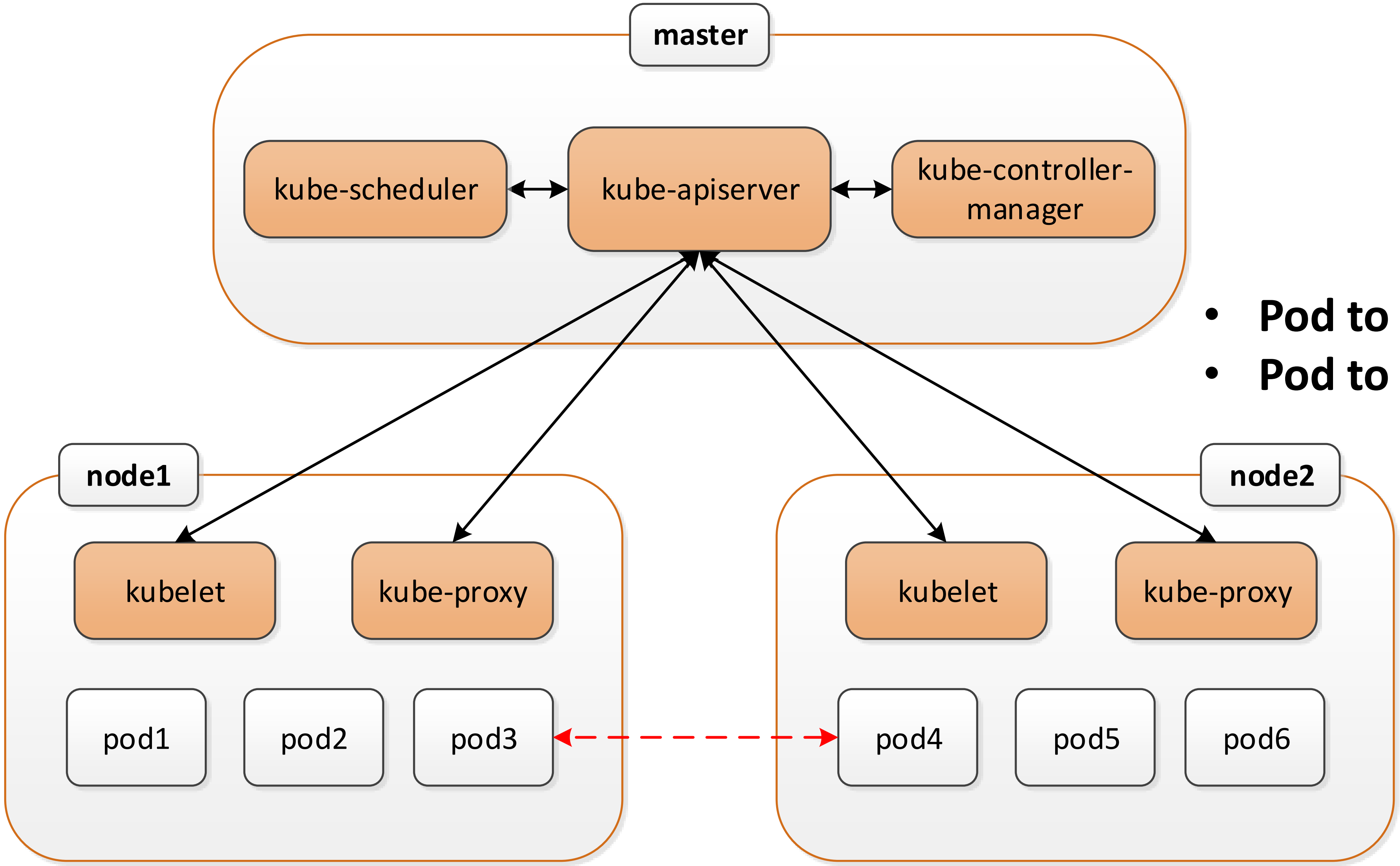
DPDK

DATA PLANE DEVELOPMENT KIT

DPDK Based Networking Products Enhance and Expand Container Networking

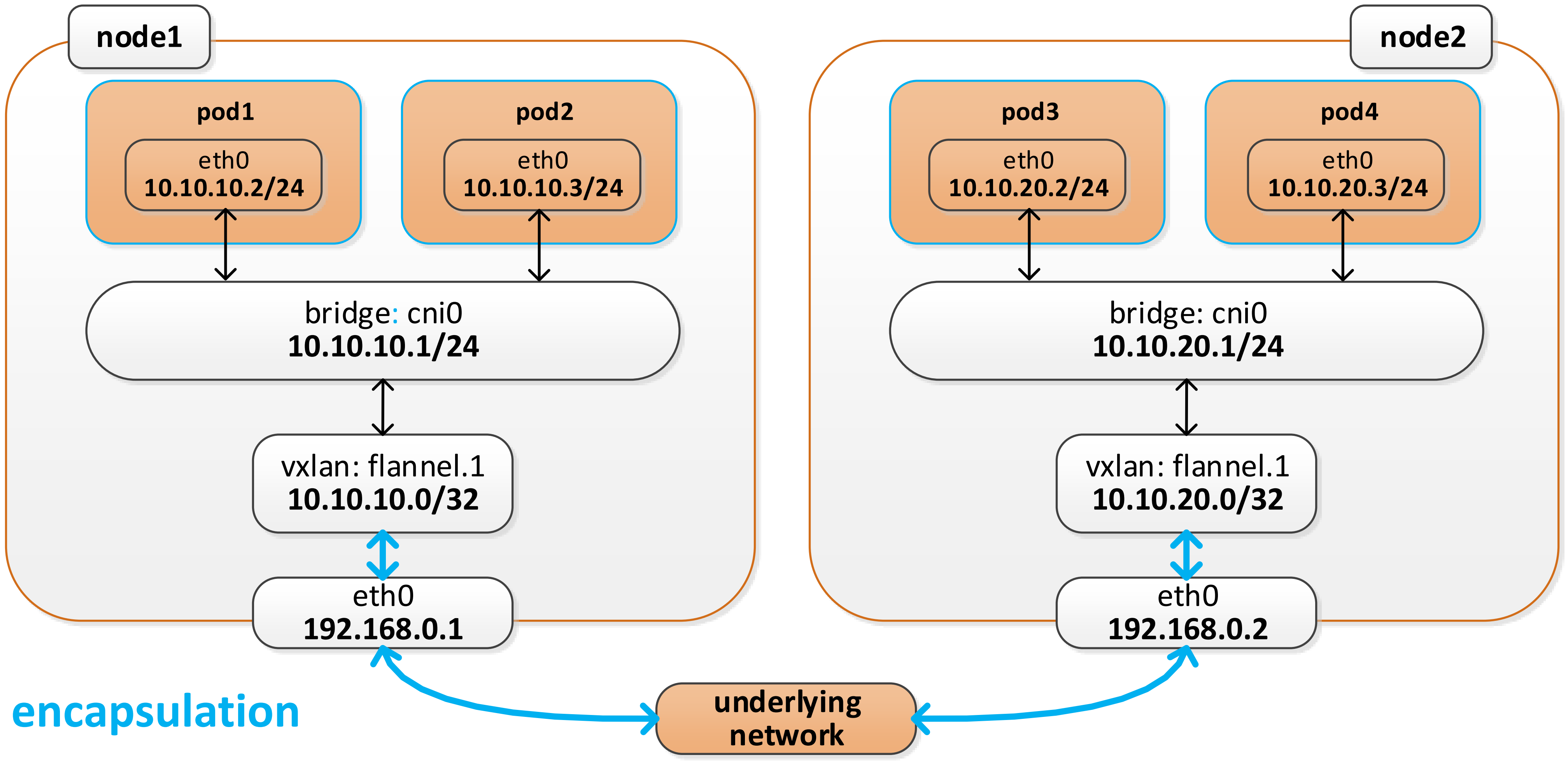
zhouzijiang@jd.com
Jingdong Digital Technology

Kubernetes Overview



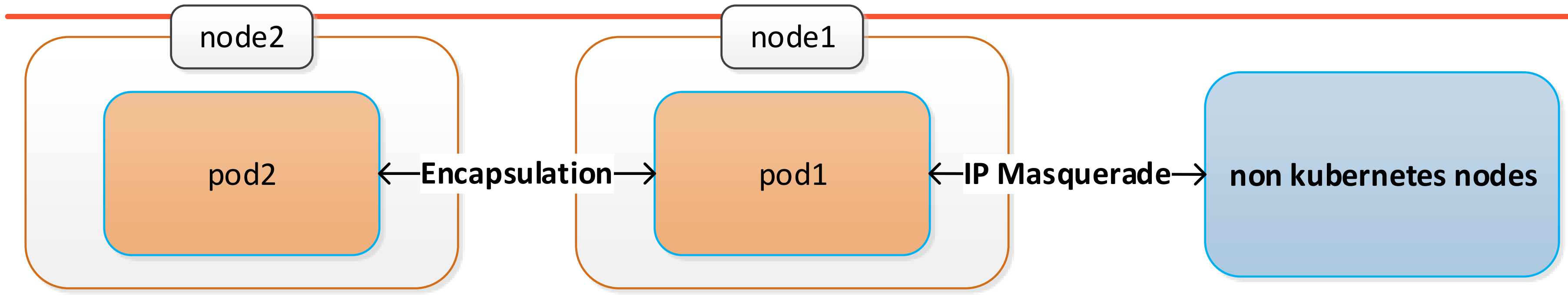
- **Pod to Pod communication**
- **Pod to Service communication**

Flannel Overview



VXLAN encapsulation

Outer Ethernet header	Outer IP header **src: 192.168.0.1** **dst: 192.168.0.2**	Outer UDP header	Vxlan header	Inner Ethernet header	Inner IP header **src: 10.10.10.2** **dst: 10.10.20.3**	Payload



1、 pods communicate with endpoints in k8s cluster, packets must be encapsulated
2、 pods communicate with endpoints out of k8s cluster, packets must be masqueraded

It will lead to extra overhead. Besides, it can't meet some demands, e.g. pod wants to access white-list enabled application outside of k8s cluster

Our goals:

