

- 云计算

- OpenStack

- Nuetron

- Neutron 概述
 - Neutron 功能
 - Neutron 管理的网络资源
 - Neutron 架构
 - Neutron 分层结构
 - Neutron ML2 解决 core plugin 的问题
 - Neutron ML2 的架构
 - Neutron Service Plugin/Agent
 - Neutron 总结

云计算

OpenStack

Nuetron

Neutron 概述

SDN -- (software-defined networking) 软件定义网络所具有的灵活性和自动化优势使其成为云时代网络管理的主流。

Neutron 的设计目标是实现“网络即服务（Networking as a Service）”。为了达到这一目标，在设计上遵循了基于 SDN 实现网络虚拟化的原则，在实现上充分利用了 Linux 系统上的各种网络相关的技术。

Neutron 功能

Neutron 为整个 OpenStack 环境提供网络支持，包括二层交换，三层路由，负载均衡，防火墙和 VPN 等。Neutron 提供了一个灵活的框架，通过配置，无论是开源还是商业软件都可以被用来实现这些功能。

- 二层交换 Switching

- Nova 的 Instance 是通过虚拟交换机连接到虚拟二层网络的。
 - Neutron 支持多种虚拟交换机，包括 Linux 原生的 Linux Bridge 和 Open vSwitch。
 - 利用 Linux Bridge 和 OVS，Neutron 除了可以创建传统的 VLAN 网络，还可以创建基于隧道技术的 Overlay 网络，比如 VxLAN 和 GRE (Linux Bridge 目前只支持 VxLAN)。

- 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。router 通过 IP forwarding, iptables 等技术来实现路由和 NAT。

- 负载均衡 Load Balancing

- 提供了将负载分发到多个 instance 的能力。LBaaS 支持多种负载均衡产品和方案，不同的实现以 Plugin 的形式集成到 Neutron，目前默认的 Plugin 是 HAProxy。

- 防火墙 Firewalling
 - Security Group 通过 iptables 限制进出 instance 的网络包。
 - Firewall-as-a-Service FWaaS, 限制进出虚拟路由器的网络包, 也是通过 iptables 实现。

Open vSwitch -- (OVS) 是一个开源的虚拟交换机, 它支持标准的管理接口和协议。

Neutron 管理的网络资源

- network
 - local 网络中的 instance 只能与位于同一节点上同一网络的 instance 通信, local 网络主要用于单机测试。
 - flat 网络是无 vlan tagging 的网络。flat 网络中的 instance 能与位于同一网络的 instance 通信, 并且可以跨多个节点。
 - VLAN 网络是具有 802.1q tagging 的网络。vlan 是一个二层的广播域, 同一 vlan 中的 instance 可以通信, 不同 vlan 只能通过 router 通信。vlan 网络可以跨节点, 是应用最广泛的网络类型。
 - VxLAN 基于隧道技术的 overlay 网络。vxlan 网络通过唯一的 segmentation ID (也叫 VNI) 与其他 vxlan 网络区分。vxlan 中数据包会通过 VNI 封装成 UDP 包进行传输。因为二层的包通过封装在三层传输, 能够克服 vlan 和物理网络基础设施的限制。
 - GRE 与 vxlan 类似的一种 overlay 网络。主要区别在于使用 IP 包而非 UDP 进行封装。不同 network 之间在二层上是隔离的。

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

network 与 Project 之间是 1对多 关系。

- subnet
 - subnet 是一个 IPv4 或者 IPv6 地址段。instance 的 IP 从 subnet 中分配。每个 subnet 需要定义 IP 地址的范围和掩码。
 - subnet 与 network 是 1对多 关系。一个 subnet 只能属于某个 network; 一个 network 可以有多个 subnet, 这些 subnet 可以是不同的 IP 段, 但不能重叠。

```
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"}
```

```
network A      subnet A-a: 10.10.1.0/24 {"start": "10.10.1.1", "end": "10.10.1.50"}  
network B      subnet B-a: 10.10.1.0/24 {"start": "10.10.1.1", "end": "10.10.1.50"}
```

- port
 - port 可以看做虚拟交换机上的一个端口。port 上定义了 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。
 - port 与 subnet 是 1对多 关系。一个 port 必须属于某个 subnet; 一个 subnet 可以有多个 port

