在线游戏场景下的kubernetes网络实践与优化

腾讯互娱 周威

2017-12-16

目录

TenC容器平台

专注容器计算资源服务

深度定制，持续优化的轻量化类vm式docker容器主机，助力成本优化。为在线业务提供资源极速调度，在线升降配，镜像扩容（故障切换）等服务。

基于kubernetes，docker的容器原生集群服务，为微服务化架构业务提供一站式管理方案，支持弹性伸缩、服务编排、镜像发布、服务发现、负载均衡、故障恢复、日志管理及性能监控等功能。

充分发挥docker容器轻量，敏捷等特点实现资源的快速调用，结合IEG资源管理规划，以docker镜像为统一交付件，为离线计算场景提供快速的计算资源弹性调度。

Part 1

网络架构

Docker网络方案

HOST模式

共用主机的网络，它的网络命名空间和主机是同一个，使用宿主机Namespace、IP和端口。

Bridge模式

类似于物理交换机的功能，在系统上默认创建一个Linux网桥，并为其分配一个子网，对每一个容器创建一个虚拟的以太网设备(veth peer)。其中一端关联到网桥上，另一端映射到容器类的网络空间中。然后从这个虚拟网段中分配一个IP地址给这个接口。

Container模式

加入另一个容器的网络空间，共享网络环境

None模式

只有loopback设备

跨主机通信方案

NAT方式

