云平台构建小组技术总结

随着时间推移，云平台小组成立数月，在小组成员努力探索下，确定了总体技术方向和演进路线。图1中即小组确立的最终建设方案，其中已实现的部分我们将尽力优化，未实现部分我们会加快脚步。

近年来，随着云计算的飞速发展，个人和企业都将其业务云化，以提升效率和节约成本。在企业构建自己的云平台时，摆在他们面前的是两条路，一条是openstack，一条是docker。这两者其实都是在做同一件事情，那就是将资源池化。虽然他们的理念一致，但实现方法不同，导致最后的应用也不尽相同。Openstack，使用传统的虚拟机技术，虚拟机技术出现这么多年了，为什么因为openstack非常火了呢？那是因为openstack赋予了大规模管理虚拟机的能力，再加上软件技术的发展，“软件定义网络”，“软件定义存储”，“软件定义数据中心”，貌似这个世界上的一切都可以用软件定义了，这两点足以让传统的虚拟机技术如虎添翼。Docker，新的容器化技术，可以说是对传统虚拟机的一种颠覆，在过去的5年内火遍了世界各地，由最开始的Geek玩物，迅速发展成为公司开发部署扩展业务的利器，再加上Google对docker的热捧，推出了kubernetes这个神器来管理docker集群，docker在云计算方面跟openstack已势均力敌。然后两个社区并不是非要你死我活，双方都抱着尝试拥抱对方的态度，在共同努力提升云计算技术上努力着。

我们小组拟将采用openstack和docker结合的方式构建云平台，这其中包含两个重点：即openstack的容器化和openstack管理容器的能力，目前我们正在尝试容器化openstack。



图1 云平台整体构建目标

接下来将从各个方面记录总结云平台构建过程中遇到的难题和新的知识点。

## 分类

云计算的核心思想是将大量用网络连接的计算资源统一管理和调度，构成一个计算资源池面向用户按需服务。云平台技术可以从多个维度进行归类，从服务类型大致分为三类，即常见的IaaS（基础设施即服务）、PaaS（平台即服务）、SaaS（软件即服务），如图2。按照云平台的所有者划分，分为公有云、私有云、混合云。

* 公有云：公有云是面向大众提供计算资源的服务，公有云并非用户所有，由第三方提供云计算资源，用户通过互联网来使用资源。公有云的提供商有Amazon、微软、阿里云、腾讯云等。
* 私有云：私有云是企业传统数据中心的延伸和优化，是为一个客户单独使用而构建的，因而能对数据、安全性和服务质量的提供最有效的控制。私有云可以部署在企业数据中心的防火墙内，也可以将他们部署在一个安全的主机托管场所，私有云的核心属性是专有资源。
* 混合云：混合云融合了公有云和私有云，是近年来云计算的主要模式和发展方向。出于安全考虑，用户更愿意将数据存放在私有云中，但是同时又希望可以获得公有云的计算资源，在这种情况下混合云越来越多的被采用，它将公有云和私有云进行混合和匹配，以获得最佳的效果，这种个性化的解决方案，到达了既省钱又安全的目的。