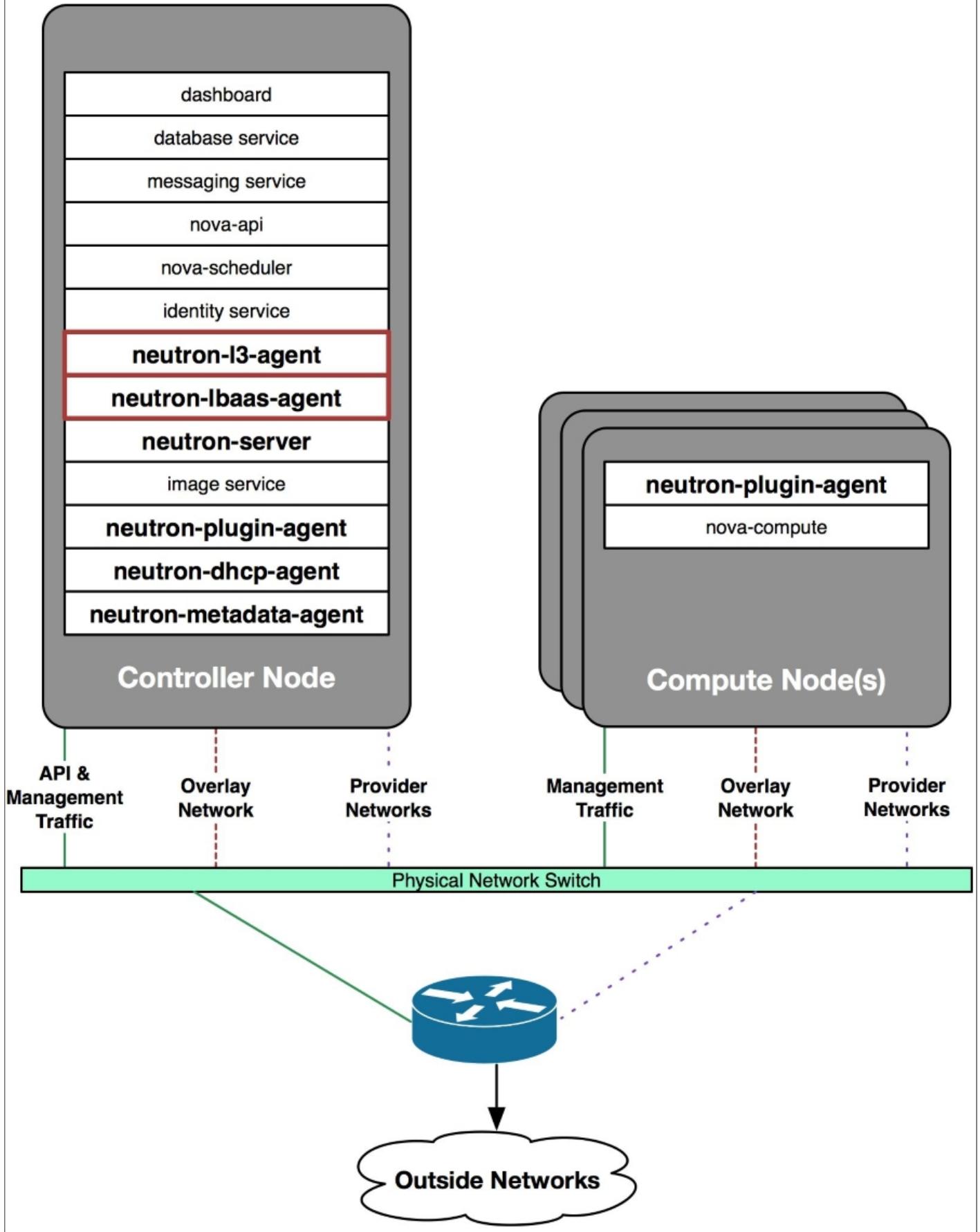
OpenStack_Framework_Neutron

- Neutron Server 对外提供 OpenStack 网络 API，接收请求，并调用 Plugin 处理请求。
- Queue Neutron Server, Plugin 和 Agent 之间通过 Messaging Queue 通信和调用。
- Neutron Plugin(s) 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。
- Neutron Agent 处理 Plugin 的请求，负责在 network provider 上真正实现各种网络功能。
- network provider 提供网络服务的虚拟或物理网络设备，例如 Linux Bridge, Open vSwitch 或者其他支持 Neutron 的物理交换机。
- Neutron Database 存放 OpenStack 的网络状态信息，包括 Network, Subnet, Port, Router 等。

