OVN (Open Virtual Network)

Ben Pfaff - @Ben_Pfaff
Justin Pettit - @Justin_D_Pettit
Russell Bryant - @russellbryant

OVN Principles

Performance

Scalability

Simplicity

Reliability

Visibility

What is OVN?

- Virtual Networking for Open vSwitch (OVS)
- Developed within the OVS project
- Linux Foundation Collaborative Project
- First release of OVN comes with OVS 2.6
- First release of Neutron integration (networking-ovn) available in OpenStack Newton

Feature Overview

- Manages overlays and physical network connectivity
- Flexible security policies (ACLs)
- Distributed L3 routing, IPv4 and IPv6
- Native support for NAT, load balancing, DHCP
- Works with Linux, DPDK, and Hyper-V
- L2 and L3 gateways
- Designed to be integrated into another system
  - OpenStack, Kubernetes, Docker, Mesos, oVirt

IPv4 and IPv6 Routing

- Native support for IPv4 and IPv6
- Distributed
- ARP/ND suppression
- Flow caching improves performance
  - Without OVN: multiple per-packet routing layers
  - With OVN: cache sets dest mac, decrements TTL
- No CMS-specific L3 agent

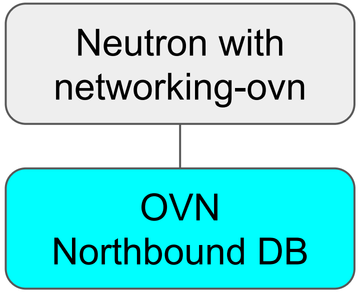
Built-in DHCP (v4 and v6)

Other solutions require running a separate central or per-HV DHCP agent.
Now, OVN includes DHCP support in ovn-controller (local OVN agent):

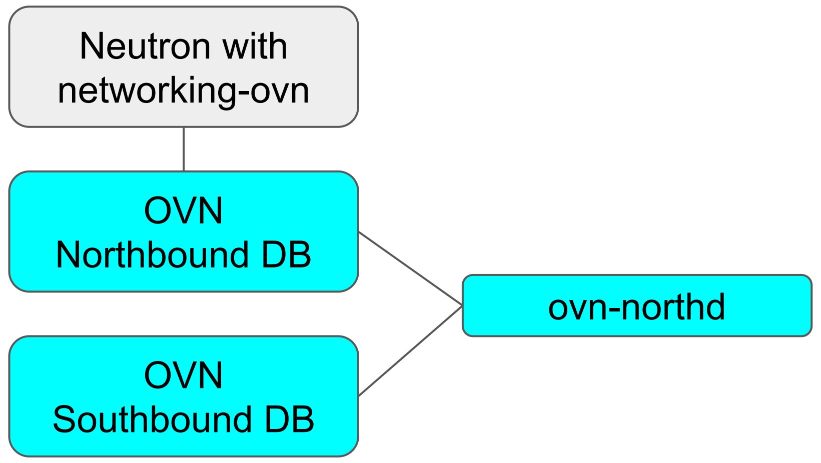
- DHCP packets never leave the hypervisor.
- Arbitrary DHCP options.
- Optional basic IPAM support.
- Meant for simple case.
- Future direction: built-in DNS server
- Bonus: “ovn-trace” debugging tool can help find issues

How does OVN work?

