OpenStack

Networking(neutron)

安装并配置controller节点

- 配置先决条件
- 安装网络服务组件
- 配置网络服务组件
- 配置Modular Layer2(ML2)插件
- 配置计算服务使用Neutron
- 完成安装
- 验证

配置先决条件

1、创建数据库，完成下列步骤：

a.使用root用户连接mysql数据库

```
# mysql -u root -p
```

b.创建neutron数据库

```
CREATE DATABASE neutron;
```

c.创建数据库用户neutron，并授予neutron用户对neutron数据库完全控制权限

```
GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'localhost' IDENTIFIED BY
'NEUTRON_DBPASS';
GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'%' IDENTIFIED BY
'NEUTRON_DBPASS';
```

d.退出数据库连接

2、执行admin环境变量脚本

```
# source admin-openrc.sh
```

3、在认证服务中创建网络服务的认证信息，完成下列步骤：

a.创建neutron用户

```
# keystone user-create --name neutron --pass NEUTRON_PASS
```

```
+----------+----------------------------------+
| Property |              Value               |
+----------+----------------------------------+
|  email   |                                  |
| enabled  |               True               |
|    id    | 7799abc098144151b7b2b62c001729fb |
|   name   |             neutron              |
| username |             neutron              |
+----------+----------------------------------+
```

b.连接neutron用户到serivce租户和admin角色

```
# keystone user-role-add --user neutron --tenant service --role admin
```

c.创建neutron服务

```
# keystone service-create --name neutron --type network --description "OpenStack Networking"
```

```
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |       OpenStack Networking       |
|   enabled   |               True               |
|      id     | 60d88f5b0a26441cb3664f6a8cc8ce34 |
|     name    |              neutron             |
|     type    |              network             |
+-------------+----------------------------------+
```

d.创建neutron服务端点

```
# keystone endpoint-create \
--service-id $(keystone service-list | awk '/ network / {print $2}') \
--publicurl http://controller.nice.com:9696 \
--adminurl http://controller.nice.com:9696 \
--internalurl http://controller.nice.com:9696 \
--region regionOne
```

```
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
|   adminurl  | http://controller.nice.com:9696  |
|      id     | 766db17ce4af44ea9223222c60317450 |
| internalurl | http://controller.nice.com:9696  |
|  publicurl  | http://controller.nice.com:9696  |
|    region   |             regionOne            |
|  service_id | 60d88f5b0a26441cb3664f6a8cc8ce34 |
+-------------+----------------------------------+
```


安装网络服务组件

```
# yum install openstack-neutron openstack-neutron-ml2 python-neutronclient which
```


配置网络服务组件

编辑/etc/neutron/neutron.conf文件，并完成下列操作：

a.编辑[database]小节，配置数据库访问

```
[database]
...
connection = mysql://neutron:NEUTRON_DBPASS@controller.nice.com/neutron
```

b.编辑[DEFAULT]小节，配置RabbitMQ消息队列访问：

```
[DEFAULT]
...
rpc_backend = rabbit
rabbit_host = controller.nice.com
rabbit_password = RABBIT_PASS
```

c.编辑[DEFAULT]和[keystone_authtoken]小节，配置认证服务访问：

```
[DEFAULT]
...
auth_strategy = keystone
[keystone_authtoken]
...
auth_uri = http://controller.nice.com:5000/v2.0
identity_uri = http://controller.nice.com:35357
admin_tenant_name = service
admin_user = neutron
admin_password = NEUTRON_PASS
```

d.编辑[DEFAULT]小节，启用Modular Layer2(ML2)插件,路由服务和重叠IP地址功能：

```
[DEFAULT]
...
core_plugin = ml2
service_plugins = router
allow_overlapping_ips = True
```

e.编辑[DEFAULT]小节，配置当网络拓扑结构发生变化时通知计算服务：

```
[DEFAULT]
...
notify_nova_on_port_status_changes = True
notify_nova_on_port_data_changes = True
nova_url = http://controller.nice.com:8774/v2
nova_admin_auth_url = http://controller.nice.com:35357/v2.0
nova_region_name = regionOne
nova_admin_username = nova
nova_admin_tenant_id = SERVICE_TENANT_ID
nova_admin_password = NOVA_PASS
```