支持分布式部署，获得足够的扩展性，架构非常灵活，层次较多，为了支持各种现有或者将来会出现的优秀网络技术。

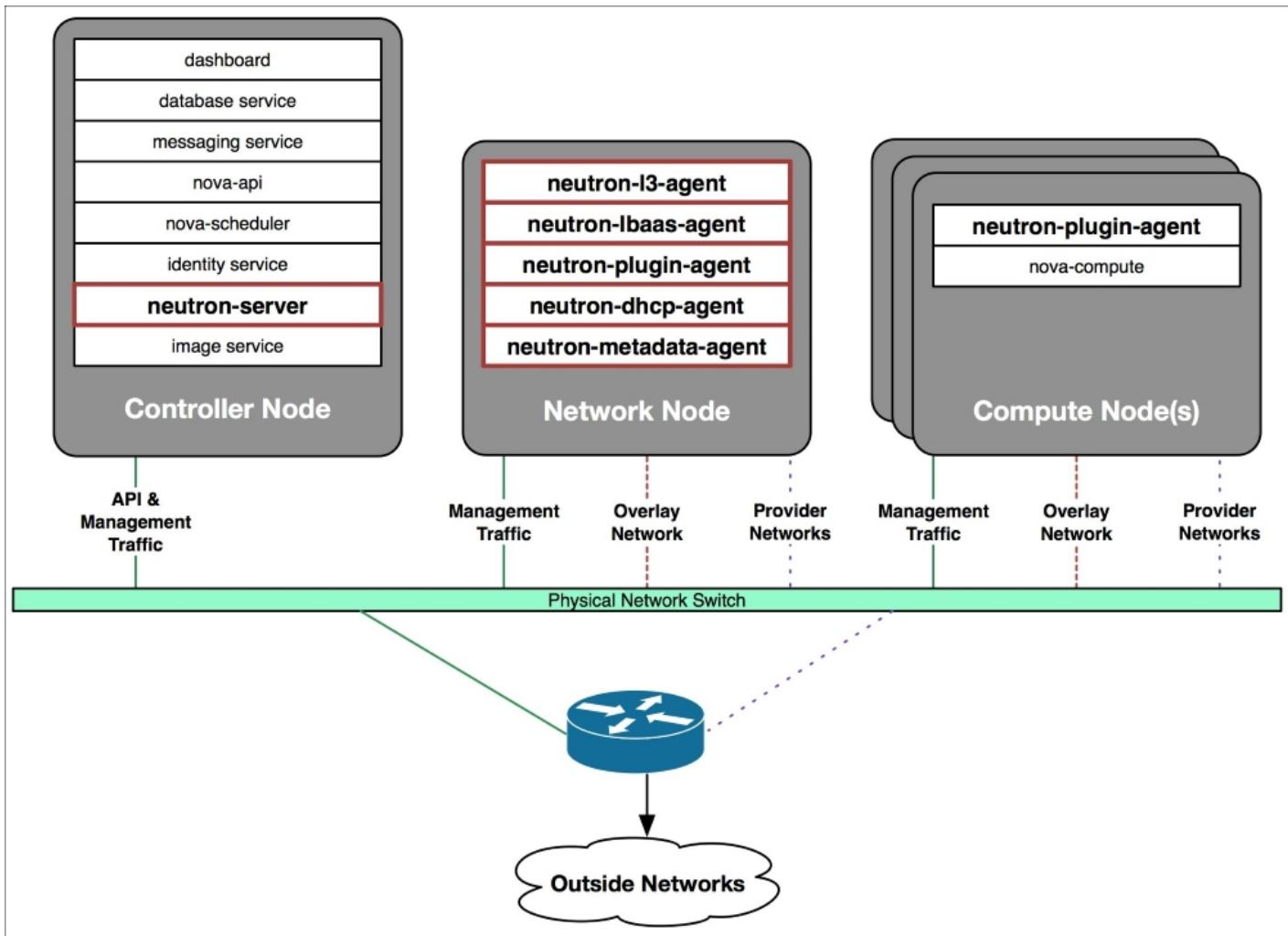
| core plugin | agent | service plugin | agent |
|---------------------------|---------------------|-------------------------------------|-------|
| network subnet port | linux bridge OVS | routing firewall load balance | |

方案A：控制节点 + 计算节点



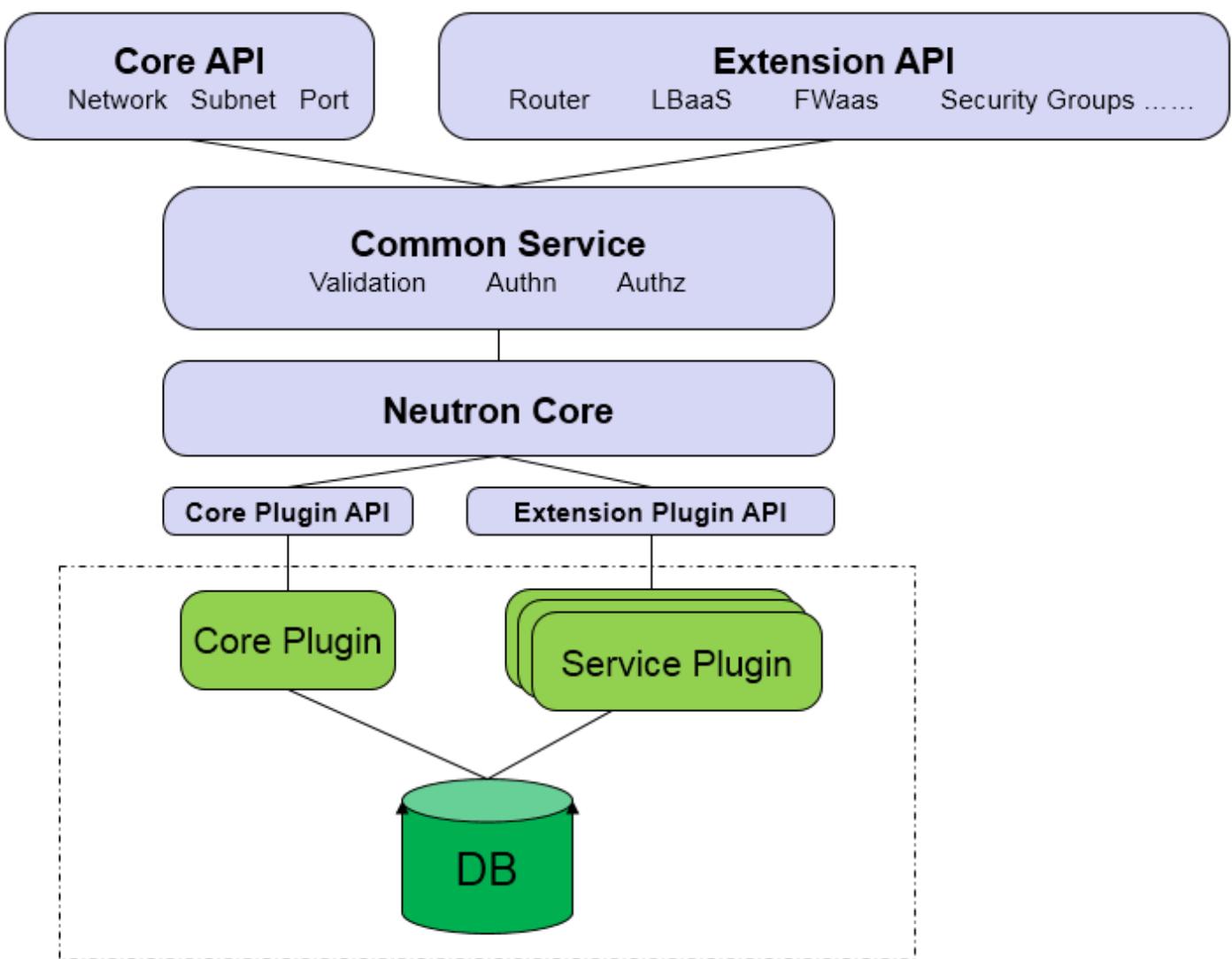
| 控制节点 | 计算节点 |
|---|-------------------|
| neutron server core plugin-agent service plugin-agent | core plugin-agent |