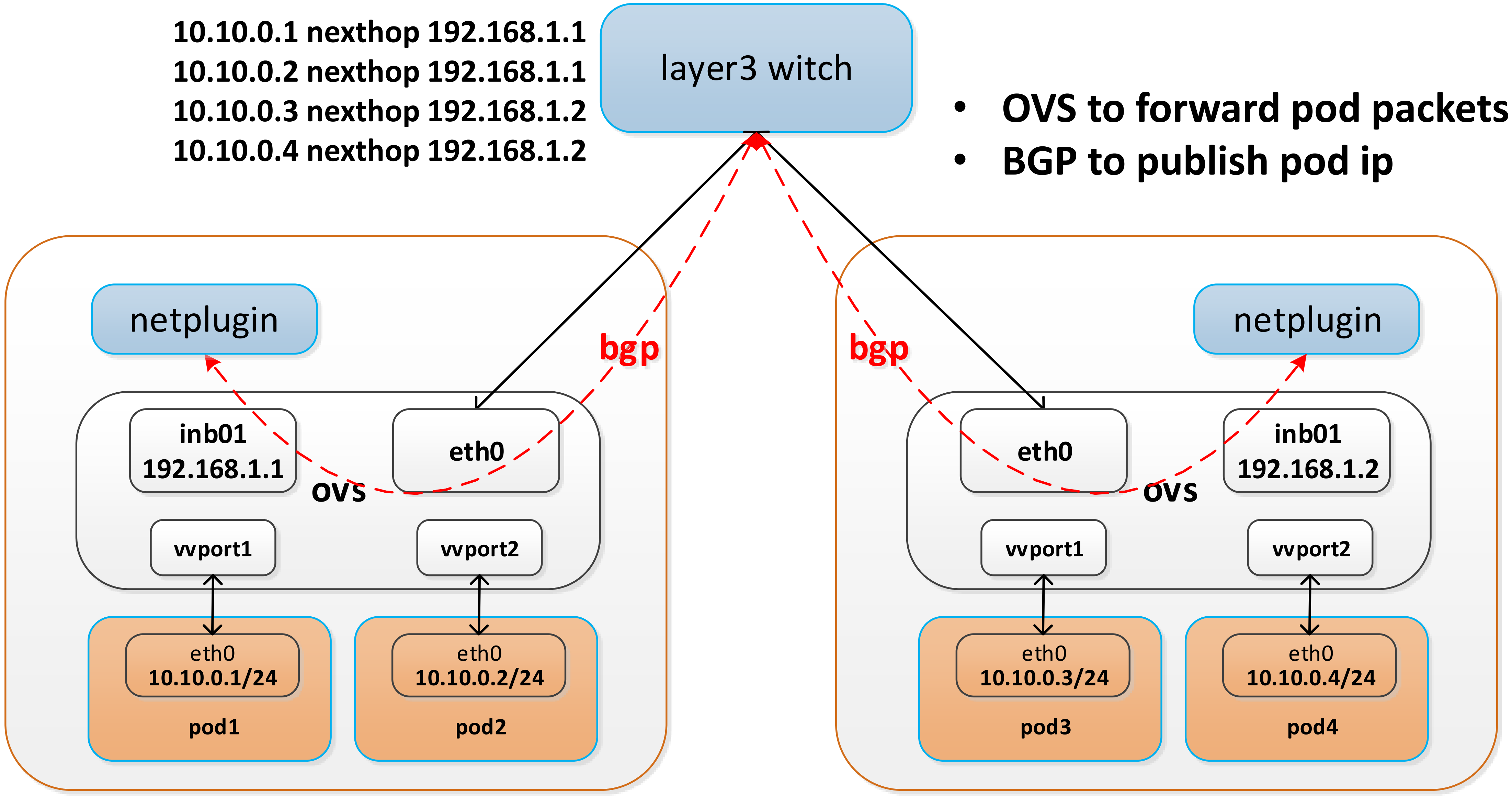
- no encapsulation
- no network address translation
- pods can be reached from everywhere directly

Our Choice:

- **contiv with layer3 routing mode**

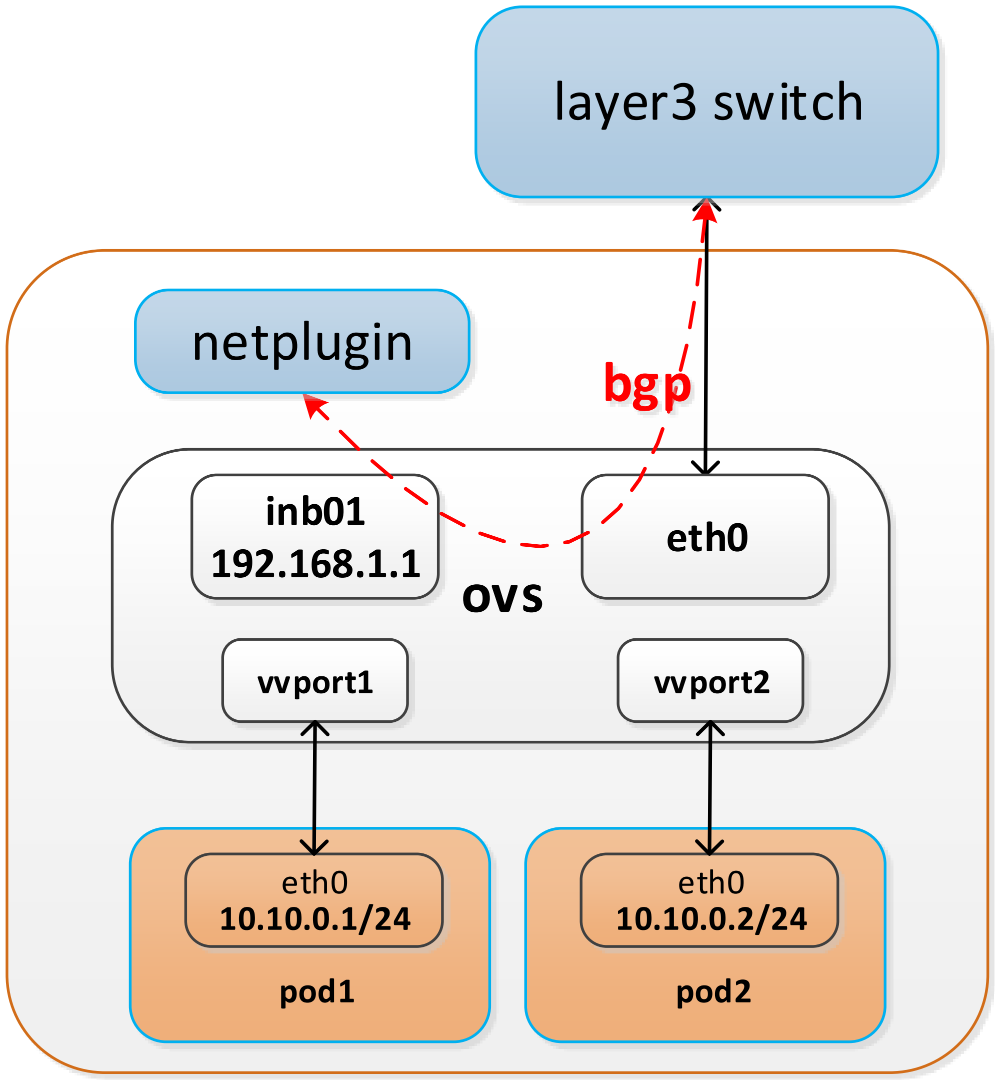
Contiv Overview

10.10.0.1 nexthop 192.168.1.1
10.10.0.2 nexthop 192.168.1.1
10.10.0.3 nexthop 192.168.1.2
10.10.0.4 nexthop 192.168.1.2



