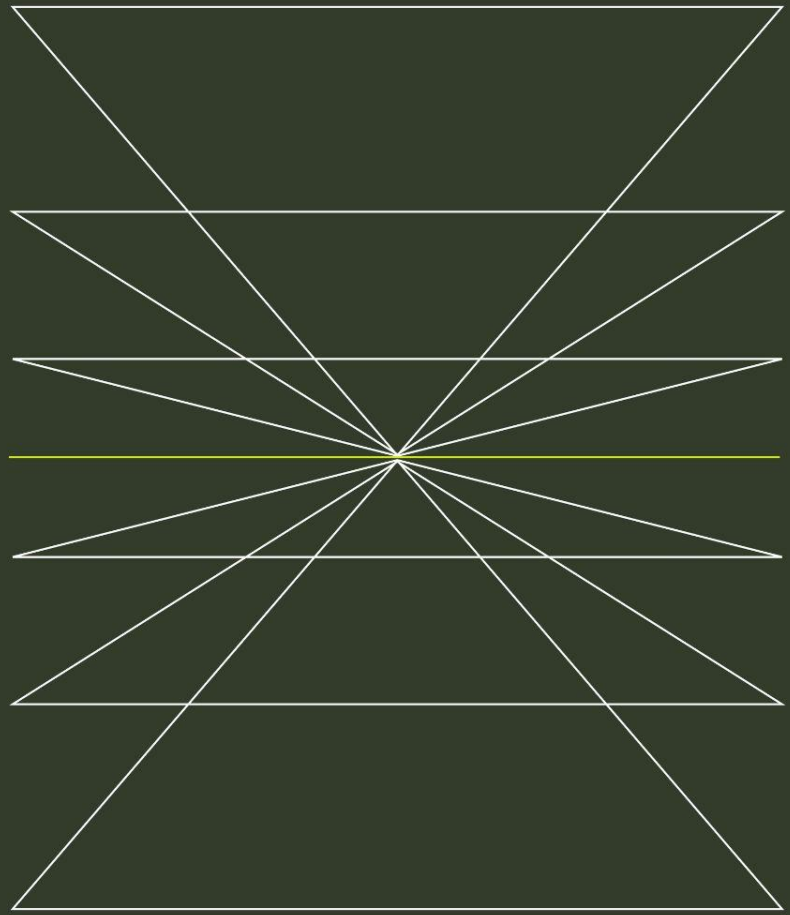
Crusoe

Programming OVS Bridges using OVN Bridge Controller



Numan Siddique

Senior Staff Engineer
Crusoe

Agenda

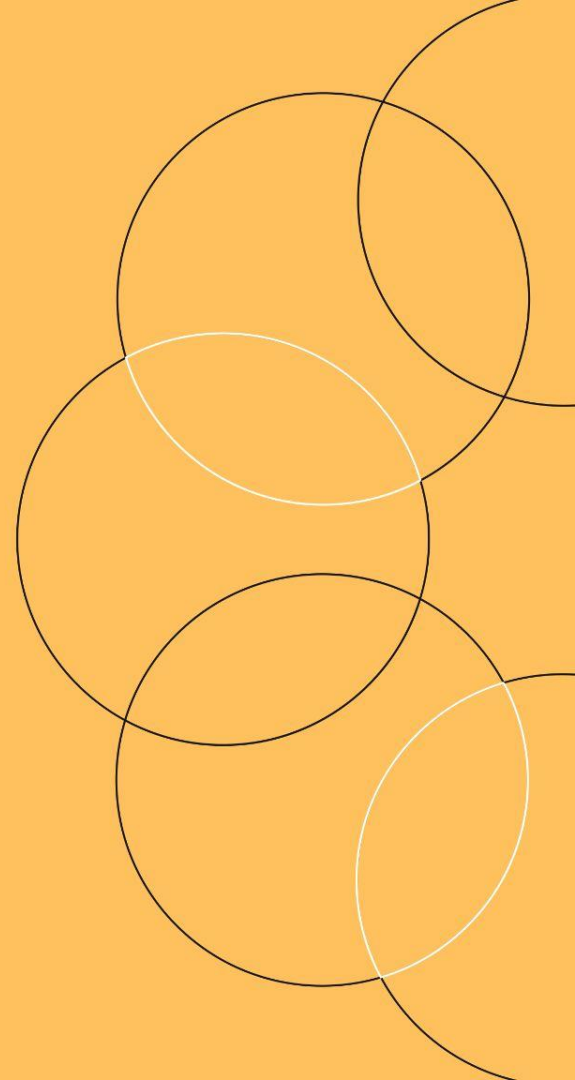
01 What is OVN Bridge Controller

02 Motivation

03 Benefits

04 Basic Architecture

05 How to use it



What is OVN Bridge Controller

- Just provides an abstraction over OpenFlow rules and protocol using OVN Logical flows.
- CMS programs the flow pipeline using OVN logical flows and writes to the OVN Bridge database.
- OVN Bridge controller translates the logical flows to OpenFlow rules.
- OVN Bridge controller and OVN Bridge database runs on each hypervisor.

Motivation and Why ?

- At Crusoe we use OVN as our SDN controller.
- We have the requirement to program OpenFlow rules in the provider bridge (after the packet leaves OVN pipeline).
- We had to write an additional controller (in golang) - which uses digitalocean/go-openvswitch to program the bridge (which internally uses ovs-ofctl).
- Why not use OVN logical flows instead.

Some of the advantages (I think of)

- OVN logical flows provide a good abstraction for OpenFlows.
- Defining your pipeline becomes faster, easier and readable*
- No need to exec ovs-ofctl for every flow add/delete
- Packet-in controller actions can be supported easily if required (like put_dhcp_opts, put_fdb etc)

* Note: Some of these things can be subjective

Some of the disadvantages (I think of)

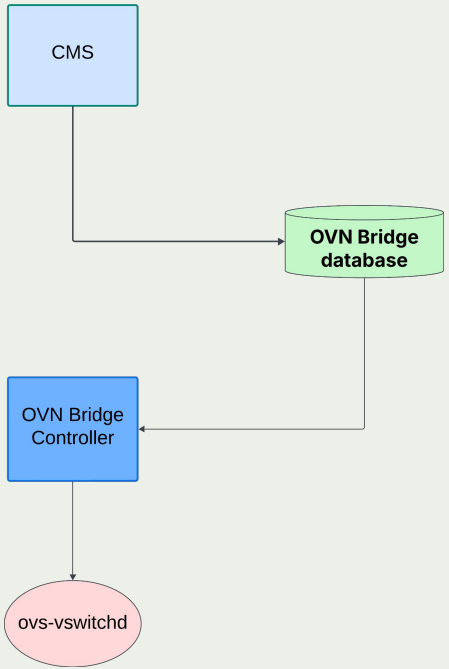