方案B：控制节点 + 网络节点 + 计算节点

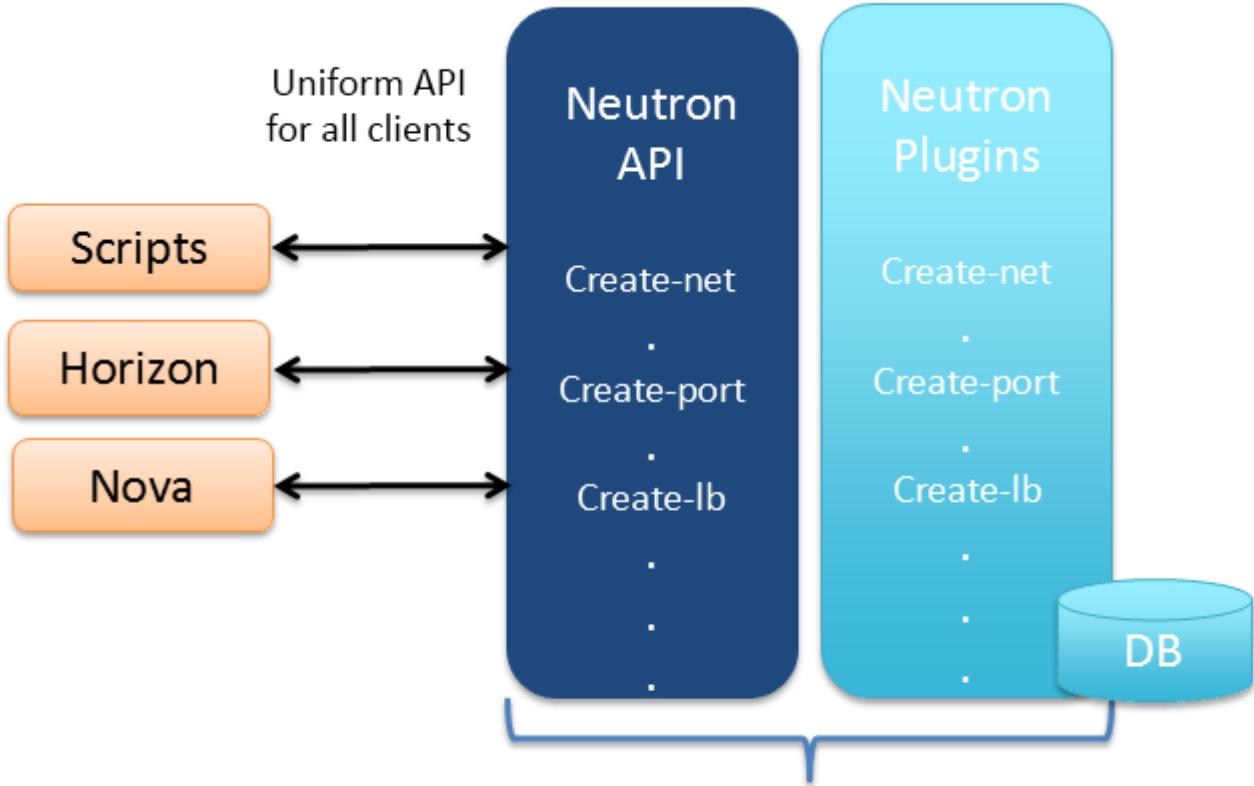


| 控制节点 | 网络节点 | 计算节点 |
|----------------|---|-------------------|
| neutron server | core plugin-agent service plugin-agent | core plugin-agent |