利用宿主机的IP和Iptables来达到容器之间的通信。容器对外IP都是宿主机的IP，NAT的性能损耗比较大。

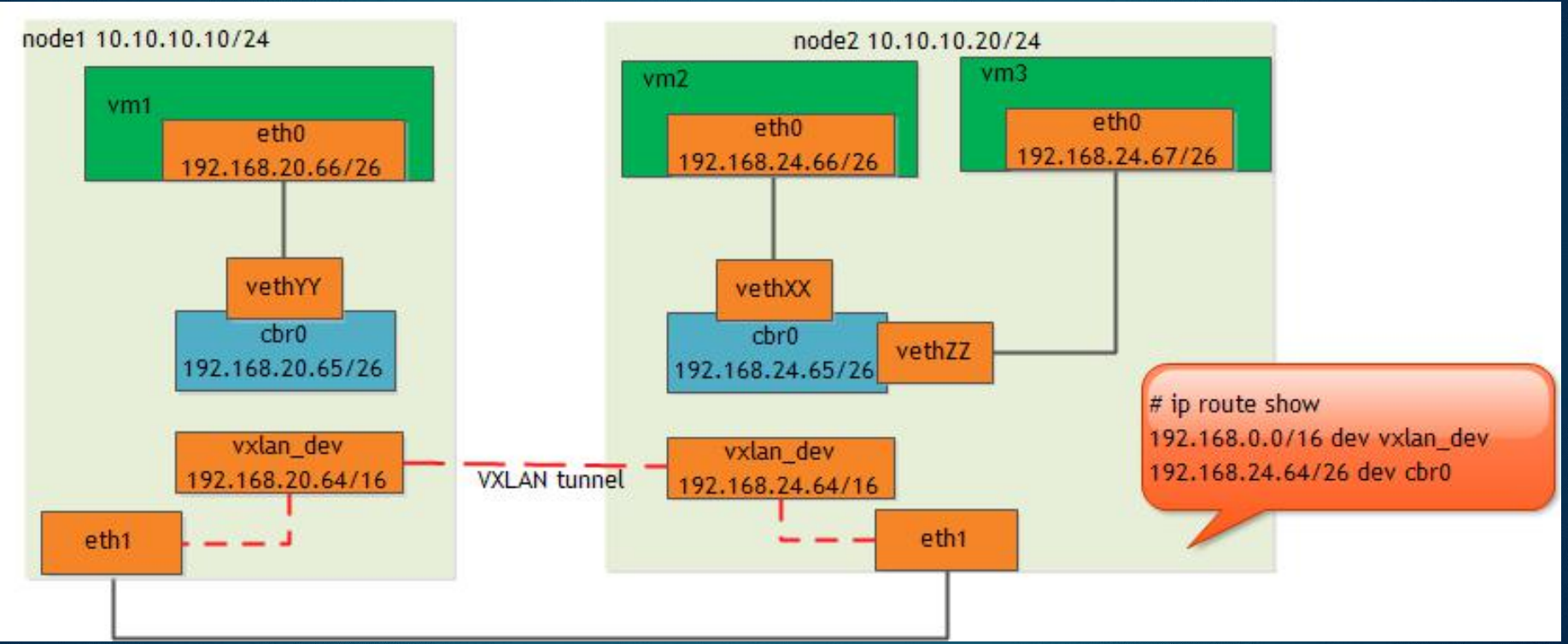
Overlay(tunnel)方式

VPN，ipip，VXLAN等都是overlay(tunnel)技术，简单讲就是在容器的数据包间封装一层或多层其他的数据协议头，达到连通的效果。

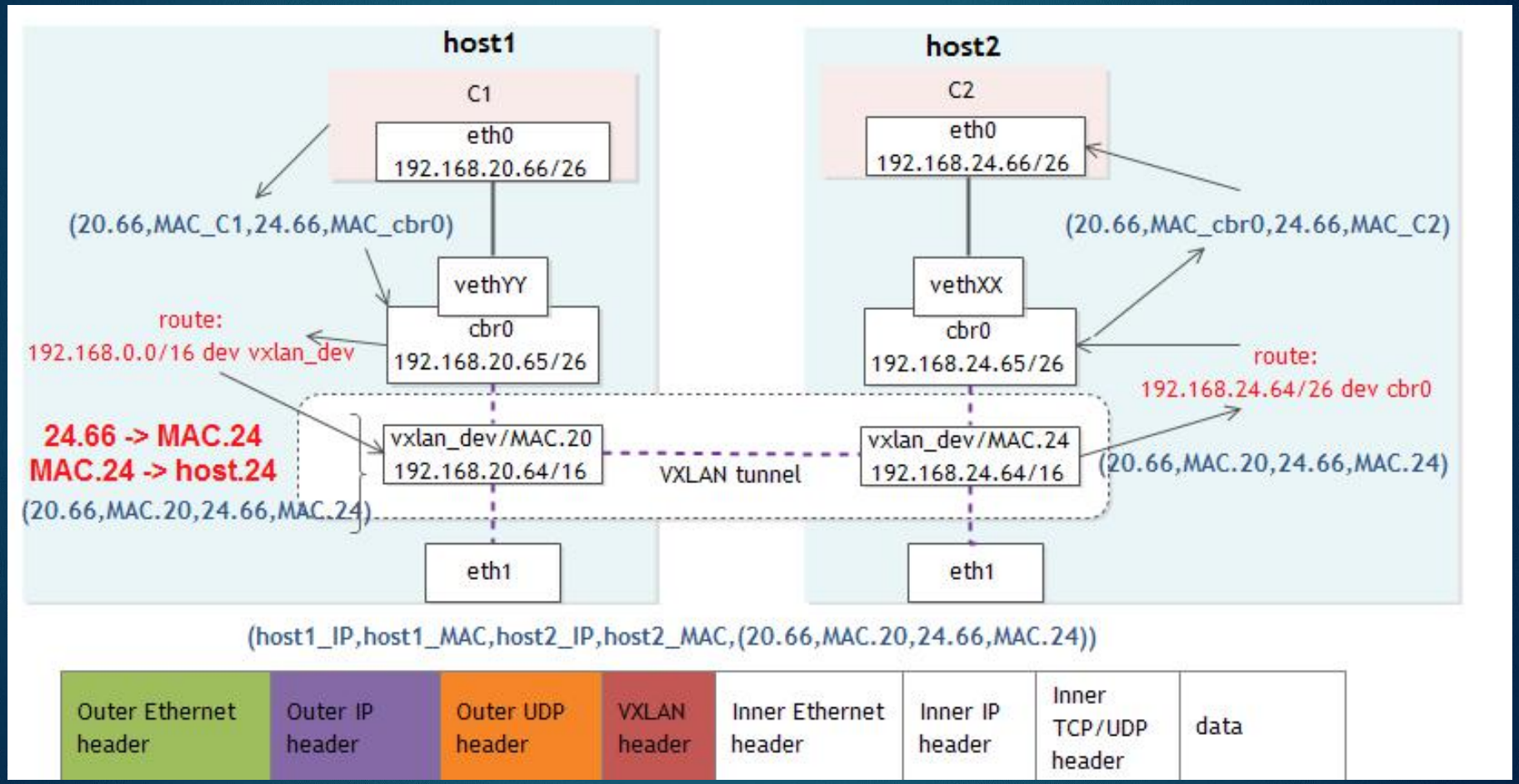
路由方式

通过路由设置的方式来互相通信，比如BGP路由方案，Macvlan。这种方式一般适用于单个数据中心，需要底层网络支持