注：可通过keystone tenant-get service，获取service租户ID。

f.（可选）在[DEFAULT]小节中配置详细日志输出。方便排错。

```
[DEFAULT]
...
verbose = True
```

配置Modular Layer 2 (ML2) plug-in

编辑/etc/neutron/plugins/ml2/ml2_conf.ini文件，并完成下列操作：

a.编辑[ml2]小节，启用flat和generic routing encapsulation (GRE)网络类型驱动，配置GRE租户网络和OVS驱动机制。

```
[ml2]
...
type_drivers = flat,gre
tenant_network_types = gre
mechanism_drivers = openvswitch
```

b.编辑[ml2_type_gre]小节，配置隧道标识范围：

```
[ml2_type_gre]
...
tunnel_id_ranges = 1:1000
```

c.编辑[securitygroup]小节，启用安全组，启用ipset并配置OVS防火墙驱动：

```
[securitygroup]
...
enable_security_group = True
enable_ipset = True
firewall_driver = neutron.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDriver
```

配置计算服务使用Neutron

默认情况下，计算服务使用传统网络，我们需要重新配置。

编辑/etc/nova/nova.conf文件，并完成下列操作：

a.编辑[DEFAULT]小节，配置API接口和驱动程序：

```
[DEFAULT]
...
network_api_class = nova.network.neutronv2.api.API
security_group_api = neutron
linuxnet_interface_driver = nova.network.linux_net.LinuxOVSInterfaceDriver
firewall_driver = nova.virt.firewall.NoopFirewallDriver
```

b.编辑[neutron]小节，配置访问参数：

```
[neutron]
...
url = http://controller.nice.com:9696
auth_strategy = keystone
admin_auth_url = http://controller.nice.com:35357/v2.0
admin_tenant_name = service
admin_username = neutron
admin_password = NEUTRON_PASS
```

完成配置

1、为ML2插件配置文件创建连接文件。

```
# ln -s /etc/neutron/plugins/ml2/ml2_conf.ini /etc/neutron/plugin.ini
```

2、初始化数据库

```
# su -s /bin/sh -c "neutron-db-manage --config-file /etc/neutron/neutron.conf --config-file /etc/neutron/plugins/ml2/ml2_conf.ini upgrade juno" neutron
```

3、重新启动计算服务

```
# systemctl restart openstack-nova-api.service openstack-nova-scheduler.service openstack-nova-conductor.service
```

4、启动网络服务并配置开机自动启动

```
# systemctl enable neutron-server.service
# systemctl start neutron-server.service
```

验证

1、执行admin环境变量脚本

```
# source admin-openrc.sh
```

2、列出加载的扩展模块，确认成功启动neutron-server进程。

```
# neutron ext-list
```

alias	name
security-group	security-group
l3_agent_scheduler	L3 Agent Scheduler
ext-gw-mode	Neutron L3 Configurable external gateway mode
binding	Port Binding
provider	Provider Network
agent	agent
quotas	Quota management support
dhcp_agent_scheduler	DHCP Agent Scheduler
l3-ha	HA Router extension
multi-provider	Multi Provider Network
external-net	Neutron external network
router	Neutron L3 Router
allowed-address-pairs	Allowed Address Pairs
extraroute	Neutron Extra Route
extra_dhcp_opt	Neutron Extra DHCP opts
dvr	Distributed Virtual Router

安装并配置network节点

- 配置先决条件
- 安装网络组件
- 配置网络通用组件
- 配置Modular Layer 2 (ML2) plug-in
- 配置Layer-3 (L3) agent
- 配置DHCP agent
- 配置metadata agent
- 配置Open vSwitch (OVS)服务
- 完成安装
- 验证

配置先觉条件

1、编辑/etc/sysctl.conf文件，包含下列参数：

```
net.ipv4.ip_forward=1
net.ipv4.conf.all.rp_filter=0
net.ipv4.conf.default.rp_filter=0
```

2、使更改生效

```
# sysctl -p
```

安装网络组件

```
# yum install openstack-neutron openstack-neutron-ml2 openstack-neutron-openvswitch
```

