# 一．Neutron

## （一）Neutron介绍

传统的网络管理方式依赖于管理员手工配置和维护各种网络硬件设备，而云环境下的网络非常复杂，在多租户的场景下，用户可能随时需要创建，修改和删除网络，因此需要自动地对网络进行修改。软件定义网络（SDN）所具有的灵活性和自动化优势成为了云时代的网络管理的主流方式。Neutron的设计目标是“网络即服务“，在设计上遵循了基于SDN实现网络虚拟化的原则，充分利用了Linux系统上的各种网络相关技术。

## （二）Neutron功能

Neutron为整个Openstack环境提供了网络支持，包括二层交换，三层路由，负载均衡，防火墙和VPN等功能。

### 1.二层交换Switching

Nova的Instance是通过虚拟交换机连接到虚拟的二层网络上的，Neutron支持多种虚拟交换机，包括Linux原生的Linux Bridge和Open vSwitch（OVS）。利用虚拟交换机，Neutron除了可以创建传统的VLAN网络外，还可以创建基于隧道技术的Overlay网络，如VxLan和GRE。

### 2.三层路由Routing

Instance可以配置不同网段的IP，Neutron的router（虚拟路由器）实现Instance 跨网段通信。虚拟路由器会通过IP forwading，iptables等技术实现路由和NAT。

### 3.负载均衡Load Balancing

Openstack在Grizzly版本引入了Load-Balancing-as-a-service(LBaaS)，提供将负载分发到多个Instance的能力。

### 4.防火墙Firewalling

Neutron通过两种方式来保障instance和网络的安全性。

#### （1）Security Group

通过iptables限制进出instance的网络包。

#### （2）Firewall-as-a-Service

限制进出虚拟路由器的网络包，通过iptables实现。

## （三）Neutron网络的基本概念

Neutron管理的网络资源包括network, subnet和port：

### 1.network

network是一个隔离的二层广播域，Neutron支持多种类型的network，包括local, fla, VLAN, VxLAN和GRE。不同的network在二层上是隔离的，但是如果借助了路由器的haunt，则network可以在三层之上通信。

注意，network必须属于某个Project（Tenant租户），一个Project可以创建多个network。

#### （1）local

local网络与其他网络和节点隔离，local网络中的instance只能与位于同一节点上的同一网络的instance通信，local主要用于单机测试。

#### （2）flat

flat网络是无vlan tagging的网络，其中的instance能够与同一网路中的instance通信，并且可以跨越多个节点。

#### （3）vlan

vlan是一个二层的广播域，同一个vlan的instance可以通信，不同的vlan只能够通过router通信，vlan网络可以跨节点，是应用最广的网络类型。

#### （4）vxlan

vxlan是基于隧道技术的overlay网络。

#### （5）gre

gre也是overlay网络，使用的是IP包进行封装。

### 2.subnet

subnet是一个IPV4或IPV6地址段，即子网。instance的IP从subnet中分配，每个subnet需要定义IP地址的范围和掩码。一个subnet只能够属于某个network，一个network可以有多个subnet，这些subnet可以是不同的IP段，但是每个subnet的CIDR不能重叠。如下所示：

network A

subnet A-a : 10.10.1.0/24 {“start” : “10.10.1.1”, “end” : “10.10.1.50”}

subnet A-b : 10.10.1.0/24 {“start” : “10.10.1.51”, “end” : “10.10.1.100”}

虽然两个子网的IP地址没有重叠，但是它们的子网段都是10.10.1.0/24，因此是重叠的。

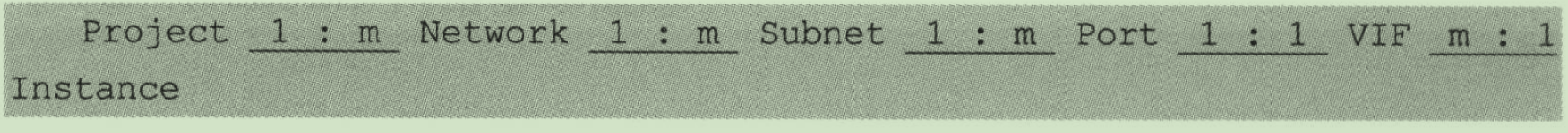
如果subnet在不同的network中，则CIDR和IP都是可以重叠的。那么这样不就会造成IP冲突了吗？

