ovn-k8s的egress-service功能介绍

Egress-service 功能简介

Egress Service 功能使得支持负载均衡器服务的 Pod 的出口流量可以使用其入口 IP 离开集群。这对于通过负载均衡器服务与运行在 Kubernetes 集群上的应用程序通信并期望来自支持该服务的 Pod 的出口流量源 IP 与它们到达目标时使用的目标 IP 相同（即负载均衡器的入口 IP）的外部系统非常有用。

通过引入新的CRD EgressService，用户可以请求将负载均衡器服务终端点中所有Pod发出的出口数据包源IP设置为其入口IP。该CRD仅适用于命名空间范围内。EgressService的名称对应于应受此功能影响的LoadBalancer Service的名称。请注意，EgressService到Kubernetes Service的映射是1对1。该功能将由“Shared”和“Local”网关模式支持，并且受影响流量将来自集群外部目标 - 这意味着不会影响Pod之间、Pod与 svc 之间或者Pod与 Node 之间的流量。

将服务对外公开（ingress traffic）由负载均衡器提供商（如MetalLB）处理，而不是由后面的OVN-Kubernetes处理。

网络详细介绍

仅将 Pod 的 IP SNAT 到其所支持的 LoadBalancer 服务入口 IP 是有问题的，因为通常负载均衡器提供商会通过多个节点公开入口 IP。这意味着我们不能只是在 Pod 离开其节点之前将 SNAT 添加到常规流量中，因为我们无法保证回复会返回到 Pod 所在的节点（即流量起源）。外部客户端通常有多条路径可以到达负载均衡器入口 IP，并且可能回复到不是 Pod 所在节点的节点 - 在这种情况下，其他节点没有正确的 CONNTRACK 条目来将回复发送回 Pod，从而导致流量丢失。出于这个原因，我们需要确保所有服务 Pods 的流量（进/出）都由单个节点处理，以便始终匹配正确的 CONNTRACK 条目并避免流量丢失。

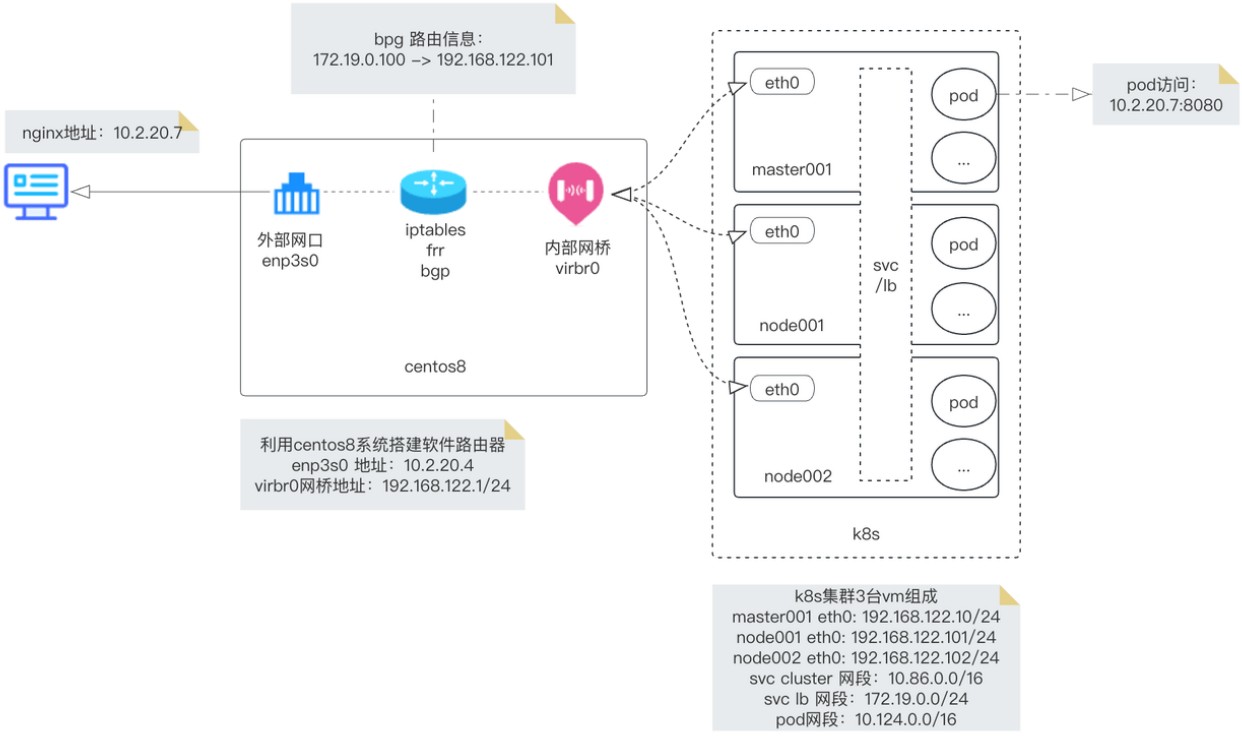
出口部分由 OVN-Kubernetes 处理，它选择一个作为入口/出口点的节点，并通过在 ovn_cluster_router 上使用逻辑路由器策略将相关 pod 的出口流量引导到其管理端口。当该流量到达节点的管理端口时，它将在传出之前使用其路由表和 iptables 进行处理。因此，它负责在所选节点上添加必要的 iptables 规则，以使从这些 pod 发送到服务入口 IP 的流量进行 SNAT。