Overlay Network



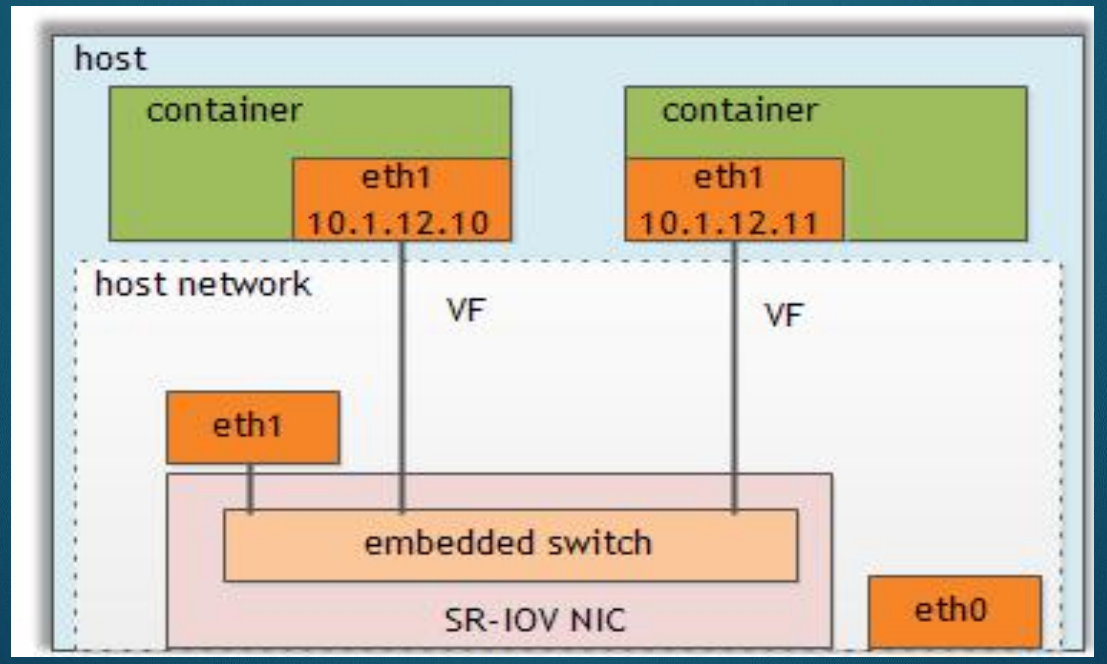
Overlay Network — 数据流



SRIOV

SR-IOV
CNI or CNM
CNI is Network Specification for Container Proposed by CoreOS

SR-IOV CNI plugin

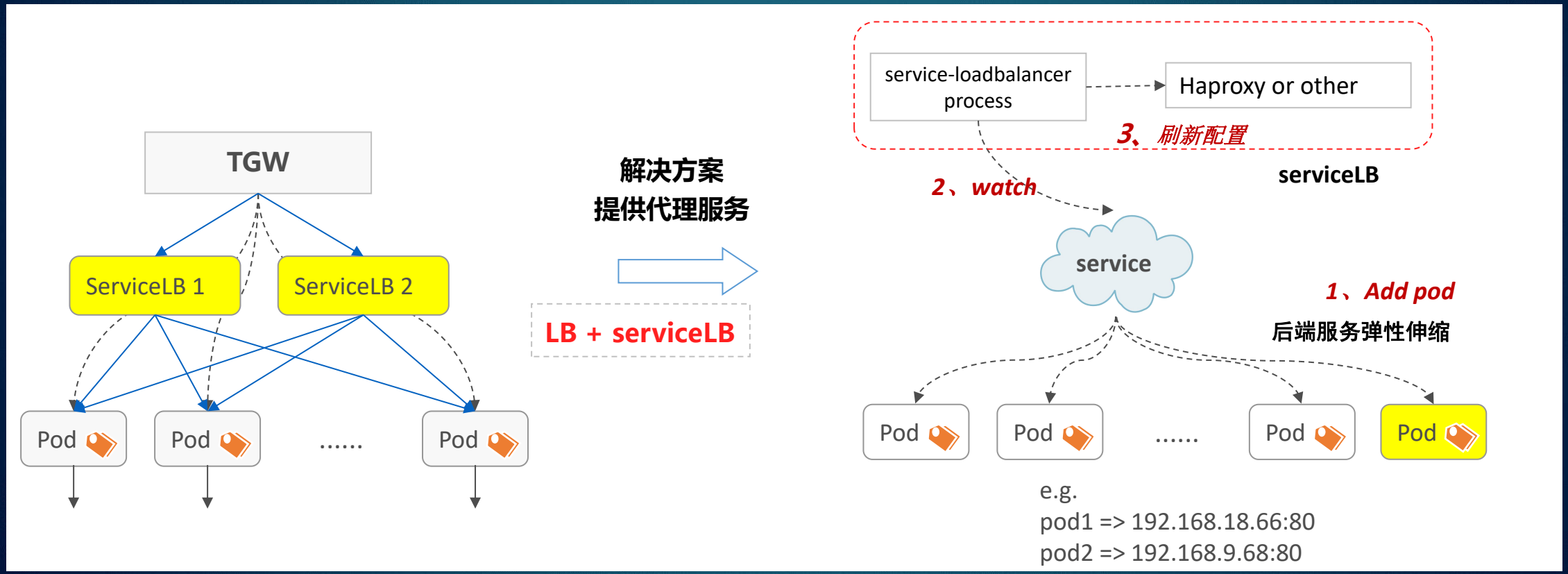


Part 2

几个问题与解决方案

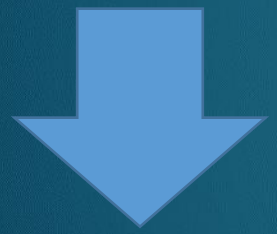
业务对外暴露服务

