OpenDayLight入门教程

北邮-李呈

更多SDN教程：**www.muzixing.com**

关于OpenDaylight

OpenDaylight is an open platform for network programmability to enable SDN and create a solid foundation for NFV for networks at any size and scale.

对于SDN而言，ODL（OpenDaylight）是一个开源的可编程的平台，简单的说是一个非常有野心开源控制器而不仅仅只是控制器。

opendaylight官网:http://www.opendaylight.org/

环境配置

- jdk需要1.6以上。

  sudo apt-get -y install openjdk-7-jdk

更多安装信息请查看:https://wiki.opendaylight.org/view/OpenDaylight_Controller:Installation

或者查看《opendaylight学习及开发初级教程-北邮天依》

- 安装maven

  sudo apt-get install maven

下载与安装

```
git clone https://git.opendaylight.org/gerrit/p/controller.git
```

编译

Controller:

```
cd   controller/opendaylight/distribution/opendaylight
mvn clean install
```

执行

```
cd controller/opendaylight/distribution/opendaylight/target/distribution.opendaylight-OSG
./run.sh
```

maven

opendaylight的工程是基于osgi的maven工程。使用maven对odl进行编译，并运行osgi架构。

opendaylight的文件目录是maven标准目录。

- src/main/java: Application/Library sources
- src/main/resources:Application/Library resources
- src/main/filters: Resource filter files
- src/main/assembly: Assembly descriptors
- src/main/config: Configuration files
- src/main/webapp: Web application sources
- src/test/java: Test sources
- src/test/resources: Test resources
- src/test/filters: Test resource filter files
- src/site: Site
- LICENSE.txt: Project's license
- README.txt: Project's readme

其中pom.xml是非常重要文件，是maven工程的核心文件，是maven的项目对象模型。用于描述资源，包括版本号，依赖关系，资源url等信息。

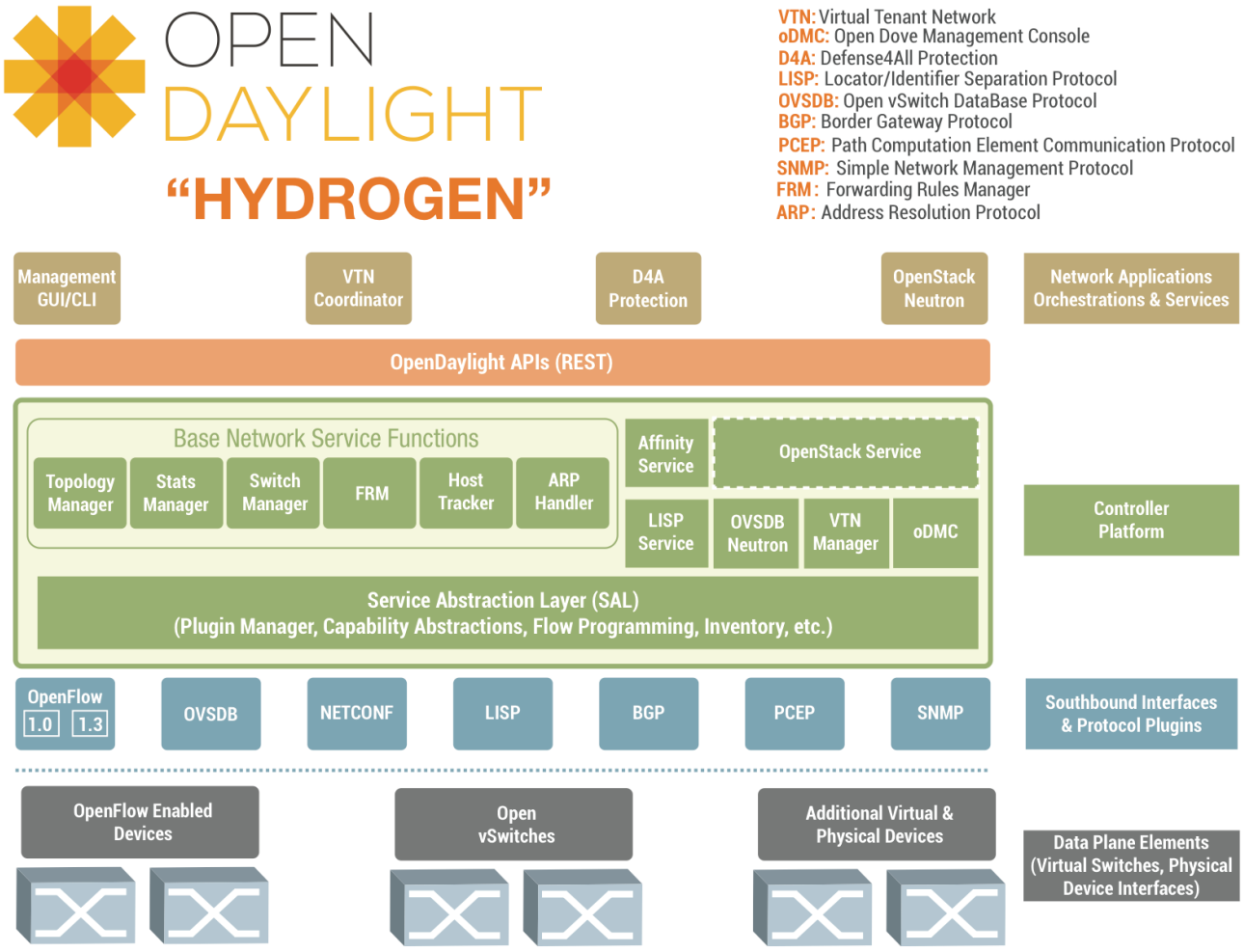
其中每一个工程由的3个坐标定位：groupId:artifactId:version定位。在ODL中groupId可以是：org.opendaylight.controller.artifactId可以是controller的一个模块名如：hosttracker.