通过引入新的资源EgressService，用户可以创建与LoadBalancer服务并行的资源，这些资源可以是空的或包含可选字段，从而实现了这些目标：

- nodeSelector: 允许限制可以选择处理服务流量的节点。当存在时，只有其标签与指定选择器匹配的节点可以被选择用于处理服务的流量，如前所述。如果未指定字段，则可以选择集群中的任何节点来管理服务的流量。此外，如果服务的ExternalTrafficPolicy设置为Local，则会添加一个附加约束，即只能选择具有终结点的节点。
- network: 配置出口和入口网络。这通常实现为VRF（Virtual Routing and Forwarding）虚拟路由映射，表示路由表的数字id或字符串名称（参考 linux vrf 实现方法），通过省略使用默认主机路由。

测试网络架构

整体网络图：



开启 egress-service

依赖：ovn-k8s 社区最新代码才支持该功能，对应镜像地址：harbor.yusur.tech/yusur_ovn/ovn-daemonset-f:new-latest

```
--egress-service-enable=true
```

```
OVN_IMAGE=harbor.yusur.tech/yusur_ovn/ovn-daemonset-f:new-latest
./daemonset.sh --image=$OVN_IMAGE --net-cidr=10.124.0.0/16 \
--svc-cidr=10.86.0.0/16 \
--gateway-mode="local" \
--gateway-options="--gateway-interface=ens3f0 --gateway-
nexthop=192.168.122.1" \
--k8s-apiserver=https://192.168.122.10:6443 \
--multicast-enabled=false \
--disable-snat-multiple-gws=true \
--disable-pkt-mtu-check=true \
--egress-service-enable=true # 开启egress-service 功能

# 安装yaml
kubectl apply -f ../yaml/ovn-setup.yaml
kubectl apply -f ../yaml/k8s.ovn.org_adminpolicybasedexternalroutes.yaml
kubectl apply -f ../yaml/k8s.ovn.org_egressservices.yaml
kubectl apply -f ../yaml/ovnkube-db.yaml
kubectl apply -f ../yaml/ovnkube-master.yaml
kubectl apply -f ../yaml/ovnkube-node.yaml


```

LB 服务安装

k8s lb服务metallb(bgp模式)介绍

06-08 14:02 更新

1， 查看 lb 地址池

```
kubectl  get  ipaddresspools -n metallb-system   example-pool
NAME           AUTO ASSIGN   AVOID BUGGY IPS   ADDRESSES
example-pool   false         false             ["172.19.0.0/24"]
```

2， 查看 bgp 宣告配置

```
kubectl  get bgpadvertisements -n metallb-system   example-bgp-adv
NAME              IPADDRESSPOOLS     IPADDRESSPOOL SELECTORS   PEERS
example-bgp-adv   ["example-pool"]
```

3，查看 bgp peers 配置

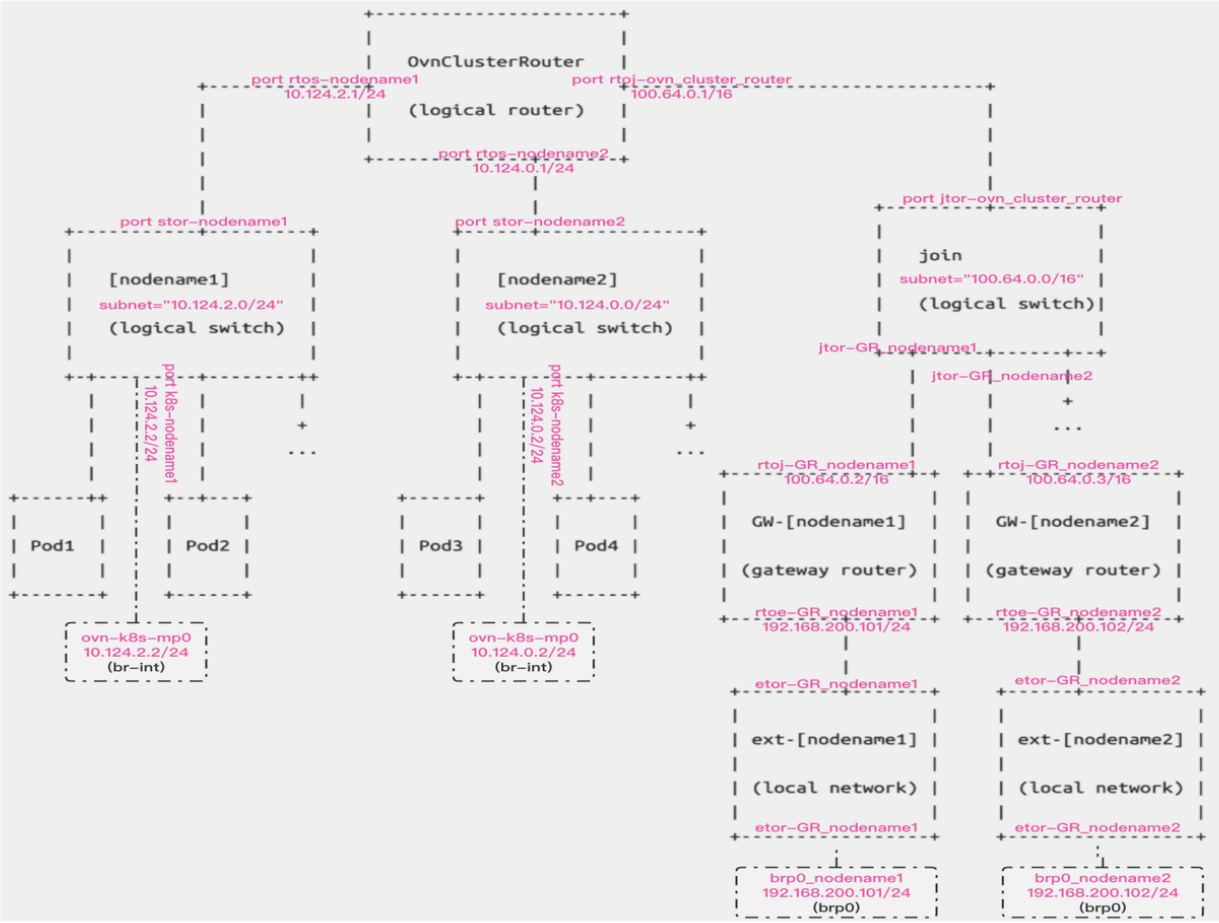
```
kubectl  get bgppeers -n metallb-system   example -o yaml
apiVersion: metallb.io/v1beta2
kind: BGPPeer
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"metallb.io/v1beta2","kind":"BGPPeer","metadata":{"annotations":{},"name":"example","namespace":"metallb-system"},"spec":{"myASN":64512,"peerASN":65001,"peerAddress":"192.168.122.1","peerPort":179}}
  creationTimestamp: "2023-06-02T06:08:49Z"
  generation: 1
  name: example
  namespace: metallb-system
  resourceVersion: "13636"
  uid: 3b952149-ed3b-4872-93d9-3e109f659ec1
spec:
  myASN: 64512
  peerASN: 65001
  peerAddress: 192.168.122.1 #宣告对端地址
  peerPort: 179
```

