# OpenDaylight Helium版本与1.0版本比较

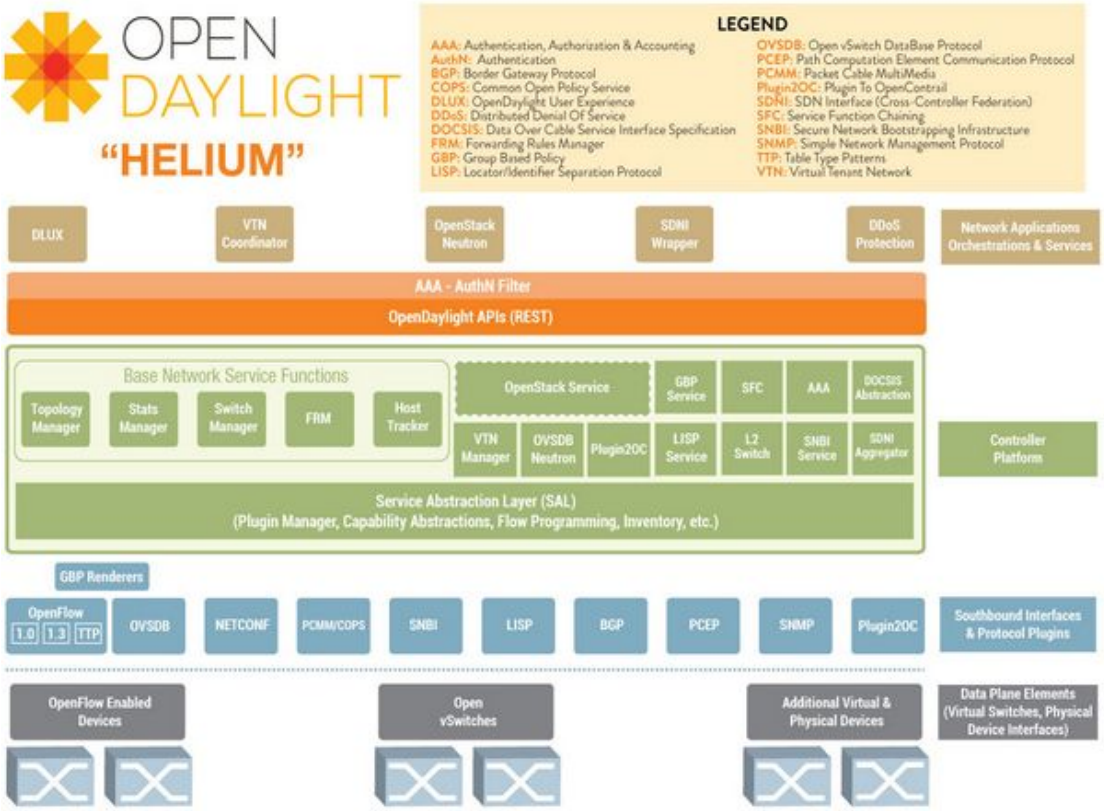
作者:周洁　　QQ：2647858980

OpenDaylight SDN 研究群

@南京-周洁

# 1 OpenDaylight Helium 版本

业界组织 OpenDaylight 联盟最近发布了其开源 SDN 软件的 2.0 版本，即 Helium（氦）版本，该 2.0 版本加入了一些有关 Helium 和 OpenDaylight 未来发展方向的新理念，相比较于 1.0 版本，致力于研发出“开放、易懂”的 SDN 解决方案。且更多的厂商新加入对 OpenDaylight 项目的支持，印证了 OpenDaylight 目前的发展和进步。




## 1.1 Helium 版本变化

Helium 版本相较于 1.0 氢版本的一些变化：

1.OpenDaylight 结合 OpenStack。在 Helium 版本中最明显的特征是 OpenDaylight 与 OpenStack 之间整合的方式，其中包括在 Open vSwitch 数据库整合项目中一些明显的改善，以及高级的 OpenStack 特征(如安全小组、分布式虚拟路由器和负载均衡即服务)的技术预览。