OSGI

ODL的每一个maven文件都是OSGI的一个bundle，需要在OSGI中注册使用。更多信息可查看：http://www.opendaylight.org/project/technical-overview

我们使用mvn clean install 编译完ODL之后，进入生成的target目录（前面有介绍）找到opendaylight目录下的run.sh（linux下为.sh,windows下为.bat）,并运行，实际上就是运行了osgi框架。然后osgi根据项目的pom.xml把相关的bundle注册运行。

OpenDayLight目录简介



本图来自：http://www.opendaylight.org

从上图可以看出ODL从层次上可分为三层:

- 南向接口和协议插件
- 控制器平台
- 网络应用 业务和服务

此处主要介绍ODL中controller各目录主要功能。

```
--controller
    --features
    --opendaylight  #主文件 包含各个模块的源文件
    --third-party
    --itest
    --LICENSE
    --NOTICE
    --README.OPENDAYLIGHT
    --pom.xml  #maven文件对象模型，用于描述项目如何工作。
```

在opendaylight中有ODL的模块源文件，各目录的功能简介如下：

- appauth:关于app授权，定义 abstract class Authrization

- archetypes:工程使用的maven的archetype
- arphandler:用于处理arp数据
- forwarding:静态路由
- forwardrulesmanager:管理流表数据和转发规则
- connectionmanager:交换机连接管理：单一，集群，轮询，负载均衡，VTN等连接管理。
- containermanager:VTN管理，每一个VTN对应一个container
- clustering:控制器集群
- commons:公共信息
- hosttracker（new）:主机发现，管理，追踪
- md-sal:

  Model-driven approach to service abstraction presents an opportunity to unify both northbound and southbound APIs and the data structures used in various services and components of an SDN Controller. MD-SAL使得在SDN控制器那些丰富的服务和模块可以使用统一的数据结构和南向和北向的API。

  In order to describe the structure of data provided by controller components a domain-specific language, YANG, is proposed as the modeling language for service and data abstractions. Such language allows to:

  为了描述控制器组件提供的数据结构，我们使用一种领域专门的语言YANG作为服务和数据抽象的建模语言。

  （下面的英文也不难，不翻译了）

  —— Modeling the structure of XML data and functionality provided by controller components

  —— Define sem antic elements and their relationships

  —— Model all the components as a single system.

  The XML nature of YANG data model presents an opportunity for self-describing data, which controller components and applications using the controller’s northbound APIs can consume in a raw format, along with the data’s schema.

  Utilizing a schema language simplifies development of controller components and application. A developer of a module that provides some functionality (a service, data, functions/procedure) can define a schema and thus create simpler, statically typed APIs for the provided functionality, and thus lower the risk of incorrect interpretation of data structures exposed through the Service Abstraction Layer.

- northbound:北向相关（控制器管理，HTTP,web UI,统计，子网，交换机管理，用户管理，网络配置，静态路由，流表编程，主机管理，连接管理等组建的北向接口）

- protocol_plugins:协议插件（openflow,拓扑）
- routing:Dijkstra算法实现
- sal:（最重要的文件之一！）服务抽象层，action，match等功能抽象
- samples:例子（流聚合，负载均衡，二三层转发等）

- security：安全相关
- statisticsmanager:统计模块接口
- switchmanager:交换机管理
- usermanager:用户管理（接口和实现）
- web：web UI

其他目录不再介绍。

Opendaylight模块开发

在ODL中开发模块需要掌握：YANG,XML,MAVEN,JAVA,RESTAPI，等多项知识。每一个模块基本上可以由四个bundle联合实现功能：

- 创建model bundle,使用YANG描述数据结构，和RPC调用
- 创建plugin bundle 用于插件的具体实现，实现model的声明
- 创建service bundle 提供连接北向实现和南向的MD-SAL定义的服务
- 创建northbound bundle 用于提供北向的接口和实现。

更具体的操作将在下一篇教程中详细介绍如何一步一步在ODL中开发模块。