配置 BGP 路由器

LB服务和BGP路由器测试
06-09 10:54 更新

创建 LB 类型 SVC（未设置 egress-service）

配置 lb 类型 svc 网络图：



1， 创建 svc

```yaml
cat <<EOF > svc.yaml
apiVersion: v1
kind: Service
metadata:
  name: example-service
  annotations:
    # 指定lb的地址池
    metallb.universe.tf/address-pool: example-pool
spec:
  selector:
    app: nginx1-ovn
  ports:
    - name: http
      protocol: TCP
      port: 80
      targetPort: 80
```

```
  type: LoadBalancer

EOF

kubectl apply -f svc.yaml
# 查看svc 的lb 信息
kubectl  describe svc example-service
Name:                     example-service
Namespace:                default
Labels:                   <none>
Annotations:              metallb.universe.tf/address-pool: example-pool
                          metallb.universe.tf/ip-allocated-from-pool:
example-pool
Selector:                 app=nginx1-ovn
Type:                     LoadBalancer
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       10.86.73.55
IPs:                      10.86.73.55
LoadBalancer Ingress:     172.19.0.100 # 分配的lb地址
Port:                     http  80/TCP
TargetPort:               80/TCP
NodePort:                 http  32541/TCP
Endpoints:                10.124.0.3:80
Session Affinity:         None
External Traffic Policy:  Cluster
```

2，查看集群 ovn 信息

ndb

```
# 查看 ovn_cluster_router 路由器规则
ovn-nbctl lr-route-list ovn_cluster_router
IPv4 Routes
Route Table <main>:
               100.64.0.2               100.64.0.2 dst-ip
               100.64.0.3               100.64.0.3 dst-ip
               100.64.0.4               100.64.0.4 dst-ip
            10.124.0.0/24               10.124.0.2 src-ip
            10.124.1.0/24               10.124.1.2 src-ip
            10.124.2.0/24               10.124.2.2 src-ip

# 查看 ovn_cluster_router 路由器的策略信息
ovn-nbctl lr-policy-list ovn_cluster_router
```

```
Routing Policies
      1004 inport == "rtos-master001" && ip4.dst == 192.168.122.10 /* master001 */         reroute                10.124.1.2
      1004 inport == "rtos-work001" && ip4.dst == 192.168.122.101 /* work001 */         reroute                10.124.0.2
      1004 inport == "rtos-work002" && ip4.dst == 192.168.122.102 /* work002 */         reroute                10.124.2.2
       102 (ip4.src == $a4548040316634674295 || ip4.src == $a13607449821398607916) && ip4.dst == $a14918748166599097711 allow               pkt_mark=1008
       102 ip4.src == 10.124.0.0/16 && ip4.dst == 10.124.0.0/16 allow
       102 ip4.src == 10.124.0.0/16 && ip4.dst == 100.64.0.0/16 allow


# 查看lb 信息
[root@master001 ~]# ovn-nbctl find Load_Balancer name=Service_default/example-service_TCP_cluster
_uuid               : a58212a2-bb2a-44b2-aa79-41a3647c4b75
external_ids        : {"k8s.ovn.org/kind"=Service, "k8s.ovn.org/owner"="default/example-service"}
health_check        : []
ip_port_mappings    : {}
name                : "Service_default/example-service_TCP_cluster"
options             : {event="false", hairpin_snat_ip="169.254.169.5 fd69::5", neighbor_responder=none, reject="true", skip_snat="false"}
protocol            : tcp
selection_fields    : []
vips                : {"10.86.210.196:80"="10.124.0.3:80", "172.19.0.0:80"="10.124.0.3:80"}

# 查看ovs 流表规则
root@work002:~# ovs-ofctl  dump-flows  br-int | grep 10.86.210.196
 cookie=0xa7f4dcc4, duration=1874.054s, table=15, n_packets=8, n_bytes=592, idle_age=1426, priority=120,ct_state=+new-rel+trk,tcp,metadata=0xa,nw_dst=10.86.210.196,tp_dst=80 actions=load:0x1->NXM_NX_REG10[3],group:9

# 查看对应group 信息
root@work002:~# ovs-ofctl  dump-groups  br-int | grep group_id=9
group_id=9,type=select,selection_method=dp_hash,bucket=bucket_id:0,weight:100,actions=ct(commit,table=16,zone=NXM_NX_REG11[0..15],nat(dst=10.124.0.3:80),exec(load:0x1->NXM_NX_CT_MARK[1],load:0x1->NXM_NX_CT_MARK[3]))
```