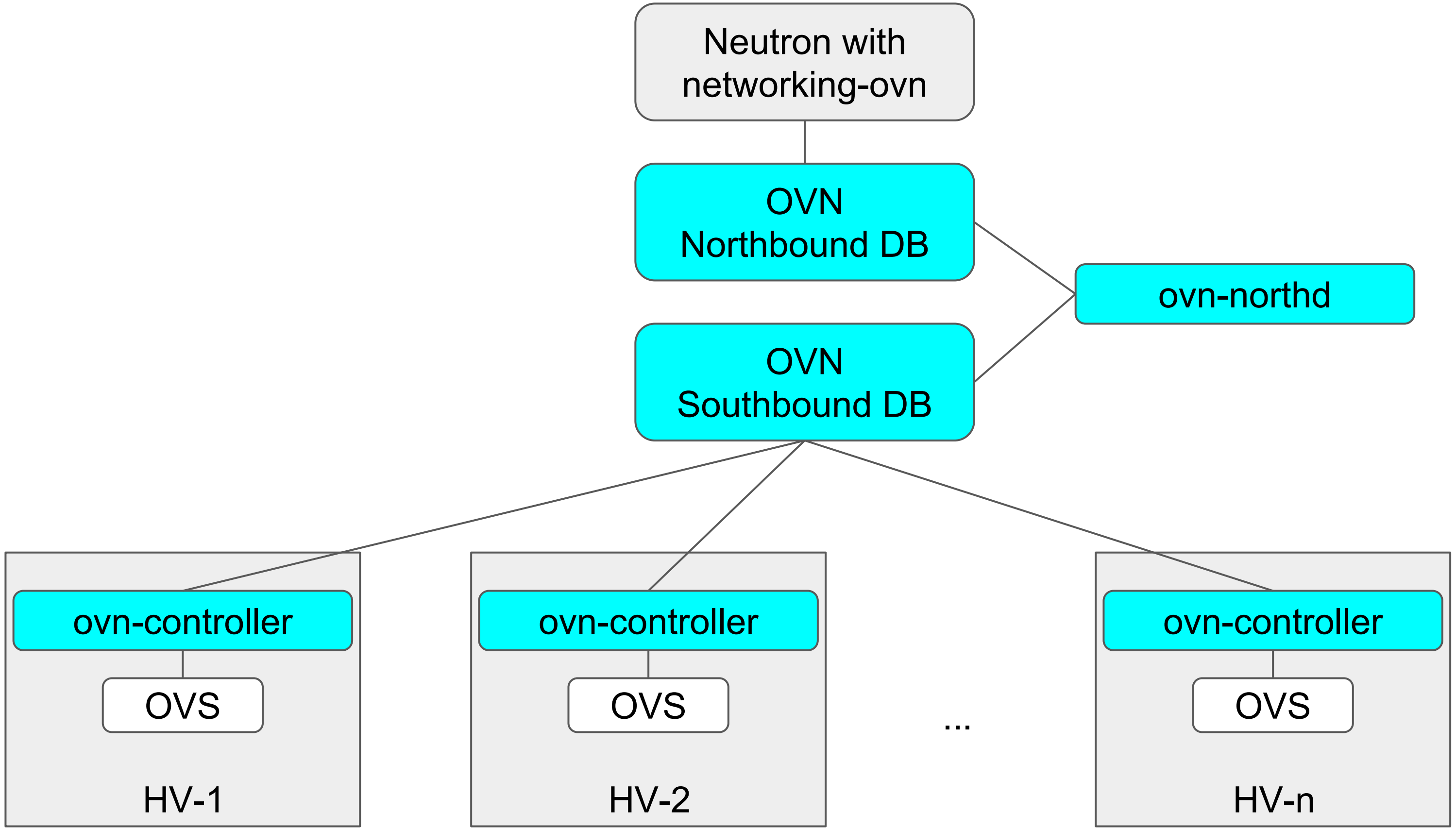
1. Logical Configuration in Northbound Database



2. ovn-northd Populates Southbound Database



2. Hypervisors Generate Physical Flows



OVN Database HA

- Based on ovsdb-server
- Supports primary and a backup
- Using pacemaker to manage which node currently acts as the primary
- Next steps: multi-master OVSDB or etcd v3

Address sets

OVN has always supported arbitrary user-defined ACLs with flexible matches, e.g.:

IF ip4.src in {*a, b, c, …*} AND ip4.dst in {*d, e, f, …*} THEN accept|drop|etc.

This works fine for small sets, but it doesn't scale well to long lists of hosts (etc.) that recur in many ACL entries. OVN now supports "address sets" to reduce the size of these lists, e.g.:

IF ip4.src in {$set1} AND ip4.dst in {$set2} THEN accept|drop|etc.

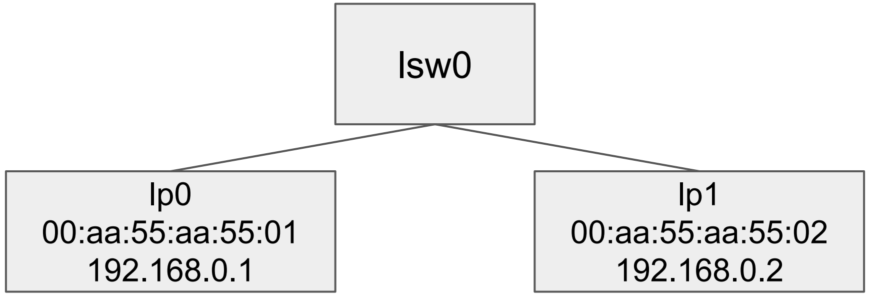
This also makes OVN setups with complex ACLs easier to understand.