- User need to learn the syntax of OVN logical flows
- 2 additional services in the hypervisor to manage

Present Status

- RFC proposal was submitted in June
- Community decided to have this service as part of OVN project instead of a separate one.
- ovn-br-controller shares a lot of code with ovn-controller.
- Patches to add this feature are merged in upstream main just recently (a week ago)
- Will be available in OVN 26.03 release.

Architecture

- An OVSDB database - OVN Bridge.
- 2 Main tables - Bridge and Logical_Flow.
- CMS creates the Bridge for each OVS bridge.
- CMS programs the logical flows.
- ovn-br-controller (very similar to ovn-controller) translates the logical flows to OpenFlow rules and programs the bridges.



Logical flow structure

- Exactly like OVN logical flow
- Has 5 fields
    - Bridge
    - Table id
    - Priority
    - Match
    - Actions

How to use: Step 1

Start OVN bridge database and ovn-br-controller service on the hypervisor/compute node

```
/usr/share/ovn/scripts/ovn-ctl  start_ovnbr_ovsdb
/usr/share/ovn/scripts/ovn-ctl start_ovnbr_controller
```

Configure the external_ids:ovn-br-remote in the OpenvSwitch table so that ovn-br-controller can connect to the ovnbr ovsdb0-server.

```
ovs-vsctl set open .
external_ids:ovn-br-remote="unix:/var/run/ovn/ovnbr_db.sock
```

How to use: Step 2

- Create a bridge in the OVN bridge database

```
ovn-brctl add-br br-ex
```

Note: CMS should create the bridge and its ports in the OVS database.