3，查看节点 iptables 信息

iptables

```
# 应用OVN-KUBE-EXTERNALIP 的链为PREROUTING, OUTPUT, # 172.19.0.0 lb的ip地址
root@work002:~# iptables -vn -t nat -L  OVN-KUBE-EXTERNALIP
Chain OVN-KUBE-EXTERNALIP (2 references)
 pkts bytes target     prot opt in     out     source               destination
    7   420 DNAT       tcp  --  *      *       0.0.0.0/0            172.19.0.0           tcp dpt:80 to:10.86.210.196:80


# 查看对应生成的nodeport 规则
root@work002:~# iptables -vn -t nat -L  OVN-KUBE-NODEPORT
Chain OVN-KUBE-NODEPORT (2 references)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DNAT       tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            ADDRTYPE match dst-type LOCAL tcp dpt:31360 to:10.86.210.196:80
    0     0 DNAT       tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            ADDRTYPE match dst-type LOCAL tcp dpt:32108 to:10.86.183.58:80
```

4，查看节点 route 信息

```
root@work001:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.122.1   0.0.0.0         UG    0      0        0 brens3f0
10.86.0.0       169.254.169.4   255.255.0.0     UG    0      0        0 brens3f0 #svc 网段流量导入ovs 的brens3f0桥，在ovs 里面做svc的 dnat到pod的 转发
10.124.0.0      0.0.0.0         255.255.255.0   U     0      0        0 ovn-k8s-mp0
10.124.0.0      10.124.0.1      255.255.0.0     UG    0      0        0 ovn-k8s-mp0
169.254.169.0   0.0.0.0         255.255.255.248 U     0      0        0 brens3f0
169.254.169.1   0.0.0.0         255.255.255.255 UH    0      0        0 brens3f0
```

```
169.254.169.3   10.124.0.1      255.255.255.255 UGH   0      0        0 ovn-k8s-mp0
192.168.100.0   0.0.0.0         255.255.255.0   U     0      0        0 ens3f1
192.168.122.0   0.0.0.0         255.255.255.0   U     0      0        0 brens3f0
192.168.200.0   0.0.0.0         255.255.255.0   U     0      0        0 ens3f2
```

流量测试

1，访问 web 服务

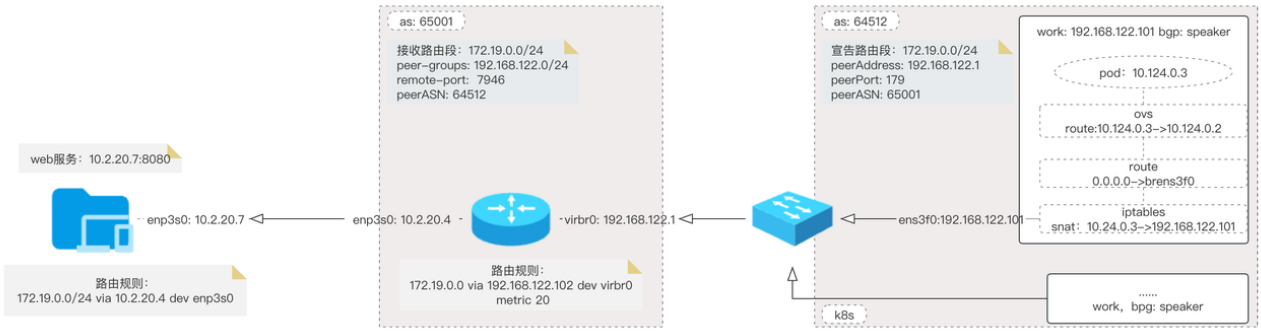
登入pod端curl 10.2.20.7:8080

```
[root@yusur-55 ~]# kubectl  exec -it nginx1-ovn -- curl http://10.2.20.7:8080/login/
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="shortcut icon" href="/static/img/favicon.ico">
    <title>Sign In</title>
    <link rel="stylesheet" href="/static/css/bootstrap.min.css">
    <link href="/static/css/webvirtmgr.css" rel="stylesheet">
    <link href="/static/css/signin.css" rel="stylesheet">
</head>