Debugging OVN

Using ovn-trace to understand OVN

“What’s happening to these packets?”
“What if…?”
ovn-trace answers these questions and more.

- Only requires access to southbound database.
- Multiple output formats with varying levels of detail.
- Physical network layout independent.
- Provides references back to source code, to make debugging easier for developers
- Omits trivial match-action tables that would otherwise clutter output.



ovn-trace example #1: detailed output

```
$ ovn-trace lsw0 ‘inport == "lp0" && eth.src == 00:aa:55:aa:55:01 && eth.dst == ff:ff:ff:ff:ff:ff’

ingress(dp="lsw0", inport="lp0")
 0. ls_in_port_sec_l2 (ovn-northd.c:2826): inport == "lp0" && eth.src == {00:aa:55:aa:55:01}, priority 50
        next(1);
13. ls_in_l2_lkup (ovn-northd.c:3071): eth.mcast, priority 100
        outport = "_MC_flood";
        output;

multicast(dp="lsw0", mcgroup="_MC_flood")
        egress(dp="lsw0", inport="lp0", outport="lp0")
        /* omitting output because inport == outport && !flags.loopback */

        egress(dp="lsw0", inport="lp0", outport="lp1")
        8. ls_out_port_sec_l2 (ovn-northd.c:3146): eth.mcast, priority 100
        output;
        /* output to "lp1", type "" */
```

ovn-trace example #1: minimal output

```
$ ovn-trace lsw0 ‘inport == "lp0" && eth.src == 00:aa:55:aa:55:01 && eth.dst == ff:ff:ff:ff:ff:ff’
```

```
output("lp1");
```

ovn-trace example #2: DHCP request processing

```
$ ovn-trace lsw0 'inport == "lp0" && eth.src == 00:aa:55:aa:55:01 && eth.dst == ff:ff:ff:ff:ff:ff
&& ip4.src == 0.0.0.0 && ip4.dst == 255.255.255.255 && udp.src == 68 && udp.dst == 67'
```

```
/* We assume that this packet is DHCPDISCOVER or DHCPREQUEST. */;
put_dhcp_opts(offerip = 192.168.0.1, netmask = 255.255.255.0,
              router = 192.168.0.254, server_id = 192.168.0.253,
              lease_time = 3600);
eth.dst = 00:aa:55:aa:55:01;
eth.src = 00:aa:55:aa:55:fd;
ip4.dst = 192.168.0.1;
ip4.src = 192.168.0.253;
udp.src = 67;
udp.dst = 68;
output("lp0");
```

Future Work

Neutron driver (networking-ovn) next steps

- Multi-node CI job in progress
- Improved upgrade CI
- Completion of SFC API
- Keeping up with OVN

OVN Next Steps

- Native DNS
- Encrypted tunnels
- Database Clustering
- BPF for OVS
- Service Function Chaining

BPF Datapath

- BPF provides a safe, virtual sandbox in the Linux kernel (as well as other platforms)
- DPDK-like performance in Linux kernel with XDP
- Potentially greater portability across kernel versions and platforms
- Insert new functionality at run-time:
  - New network and tunneling protocols
  - Push OVN-specific actions into the fastpath

SFC (Service Function Chaining)

- Working POC with OVN and the networking-sfc API
- “Delivering OpenStack NFV Service Chaining at Scale with Networking-SFC and Networking-OVN”
  - Thursday, 1:50pm-2:30pm

Other Resources

- OVS/OVN Repository
  - https://github.com/openvswitch/ovs
- Kubernetes OVN Plugin
  - https://github.com/openvswitch/ovn-kubernetes
- OVS Orbit Podcast
  - https://ovsorbit.benpfaff.org/
- OVS Conference, 7-8 November 2016 in San Jose, CA
  - http://events.linuxfoundation.org/events/open-vswitch-2016-fall-conference

Thank you!

Ben Pfaff - @Ben_Pfaff
Justin Pettit - @Justin_D_Pettit
Russell Bryant - @russellbryant