- OVS to forward pod packets
- BGP to publish pod ip

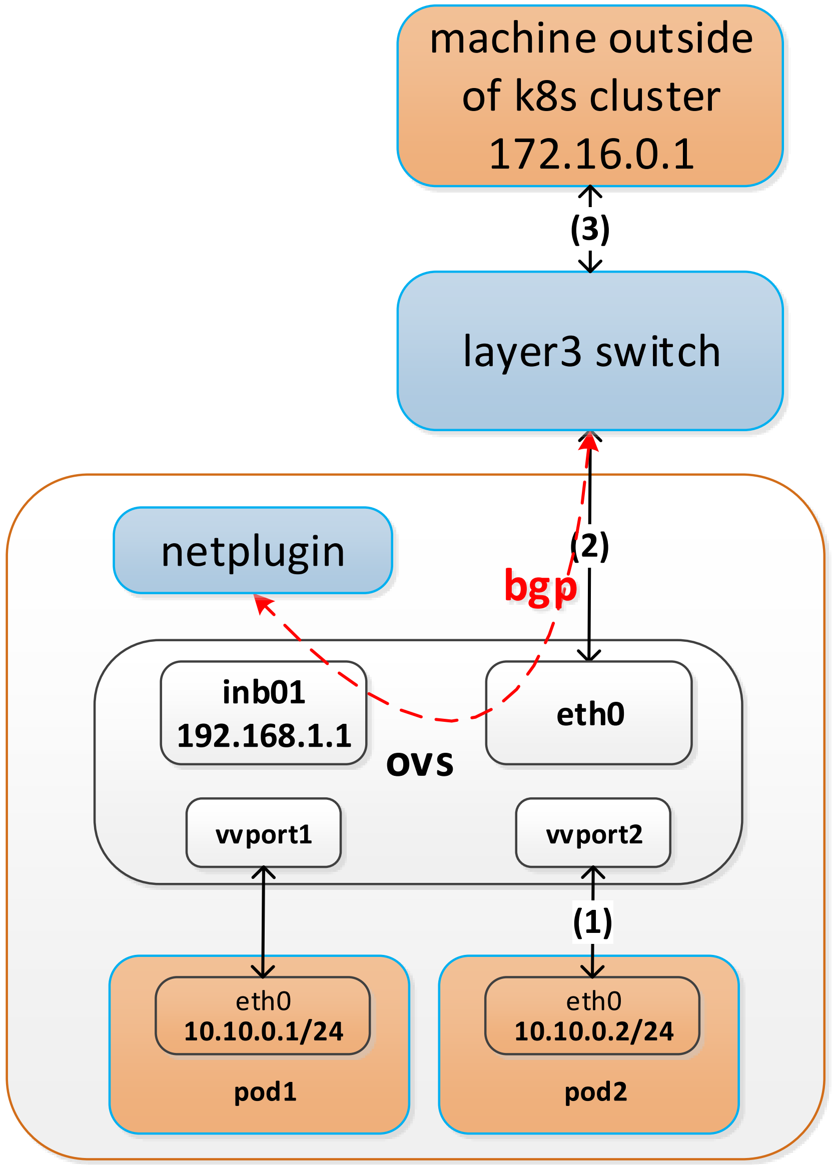
Contiv Implementation Detail



1、 user creates a new pod in k8s cluster
2、 netplugin requests a free ip 10.10.0.1 from netmaster
3、 netplugin creates a veth pair, such as vport1 and vvport1
4、 netplugin moves interface vport1 to pod network namespace and rename it to eth0
5、 netplugin sets ip and route in the pod network namespace
6、 netplugin adds vvport1 to ovs
7、 netplugin publishes 10.10.0.1/32 to bgp neighbor switch

- nw_dst=10.10.0.1 output:vvport1
- nw_dst=10.10.0.2 output:vvport2

Pod IP is Reachable in IDC Scope



10.10.0.2(in cluster) ping 172.16.0.1(outside cluster)

1、 pod2 sends out packet through its eth0

Ethernet header	src: 10.10.0.2 dst: 172.16.0.1	Payload

2、 ovs receives packet from vvport2 and forwards it to host eth0

Ethernet header	src: 10.10.0.2 dst: 172.16.0.1	Payload

3、 switch receives packet and forwards it to host 172.16.0.1

Ethernet header	src: 10.10.0.2 dst: 172.16.0.1	Payload

in the pod, in the host, in the underlying infrastructure，packet ip header is always the same