```

2，转发流程图

gw 为 local模式转发图：



3， 节点上抓包

ovn-k8s-mp0 122.101上

```
root@work001:~# tcpdump  port 8080 -nnvv -i ovn-k8s-mp0
tcpdump: listening on ovn-k8s-mp0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
06:28:39.096725 IP (tos 0x0, ttl 63, id 18384, offset 0, flags [DF], proto TCP (6), length 60)
    10.124.0.3.41354 > 10.2.20.7.8080: Flags [S], cksum 0x7d65 (correct), seq 162597100, win 65280, options [mss 1360,sackOK,TS val 599960572 ecr 0,nop,wscale 7], length 0
```

4， 集群外抓包

路由器122.1上

```
[root@yusur-55 ~]# tcpdump -i virbr0 port 8080 -nnvv
dropped privs to tcpdump
tcpdump: listening on virbr0, link-type EN10MB (Ethernet), capture size 262144 bytes
10:55:53.203505 IP (tos 0x0, ttl 62, id 31089, offset 0, flags [DF], proto TCP (6), length 60)
    192.168.122.101.58332 > 10.2.20.7.8080: Flags [S], cksum 0x0de1 (correct), seq 1315082024, win 65280, options [mss 1360,sackOK,TS val 587193972 ecr 0,nop,wscale 7], length 0

# src ip 更改为所在宿主机ip（192.168.122.101）
```

5，查看 nat 信息

ovn和iptables

```
# 查看ovn，nat 信息
[root@master001 ~]# ovn-nbctl lr-nat-list GR_work001
TYPE             GATEWAY_PORT          EXTERNAL_IP        EXTERNAL_PORT    LOGICAL_IP          EXTERNAL_MAC         LOGICAL_PORT
snat                                   192.168.122.101                     10.124.0.4
snat                                   192.168.122.101                     10.124.0.3

# 查看iptables，nat 信息
root@work001:~# iptables -nv -t nat  -L POSTROUTING
Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)

```

```
 pkts bytes target     prot opt in     out     source               destination
90538 6039K OVN-KUBE-EGRESS-SVC  all  --  *      *       0.0.0.0/0            0.0.0.0/0
16490  989K OVN-KUBE-SNAT-MGMTPORT  all  --  *      ovn-k8s-mp0  0.0.0.0/0          0.0.0.0/0
74192 5059K KUBE-POSTROUTING  all  --  *      *       0.0.0.0/0            0.0.0.0/0            /* kubernetes postrouting rules */
    0     0 MASQUERADE  all  --  *      *       169.254.169.1        0.0.0.0/0
   28  2016 MASQUERADE  all  --  *      *       10.124.0.0/24        0.0.0.0/0
```