Neutron 分层结构



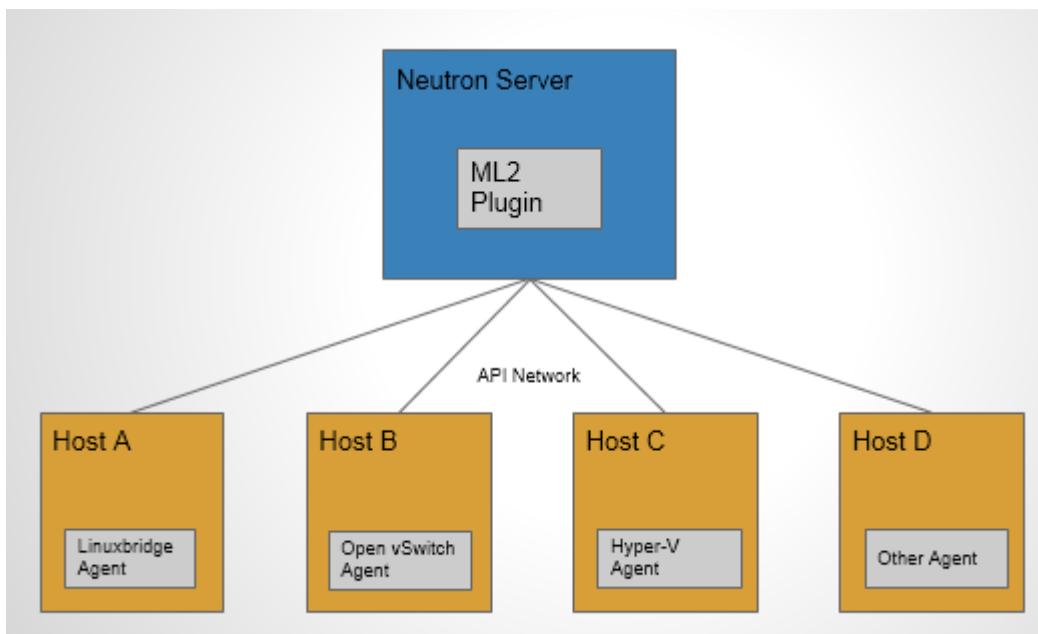
- Core API -- 对外提供管理 network, subnet 和 port 的 RESTful API。
- Extension API -- 对外提供管理 router, load balance, firewall 等资源的 RESTful API。
- Common Service -- 认证和校验 API 请求。
- Neutron Core -- Neutron server 的核心处理程序，通过调用相应的 Plugin 处理请求。
- Core Plugin API -- 定义了 Core Plgin 的抽象功能集合，Neutron Core 通过该 API 调用相应的 Core Plgin。
- Extension Plugin API -- 定义了 Service Plgin 的抽象功能集合，Neutron Core 通过该 API 调用相应的 Service Plgin。
- Core Plugin -- 实现了 Core Plugin API，在数据库中维护 network, subnet 和 port 的状态，并负责调用相应的 agent 在 network provider 上执行相关操作，比如创建 network。
- 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 上执行相关操作，比如创建 router。