Contiv Optimization

1、 multiple bgp neighbors support

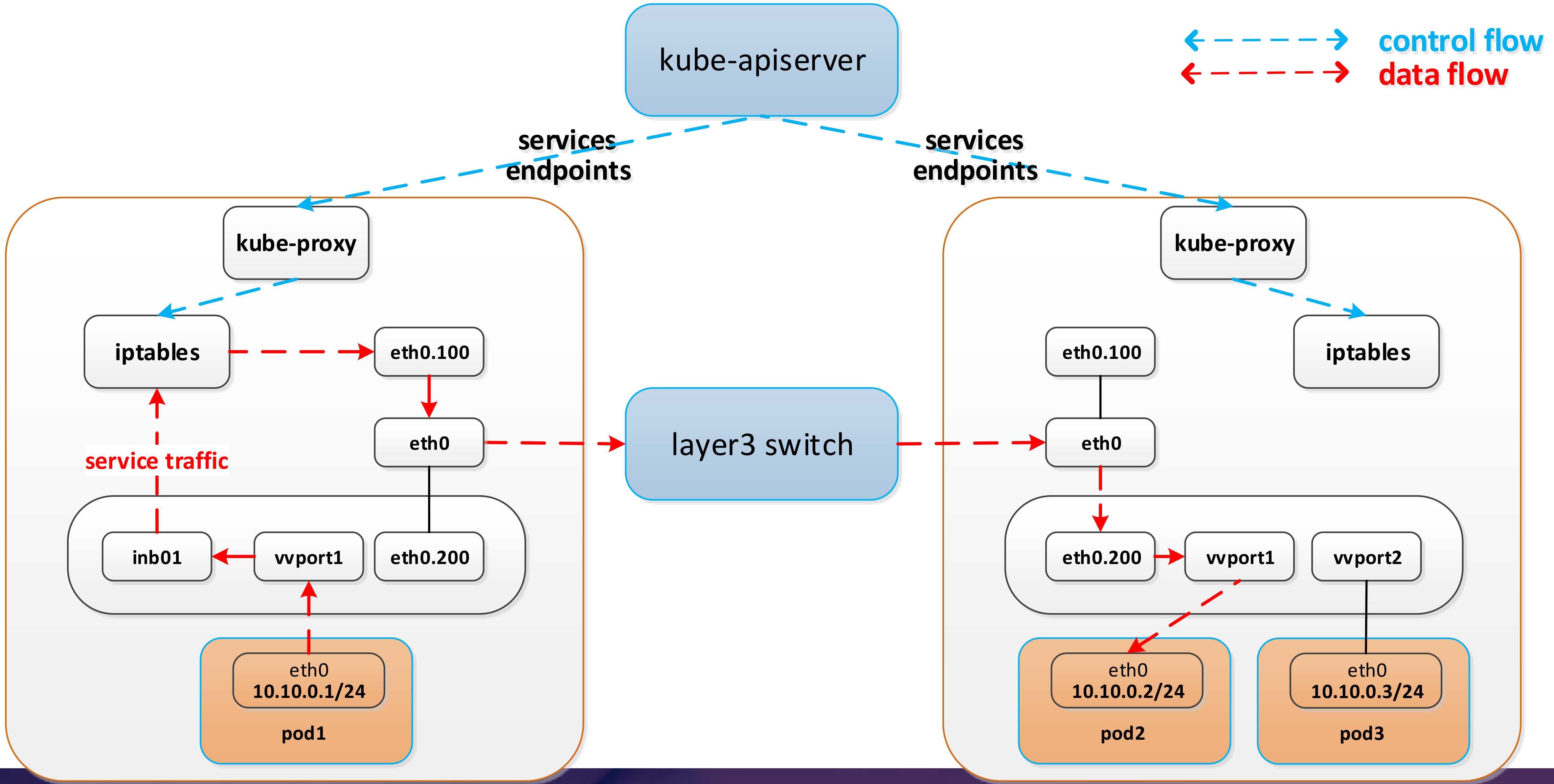
2、 reduce number of node’s ovs rules from magnitude of cluster to node

3、 remove dns and load balance module from netplugin

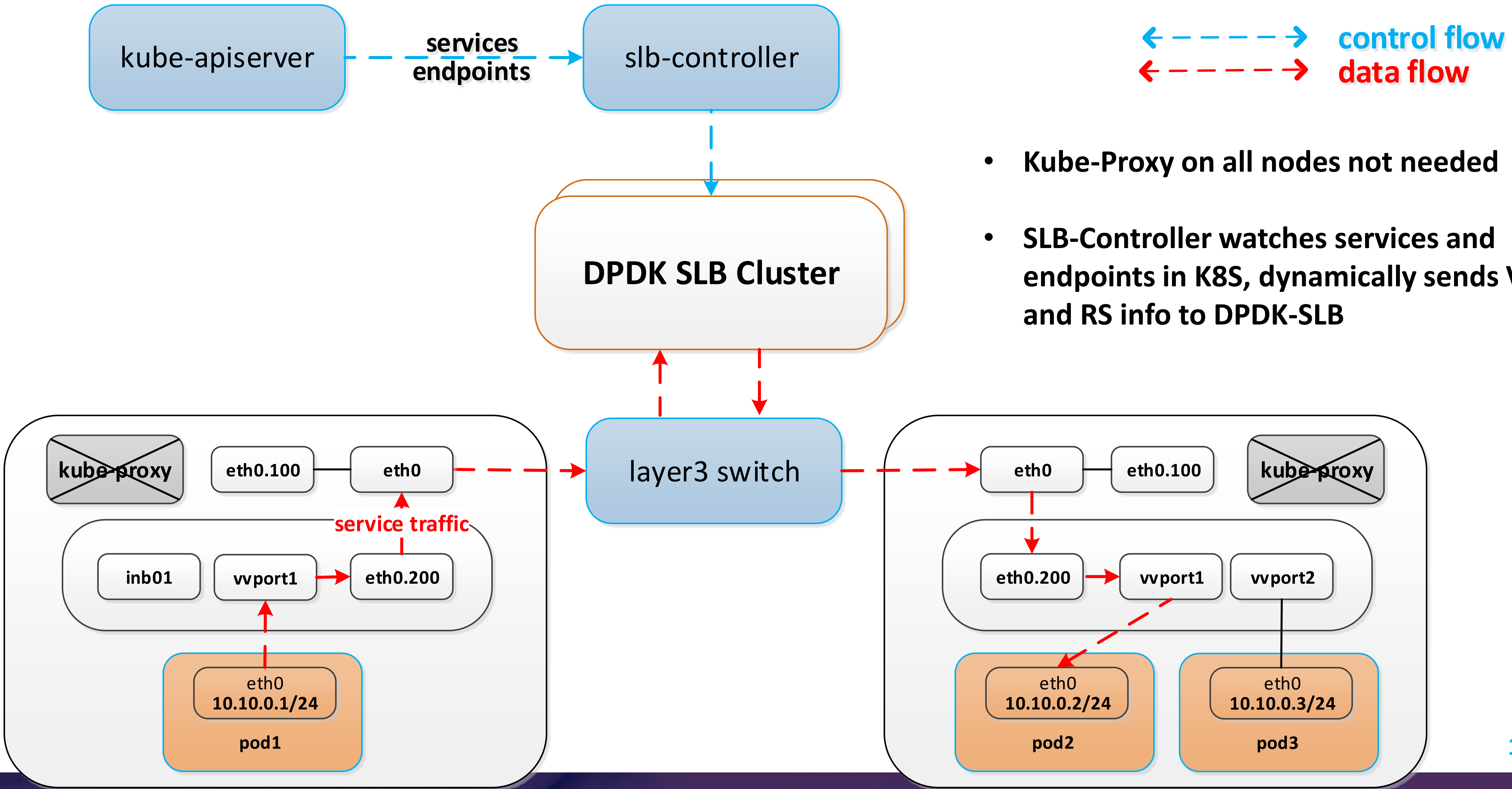
4、 add non-docker container runtime support, e.g. containerd