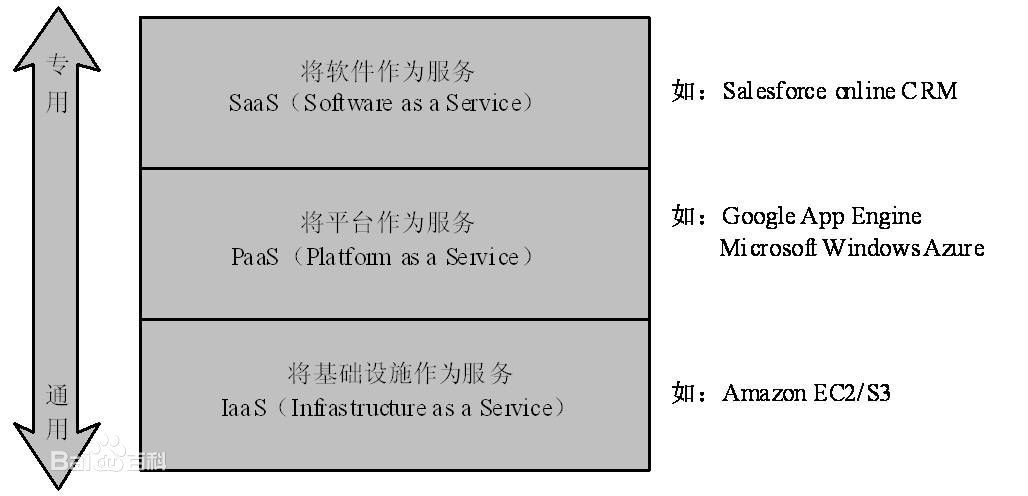


图2 云计算按服务类型分类

## 云平台构建之openstack入门

按照图2的划分，构建云平台首先需要实现的是IaaS，将传统资源统一构建成资源池。openstack是构建IaaS的不二之选，其社区活跃，国内外大公司譬如华为、IBM等都在社区有巨大贡献。Openstack是一整套开源软件项目的综合，它允许企业或服务提供者建立、运行自己的计算和存储设施。Openstack包括7个核心组件：Compute（计算）、Object Storage（对象存储）、Identity（身份认证）、Dashboard（仪表盘）、Block Storage（块存储）、Network（网络）、Image Service（镜像服务），他们之间的关系如图3和图4所示。

* Compute：代号为Nova，最核心的组件，管理计算资源。
* Identity：代号为Keystone，提供身份认证和授权的组件。
* Dashboard：代号为Horizon，为所有的Openstack服务提供了一个模块化的用户界面，使用这个界面，可以完成绝大多数操作，例如启动实例、分配ip，设置访问控制等。
* Image Service：代号为Glance，镜像的存储服务。
* Network：代号为Neutron，网络管理组件，实现了软件定义网络。
* Object Storage：代号为Swift，对象存储组件。
* Block Storage：代号为Cinder，提供块存储服务，实现云硬盘的热插拔功能。

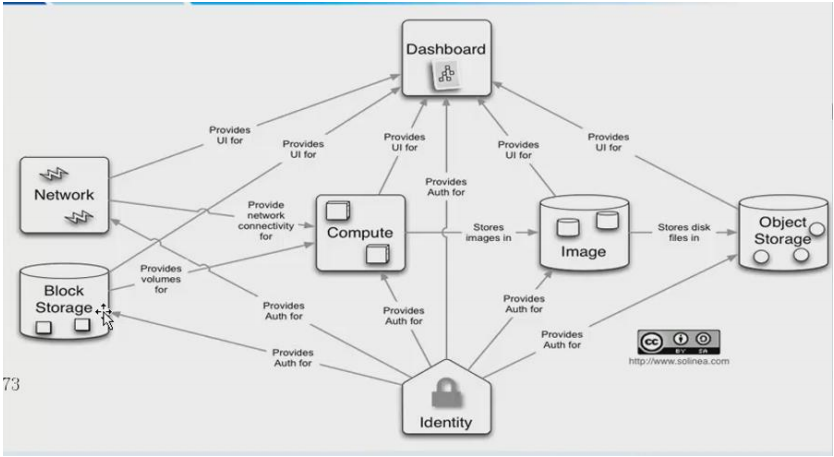


图3 openstack各个组件的关系

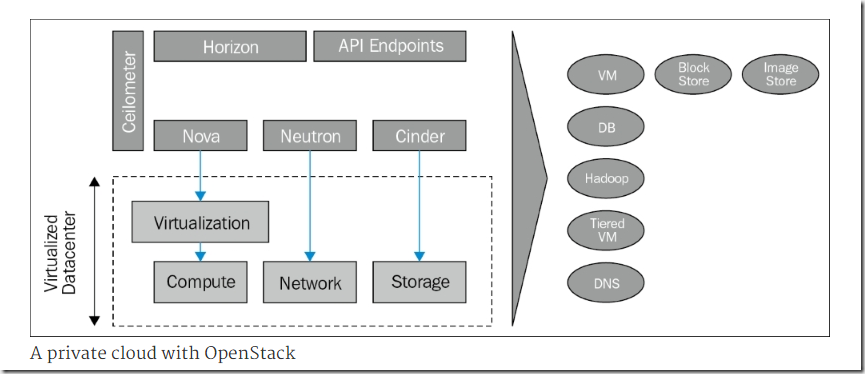


图4 使用openstack搭建的私有云

## 云平台之openstack搭建

部门提供3台主机搭建小规模的云计算平台，规划如图5：

图5 云计算平台构建规划

逻辑上，云计算平台由一个controller节点和三个compute节点构成。Controller节点中运行着各类基础组件，例如mysql、rabbitmq等，如果需要可以在controller中运行nova-compute，让其同时充当计算节点，compute节点中运行计算组件和网络组件。具体构建步骤详见https://www.cnblogs.com/kevingrace/p/5707003.html