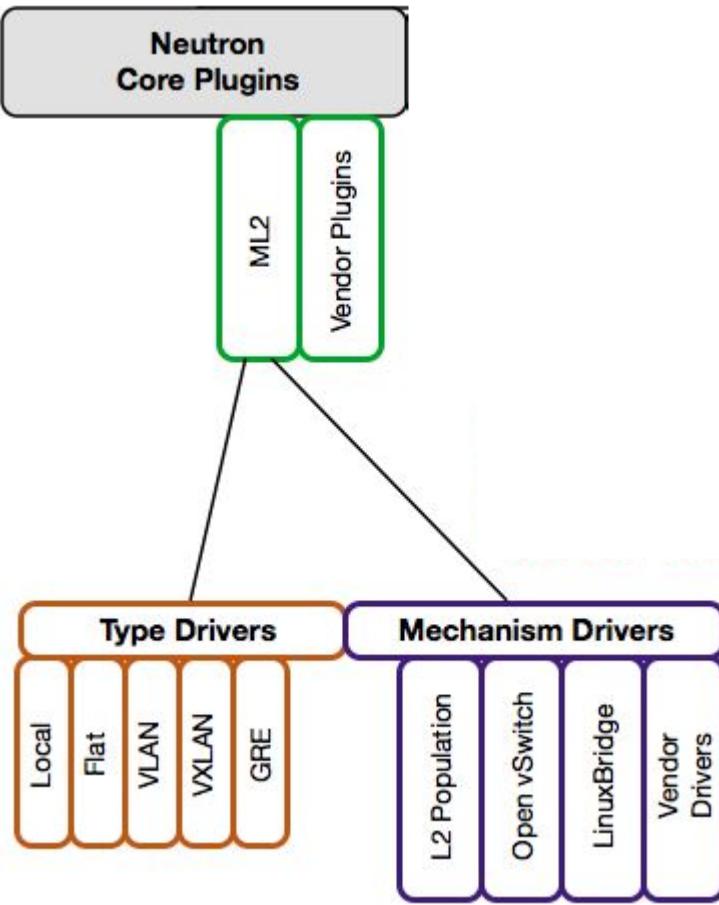
Neutron Server = API + Plugins

Neutron ML2 解决 core plugin 的问题

- Core plugin 负责管理和维护 Neutron 的 network, subnet 和 port 的状态信息，这些信息是全局的，只需要也只能由一个 core plugin 管理。
- 所有传统的 core plugin 都需要编写大量重复和类似的数据库访问的代码，大大增加了 plugin 开发和维护的工作量。



- 如上图所示，采用 ML2 plugin 后，可以在不同节点上分别部署 linux bridge agent, open vswitch agent, hyper-v agent 以及其他第三方 agent。
- ML2 不但支持异构部署方案，同时能够与现有的 agent 无缝集成：以前用的 agent 不需要变，只需要将 Neutron server 上的传统 core plugin 替换为 ML2。
- 有了 ML2，要支持新的 network provider 就变得简单多了：无需从头开发 core plugin，只需要开发相应的 mechanism driver，大大减少了要编写和维护的代码。



- Type Driver
 - Neutron 支持的每一种网络类型都有一个对应的 ML2 type driver。
 - type driver 负责维护网络类型的状态，执行验证，创建网络等。
 - ML2 支持的网络类型包括 local, flat, vlan, vxlan 和 gre。
- Mechanism Driver
 - Neutron 支持的每一种网络机制都有一个对应的 ML2 mechanism driver。
 - mechanism driver 负责获取由 type driver 维护的网络状态，并确保在相应的网络设备（物理或虚拟）上正确实现这些状态。

type 和 mechanism 都太抽象，现在我们举一个具体的例子：

type driver 为 vlan，mechanism driver 为 Linux bridge，我们要完成的操作是创建 network vlan100，那么：

vlan type driver 会确保将 vlan100 的信息保存到 Neutron 数据库中，包括 network 的名称，vlan ID 等。

linux bridge mechanism driver 会确保各节点上的 linux brige agent 在物理网卡上创建 ID 为 100 的 vlan 设备和 brige 设备，并将两者进行桥接。

mechanism driver 有三种类型：

- Agent-based -- 包括 linux bridge, open vswitch 等。
- Controller-based -- 包括 OpenDaylight, VMWare NSX 等。
- 基于物理交换机 -- 包括 Cisco Nexus, Arista, Mellanox 等。

比如前面那个例子如果换成 Cisco 的 mechanism driver，则会在 Cisco 物理交换机的指定 trunk 端口上添加 vlan100。

linux bridge 和 open vswitch 的 ML2 mechanism driver 其作用是配置各节点上的虚拟交换机。

linux bridge driver 支持的 type 包括 local, flat, vlan, and vxlan。

open vswitch driver 除了这 4 种 type 还支持 gre。

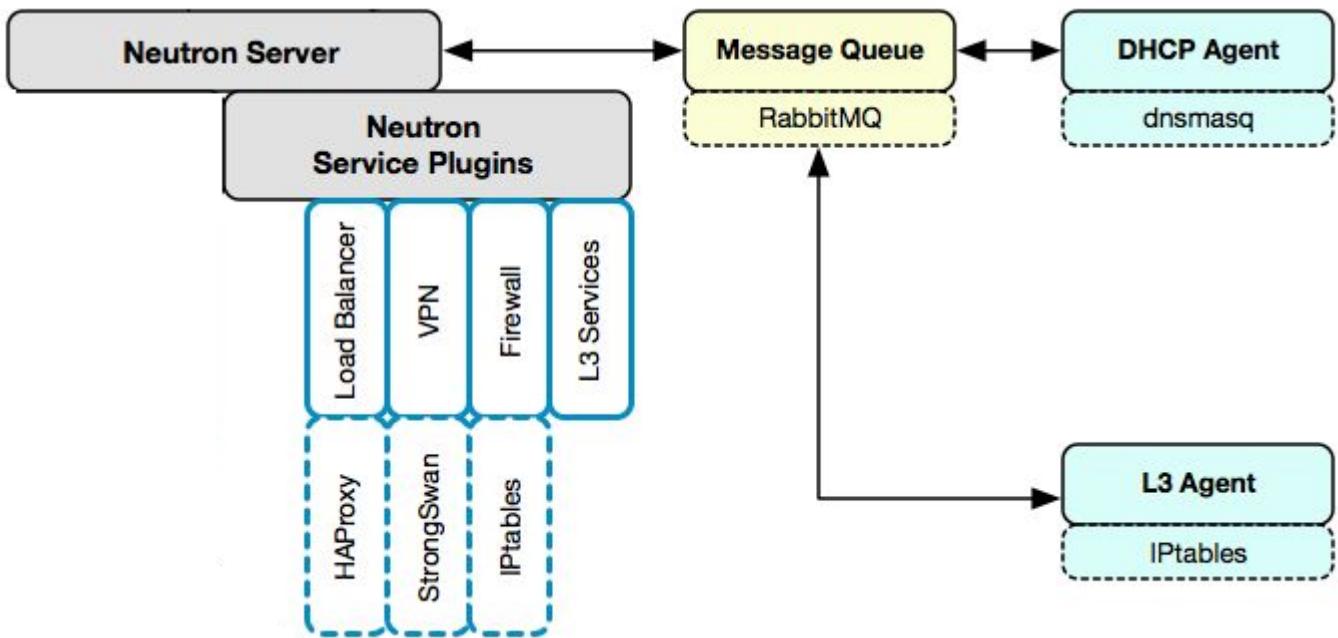
L2 population driver 作用是优化和限制 overlay 网络中的广播流量。