配置网络通用组件

网络通用组件配置包含认证机制，消息队列及插件。

编辑/etc/neutron/neutron.conf文件并完成下列操作：

a.编辑[database]小节，注释任何connection选项。因为network节点不能直接连接数据库。

b.编辑[DEFAULT]小节，配置RabbitMQ消息队列访问

```
[DEFAULT]
...
rpc_backend = rabbit
rabbit_host = controller.nice.com
rabbit_password = RABBIT_PASS
```

c.编辑[DEFAULT]和[keystone_authtoken]小节，配置认证服务访问：

```
[DEFAULT]
...
auth_strategy = keystone

[keystone_authtoken]
...
auth_uri = http://controller.nice.com:5000/v2.0
identity_uri = http://controller.nice.com:35357
admin_tenant_name = service
admin_user = neutron
admin_password = NEUTRON_PASS
```

d.编辑[DEFAULT]小节，启用Modular Layer2(ML2)插件,路由服务和重叠IP地址功能：

```
[DEFAULT]
...
core_plugin = ml2
service_plugins = router
allow_overlapping_ips = True
```

e.（可选）在[DEFAULT]小节中配置详细日志输出。方便排错。

```
[DEFAULT]
...
verbose = True
```

配置Modular Layer 2 (ML2) plug-in

ML2插件使用Open vSwitch (OVS)机制为虚拟机实例提供网络框架。
编辑/etc/neutron/plugins/ml2/ml2_conf.ini文件并完成下列操作：

a.编辑[ml2]小节，启用flat和generic routing encapsulation (GRE)网络类型驱动，配置GRE租户网络和OVS驱动机制。

```
[ml2]
...
type_drivers = flat,gre
tenant_network_types = gre
mechanism_drivers = openvswitch
```

b.编辑[ml2_type_flat]小节，配置外部网络：

```
[ml2_type_flat]
...
flat_networks = external
```

c.编辑[ml2_type_gre]小节，配置隧道标识范围：

```
[ml2_type_gre]
...
tunnel_id_ranges = 1:1000
```

d.编辑[securitygroup]小节，启用安全组，启用ipset并配置OVS防火墙驱动：

```
[securitygroup]
...
enable_security_group = True
enable_ipset = True
firewall_driver = neutron.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDriver
```

e.编辑[ovs]小节,配置Open vSwitch (OVS) 代理

```
[ovs]
...
local_ip = INSTANCE_TUNNELS_INTERFACE_IP_ADDRESS
tunnel_type = gre
enable_tunneling = True
bridge_mappings = external:br-ex
```

配置Layer-3 (L3) agent

编辑/etc/neutron/l3_agent.ini文件并完成下列配置：

a.编辑[DEFAULT]小节，配置驱动，启用网络命名空间，配置外部网络桥接

```
[DEFAULT]
...
interface_driver = neutron.agent.linux.interface.OVSInterfaceDriver
use_namespaces = True
external_network_bridge = br-ex
```

b.（可选）在[DEFAULT]小节中配置详细日志输出。方便排错。

```
[DEFAULT]
...
verbose = True
```

配置DHCP agent

1、编辑/etc/neutron/dhcp_agent.ini文件并完成下列步骤：

a.编辑[DEFAULT]小节，配置驱动和启用命名空间

```
[DEFAULT]
...
interface_driver = neutron.agent.linux.interface.OVSInterfaceDriver
dhcp_driver = neutron.agent.linux.dhcp.Dnsmasq
use_namespaces = True
```

b.（可选）在[DEFAULT]小节中配置详细日志输出。方便排错。

```
[DEFAULT]
...
verbose = True
```

2、(可选,在VMware虚拟机中可能是必要的!)配置DHCP选项，将MUT改为1454bytes，以改善网络性能。

a.编辑/etc/neutron/dhcp_agent.ini文件并完成下列步骤：

编辑[DEFAULT]小节，启用dnsmasq配置：

```
[DEFAULT]
...
dnsmasq_config_file = /etc/neutron/dnsmasq-neutron.conf
```

b.创建并编辑/etc/neutron/dnsmasq-neutron.conf文件并完成下列配置：

启用DHCP MTU选项(26)并配置值为1454bytes

```
dhcp-option-force=26,1454
user=neutron
group=neutron
```