5、 add ipv6 support

Load Balance: Native KubeProxy

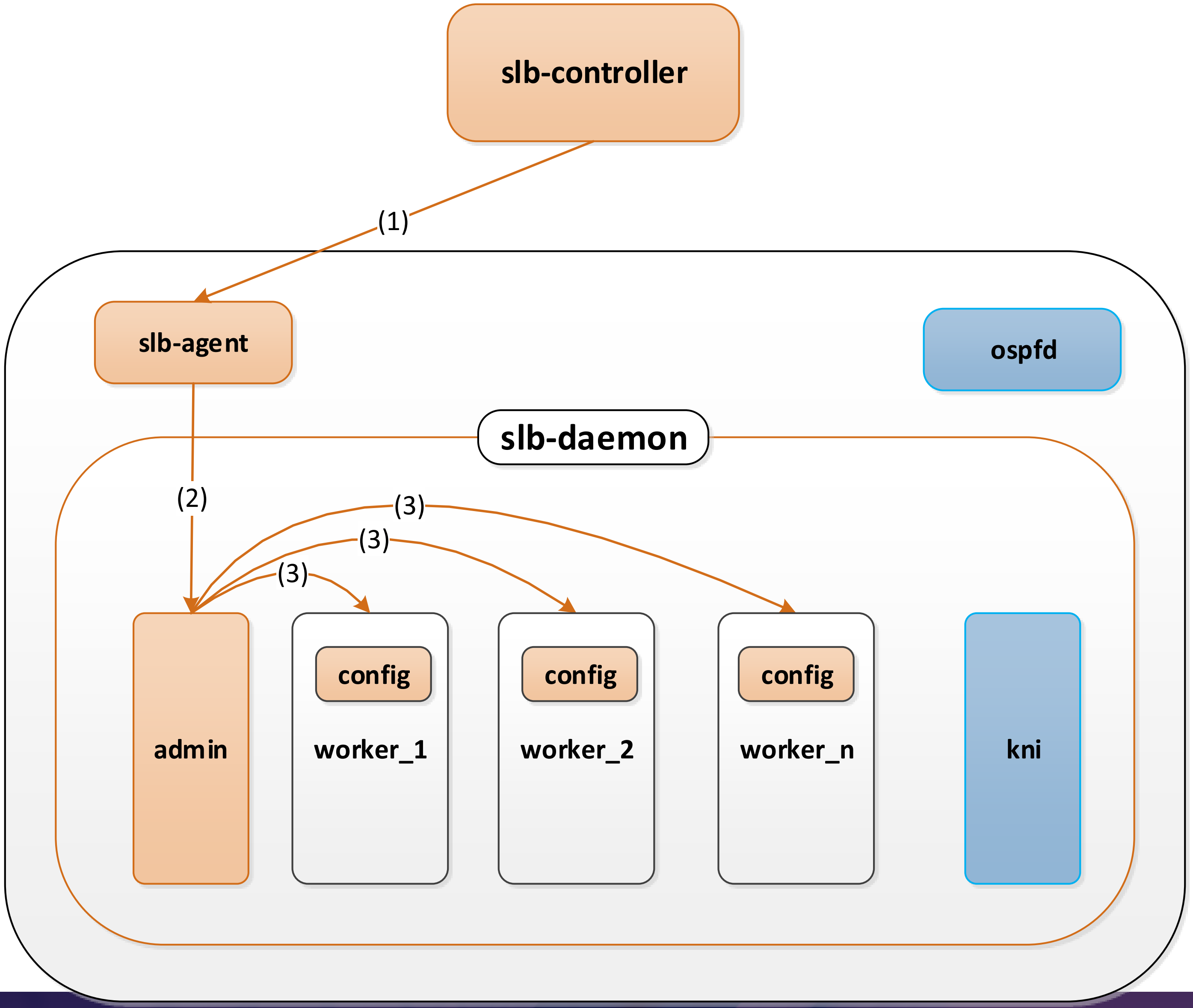


Load Balance: DPDK-SLB



- **Kube-Proxy on all nodes not needed**
- **SLB-Controller watches services and endpoints in K8S, dynamically sends VS and RS info to DPDK-SLB**

