

Neutron 的架构是非常开放的，可以支持多种 network provider，只要遵循一定的设计原则和规范。本节我们将开始讨论这个主题。  
  
先讨论一个简单的场景：在 Neutorn 中使用 linux bridge 这一种 network provider。

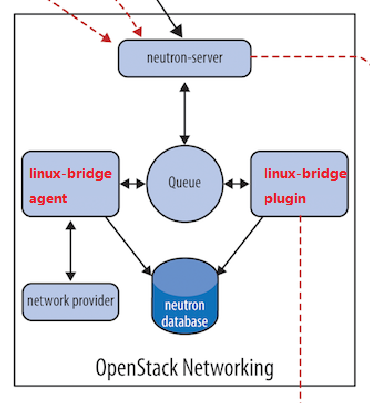
根据我们上一节讨论的 Neutron Server 的分层模型，我们需要实现两个东西：linux bridge core plugin 和 linux bridge agent。

**linux bridge core plugin**

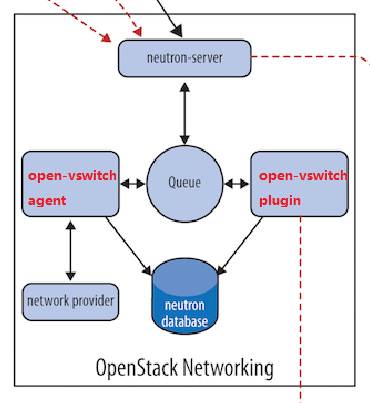
1. 与 neutron server 一起运行。
2. 实现了 core plugin API。
3. 负责维护数据库信息。
4. 通知 linux bridge agent 实现具体的网络功能。

**linux bridge agent**

1. 在计算节点和网络节点（或控制节点）上运行。
2. 接收来自 plugin 的请求。
3. 通过配置本节点上的 linux bridge 实现 neutron 网络功能。



同样的道理，如果要支持 open vswitch，只需要实现 open vswitch plugin 和 open vswitch agent。



由此可见：Neutron 可以通过开发不同的 plugin 和 agent 支持不同的网络技术。这是一种相当开放的架构。

不过随着支持的 network provider 数量的增加，开发人员发现了两个突出的问题：

1. 只能在 OpenStack 中使用一种 core plugin，多种 network provider 无法共存。
2. 不同 plugin 之间存在大量重复代码，开发新的 plugin 工作量大。

下一节将深入讨论这两个问题的成因以及解决方案。