6，ovs 中流表信息

Ova

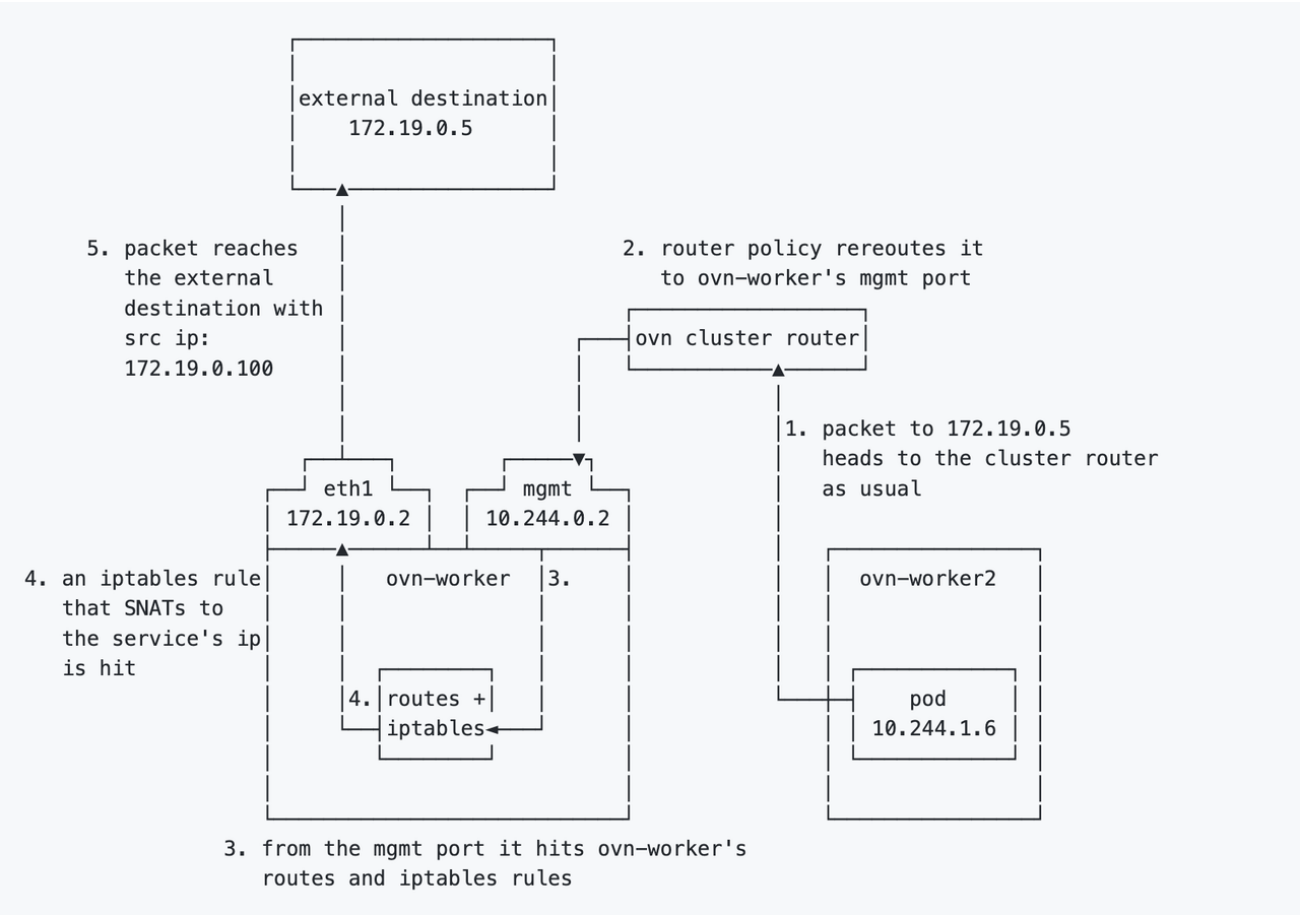
```
# work001上，pod所在的宿主机：
# #分析：src=10.124.0.3,dst=10.0.0.0/255.192.0.0 转到 ovn-k8s-mp0 口
ufid:c1ecc9ce-7d5d-467a-b71c-1b3561f8a872, recirc_id(0),dp_hash(0/0),skb_priority(0/0),in_port(cfce2e228d32569),skb_mark(0/0),ct_state(0/0),ct_zone(0/0),ct_mark(0/0),ct_label(0/0),eth(src=0a:58:0a:7c:00:03,dst=0a:58:0a:7c:00:01),eth_type(0x0800),ipv4(src=10.124.0.3,dst=10.0.0.0/255.192.0.0,proto=6,tos=0/0,ttl=0/0,frag=no),tcp(src=0/0,dst=0/0),tcp_flags(0/0), packets:8, bytes:612, used:2.840s, flags:FP., dp:ovs, actions:ct(zone=12,nat),recirc(0x1942)
ufid:d3c12d9a-6617-42ca-8482-17e56d604799, recirc_id(0x1943),dp_hash(0/0),skb_priority(0/0),in_port(cfce2e228d32569),skb_mark(0/0),ct_state(0x21/0x3f),ct_zone(0/0),ct_mark(0/0x1),ct_label(0/0),eth(src=0a:58:0a:7c:00:01,dst=00:00:00:00:00:00/01:00:00:00:00:00),eth_type(0x0800),ipv4(src=10.124.0.3,dst=0.0.0.0/0.0.0.0,proto=0/0,tos=0/0,ttl=0/0,frag=no), packets:0, bytes:0, used:never, dp:ovs, actions:ct(commit,zone=7,mark=0/0x1,nat(src)),ovn-k8s-mp0

# work002上，无包
ovs-appctl dpctl/dump-flows -m | grep 10.124.0.3

```

创建 LB 类型 SVC（设置 egress-service）

配置 egress-service 的网络图：



1，创建 egress-service

```
cat <<EOF > egress-service.yaml
apiVersion: k8s.ovn.org/v1
kind: EgressService
metadata:
  name: example-service
spec:
  nodeSelector:
    matchLabels:
      node-role.kubernetes.io/worker: ""
      kubernetes.io/hostname: work002
EOF
```

2，更新 bgp adv

```
cat <<EOF > adv.yaml
apiVersion: metallb.io/v1beta1
kind: BGPAdvertisement
metadata:
  name: example-bgp-adv
  namespace: metallb-system
spec:
  ipAddressPools:
  - example-pool
  nodeSelectors:
  - matchLabels:
      egress-service.k8s.ovn.org/default-example-service: ""
EOF

# 只选择具备egress-service.k8s.ovn.org/default-example-service: "" 标签的node节点进行bgp 路由宣告
kubectl apply -f adv.yaml
```

3，查看集群 ovn 信息

路由发生变化

```
[root@master001 ~]# ovn-nbctl lr-policy-list  ovn_cluster_router
Routing Policies
      1004 inport == "rtos-master001" && ip4.dst == 192.168.122.10 /* master001 */         reroute                10.124.1.2
      1004 inport == "rtos-work001" && ip4.dst == 192.168.122.101 /* work001 */         reroute                10.124.0.2
      1004 inport == "rtos-work002" && ip4.dst == 192.168.122.102 /* work002 */         reroute                10.124.2.2
       102 (ip4.src == $a4548040316634674295 || ip4.src == $a13607449821398607916) && ip4.dst == $a14918748166599097711 allow               pkt_mark=1008
       102 ip4.src == 10.124.0.0/16 && ip4.dst == 10.124.0.0/16 allow
       102 ip4.src == 10.124.0.0/16 && ip4.dst == 100.64.0.0/16 allow
       101                                ip4.src == 10.124.0.3 reroute               10.124.2.2

# 新增加一条路由策略ip4.src == 10.124.0.3   reroute   10.124.2.2
```