## Openstack新建云主机流程

Openstack平台搭建完成之后，可以通过UI创建云主机，其内部流程如图6所示。

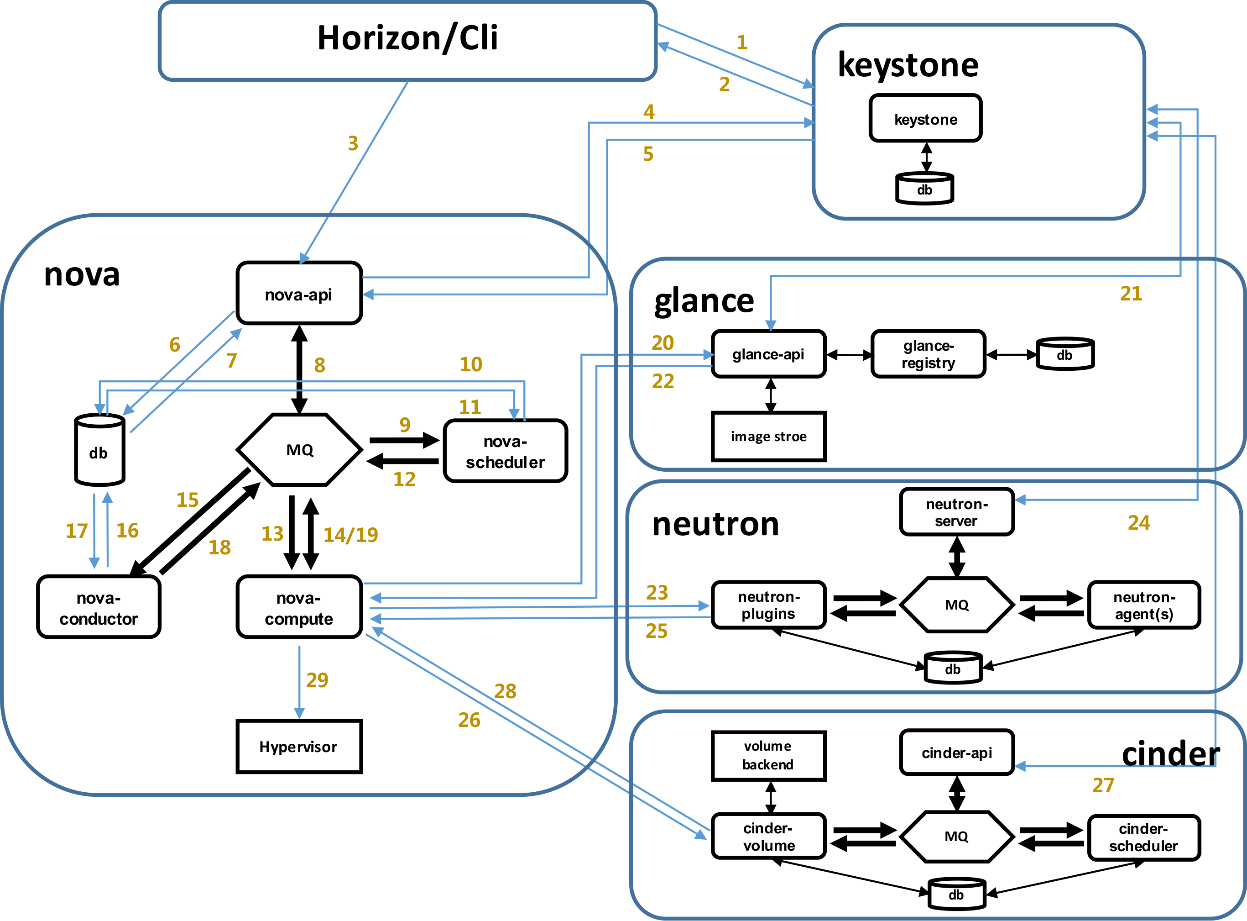


图6 openstack创建云主机流程

1. 界面或命令行通过RESTful API向keystone获取认证信息。
2. keystone通过用户请求认证信息，并生成auth-token返回给对应的认证请求。
3. 界面或命令行通过RESTful API向nova-api发送一个boot instance的请求（携带auth-token）。
4. nova-api接受请求后向keystone发送认证请求，查看token是否为有效用户和token。
5. keystone验证token是否有效，如有效则返回有效的认证和对应的角色（注：有些操作需要有角色权限才能操作）。
6. 通过认证后nova-api和数据库通讯。
7. 初始化新建虚拟机的数据库记录。
8. nova-api通过rpc.call向nova-scheduler请求是否有创建虚拟机的资源(Host ID)。
9. nova-scheduler进程侦听消息队列，获取nova-api的请求。
10. nova-scheduler通过查询nova数据库中计算资源的情况，并通过调度算法计算符合虚拟机创建需要的主机。
11. 对于有符合虚拟机创建的主机，nova-scheduler更新数据库中虚拟机对应的物理主机信息。
12. nova-scheduler通过rpc.cast向nova-compute发送对应的创建虚拟机请求的消息。
13. nova-compute会从对应的消息队列中获取创建虚拟机请求的消息。
14. nova-compute通过rpc.call向nova-conductor请求获取虚拟机消息。（Flavor）
15. nova-conductor从消息队队列中拿到nova-compute请求消息。
16. nova-conductor根据消息查询虚拟机对应的信息。
17. nova-conductor从数据库中获得虚拟机对应信息。
18. nova-conductor把虚拟机信息通过消息的方式发送到消息队列中。
19. nova-compute从对应的消息队列中获取虚拟机信息消息。
20. nova-compute通过keystone的RESTful API拿到认证的token，并通过HTTP请求glance-api获取创建虚拟机所需要镜像。
21. glance-api向keystone认证token是否有效，并返回验证结果。
22. token验证通过，nova-compute获得虚拟机镜像信息(URL)。