vxlan 和 gre 都属于 overlay 网络。

ML2 core plugin 已经成为 OpenStack Neutron 的首选 plugin，本教程后面会讨论如何在实验环境中配置 ML2 的各种 type 和 mechanism。

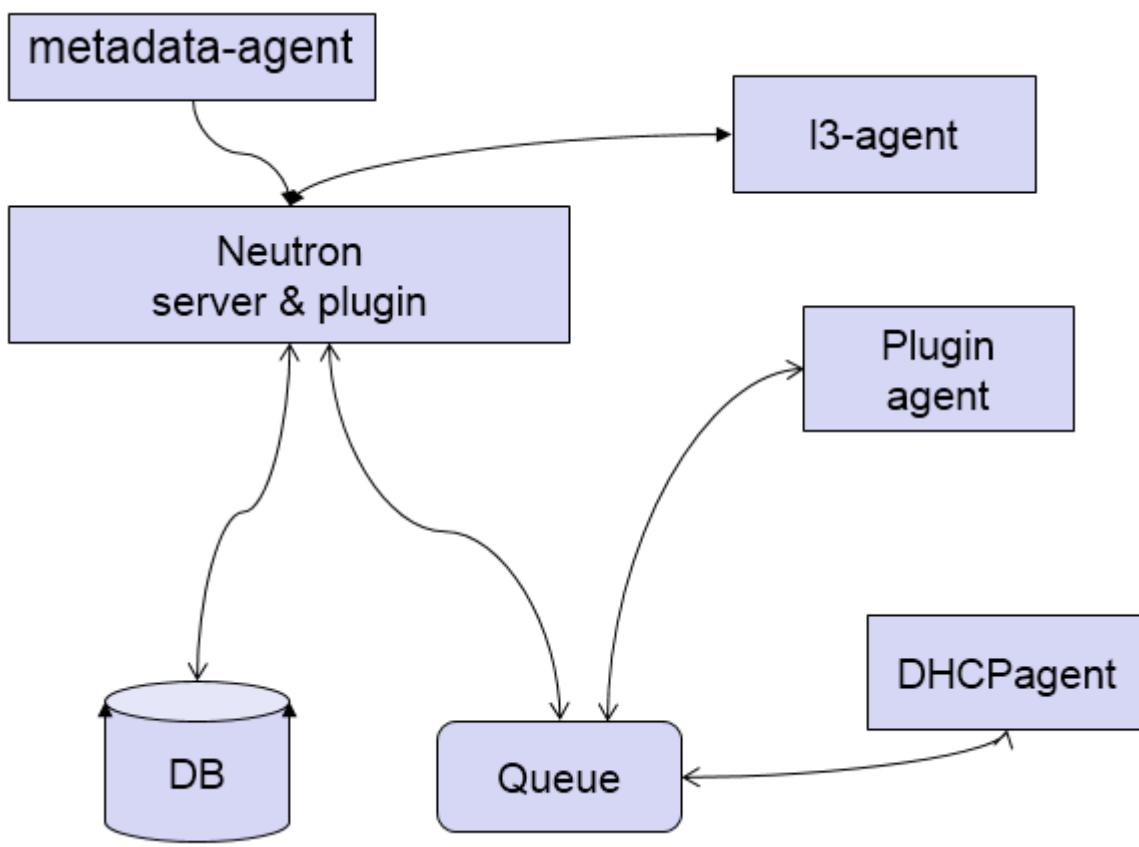
Neutron Service Plugin/Agent

- Core Plugin/Agent -- 负责管理核心实体：net, subnet 和 port。将 instance 连接到 OpenStack layer 2 虚拟网络
- Service Plugin/Agent -- 负责管理更高级的网络服务扩展功能，route, load balance, firewall 等