答案是不会的，因为Neutron的router是通过Linux network namespace实现的，network namespace是一种网络隔离机制，通过它，每个router有自己独立的路由表。因此，如果两个subnet是通过同一个router路由，根据router的配置，只有指定的一个subnet可以被路由；如果两个subnet是通过不同的router路由的，因为router的路由表是独立的，所以两个subnet都可以被路由。

### 3.port

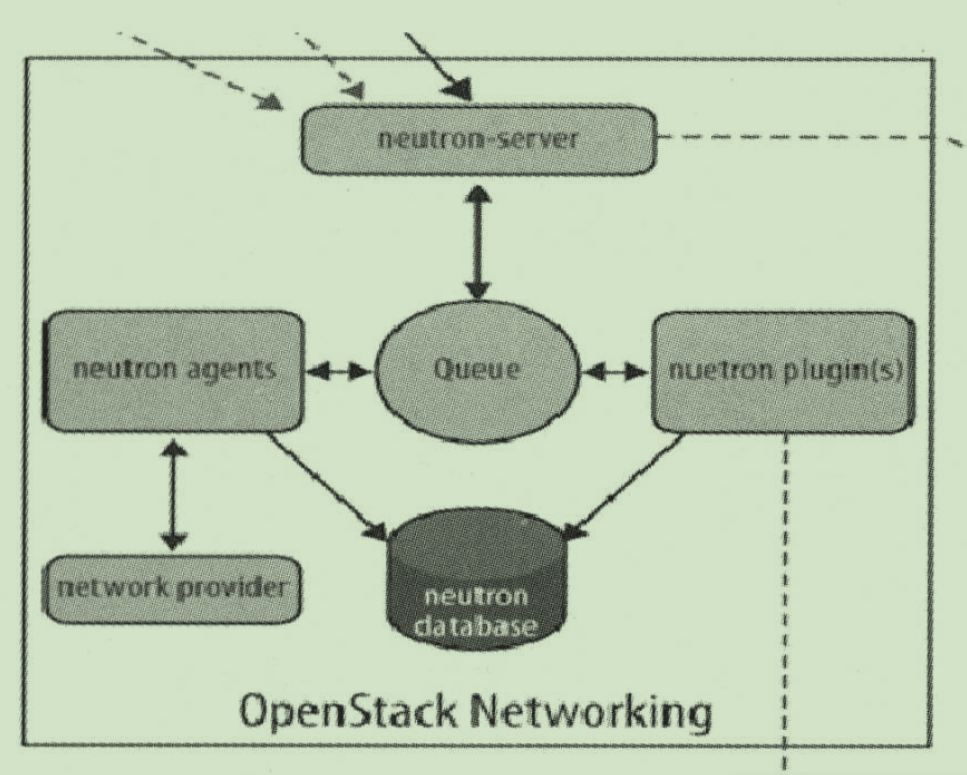
port是虚拟交换机上的一个端口，定义了MAC地址和IP地址，当instance的虚拟网卡VIF（virtual Interface）绑定到port时，port会将MAC和IP分配给VIF。一个port必须属于某个subnet，一个subnet可以有多个port。

### 4.Project，Network，Subnet，Port和VIP的关系



## （四）Neutron的架构

### 1.Neutron的架构图



#### （1）Neutron Server

对外提供Openstack网络的API，并调用Plugin处理请求。

#### （2）Plugin

处理Neutron Server发来的请求，维护Openstack逻辑网络的状态，并调用Agent处理请求。按照功能分为两类：core plugin和service plugin，core plugin维护Neutron的network，subnet和port等信息；service plugin提供routing, firewall, load balance等服务。

#### （3）Agent

处理Plugin的请求，负责在network provider上真正实现各种网络功能。

#### （4）network provider

提供网络服务的虚拟或物理网络设备，例如Linux Bridge, Open vSwitch等虚拟交换机或物理交换机。

#### （5）Queue

Neutron Server , Plugin 和Agent之间通过RabbitMQ通信和调用。

#### （6）Database

存放Openstack的网络状态信息，包括network , subnet, port, router等。

注意，plugin，agent和network provider是配套使用的，如过network provider是Linux Bridge，那么就需要使用Linux Bridge 的 plugin 和 agent。

