Baidu Vswitch Hotupgrade
A new way to upgrade vswitch with nearly zero downtime

Yuan Linsi
Baidu AI Cloud
Agenda

• Evolution of Virtual Network Data Plane
• Challenge
• Optional Solutions
• Our Solutions
  • The requirement and design goals
  • design
  • benefits
• Further work
Evolution of Virtual Network Data Plane
Evolution of Virtual Network Data Plane

Key advantages
• High Performance
• Low latency
• Lower CPU overhead, higher efficiency

*co-work with Mellanox
How to upgrade?

• Need to work for different scenario, especially for the Smart Nic
• upgrade do not affect customer's service
• The larger the cluster scale is, the more complexity the problem will be
Optional Solutions

Solution 1: restart process

upgrade procedure:
- Saving flows
- Exiting ovsdb-server
- Starting ovsdb-server
- stop forwarding
- flow restore wait
- start_forwarding
- restore flow
- flow restore complete

Advantage:
- work for both ovs-kernel and ovs-dpdk
- no extra resource required

Problem:
- break time is too long to be acceptable
- break time is unpredictable
Optional Solutions

Solution 2: Two-process backup

Upgrade procedure
- primary process hold the resource, secondary process deal with the traffic
- directly restart the secondary vswitch
- skip the initialization

Advantage:
- break time is predictable
- no extra resource required

Problem:
- only works for ovs-dpdk
- break time still in seconds
Optional Solutions

Solution 3: dual main-process

Upgrade procedure
- running on top of VF
- start new process and restore memory status
- switch traffic to the new one

Advantage:
- break time is predictable
- Millisecond break time

Problem:
- only works for ovs-dpdk
- require extra resource
Requirement and Design Goals

- Solutions need to be work for multiple different scenarios
- no extra resource required
- the break time is predictable and minimal
Summary of three Solutions

All of the solutions share something in common:

- All operations are process-based
- The essence of restore operations is trying to restore the memory status
Hot upgrade Design Overview

• Key points
  • restart threads instead of processes
  • hot upgrade via dynamic library hot replace
  • memory status sync up
Hot upgrade Solutions

- Key points 1: restart threads instead of processes

![Diagram showing the thread life cycle, quiescent state, and hot replace process. The diagram includes labels for stop thread, quiescent state, hot replace, runtime memory status, thread life cycle, and start thread.]

```
liba.so.0.0.1

stop thread

thread life cycle

quiescent state

hot replace

runtime memory status

runtime memory status

start thread

liba.so.0.0.2
```
Hot upgrade Solutions

- Key points 2: dynamic library hot replace
Hot upgrade Solutions

- Key points 3: memory status sync up
  - What kind of memory?
    -- only statically allocated memory
  - Why?

```
liba.0.0.so
.data
int xyz = 100
.bss

liba.0.1.so
.data
int xyz = 0
.bss
```

```
000000000000683 .gpp  
683 55           push %rbp
684 48 89 e5     mov %rsp,%rbp
687 8b 05 d3 03 20 00  mov 0x2003d3,%rip,%eax  # 200a60
xyz.2057>
```
Hot upgrade Break Time

- **prepare**
  - parse executable file
    - ELF sym info
  - load new library
  - parse library file
    - ELF sym info

- **hotupgrade**
  - stop thread
  - memory status sync up
  - hot replace
  - start thread

- **cleanup**
  - clean old ELF sym

**Break time**
- 700us
- 600us
Hot upgrade Break time in different scenario

Best case: can be nearly zero

Optimize the pmd reload

Best case: can be nearly zero
Hot upgrade Further Work

- prepare
  - parse executable file
    - ELF sym info
  - load new library
  - parse library file
    - ELF sym info

- hot upgrade
  - stop thread
  - memory status sync up
  - hot replace
  - start thread

- cleanup
  - clean old ELF sym

We can even do it much more better:
- sync up the memory status during compiling

break time
600us
Hot upgrade Advantage

- Work for both ovs-kernel and ovs-dpdk
- no extra resource required
- break time nearly zero
Acknowledgement

- Zhang Yu
- Mao YingMing
- Wang Li

Welcome to join Baidu AI Cloud!

yuanlinsi01@baidu.com
计算无限可能