1、与tgw一定程度上的解耦；
2、代理层自动感知后端服务的伸缩



iptables性能堪忧

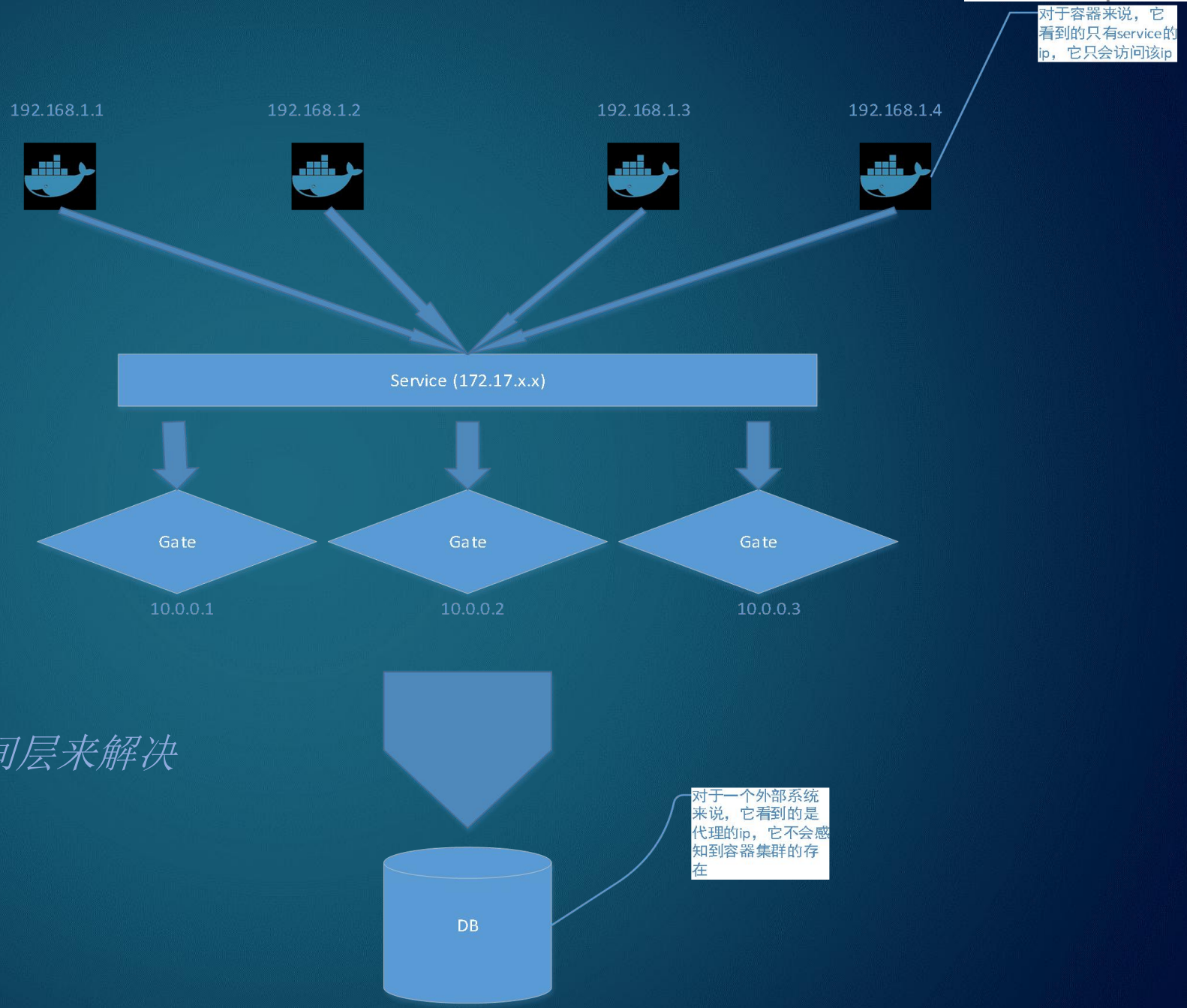
1. 游戏模块多
2. 微信，QQ，安卓，IOS，电信，网通 ….
3. 集群内多个业务
4. 每个service平均需要8条iptables规则
5. 10000+ iptables 规则，增加一条规则需要时间> 5min



1. 拆多个集群 ----- 管理成本高 ×
2. Headless service ----- DNS轮询不可控 ×
3. IPVS ------- 基于hash，增加一条2ms √