### 2.Neutron子服务的物理部署方案

#### （1）第一种方案

控制节点 + 计算节点。注意，由于core plugin 和 service plugin已经集成到了neutron server中，不需要运行独立的plugin服务。控制节点和计算节点都需要部署core plugin的agent，因为通过该agent控制节点与计算节点才能够建立二层连接。

##### a）控制节点

neutron server, core plugin的agent和service plugin的agent。

##### b）计算节点

core plugin的agent，负责提供二层网络功能。

#### （2）第二种方案

控制节点 + 网络节点 + 计算节点。将所有的agent从控制节点分离出来，部署到独立的网络节点上。可以增加网络节点承担更大的网络负载，适合规模较大的Openstack环境。

##### a）控制节点

neutron server服务。

##### b）网络节点

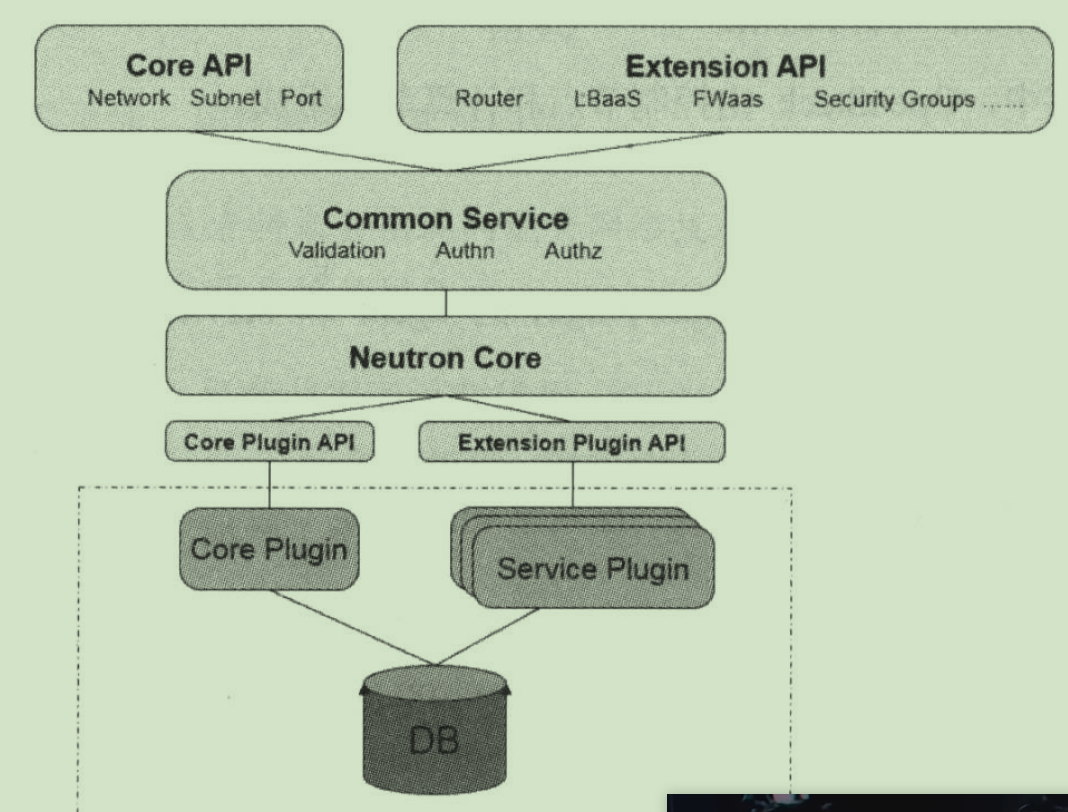
core plugin的agent和service plugin 的agent。