23. nova-compute通过keystone的RESTful API拿到认证的token，并通过HTTP请求neutron-server获取创建虚拟机所需要的网络信息。
24. neutron-server向keystone认证token是否有效，并返回验证结果。
25. token验证通过，nova-compute获得虚拟机网络信息。
26. nova-compute通过keystone的RESTful API拿到认证的token，并通过HTTP请求cinder-api获取创建虚拟机所需要的持久化存储信息。
27. cinder-api向keystone认证token是否有效，并返回验证结果。
28. token验证通过，nova-compute获得虚拟机持久化存储信息。
29. nova-compute根据instance的信息调用配置的虚拟化驱动来创建虚拟机。

## Openstack网络neutron解析

Neutron是openstack核心项目之一，提供云计算环境下的虚拟网络功能。Neutron管理openstack环境中所有虚拟网络基础设施（VNI），物理网络基础设施（PNI）的接入层。管理员创建和管理neutron外部网络，是租户虚拟机与互联网信息交互的桥梁，具体来说，外部网络会分出一个子网，它是一组在互联网上可寻址的IP地址。一般情况下，外部网络只有一个，且由管理员创建。租户虚拟机创建和管理租户网络，每个网络可以根据需要划分成多个子网。各个子网通过路由器与neutron外部网络（图7中子网A）连接。路由器的gateway网管端连接外部网络的子网，interfaces接口端有多个，连接租户网络的子网。路由器及interface接口端连接的网络都是由租户根据需要自助创建，管理者只创建和管理neutron外部网络部分。

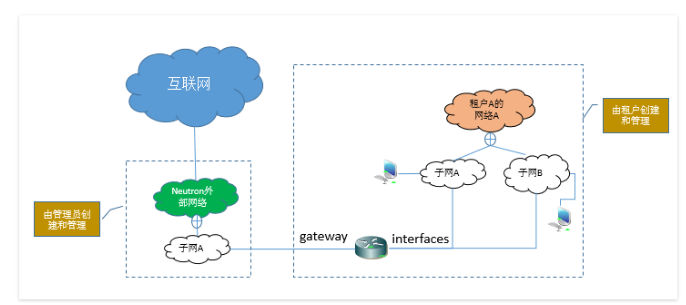


图7 openstack网络结构图

总结来看，创建一个neutron网络的过程如下：

1. 管理员拿到一组可以在互联网上寻址的IP地址，并创建一个外部网络和子网
2. 租户创建一个网络和子网
3. 租户创建一个路由器并且连接租户子网和外部网络
4. 租户创建虚拟机

Openstack网络分为内部网络（管理网络，数据网络）和外部网络（外部网络，API网络）。如图8，他们的作用如下：

1. 管理网络：用于openstack各组件之间的内部通信。
2. 数据网络：用于云部署中虚拟数据之间的通信。
3. 外部网络：公用网络，外部或internet可以访问的网络。
4. API网络：暴露所有的openstack APIs，包括openstack网络给租户们。