4，查看 iptables 信息

添加新规则OVN-KUBE-EGRESS-SVC, POSTROUTING链应用

```
root@work002:~# iptables -vn -t nat -L OVN-KUBE-EGRESS-SVC
Chain OVN-KUBE-EGRESS-SVC (1 references)
 pkts bytes target     prot opt in     out     source               destination
    0     0 RETURN     all  --  *      *       0.0.0.0/0            0.0.0.0/0            mark match 0x3f0 /* DoNotSNAT */
    0     0 SNAT       all  --  *      *       10.124.0.3           0.0.0.0/0            /* default/example-service */ to:172.19.0.0

# 新增iptables 链OVN-KUBE-EGRESS-SVC，并且egress-svc 选择的node上会增加 snat规则： 10.124.0.3 -> 172.19.0.0
```

流量测试

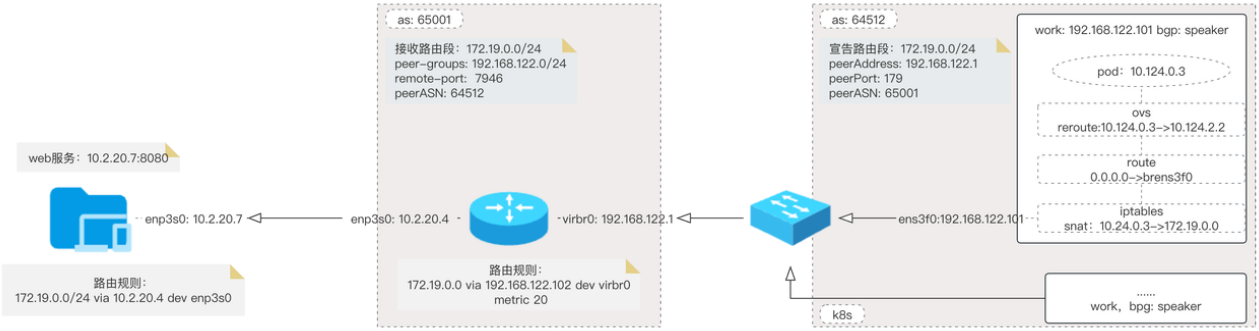
1，访问 web 服务

登入pod端curl 10.2.20.7:8080

```
[root@yusur-55 ~]# kubectl  exec -it nginx1-ovn -- curl http://10.2.20.7:8080/login/
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="shortcut icon" href="/static/img/favicon.ico">
    <title>Sign In</title>
    <link rel="stylesheet" href="/static/css/bootstrap.min.css">
    <link href="/static/css/webvirtmgr.css" rel="stylesheet">
    <link href="/static/css/signin.css" rel="stylesheet">
</head>

```

2，转发流程图



3，节点上抓包

ovn-k8s-mp0

```
# 在work001 上，无包
tcpdump port 8080 -nnvv -i ovn-k8s-mp0

# 在work002 上
tcpdump port 8080 -nnvv -i ovn-k8s-mp0
06:45:05.048729 IP (tos 0x0, ttl 63, id 55733, offset 0, flags [DF], proto
TCP (6), length 60)
    10.124.0.3.53528 > 10.2.20.7.8080: Flags [S], cksum 0xaf1d (correct),
seq 2274303372, win 65280, options [mss 1360,sackOK,TS val 600946728 ecr
0,nop,wscale 7], length 0

```

4，集群外抓包

路由器122.1

```
[root@yusur-55 ~]# tcpdump -i virbr0 port 8080 -nnvv
dropped privs to tcpdump
tcpdump: listening on virbr0, link-type EN10MB (Ethernet), capture size
262144 bytes
13:45:41.708587 IP (tos 0x0, ttl 62, id 23688, offset 0, flags [DF], proto
TCP (6), length 60)
    172.19.0.0.38168 > 10.2.20.7.8080: Flags [S], cksum 0x64c4 (correct),
seq 2136567277, win 65280, options [mss 1360,sackOK,TS val 597382392 ecr
0,nop,wscale 7], length 0

# src ip 更改为对应lb地址（172.19.0.0）
```

5，查看 nat 信息

iptables

```
root@work002:~# iptables -vn -t nat -L POSTROUTING
Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source
destination
92283 6152K OVN-KUBE-EGRESS-SVC  all  --  *      *       0.0.0.0/0
   0.0.0.0/0
16766 1006K OVN-KUBE-SNAT-MGMTPORT  all  --  *      ovn-k8s-mp0  0.0.0.0/0
           0.0.0.0/0
```

```
75653 5155K KUBE-POSTROUTING  all  --  *      *       0.0.0.0/0
0.0.0.0/0            /* kubernetes postrouting rules */
    0     0 MASQUERADE  all  --  *      *       169.254.169.1
0.0.0.0/0
  216 10580 MASQUERADE  all  --  *      *       10.124.2.0/24
0.0.0.0/0

# OVN-KUBE-EGRESS-SVC 在最前面，优先匹配
root@work002:~# iptables -vn -t nat -L OVN-KUBE-EGRESS-SVC
Chain OVN-KUBE-EGRESS-SVC (1 references)
 pkts bytes target     prot opt in     out     source
destination
    0     0 RETURN     all  --  *      *       0.0.0.0/0
0.0.0.0/0            mark match 0x3f0 /* DoNotSNAT */
    2   120 SNAT       all  --  *      *       10.124.0.3
0.0.0.0/0            /* default/example-service */ to:172.19.0.0
```