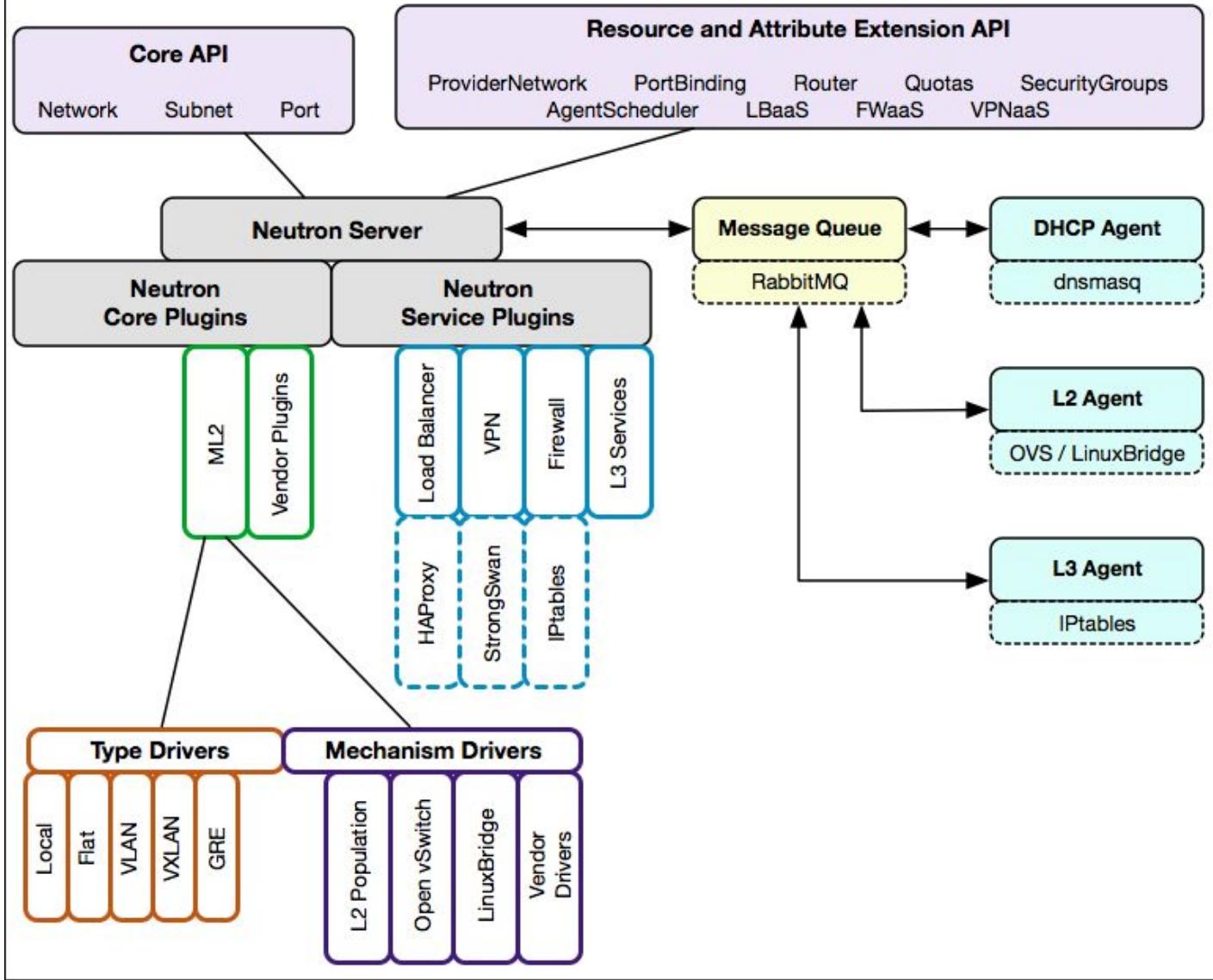
- DHCP -- dhcp agent 通过 dnsmasq 为 instance 提供 dhcp 服务。
- Routing -- l3 agent 可以为 project (租户) 创建 router，提供 Neutron subnet 之间的路由服务。路由功能默认通过 IPtables 实现。
- Firewall -- l3 agent 可以在 router 上配置防火墙策略，提供网络安全防护。
- Security Group -- 通过 IPtables 实现。
 - Firewall 安全策略位于 router，保护的是某个 project 的所有 network。
 - Security Group 安全策略位于 instance，保护的是单个 instance。
- Load Balance -- Neutron 默认通过 HAProxy 为 project 中的多个 instance 提供 load balance 服务。

Neutron 总结



Neutron 采用的是分布式架构，包括 Neutron Server、各种 plugin/agent、database 和 message queue。

- Neutron server 接收 api 请求。
- plugin/agent 实现请求。
- database 保存 neutron 网络状态。
- message queue 实现组件之间通信。



- Neutron 通过 plugin 和 agent 提供的网络服务。
- plugin 位于 Neutron server，包括 core plugin 和 service plugin。
- agent 位于各个节点，负责实现网络服务。
- core plugin 提供 L2 功能，ML2 是推荐的 plugin。
- 使用最广泛的 L2 agent 是 linux bridge 和 open vswitch。
- service plugin 和 agent 提供扩展功能，包括 dhcp, routing, load balance, firewall, vpn 等。