c.终止任何已经存在的dnsmasq进行

```
# pkill dnsmasq
```

配置metadata agent

1、编辑/etc/neutron/metadata_agent.ini文件并完成下列配置：

a.编辑[DEFAULT]小节，配置访问参数：

```
[DEFAULT]
...
auth_url = http://controller.nice.com:5000/v2.0
auth_region = regionOne
admin_tenant_name = service
admin_user = neutron
admin_password = NEUTRON_PASS
```

b.编辑[DEFAULT]小节,配置元数据主机：

```
[DEFAULT]
...
nova_metadata_ip = controller.nice.com
```

c.编辑[DEFAULT]小节，配置元数据代理共享机密暗号：

```
[DEFAULT]
...
metadata_proxy_shared_secret = METADATA_SECRET
```

b.（可选）在[DEFAULT]小节中配置详细日志输出。方便排错。

```
[DEFAULT]
...
verbose = True
```

2、在**controller节点**，编辑/etc/nova/nova.conf文件并完成下列配置

编辑[neutron]小节，启用元数据代理并配置机密暗号：

```
[neutron]
...
service_metadata_proxy = True
metadata_proxy_shared_secret = METADATA_SECRET
```

3、在**controller节点**，重新启动compute API服务

```
# systemctl restart openstack-nova-api.service
```

配置Open vSwitch (OVS)服务

1、启动VOS服务并配置开机自动启动：

```
# systemctl enable openvswitch.service
# systemctl start openvswitch.service
```

2、添加外部网桥(external birdge)

```
# ovs-vsctl add-br br-ex
```

3、添加一个端口到外部网桥，用于连接外部物理网络

```
# ovs-vsctl add-port br-ex INTERFACE_NAME
```

注：将*INTERFACE_NAME*换成实际连接外部网卡接口名。如：eth2或eno50332208。

完成安装

1、创建网络服务初始化脚本的符号连接

```
# ln -s /etc/neutron/plugins/ml2/ml2_conf.ini /etc/neutron/plugin.ini

# cp /usr/lib/systemd/system/neutron-openvswitch-agent.service
/usr/lib/systemd/system/neutron-openvswitch-agent.service.orig

# sed -i 's,plugins/openvswitch/ovs_neutron_plugin.ini,plugin.ini,g'
/usr/lib/systemd/system/neutron-openvswitch-agent.service
```

2、启动网络服务并设置开机自动启动

```
# systemctl enable neutron-openvswitch-agent.service neutron-l3-agent.service neutron-
dhcp-agent.service neutron-metadata-agent.service neutron-ovs-cleanup.service
# systemctl start neutron-openvswitch-agent.service neutron-l3-agent.service neutron-
dhcp-agent.service neutron-metadata-agent.service
```

验证(在controller节点执行下列命令)

1、执行admin环境变量脚本

```
# source admin-openrc.sh
```

2、列出neutron代理，确认启动neutron agents成功。

```
# neutron agent-list
```

```
+--------------------------------------+--------------------+-----------------+-------+----------------+---------------------------+
| id                                   | agent_type         | host            | alive | admin_state_up | binary                    |
+--------------------------------------+--------------------+-----------------+-------+----------------+---------------------------+
| 1f12c88c-f408-406c-b6ff-bfdfc7657f1d | L3 agent           | network.nice.com | :-)   | True           | neutron-l3-agent          |
| 4da69d94-821f-4ffa-8a20-33efaa2e95bf | Open vSwitch agent | network.nice.com | :-)   | True           | neutron-openvswitch-agent |
| 91da0d86-1498-4a93-b4a0-b618ce17704d | DHCP agent         | network.nice.com | :-)   | True           | neutron-dhcp-agent        |
| ecd59997-22f1-467c-b3d7-ea5a90bb13a2 | Metadata agent     | network.nice.com | :-)   | True           | neutron-metadata-agent    |
+--------------------------------------+--------------------+-----------------+-------+----------------+---------------------------+
```

安装并配置compute1节点

- 配置先决条件
- 安装网络组件
- 配置网络通用组件
- 配置Modular Layer 2 (ML2) plug-in
- 配置Open vSwitch (OVS) service
- 配置计算服务使用网络
- 完成安装
- 验证