2.组策略插件(Group Policy Plugin)。能够在以策略为重点的北向 API 中提供比 Affinity 更好的体验，提供多样性选择，可以实行单一的模型也可以同时实行两种模型。

3.DLUX(openDayLight User eXperience)。通过更多拖放功能的图形用户界面增强用户体验，更易拖拽，且设备图形显示更美观。

4.NFV（网络功能虚拟化）。推动 SDN 与 NFV 的发展，重新构建网络，且新增 11 个新的协议、应用及技术到 SDN 和 NFV 平台中，使之更灵活的、互操作性更可用。

5.变化最大的是配备了一个新的用户界面和一个更简单并可定制的功能安装过程，并使用了 Apache Karaf 容器，提供开发者更方便测试和管理 SDN 生产环境的平台。通过 Karaf 容易后，ODL 的启动方式也有了很大的改变，如，进入 OpenDaylight 目录，切换到 bin 目录下，执行启动命令：

```
./karaf  或 ./karaf  clean
```

启动成功后安装各功能模块也较简单，直接通过 feature 命令进行安装，如安装 L2switch 功能：

```
feature:install odl-l2switch-switch
```

安装模块简单易操作。

新的用户界面显示相比较 1.0 版本完全变化，直接将 Nodes、YangUI、Topology、Network、Connection manager 、Flows 等模块功能显示，更显直观，如下图部分功能显示：


6. 更高的可用性，以及加强和增加新的协议，如 OpenFlow 的表格型模式、PacketCable 多媒体、应用程序的政策框架和工具、服务功能链接等。

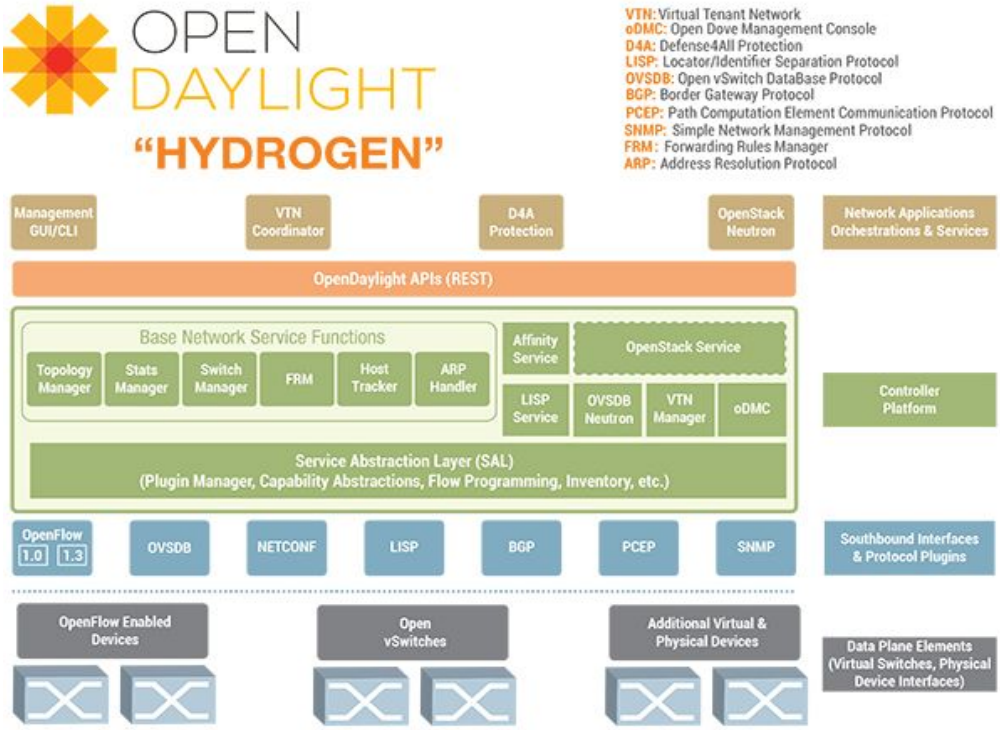
7. 安全性以及授权功能方面有所增强。拥有更出色的集群化故障转移机制，新的 Secure Network Bootstrapping Infrastructure（即安全网络引导基础设施，简称 SNBI）允许用户对一整套控制器与网络设备安全集进行定义与启用。

