Red Hat
Enterprise Linux

OVN issues in the field

Numan Siddique
Red Hat, Bengaluru
@numansiddique

Red Hat

What we'll be discussing today

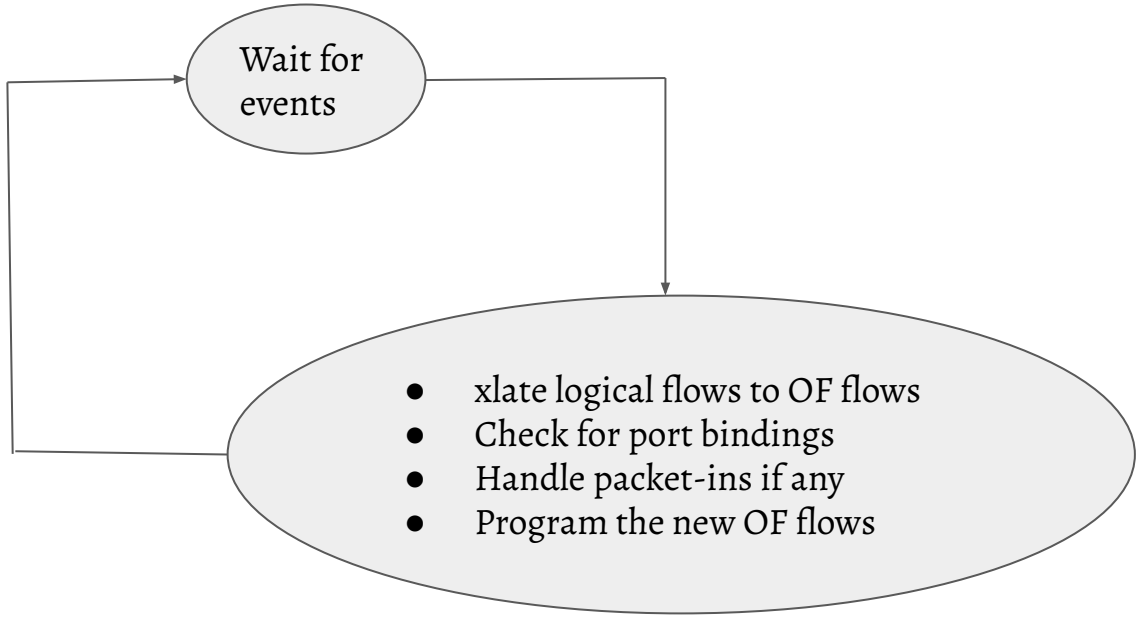
- Some of the OVN issues seen in production
- How we solved or mitigated it
- How can we improve OVN further

Deployment

- OpenStack deployment using OSP13 (Queens)
- OpenvSwitch 2.9
- OVN 2.9
- Later moved to OVN 2.12

ovn-controller design

(before Incremental processing)



ovn-controller design

Before incremental processing (I-P)

- Main while loop which handles events
- In each run
  - Translates logical flows to OpenFlow rules - lflow_run()
  - lflow_run() is called even for pinctrl (packet-ins) events.

After incremental processing (I-P)

- Main while loop which handles events
- In each run
  - Translates only required logical flows to OpenFlow rules.
  - Pinctrl events doesn't cause flow translation.

Issue #1 - ovn-controller 100% CPU
continuous reconnection to SB ovsdb-server

Cause:

- ovn-controller takes >5 seconds to process logical flows in one run
- The IDL connection to ovsdb-server sends probes periodically.
- The default value is 5 seconds.
- If lflow_run() takes > 5 seconds, the IDL connection is closed and reopened.
- Results in snowball effect.

Resolution:

- Increase the default probe interval time - ovn-remote-probe-interval

Eg. To set 180 seconds probe interval.

```
ovs-vsctl set open .
external_ids:ovn-remote-probe-interval=180000
```

Issue #2 - ovn-controller 100% CPU
connection drops from ovsdb-server to its clients

Cause:

- ovn-controller takes >5 seconds to process logical flows in one run
- ovsdb-server sends probes periodically to all its clients.
- The default value is 5 seconds.
- If lflow_run() takes > 5 seconds, the IDL connection is closed and reopened.
- Results in snowball effect.

Resolution:

- Increase the default probe interval time.
- ovn-sbctl set-connection ptcp:6642:IP

To set 180 seconds probe interval.

- ovn-sbctl set connection . inactivity_probe=180000

Issue #3 - ovn-controller 100% CPU

continuous reconnection to openflow connection

Cause:

- ovn-controller takes >5 seconds to process logical flows in one run
- The openflow connection to ovs-vswitchd sends probes periodically.
- The default value is 5 seconds.
- If lflow_run() takes > 5 seconds, the openflow connection is closed and reopened.
- Results in snowball effect.

Resolution:

- Added a new configuration option - ovn-openflow-probe-interval.

Eg. To set 60 seconds probe interval.

ovs-vsctl set open . external_ids:ovn-openflow-probe-interval=60

Issue #4 - ovn-controller 100% CPU
continuous DHCP packet-ins

Cause:

- Any packet-in wakes up ovn-controller main loop.
- It calculates logical flows
- Handles the packet-in and responds.
- If the response is slow, the VIF can retransmit.
- Resulting in snowball effect.
- This issue was observed with DHCP requests (with OVN 2.9)

Resolution:

- Added a new thread in ovn-controller - pinctrl thread to handle packet-ins.