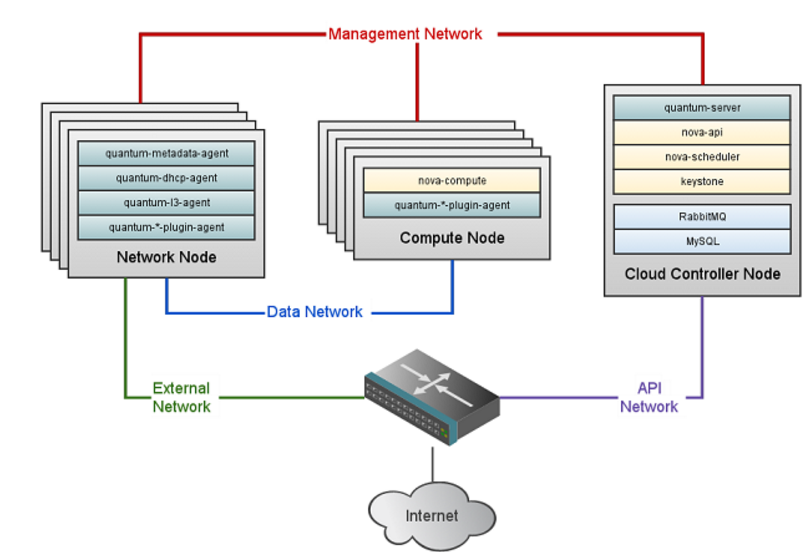


图8 openstack网络结构图

Openstack网络模式有如下几种：

* Flat模式：网桥模式，所有的都需手工配置，属于平面网络，所有势力都属于同一网络
* Flat DHCP：网桥模式，在网关出单独取一个DHCP进程，可以辅助用户进行网络配置。
* VLAN：为每个不同的租户设置了不同的虚拟子网，在这个虚拟子网中，用户可以有自己的ip。
* VXLAN：类似VLAN，但更灵活、功能更加强大。

TIP1：VXLAN与VLAN相比具有以下几个优势：

1. 支持更多的二层网段。VLAN使用12bit标记VLAN ID，最多支持4096个VLAN，这对于大型云部署会成为瓶颈。VXLAN的ID则使用24bit进行标记，支持更多网段。
2. 能更好地利用已有的网络路径。VLAN使用Spanning Tree Protocol避免环路，这会导致有一般的网络路径被block掉。VXLAN的数据包是封装到UDP通过三层传输和转发的，可以使用所有的路径。
3. 避免物理交换机MAC表耗尽。由于采用隧道机制，TOR（Top On Rack）交换机无需在MAC表中记录虚拟机的信息。

TIP2：VXLAN包的转发如图9所示：

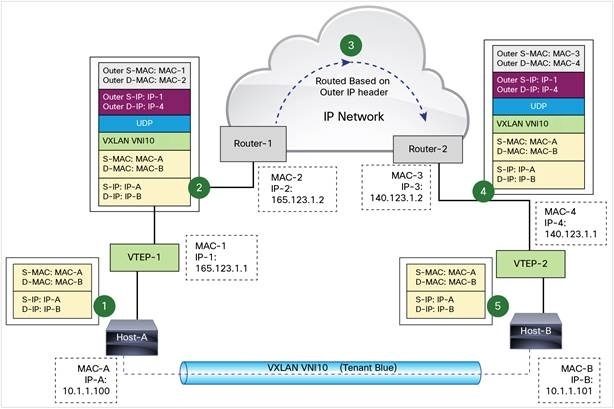


图9 VXLAN数据转发示意图

图中 Host-A 和 Host-B 位于 VNI 10 的 VXLAN，通过 VTEP-1 和 VTEP-2 之间建立的 VXLAN 隧道通信。

数据传输过程如下：

1. Host-A 向 Host-B 发送数据时，Host-B 的 MAC 和 IP 作为数据包的目标 MAC 和 IP，Host-A 的 MAC 作为数据包的源 MAC 和 IP，然后通过 VTEP-1 将数据发送出去。
2. VTEP-1 从自己维护的映射表中找到 MAC-B 对应的 VTEP-2，然后执行 VXLAN 封装，加上 VXLAN 头，UDP 头，以及外层 IP 和 MAC 头。此时的外层 IP 头，目标地址为 VTEP-2 的 IP，源地址为 VTEP-1 的 IP。同时由于下一跳是 Router-1，所以外层 MAC 头中目标地址为 Router-1 的 MAC。
3. 数据包从 VTEP-1 发送出去后，外部网络的路由器会依据外层 IP 头进行包路由，最后到达与 VTEP-2 连接的路由器 Router-2。
4. Router-2 将数据包发送给 VTEP-2。VTEP-2 负责解封数据包，依次去掉外层 MAC 头，外层 IP 头，UDP 头 和 VXLAN 头。
5. VTEP-2 依据目标 MAC 地址将数据包发送给 Host-B。