配置先决条件

1、编辑/etc/sysctl.conf文件，使其包含下列参数：

```
net.ipv4.conf.all.rp_filter=0
net.ipv4.conf.default.rp_filter=0
```

2、使/etc/sysctl.conf文件中的更改生效：

```
# sysctl -p
```

安装网络组件

```
#  yum install openstack-neutron-ml2 openstack-neutron-openvswitch
```

配置网络通用组件

编辑/etc/neutron/neutron.conf文件并完成下列操作：

a.编辑[database]小节，注释左右connection配置项。因为计算节点不能直接连接数据库。

b.编辑[DEFAULT]小节，配置RabbitMQ消息代理访问：

```
[DEFAULT]
...
rpc_backend = rabbit
rabbit_host = controller.nice.com
rabbit_password = RABBIT_PASS
```

c.编辑[DEFAULT]和[keystone_authtoken]小节，配置认证服务访问：

```
[DEFAULT]
...
auth_strategy = keystone

[keystone_authtoken]
...
auth_uri = http://controller.nice.com:5000/v2.0
identity_uri = http://controller.nice.com:35357
admin_tenant_name = service
admin_user = neutron
admin_password = NEUTRON_PASS
```

d.编辑[DEFAULT]小节，启用Modular Layer2(ML2)插件，路由服务和重叠ip地址功能：

```
[DEFAULT]
...
core_plugin = ml2
service_plugins = router
allow_overlapping_ips = True
```

b.（可选）在[DEFAULT]小节中配置详细日志输出。方便排错。

```
[DEFAULT]
...
verbose = True
```

配置Modular Layer 2 (ML2) plug-in

编辑/etc/neutron/plugins/ml2/ml2_conf.ini文件并完成下列操作：

a.编辑[ml2]小节，启用flat和generic routing encapsulation (GRE)网络类型驱动，GRE租户网络和OVS机制驱动：

```
[ml2]
...
type_drivers = flat,gre
tenant_network_types = gre
mechanism_drivers = openvswitch
```

b.编辑[ml2_type_gre]小节，配置隧道标识符(id)范围：

```
[ml2_type_gre]
...
tunnel_id_ranges = 1:1000
```

c.编辑[securitygroup]小节，启用安装组，ipset并配置OVS iptables防火墙驱动：

```
[securitygroup]
...
enable_security_group = True
enable_ipset = True
firewall_driver = neutron.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDriver
```

d.编辑[ovs]小节，配置Open vSwitch (OVS) agent：

```
[ovs]
...
local_ip = INSTANCE_TUNNELS_INTERFACE_IP_ADDRESS
tunnel_type = gre
enable_tunneling = True
```

配置Open vSwitch (OVS) service

启动OVS服务并设置开机自动启动：

```
# systemctl enable openvswitch.service
# systemctl start openvswitch.service
```

配置计算服务使用网络

编辑/etc/nova/nova.conf文件并完成下列操作：

a.编辑[DEFAULT]小节,配置API接口和驱动：

```
[DEFAULT]
...
network_api_class = nova.network.neutronv2.api.API
security_group_api = neutron
linuxnet_interface_driver = nova.network.linux_net.LinuxOVSInterfaceDriver
firewall_driver = nova.virt.firewall.NoopFirewallDriver
```

b.编辑[neutron]小节，配置访问参数：

```
[neutron]
...
url = http://controller.nice.com:9696
auth_strategy = keystone
admin_auth_url = http://controller.nice.com:35357/v2.0
admin_tenant_name = service
admin_username = neutron
admin_password = NEUTRON_PASS
```

完成安装

1、创建网络服务初始化脚本的符号连接

```
# ln -s /etc/neutron/plugins/ml2/ml2_conf.ini /etc/neutron/plugin.ini

# cp /usr/lib/systemd/system/neutron-openvswitch-agent.service
/usr/lib/systemd/system/neutron-openvswitch-agent.service.orig

# sed -i 's,plugins/openvswitch/ovs_neutron_plugin.ini,plugin.ini,g'
/usr/lib/systemd/system/neutron-openvswitch-agent.service
```