访问外部模块权限扩大

如果容器直接通过nat方式访问外部系统，由于容器调度的缘故，需要将所有母机加入白名单



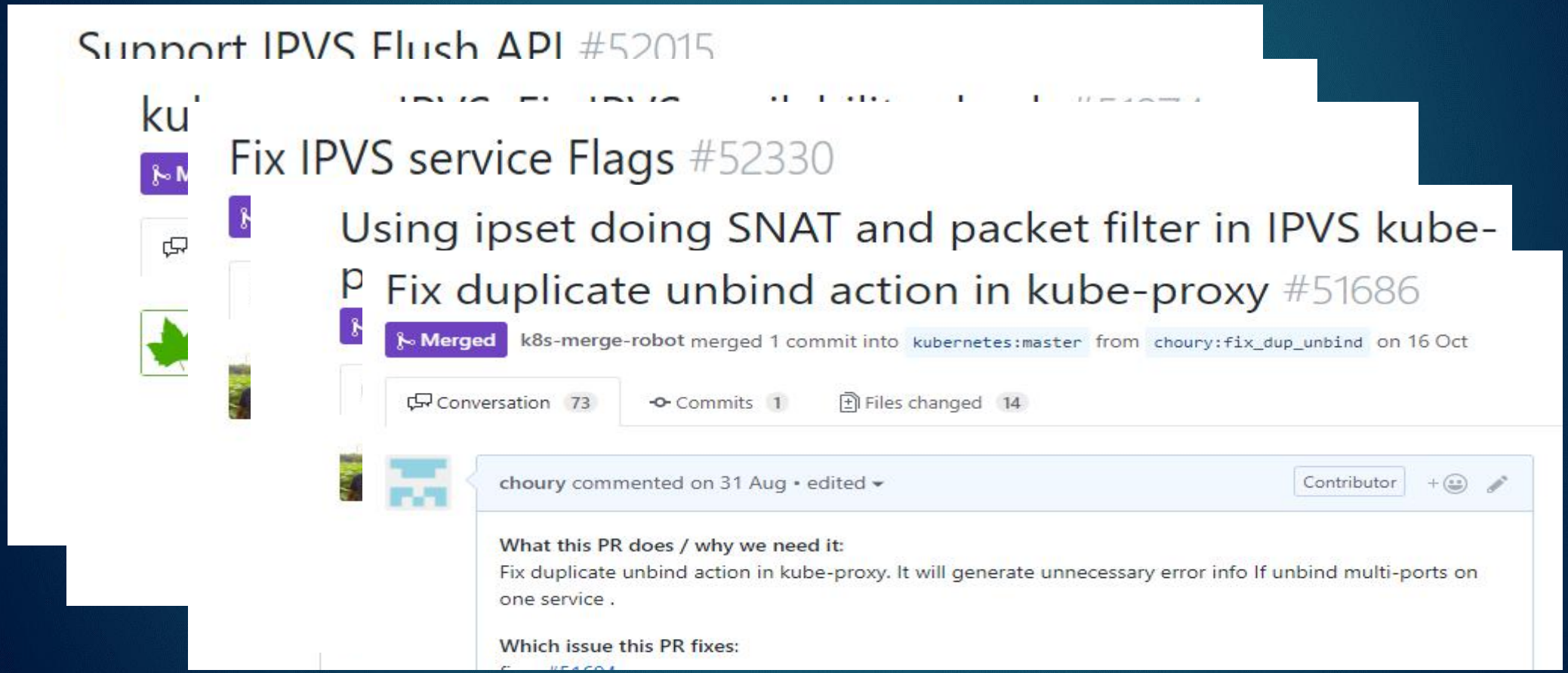
软件问题都可以通过增加一个中间层来解决

Part 3

踩的一些坑

K8s 1.8版本 ipvs功能尚不成熟

为了引入IPVS支持，我们使用k8s 1.8 。但是1.8版本的IPVS属于alpha功能，backport一系列patch，也有自己打的patch（已merge到1.9）。1.9版本IPVS已经成为beta特性，推荐使用1.9。