##### c）计算节点

部署core plugin的agent，负责提供二层网络功能。

### 3.Neutron Server

包括两部分功能，提供API服务与运行Plugin。以下是它的架构图：



从上而下的组件：

#### （1）Core API

对外提供管理network，subnet和port的RESTful API。

#### （2）Extension API

对外提供管理router, load balance, firewall等资源的RESTful API。

#### （3）Common Service

认证和校验API请求。

#### （4）Neutron Core

核心处理程序，调用相应的Plugin处理请求。

#### （5）Core Plugin API

定义了Core Plugin的抽象功能集合, Neutron Core通过该API调用相应的Core Plugin。

#### （6）Extension Plugin API

定义了Service Plugin的抽象功能集合，Neutron Core通过该API调用响应的Service Plugin。

#### （7）Core Plugin

实现了Core Plugin API，在数据库中维护netwok ,subnet 和 port的状态，并负责调用相应的agent在network provider上执行操作。

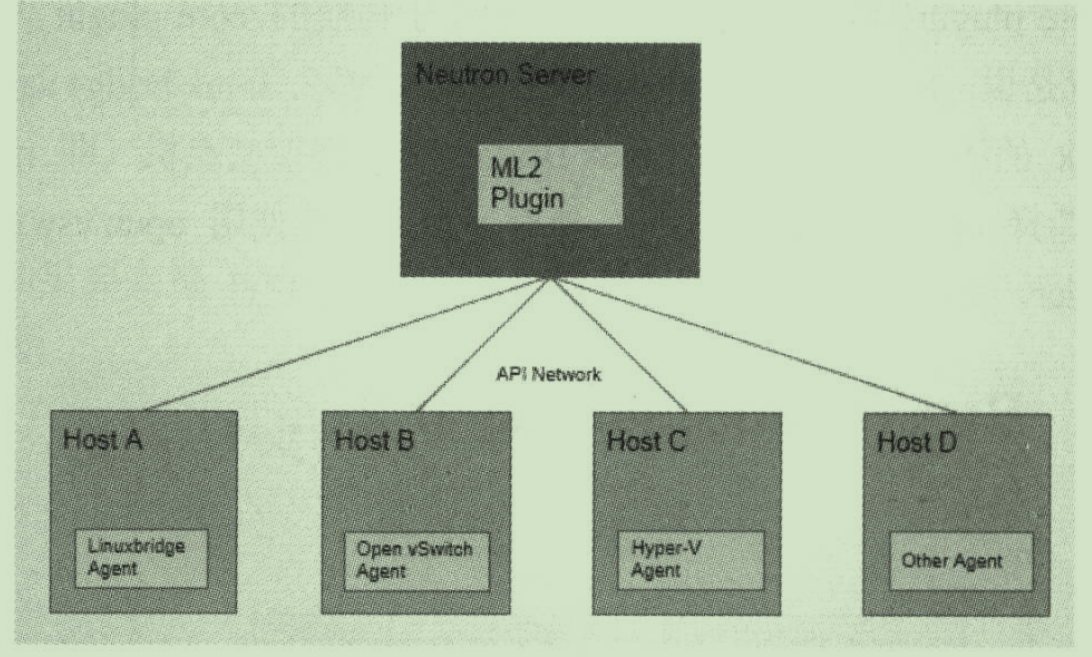
#### （8）Service Plugin

实现了Extension Plugin API，在数据库中维护router, load balance, security group等资源的状态，并负责调用相应的agent在network provider上执行操作。

