/CMS Plugin

OVN Northbound DB

ovn-northd

Southbound DB

ovn-controller

ovsdb-server

ovs-vswitchd

HV-1

...

ovn-controller

ovsdb-server

ovs-vswitchd

HV-n
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

• Central: ovn-northd
  – Converts from the high-level northbound DB to the run-time southbound DB
  – Generates logical flows based on high-level configuration

• Per-hypervisor: 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

<table>
<thead>
<tr>
<th>Name</th>
<th>Ports</th>
</tr>
</thead>
<tbody>
<tr>
<td>LS1</td>
<td>LP1,LP2</td>
</tr>
</tbody>
</table>

#### Logical Port

<table>
<thead>
<tr>
<th>Name</th>
<th>MAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>LP1</td>
<td>AA</td>
</tr>
<tr>
<td>LP2</td>
<td>BB</td>
</tr>
</tbody>
</table>

#### Chassis (ovn-controller)

<table>
<thead>
<tr>
<th>Name</th>
<th>Encap</th>
<th>IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>HV1</td>
<td>Geneve</td>
<td>10.0.0.10</td>
</tr>
<tr>
<td>HV2</td>
<td>Geneve</td>
<td>10.0.0.11</td>
</tr>
</tbody>
</table>

#### Bindings (ovn-controller)

<table>
<thead>
<tr>
<th>Name</th>
<th>Chassis</th>
</tr>
</thead>
<tbody>
<tr>
<td>LP1</td>
<td>HV1</td>
</tr>
</tbody>
</table>

#### Pipeline (ovn-northd)

<table>
<thead>
<tr>
<th>Datapath</th>
<th>Match</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>LS1</td>
<td>eth.dst = AA</td>
<td>LP1</td>
</tr>
<tr>
<td>LS1</td>
<td>eth.dst = BB</td>
<td>LP2</td>
</tr>
<tr>
<td>LS1</td>
<td>eth.dst = &lt;broadcast&gt;</td>
<td>LP1,LP2</td>
</tr>
</tbody>
</table>
### LP2 Arrives on HV2

#### Logical Switch

<table>
<thead>
<tr>
<th>Name</th>
<th>Ports</th>
</tr>
</thead>
<tbody>
<tr>
<td>LS1</td>
<td>LP1,LP2</td>
</tr>
</tbody>
</table>

#### Logical Port

<table>
<thead>
<tr>
<th>Name</th>
<th>MAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>LP1</td>
<td>AA</td>
</tr>
<tr>
<td>LP2</td>
<td>BB</td>
</tr>
</tbody>
</table>

#### Chassis (ovn-controller)

<table>
<thead>
<tr>
<th>Name</th>
<th>Encap</th>
<th>IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>HV1</td>
<td>Geneve</td>
<td>10.0.0.10</td>
</tr>
<tr>
<td>HV2</td>
<td>Geneve</td>
<td>10.0.0.11</td>
</tr>
</tbody>
</table>

#### Bindings (ovn-controller)

<table>
<thead>
<tr>
<th>Name</th>
<th>Chassis</th>
</tr>
</thead>
<tbody>
<tr>
<td>LP1</td>
<td>HV1</td>
</tr>
<tr>
<td>LP2</td>
<td>HV2</td>
</tr>
</tbody>
</table>

#### Pipeline (ovn-northd)

<table>
<thead>
<tr>
<th>Datapath</th>
<th>Match</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>LS1</td>
<td>eth.dst = AA</td>
<td>LP1</td>
</tr>
<tr>
<td>LS1</td>
<td>eth.dst = BB</td>
<td>LP2</td>
</tr>
<tr>
<td>LS1</td>
<td>eth.dst = &lt;broadcast&gt;</td>
<td>LP1,LP2</td>
</tr>
</tbody>
</table>
Security Groups

- **Security group**: a firewall policy that typically allows all outbound connections plus inbound return traffic.
- Legacy OVS plugin uses namespaces and iptables
  - Slow and badly integrated because of extra layers
- New OVS support for kernel-based connection state tracking
  - Much faster (see OpenStack Vancouver presentation)
  - Also being added to OVS DPDK switch
- OVN will use this new OVS feature to implement reflexive ACLs and construct security groups from them
Based on “vtep” OVSDB schema included with OVS

- Hardware: Arista, Brocade, Cumulus, Dell, HP, Juniper, Lenovo
- Software: Implement “vtep” schema in software, via DPDK
  - Will become a reference for building OVS DPDK applications
- Later: move beyond the capabilities of the “vtep” schema to support fail-over, scale-out, and more stateful services
Trying out OVN
Test #1 - ovs-sandbox

$ git clone http://github.com/openvswitch/ovs.git
$ cd ovs
$ ./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=00:00:00:00:00:00:02 -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
Status

• From start of coding to first ping: 6 weeks
• Limited testing so far:
  • Small numbers of hypervisors and logical networks
  • Simulated scale testing up to 500 hypervisors
• Feature progress:
  • Gateways: In code review
    • Connection tracking: RFC patches
    • Security groups: In development
  • L3: to-do
Features for 2016?

• Native IP management
  • Integrate DHCP server into ovn-controller
• NAT
• Load-balancing
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 “master” branch of the main OVS repo:
  – https://github.com/openvswitch/ovs
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!

Justin Pettit (@Justin_D_Pettit)
Ben Pfaff (@Ben_Pfaff)