6，ovs 中流表信息

ovs

```
# work001上，pod所在的宿主机：
#分析：src=10.124.0.3,dst=10.0.0.0/255.192.0.0 转到 -> dst=192.168.122.102 work002(bgp speaker 宣告节点和 egress-service作用节点)
# ovs-appctl dpctl/dump-flows -m | grep 10.124.0.3
ufid:868fbc3a-1e98-4c6b-8b1d-94a9988533a8,
recirc_id(0x1906),dp_hash(0/0),skb_priority(0/0),in_port(cfce2e228d32569),skb_mark(0/0),ct_state(0x22/0x3f),ct_zone(0/0),ct_mark(0/0xf),ct_label(0/0),eth(src=0a:58:0a:7c:00:03,dst=0a:58:0a:7c:00:01),eth_type(0x0800),ipv4(src=10.124.0.3,dst=10.0.0.0/255.192.0.0,proto=6,tos=0/0x3,ttl=64,frag=no),tcp(src=0/0,dst=0/0),tcp_flags(0/0), packets:5, bytes:330, used:2.100s, flags:F., dp:ovs,
actions:ct_clear,set(tunnel(tun_id=0x1,dst=192.168.122.102,ttl=64,tp_dst=6081,geneve({class=0x102,type=0x80,len=4,0x20007}),flags(df|csum|key))),set(eth(src=0a:58:0a:7c:02:01,dst=9a:8c:6a:42:90:60)),set(ipv4(ttl=63)),genev_sys_6081
ufid:094209d8-3f78-479c-98ba-2bfb53c38de0,
recirc_id(0x1906),dp_hash(0/0),skb_priority(0/0),in_port(cfce2e228d32569),skb_mark(0/0),ct_state(0x21/0x3f),ct_zone(0/0),ct_mark(0/0xf),ct_label(0/0),eth(src=0a:58:0a:7c:00:03,dst=0a:58:0a:7c:00:01),eth_type(0x0800),ipv4(src=10.124.0.3,dst=10.0.0.0/255.192.0.0,proto=6,tos=0/0x3,ttl=64,frag=no),tcp(src=0/0,dst=0/0),tcp_flags(0/0), packets:0, bytes:0, used:never, dp:ovs,
actions:ct(commit,zone=12,mark=0/0x1,nat(src)),set(tunnel(tun_id=0x1,dst=192.168.122.102,ttl=64,tp_dst=6081,geneve({class=0x102,type=0x80,len=4,0x2000
```

```
7}),flags(df|csum|key))),set(eth(src=0a:58:0a:7c:02:01,dst=9a:8c:6a:42:90:60)),set(ipv4(ttl=63)),genev_sys_6081

# work002(bgp speaker 宣告节点和 egress-service作用节点)上:
# 分析: src=10.0.0.0/255.192.0.0,dst=10.124.0.3 转到 -> dst=192.168.122.101
# ovs-appctl dpctl/dump-flows -m | grep 10.124.0.3
ufid:ffe8db69-5c7e-4d72-881f-b279974287e8,
recirc_id(0xd),dp_hash(0/0),skb_priority(0/0),in_port(ovn-k8s-mp0),skb_mark(0/0),ct_state(0x2a/0x3f),ct_zone(0/0),ct_mark(0/0xf),ct_label(0/0),eth(src=9a:8c:6a:42:90:60,dst=0a:58:0a:7c:02:01),eth_type(0x0800),ipv4(src=10.0.0.0/255.192.0.0,dst=10.124.0.3,proto=6,tos=0/0x3,ttl=61,frag=no),tcp(src=0/0,dst=0/0),tcp_flags(0/0), packets:6, bytes:3209, used:4.508s, flags:FP., dp:ovs,
actions:ct_clear,set(tunnel(tun_id=0x1,dst=192.168.122.101,ttl=64,tp_dst=6081,geneve({class=0x102,type=0x80,len=4,0x60003}),flags(df|csum|key))),set(eth(src=0a:58:0a:7c:00:01,dst=0a:58:0a:7c:00:03)),set(ipv4(ttl=60)),genev_sys_6081
```

总结Egress Service：

功能：pod 访问集群外部服务时，源 ip 设置成对应 lb 类型 svc 的 ip 地址（默认情况下源 ip 为宿主机 ip）

区别：在ovn_cluster_router 路由器中增加一条策略路由，重定向(reroute) 到指定的 work 节点, 做 snat 出去集群网络，eg：101 ---> ip4.src == 10.124.0.3 ---> reroute ---> 10.124.2.2

场景：集群外部服务在配置只允许 lb ip 能通过访问 ACL 规则场景下，可使用该功能