DPDK-SLB: Control Plane

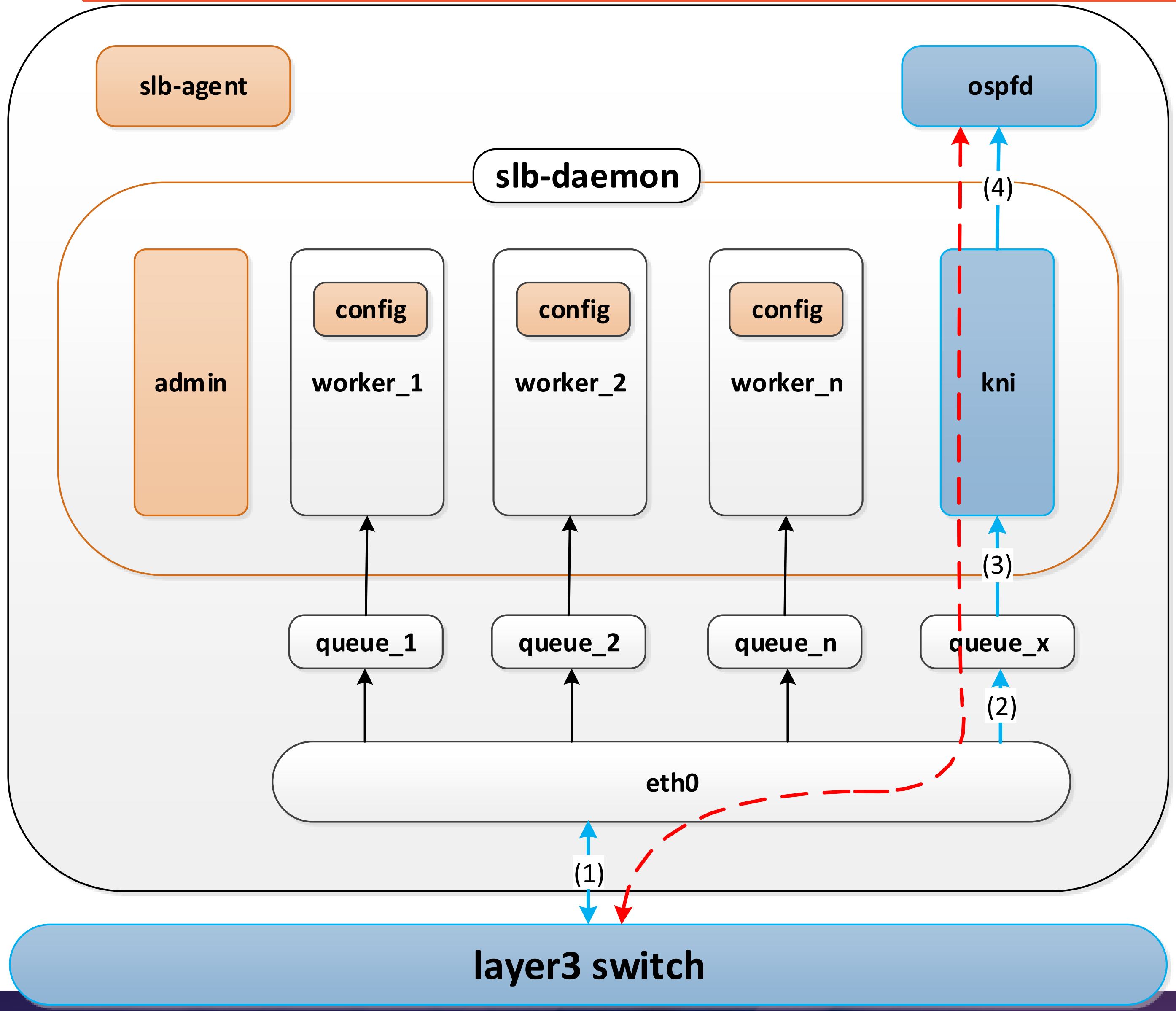


- **SLB-Daemon: core process which does load balance and full NAT**
- **SLB-Agent monitors and configures SLB-Daemon**
- **OSPFD publishes service subnets to layer3 switch**

- **Admin core configures VS and RS info to worker cores**
- **KNI core forwards OSPF packets to kernel, the kernel then sends them to OSPFD**
- **Worker cores do the load balance**

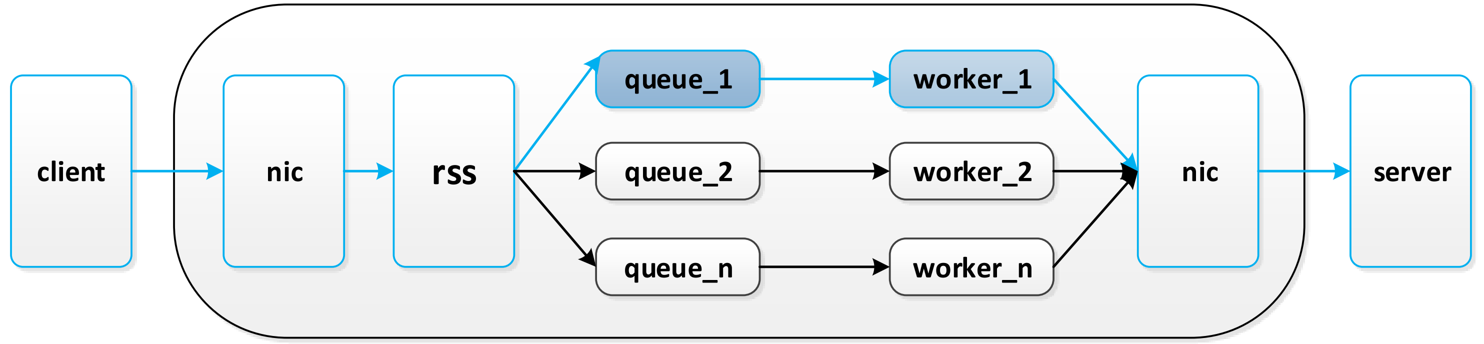
All data (config data, session data, local addrs) is per CPU, fully parallelizing packets processing

DPDK-SLB: OSPF Neighbor



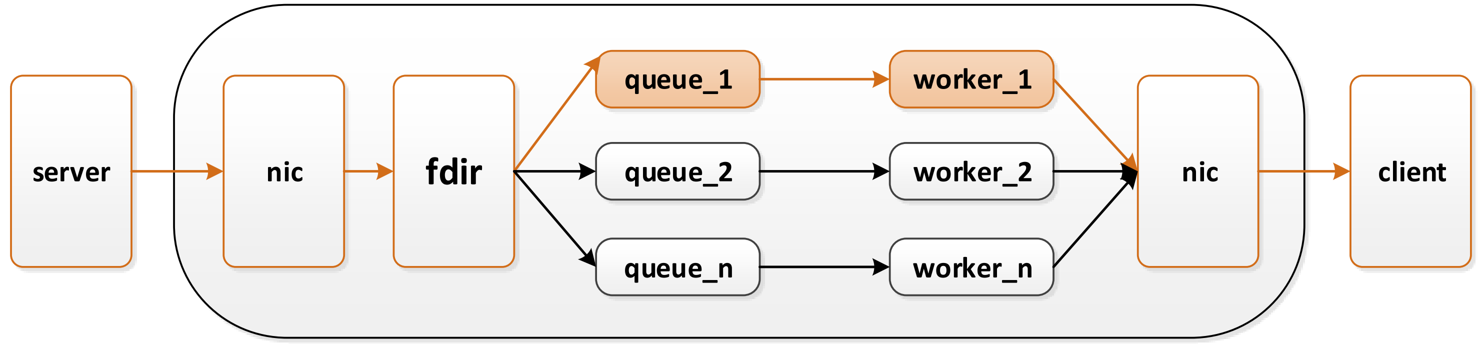
- OSPF uses multicast address 224.0.0.5
- Flow Director: destination ip 224.0.0.5 bound to queue_x
- Dedicated KNI core to process OSPF packets
- OSPFD publishes service subnets to layer3 switch