IPVS的跨namespace转发问题

```
1574 /*
1575  *  Check if it's for virtual services, look it up,
1576  *  and send it on its way...
1577  */
1578 static unsigned int
1579 ip_vs_in(
1580 {
1581     struct
1582     struct
1583     struct
1584     struct
1585     struct
1586     int re
1587     struct
1588
1589     /* Al
1590     if (sk
1591         re
1592
1593     /*
1594      *  Bi
1595      *
```

安全 | https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=2b5ec1a5f9738ee7bf8f5ec0526e75e00362c48f

about summary refs log tree commit diff stats

author	Ye Yin <hustcat@gmail.com>	2017-10-26 16:57:05 +0800
committer	David S. Miller <davem@davemloft.net>	2017-11-04 22:37:42 +0900
commit	2b5ec1a5f9738ee7bf8f5ec0526e75e00362c48f (patch)	
tree	ad668d857492b840266537b4d64f7b4e96ee97a8	
parent	24de79e5008a928beb2c7ccc2396f15065613363 (diff)	
download	linux-2b5ec1a5f9738ee7bf8f5ec0526e75e00362c48f.tar.gz	

netfilter/ipvs: clear ipvs_property flag when SKB net namespace changed

When run ipvs in two different network namespace at the same host, and one
ipvs transport network traffic to the other network namespace ipvs.
'ipvs_property' flag will make the second ipvs take no effect. So we should
clear 'ipvs_property' when SKB network namespace changed.

Fixes: 621e84d6f373 ("dev: introduce skb_scrub_packet()")
Signed-off-by: Ye Yin <hustcat@gmail.com>
Signed-off-by: Wei Zhou <chouryzhou@gmail.com>
Signed-off-by: Julian Anastasov <ja@ssi.bg>
Signed-off-by: Simon Horman <horms@verge.net.au>
Signed-off-by: David S. Miller <davem@davemloft.net>

未预留网卡中断CPU导致丢包问题

```
[root@TENCENT64 ~]#ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 0A:58:C0:A8:17:8F
          inet addr:192.168.23.143  Bcast:0.0.0.0  Mask:255.255.255.192
          UP BROADCAST RUNNING
          RX packets:871160468
          TX packets:776195784
          collisions:0 txqueuel
          RX bytes:421714001994
```

```
Cpu(s): 22.0%us,  1.8%sy,  0.0%ni, 72.2%id,  0.0%wa,  0.0%hi,  4.1%si,  0.0%st
Mem:  131655204k total, 36042220k used, 95612984k free,   330604k buffers
Swap:  2104508k total,        8k used,  2104500k free, 12156792k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
    3 root      20   0     0    0    0 R 99.2  0.0  23:26.09 ksoftirqd/0
149859 root     20   0 1332m 189m  12m R 92.9  0.1  14:41.45 node
148997 root     20   0 1334m 191m  12m R 92.6  0.1  14:51.15 node
149571 root     20   0 1328m 186m  12m R 91.6  0.1  15:23.09 node
149233 root     20   0 1336m 193m  12m R 86.6  0.2  14:33.32 node
150506 root     20   0 1341m 198m  12m R 86.3  0.2  12:13.58 node
150056 root     20   0 1340m 196m  12m R 85.0  0.2  14:33.25 node
149165 root     20   0 1329m 184m  12m S 81.0  0.1  14:04.86 node
150995 root     20   0 1340m 195m  12m R 72.7  0.2  10:26.06 node
149103 root     20   0 1335m 190m  12m R 71.7  0.1  14:22.60 node
151060 root     20   0 1344m 199m  12m R 69.1  0.2  10:29.41 node
```

网卡中断默认是使用CPU0，容器也使用了CPU0的话处理网卡中断不及时，导致丢包
开启了SRIOV的网卡PF有4个队列，我们按照NUMA结构将前4个core预留出来，并绑定网卡中断，这样就解决了丢包问题

The End !

谢 谢!