2、重启计算服务：

```
# systemctl restart openstack-nova-compute.service
```

3、启动OVS代理服务并设置开机自动启动：

```
# systemctl enable neutron-openvswitch-agent.service
# systemctl start neutron-openvswitch-agent.service
```

验证(在controller节点执行下列命令)

1、执行admin环境变量脚本

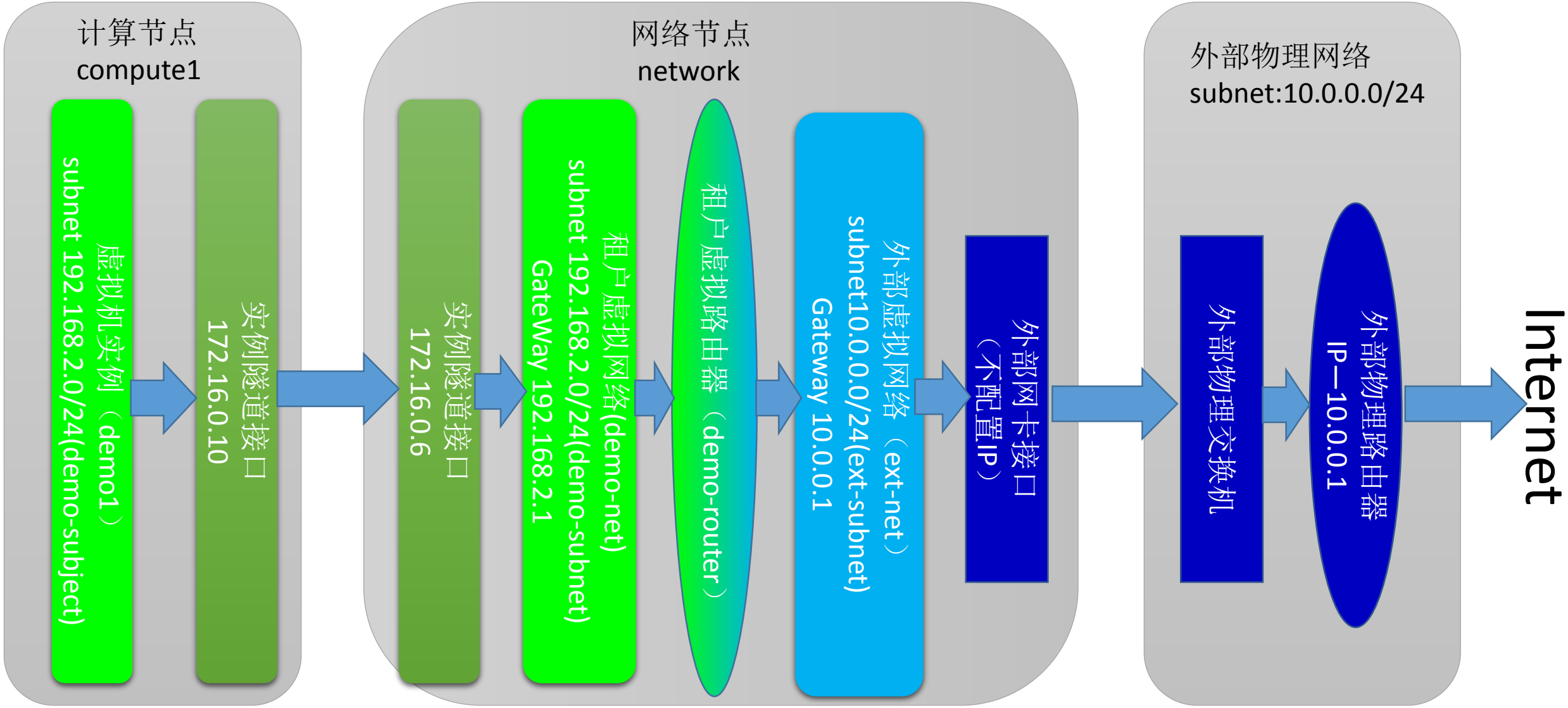
```
# source admin-openrc.sh
```

2、列出neutron代理，确认启动neutron agents成功。

```
# neutron agent-list
```

```
+--------------------------------------+--------------------+------------------+-------+----------------+---------------------------+
| id                                   | agent_type         | host             | alive | admin_state_up | binary                    |
+--------------------------------------+--------------------+------------------+-------+----------------+---------------------------+
| 1f12c88c-f408-406c-b6ff-bfdfc7657f1d | L3 agent           | network.nice.com | :-)   | True           | neutron-l3-agent          |
| 4da69d94-821f-4ffa-8a20-33efaa2e95bf | Open vSwitch agent | network.nice.com | :-)   | True           | neutron-openvswitch-agent |
| 91da0d86-1498-4a93-b4a0-b618ce17704d | DHCP agent         | network.nice.com | :-)   | True           | neutron-dhcp-agent        |
| ecd59997-22f1-467c-b3d7-ea5a90bb13a2 | Metadata agent     | network.nice.com | :-)   | True           | neutron-metadata-agent    |
| f2962e79-72b5-4626-8ddd-3c3d94ab7f91 | Open vSwitch agent | compute1.nice.com | :-)   | True           | neutron-openvswitch-agent |
+--------------------------------------+--------------------+------------------+-------+----------------+---------------------------+
```

创建第一个网络



openstack

配置外部网络(在controller节点执行后面的命令)

- 创建一个外部网络
- 创建一个外部网络的子网

创建一个外部网络

1、执行admin环境变量脚本

```
# source admin-openrc.sh
```

2、创建网络

```
# neutron net-create ext-net --shared --router:external True --provider:physical_network external --provider:network_type flat
```

Field	Value
admin_state_up	True
id	10f74979-8ba7-4127-ab3d-a4e9e8c95b18
name	ext-net
provider:network_type	flat
provider:physical_network	external
provider:segmentation_id	
router:external	True
shared	True
status	ACTIVE
subnets	
tenant_id	28b6940857244f8fa0e4178f07d1f125

创建一个外部网络的子网

创建子网：

```