Issue #5 - ovn-controller 100% CPU
continuous ARP packet-ins

Issue:

- Periodic GARPs are received from the fabric every 10 seconds.
- Resulting in the logical flow computation.
- If lflow_run() takes > 10 seconds then 100% CPU usage.

Resolution:

- OVN 2.12 (which has incremental processing support)
- I-P helped.
- We added new OVN actions - *lookup_arp/lookup_nd*
- Send the packet to ovn-controller only if required.

Issue #6 - VMs failed to spawn in some compute nodes.

Setup:

- OpenStack deployed using tripleo.
- OVN ovsdb-servers deployed in active/standby using pacemaker and a VIP

Cause:

- OVN ovsdb-server VIP moves from one node to another (due to failover)
- All the ovn-controllers connect to the new SB ovsdb-server.
- But some have read-only connection to SB ovsdb-server due to bug in ovsdb-server.
- Results in transaction failovers and 100% CPU usage.

Resolution:

- Fixed the issue in ovsdb-server to handle the existing connections' read-only status properly.

Issue #7 - MAC Binding update failures.

Issue:

- Ping from one VM to another VM using its floating ip (dnat) fails if these VMs are connected to different logical routers.

Cause:

- Same as the previous case.
- If ovn-controller which learns the mac_binding has a read-only connection to SB ovsdb-server, mac_binding update fails.
- Also results in 100% CPU.

Resolution:

- Fixed the issue in ovsdb-server to handle the existing connections' read-only status properly.

Issue #8 - Failure in VRRP workloads

Setup:

- OpenStack deployed using tripleo.
- VRRP using keepalived.

Issue:

- When the VIP moves, OVN doesn't update the MAC_Binding table with the new MAC.

Cause:

- When the VIP moves, the VM sends out a GARP.
- Pinctrl thread updates the local mac_binding cache.
- Main ovn-controller thread discards the learnt mac without updating if it is older than 1 second.

Resolution:

- Removed this time check condition.

Issue #9 - Issues in conjunction flows

Issue:

- OVN generates conjunction flows from logical flows.

  Eg

  match = “inport == {port group 1 } && ip4.src == {IP1, IP2, IP3} && tcp.dst >= 2000 && tcp.dst <= 3000”
  action = drop

  match = “inport == {port group 1 } && ip4.src == {IP1, IP2, IP3} && tcp.dst >= 3000 && tcp.dst <= 4000”
  action = allow

  OF flows:
  inport == {pg1} action=conjunction(1, 1/3)
  ip4.src == {IP1, IP2, IP3} action=conjunction(1, 2/3)
  tcp.dst >= 2000 && tcp.dst <= 3000, conjunction=(1, 3/3)
  match = conj_id=1, action = drop

  inport == {pg1} action=conjunction(2, 1/3)
  ip4.src == {IP1, IP2, IP3} action=conjunction(2, 2/3)
  tcp.dst >= 3000 && tcp.dst <= 4000, conjunction=(2, 3/3)
  match = conj_id=2, action = allow

Issue #9 - Issues in conjunction flows (cont)

- We resolved it by disabling conjunction in OVN.
- But this resulted in huge amount of OF rules and lflow_run()took 20x more time.
- Finally resolved the issue in OVN by generating proper OF rules

OF flows:

```
inport == {pg1} action=conjunction(1, 1/3), conjunction(2, 1/3)
ip4.src == {IP1, IP2, IP3} action=conjunction(1, 2/3), conjunction(2, 2/3)
tcp.dst >= 2000 && tcp.dst <= 3000, conjunction=(1, 3/3)
match = conj_id=1, action = drop
tcp.dst >= 3000 && tcp.dst <= 4000, conjunction=(2, 3/3)
match = conj_id=2, action = allow
```

Bottlenecks/Future Improvements

Improve logical flow processing

- lflow_run() takes lot of time in processing the logical flows.
- For approx. 35000 logical flows, lflow_run() takes ~10 seconds.
- We have seen setups where it takes more than 30 seconds too.
- Most of the time is spent in malloc and its friends in lib/expr.c

We need to

- Improve/rewrite expr.c
- Or cache expr parsing
- Or improve I-P engine

Improve Incremental Processing engine

- I-P engine can be improved further
- It triggers recomputations
  - For local ovs database changes
  - when a logical switch/logical router is created.
  - When a port is bound on the chassis
  - When a gateway chassis redirect port moves
  - …

Improve debugging

- Debugging flows/tracing packets is hard.
- We need tools to visualize the logical network. (may be Skydive)
- Dumitru Ceara added few patches in this regard
  - eb25a7da639e ("Improve debuggability of OVN to OpenFlow translations.")
  - 8051499a6c1b ("ovn-detrace: Add support for other types of SB cookies.")
- And we need to add more

Separate pinctrl process ?

- Pinctrl thread delegates updating MAC_Binding table to the main thread
- Doesn't access the SB DB IDL contents.
- Instead maintains a local cache of DNS table, ARP entries, IGMP entries etc

Separate process

- Having a separate process will avoid all the above. It can have its own IDL connection.
- But will increase the load on the ovsdb-server as the number of connections to SB ovsdb-server will be (N * 2) where N is number of chassis in the deployment.
- Mark submitted a RFC patches a while bck to separate the pinctrl process.

Thank you

Questions?

linkedin.com/company/red-hat

youtube.com/user/RedHatVideos

facebook.com/redhatinc

twitter.com/RedHat

Red Hat