DPDK-SLB: Data Plane



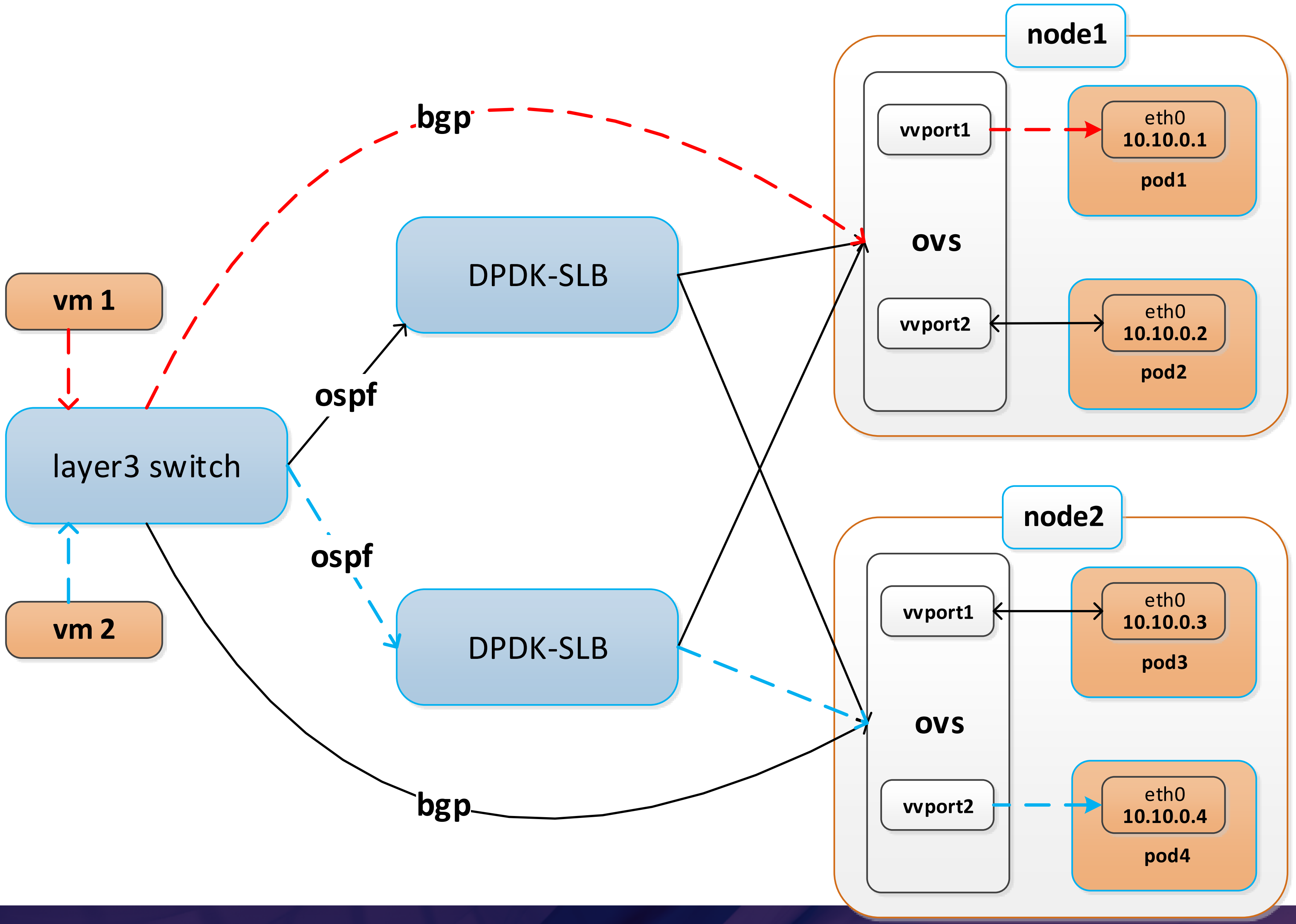
1、{client_ip, client_port, vip,vport}
2、rss selects a queue according to 5 tuple
3、worker_1 does fullnat {local_ip1, local_port, server_ip, server_port}
4、worker_1 saves session {cip,cport,vip,vport,lip1,lport,sip,sport}

the key point is that server-to-client packet must be placed on queue1, because only worker_1 has the session



1、{server_ip, server_port, local_ip1, local_port}
2、fdir selects a queue according to destination ip addr(local_ip1 bound to queue_1)
3、worker_1 lookups session {cip,cport,vip,vport,lip1,lport,sip,sport}
4、worker_1 does fullnat {vip, vport, client_ip, client_port}

Make Apps Run in the Container Cloud Seamlessly



- layer3 switch routes:
  10.10.0.1 nexthop node1
  10.10.0.4 nexthop node2
  service subnets nexthop dpdk-slb
- Pod IP can be reachable from vm1 outside k8s cluster
- Service IP can be reachable from vm2 outside k8s cluster
- Help apps to run in the container cloud and traditional environment at the same time

“

Thank You!

Q & A

”

zhouzijiang@jd.com
Jingdong Digital Technology