# neutron subnet-create ext-net --name ext-subnet \
--allocation-pool start=FLOATING_IP_START,end=FLOATING_IP_END \
--disable-dhcp --gateway EXTERNAL_NETWORK_GATEWAY  EXTERNAL_NETWORK_CIDR
```

FLOATING_IP_STAR=起始IP

FLOATING_IP_END=结束IP

EXTERNAL_NETWORK_GATEWAY=外部网络网关

EXTERNAL_NETWORK_CIDR=外部网络网段

例如，外网网段为：10.0.0.0/24，浮动地址范围为：10.0.0.100~10.0.0.200，网关为：10.0.0.1:

```
# neutron subnet-create ext-net --name ext-subnet \
  --allocation-pool start=10.0.0.100,end=10.0.0.200 \
  --disable-dhcp --gateway 10.0.0.1 10.0.0.0/24
```

```
+-------------------+--------------------------------------------------+
| Field             | Value                                            |
+-------------------+--------------------------------------------------+
| allocation_pools  | {"start": "10.0.0.100", "end": "10.0.0.200"}     |
| cidr              | 10.0.0.0/24                                      |
| dns_nameservers   |                                                  |
| enable_dhcp       | False                                            |
| gateway_ip        | 10.0.0.1                                         |
| host_routes       |                                                  |
| id                | dbe681f0-1287-4e68-aaa8-fa2c319049b8             |
| ip_version        | 4                                                |
| ipv6_address_mode |                                                  |
| ipv6_ra_mode      |                                                  |
| name              | ext-subnet                                       |
| network_id        | 10f74979-8ba7-4127-ab3d-a4e9e8c95b18             |
| tenant_id         | 28b6940857244f8fa0e4178f07d1f125                 |
+-------------------+--------------------------------------------------+
```

配置租户网络(在controller节点执行后面的命令)

- 创建租户网络
- 创建租户网络的子网
- 在租户网络创建一个路由器，用来连接外部网和租户网

openstack

创建一个租户网络

1、执行demo环境变量脚本

```
# source demo-openrc.sh
```

2、创建租户网络

```
# neutron net-create demo-net
```

Field	Value
admin_state_up	True
id	d36f6eb0-e59a-42b9-9209-5547e022484b
name	demo-net
provider:network_type	gre
provider:physical_network	
provider:segmentation_id	1
router:external	False
shared	False
status	ACTIVE
subnets	c7822bf6-5750-4f11-935b-3028cd95af04
tenant_id	a59e18303bc246eb92b86a1492db462b