- Add logical flows to the bridge

```
ovn-brctl add-flow br-ex 0 1000 “inport == \”p1\” && ip4 && tcp”
“drop;”
ovn-brctl add-flow br-ex 0 0 “ip4“ “next;”
ovn-brctl add-flow br-ex 1 100 “inport == \”p2\” && eth.dst ==
00:00:01:01:00:01” “eth.src = 00:00:41:11:00:a1; outport = \”p3\”;
output;”
ovn-brctl add-flow br-ex 1 0 “1“ “output;”
```

Repeat Step 2 for additional bridges

OVN Bridge controller supports managing and programming multiple OVS bridges

OVN Bridge controller behaviour

- Opens an openflow connection for each bridge configured.

- Connects to the local OVS database to get the OVS interfaces added to the bridges.
- Translates the logical flows to OpenFlow rules
- Also adds a few OpenFlow rules to load the interface ofport to the registers
- Always adds a flow with NORMAL action.

OVN Bridge controller - physical flows

Let's say we have an OVS bridge with the below ports:

```
Bridge br-ex
        datapath_type: system
        Port br-ex
            Interface br-ex
                type: internal
        Port eth2
            Interface eth2
        Port patch-ln-public1-to-br-int
            Interface patch-ln-public1-to-br-int
                type: patch
                options: {peer=patch-br-int-to-ln-public1}
```

OVN Bridge controller - physical flows

Adds the below OpenFlow rules:

```
table=0, priority=100,in_port="patch-ln-public" actions=load:0x2->NXM_NX_REG14[],resubmit(,8
table=0, priority=100,in_port=eth2 actions=load:0x1->NXM_NX_REG14[],resubmit(,8)

table=0, priority=0 actions=NORMAL

table=120,priority=100,reg14=0x2,reg15=0x2
actions=push:NXM_OF_IN_PORT[],load:0xffff->NXM_OF_IN_PORT[],resubmit(,121),pop:NXM_OF_IN_PORT[]

table=120, priority=100,reg14=0x1,reg15=0x1
actions=push:NXM_OF_IN_PORT[],load:0xffff->NXM_OF_IN_PORT[],resubmit(,121),pop:NXM_OF_IN_PORT[]

table=120 priority=0 actions=resubmit(,121)

table=121, priority=100,reg15=0x1 actions=output:eth2
table=121, priority=100,reg15=0x2 actions=output:"patch-ln-public"

table=121, priority=0 actions=NORMAL
```

OVN Bridge controller - Logical flows

Translates the logical flows to OpenFlow rules

```
# ovn-brctl add-flow br-ex 0 100 "inport == \"eth2\" && eth.dst ==
ae:f0:00:00:00:01" "eth.dst = be:f0:00:00:00:01; outport =
\"patch-ln-public1-to-br-int\"; output;"
```

Corresponding OVS flow

```
table=8, priority=100,reg14=0x1,dl_dst=ae:f0:00:00:00:01
actions=mod_dl_dst:be:f0:00:00:00:01,load:0x2->NXM_NX_REG15[],resubmit(,120)
```

Limitations

- Doesn’t restrict adding “br-int” to the OVN Bridge table.
- Doesn’t restrict using OVN specific logical actions which translates to controller action Eg. put_dhcp_opts, put_fdb, dns_lookup etc.

These limitations will be addressed soon.

Future improvements

- Address the limitations mentioned earlier.
- Add incremental processing support.
- Reduce code duplication by moving the flow management and ofctrl handling to the lib so that both. ovn-controller and ovn-br-controller can use them.
- Add any missing OVS match and actions.

Thank you

Questions ?

Please try out ovn-br-controller and provide any feedback.