TIP3：linux对vxlan的支持情况

VTEP可以由专有硬件来实现，也可以使用纯软件来实现，目前比较成熟的VTEP软件包括：带VXLAN内核模块的linux、open vSwitch。Linux对vxlan的支持如图10所示：

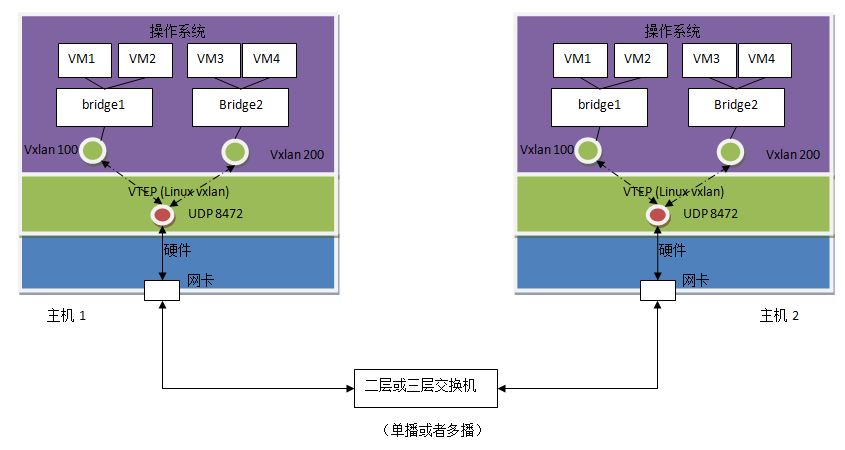


图10 linux对vxlan的支持

实现步骤：

1. Linux vxlan创建一个UDP Socket，默认在8472端口监听。
2. Linux vxlan在UDP Socket上接收到vxlan包后，解包，然后根据其中的vxlan ID将它转给某个vxlan interface，然后再通过它所连接的linux bridge转给虚拟机。
3. Linux vxlan在收到虚拟机发来的数据包后，将其封装为多播UDP包，从网卡发出。

前面大部分在总结通用网络知识，openstack网络模块neutron由哪些部分组成的呢？如图11所示

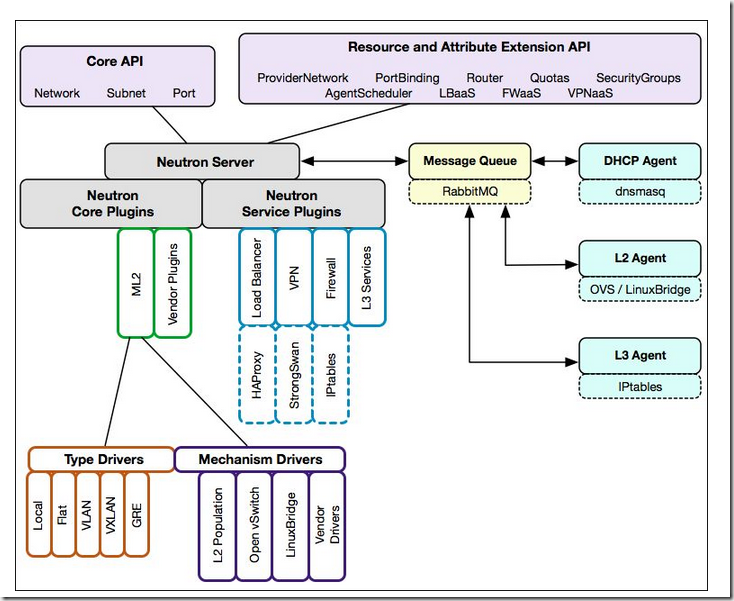


图11 neutron的内部结构

在实际云平台搭建过程中，主要配置了ML2插件，使用linuxBridge来实现了flat模式和vxlan模式的网络。