8. 利用 BGP 实现的控制器交叉联合机制在 SDNi（软件定义网络接口）在项目中同样得以实现。

9.模型驱动的服务抽象层在 OpenDaylight SDN 中扮演一个关键角色，其可以通过自动化减少人类的干预和人为错误。与此同时，自动化还可以帮助为南向和北向接口提供更好的一致性。

# 2 OpenDaylight 氢版本

OpenDaylight 氢版本提供了三种不同的版本，可以帮助用户广泛地学习，并尽可能快地运行。OpenDaylight 使用的 1.0 氢版本，主要是 AD-SAL 模型以及 release 版本的 Yang 工具相关的 MD-SAL 模型、OF1.0 协议的支持以及 SNMP4SDN 项目。基本上弄清楚 ODL 1.0 版本控制器与交换机等通信的流程，具体谈一下此版本的优缺点以及怎样研究学习。OpenDaylight 的架构纵览，可看出 ODL 是一个可插拔的控制器平台，它提供北向 API 和南向服务抽象层(SAL)。




## 2.1 OpenDaylight 1.0 版本优点

在主要的设计原则和开源项目方面，OpenDaylight 1.0 版本的主要优点是：

1.OSGi 体系结构。采用 OSGi 体系结构，可对功能进行隔离，可以动态的加载和卸载，解决了可扩展、热部署等问题，实现了一个优雅、完整和动态的组件模型,应用程序（Bundle）无需重新引导可以动态地被远程安装、启动、升级和卸载，给使用者提供了方便。框架分为模块层、生命周期层和服务层三个层次。
2.SAL(服务抽象层)。整个架构中引入了服务抽象层，使得北向 REST API 和南向接口之间的调用相互隔离，支持南向插件通过各种协议(包括 OpenFlow)与网络设备通信，并能够像整合模块一样整合控制器应用程序，以及一组提供控制器功能的 REST API。
3.MD(Model Drive)。使用 Yang 工具，使用业务驱动模型工具来设计接口、实现业务功能，根据 yang 文件，Yang 工具可直接生成业务管理的“骨架”，使开发者真正专注于具体业务。
4.Infinispan（分布式内存数据网格）。实现数据的存储、查找及监听，用开源的数据网格平

台实现 Controller 的集群。它公开了一个简单的数据结构（一个 Cache）来存储对象，可以本地模式下运行，但其最大的特点在于支持分布式。
5.Netty。南向使用 Netty 来管理底层的并发 IO，监听消息服务。

## 2.2 OpenDaylight 1.0 版本一些缺点

OpenDaylight 1.0 版本存在的一些问题：

1.版本演进过程没有清晰的说明，且存在很多废弃的代码，使开发者耗费大量精力，二次开发较困难。

2.版本功能较多，没有如何定制和研究的相关说明，整个 ODL 给人很难研究和很茫然的感觉。

3.官方提供的文档不够完善，且文档较少，看代码等分析非常困难。

4.相关的测试环境、工具等还不够成熟，需待提升和完善。

5.基本上还是处于研究阶段，没有基于此平台的原型或者示例，很难直观的感受 SDN。

# 3 总结

OpenDaylight 氦版本在 1.0 氢版本的基础上提升优化了很多，提供开发者更方便测试和管理 SDN 生产环境的平台。氦版本在项目中提到协作开发的重要性，培养协作精神以实现 SDN 平台和相关组件，如果能够在大多使用案例中被大多人应用到实际生产环境当中，将具有重大的意义。此外测试自动化是基础设施中的优秀组成部分，也是最先进的开源项目，对于 OpenDaylight 来说也非常关键，希望将尽早引入测试自动化，促进 OpenDaylight 项目的发展。