### 4.Neutron怎样支持多种network provider

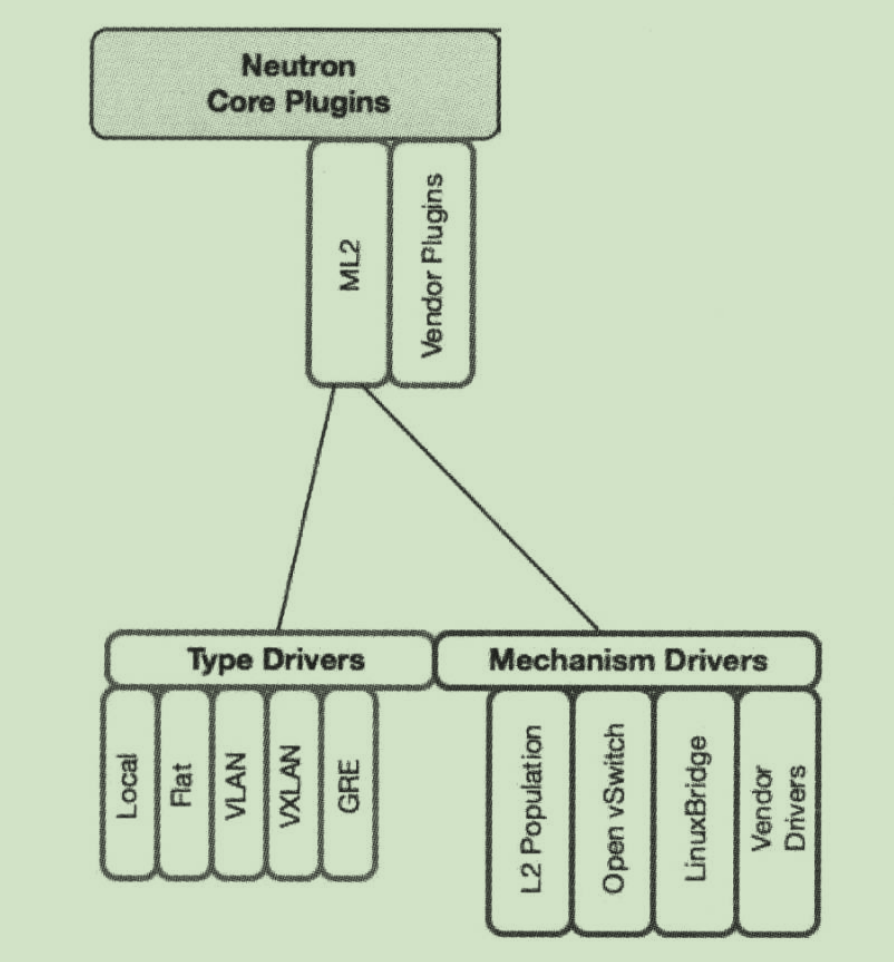
由于network provider, core plugin, agent都是配套使用的，如一个Linux Bridge就只能使用linux Bridege core plugin和linux bridge agent。由于core plugin负责管理和维护network, subnet和port的状态信息，这些信息是全局的，因此只能使用一个core plugin，多种network provider无法共存，且不同的plugin之间存在大量重复代码，开发新的plugin工作量大。

Moduler Layer2（ML2）是Neutron在Havana版本实现的一个新的core plugin，用于替代原有的linux bridge plugin 和 open vswitch plugin。ML2提供了一个框架，允许在OpenStack网络中同时使用多种Layer2网络技术，不同的节点可以使用不同的network provider。



采用ML2 plugin之后，可以在不同的节点上分别部署linux bridge agent, open vswitch agent, hyper-v agent等。

ML2对二层网络进行了抽象和建模，引入了type driver和mechansim driver，这两类driver解耦了Neutron所支持的网络类型type和访问这些网络类型的机制mechanisum，使得ML2有很好的弹性。



#### （1）Type Driver

Neutron支持的每一种网络类型都有一个对应的ML2 Type Driver，负责维护网络类型的状态，执行验证，创建网络。

#### （2）Mechansim Driver

每一种网络类型都有一个对应的ML2 Mechansim Driver。它负责获取网络状态，并确保在相应的网络设备上正确实现这些状态。

### 5.Service Plugin/Agent

之前我们说到Neutron core plugin及其agent负责将instance连接到layer2虚拟网络，所提供的资源包括network, sbunet和port。

而Service Plugin及其agent提供更多扩展功能，包括router, load balance, firewall。

#### （1）DHCP

DHCP agent通过dnsmasq为instance提供DHCP服务。

#### （2）Routing

L3 Agent可以为Project创建router，提供subnet之间的路由服务。路由的功能默认由Iptables实现。

#### （3）firewall

L3 Agent可以在router上配置防火墙策略，位于router上，保护的是某个project上的所有network。另一个安全功能是security group，也是通过iptables实现的，位于单个instance上。

#### （4）Load Balance

默认通过HAProxy为Project中的多个instance提供load balance服务。