创建一个租户网络的子网

创建子网：

```
#neutron subnet-create demo-net --name demo-subnet  |
  --gateway TENANT_NETWORK_GATEWAY TENANT_NETWORK_CIDR
```

TENANT_NETWORK_GATEWAY=租户网的网关

TENANT_NETWORK_CIDR=租户网的网段

例如，租户网的网段为192.168.2.0/24，网关为192.168.2.1(网关通常默认为.1)

```
# neutron subnet-create demo-net --name demo-subnet \
    --gateway 192.168.2.1 192.168.2.0/24
```

```
+-------------------+-------------------------------------------------+
| Field             | Value                                           |
+-------------------+-------------------------------------------------+
| allocation_pools  | {"start": "192.168.2.2", "end": "192.168.2.254"} |
| cidr              | 192.168.2.0/24                                  |
| dns_nameservers   |                                                 |
| enable_dhcp       | True                                            |
| gateway_ip        | 192.168.2.1                                     |
| host_routes       |                                                 |
| id                | c7822bf6-5750-4f11-935b-3028cd95af04            |
| ip_version        | 4                                               |
| ipv6_address_mode |                                                 |
| ipv6_ra_mode      |                                                 |
| name              | demo-subnet                                     |
| network_id        | d36f6eb0-e59a-42b9-9209-5547e022484b            |
| tenant_id         | a59e18303bc246eb92b86a1492db462b                |
+-------------------+-------------------------------------------------+
```

在租户网络创建一个路由器，用来连接外部网和租户网

1、创建路由器

```
# neutron router-create demo-router
```

```
+-----------------------+--------------------------------------+
| Field                 | Value                                |
+-----------------------+--------------------------------------+
| admin_state_up        | True                                 |
| external_gateway_info |                                      |
| id                    | 5c0e67e7-337f-4309-8bc5-b1eae9088f02 |
| name                  | demo-router                          |
| routes                |                                      |
| status                | ACTIVE                               |
| tenant_id             | a59e18303bc246eb92b86a1492db462b     |
+-----------------------+--------------------------------------+
```

2、附加路由器到demo租户的子网

```
# neutron router-interface-add demo-router demo-subnet
```

```
[root@controller ~]# Added interface b1a894fd-aee8-475c-9262-4342afdc1b58 to router demorouter.
```

3、通过设置网关，使路由器附加到外部网

```
# neutron router-gateway-set demo-router ext-net
```

```
[root@controller ~]# Set gateway for router demo-router
```


确认连接

1、查看路由器获取到的IP。

```
# neutron router-list
+--------------------------------------+-------------+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+-------------+-------+
| id                                   | name        | external_gateway_info                                                                                                                                                                                        | distributed | ha    |
+--------------------------------------+-------------+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+-------------+-------+
| 5c0e67e7-337f-4309-8bc5-b1eae9088f02 | demo-router | {"network_id": "10f74979-8ba7-4127-ab3d-a4e9e8c95b18", "enable_snat": true, "external_fixed_ips": [{"subnet_id": "dbe681f0-1287-4e68-aaa8-fa2c319049b8", "ip_address": "10.0.0.100"}]} | False       | False |
+--------------------------------------+-------------+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+-------------+-------+
```

2、在任何一台外部主机上ping路由器获取到的外部地址

```
C:\Windows\system32>ping 10.0.0.100

正在 Ping 10.0.0.100 具有 32 字节的数据:
来自 10.0.0.100 的回复: 字节=32 时间<1ms TTL=64
来自 10.0.0.100 的回复: 字节=32 时间<1ms TTL=64
来自 10.0.0.100 的回复: 字节=32 时间<1ms TTL=64
来自 10.0.0.100 的回复: 字节=32 时间<1ms TTL=64

10.0.0.100 的 Ping 统计信息:
    数据包: 已发送 = 4，已接收 = 4，丢失 = 0 (0% 丢失)，
往返行程的估计时间(以毫秒为单位):
    最短 = 0ms，最长 = 0ms，平均 = 0ms
```