OpenDaylight as a Platform for Network Programmability

FOSDEM, 3 February 2018

Charles Eckel, Cisco DevNet
eckelcu@cisco.com
Agenda

• What is SDN
• What is OpenDaylight
• Network programmability
• Installation
• Example use case (VPP)
• Conclusions
What is SDN
Software Defined Networking (SDN)

- Control & Data Planes separation?
  - OpenFlow?
  - Logically centralized control Plane?
  - White label switches?

- This a valid & useful SDN use case, but...

- SDN can be defined more broadly:
  - Network is a source of vast amount of data...
  - ..that can be utilized by variety of SDN applications

- True power of SDN is network programmability
What Do We Need from an SDN Controller?

• A platform for deploying SDN applications
• Provide an SDN application development environment
  • Developer-friendly APIs to network elements (REST/JSON, pub/sub, etc.)
  • Network-level abstraction through topologies
  • Protocol independence for network-facing applications
What is OpenDaylight
OpenDaylight as a Platform for Network Programmability

© 2018 Cisco and/or its affiliates. All rights reserved. Cisco Public

9
Software Architecture

- Java - enterprise-grade, cross-platform compatible language
- Java Interfaces - for event listening, specifications and forming patterns
- Maven – build system
- Karaf – based on OSGi, provides:
  - dynamic loading of bundles
  - registering dependencies and services exported
  - exchanging information across bundles
Network programmability
The Need for Something Better

- SNMP had failed
  - For configuration, that is
  - Extensive use in fault handling and monitoring
- CLI scripting
  - “Market share” 70%+

RFC 3535

Abstract

This document provides an overview of a workshop held by the Internet Architecture Board (IAB) on Network Management. The workshop was hosted by CNRI in Reston, VA, USA from June 4 thru June 6, 2002. The goal of the workshop was to continue the important dialog started between network operators and protocol developers, and to guide the IETFs focus on future work regarding network management.
Best Practices Coming Together

CLI Best Practices

SNMP Experience

Operations Requirements

NETCONF, RESTCONF and YANG
YANG
Data Modeling Language for Networking
• Modeling language, defined in RFC 6020
• Represents operational state, configuration, transactions, and notifications
• Defines semantics
  • Constraints (i.e. “MUSTs”)
  • Reusable structures
  • Built-in and derived types

In Summary:
YANG is a full, formal contract language with rich syntax and semantics for network data
YANG Model Example

- Screenshot from ietf-interfaces.yang
- Container 'interfaces' with list of 'interface' items
- List items (leafs) have a 'name' which is also the key for the list

```
/*
 * Configuration data nodes
 */

container interfaces {
  description
    "Interface configuration parameters."
;

  list interface {
    key "name";

    description
      "The list of configured interfaces on the device."

    The operational state of an interface is available in the
    /interfaces-state/interface list. If the configuration of a
    system-controlled interface cannot be used by the system
    (e.g., the interface hardware present does not match the
    interface type), then the configuration is not applied to
    the system-controlled interface shown in the
    /interfaces-state/interface list. If the configuration
    of a user-controlled interface cannot be used by the system,
    the configured interface is not instantiated in the
    /interfaces-state/interface list."
;

    leaf name {
      type string;

      description
        "The name of the interface."

      A device MAY restrict the allowed values for this leaf,
      possibly depending on the type of the interface.
      For system-controlled interfaces, this leaf is the
      device-specific name of the interface. The 'config false'
      list /interfaces-state/interface contains the currently
      existing interfaces on the device.
    }
  }
}
```
Tools to work with YANG Models

- **pyang** - An extensible YANG validator and converter in python
  - Source Code - [https://github.com/mbj4668/pyang](https://github.com/mbj4668/pyang)
  - Python Package - [https://pypi.python.org/pypi/pyang](https://pypi.python.org/pypi/pyang)

- **YANG Explorer** - YANG Browser / RPC Builder
  - [https://github.com/CiscoDevNet/yang-explorer](https://github.com/CiscoDevNet/yang-explorer)

- **OpenDaylight YANG Tools** – Tools supporting NETCONF and YANG, code generation from YANG models
  - [https://wiki.opendaylight.org/view/YANG_Tools:Main](https://wiki.opendaylight.org/view/YANG_Tools:Main)
Building a Plugin/Application

1. YANG Tools
   - YANG Model
   - Generate APIs

2. Maven Build Tools
   - Generated API Definition
   - Create API Bundle

3. Maven Build Tools
   - Plugin source code
   - Create Plugin Bundle

4. Deploy
   - “API” OSGI Bundle
   - “Plugin” OSGI Bundle

OpenDaylight as a Platform for Network Programmability
High Level Manageability Architecture

Application
- ANY (C, Java, Python)
  - NETCONF client

Transport
- YANG-based XML
  - NETCONF server
  - SSH / TLS
  - HTTPS

Network Device
- Manageability Infra
  - Config DB
  - BGP
  - QoS
  - VXLAN

- ANY (Java, Python, Perl, PHP)
  - RESTCONF client
- YANG-based XML/JSON
  - YANG-based XML
  - RESTCONF client
  - RESTCONF server

OpenDaylight as a Platform for Network Programmability
Mounting YANG Datastores

OpenDaylight NETCONF Node “Discovery”

- Nodes added by POSTing to config:modules
- OpenDaylight connects to each node
- OpenDaylight learns capabilities (YANG modules) and stores to cache
  - Cache at ~/cache/schema. Filenames of form yang-model@2016-07-12.yang.
Installation
### Distributions

[https://www.opendaylight.org/technical-community/getting-started-for-developers/downloads-and-documentation](https://www.opendaylight.org/technical-community/getting-started-for-developers/downloads-and-documentation)

#### Downloads

<table>
<thead>
<tr>
<th>Release</th>
<th>Release date</th>
<th>Downloads</th>
<th>Documentation</th>
</tr>
</thead>
</table>
| Carbon SR2   | October 16, 2017 | • Pre-Built Tar  
• Pre-Built Zip  
• NeXT UI  
• Virtual Tenant Network (VTN) Coordinator | • Getting Started Guide  
• Developers Guide  
• User Guide  
• Installation Guide  
• Using OpenDaylight with OpenStack  
• Release Notes |
| Nitrogen SR1 (Current Release) | November 26, 2017 | • Pre-Built Tar  
• Pre-Built Zip  
• Virtual Tenant Network (VTN) Coordinator  
• OpFlex | • Getting Started Guide  
• Developers Guide  
• User Guide  
• Installation Guide  
• Using OpenDaylight with OpenStack  
• Release Notes |
$ unzip karaf-0.7.1.zip
Archive: karaf-0.7.1.zip
  creating: karaf-0.7.1/system/ ...
$ cd karaf-0.7.1
$ ./bin/karaf
karaf: Enabling Java debug options: -Xdebug -Xnoagent -Djava.compiler=None
      -Xrunjdwp:transport=dt_socket,server=y,suspend=n,address=5005
Listening for transport dt_socket at address: 5005
Apache Karaf starting up. Press Enter to open the shell now...
100% [========================================================================]
Karaf started in 0s. Bundle stats: 10 active, 10 total

Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit '<ctrl-d>' or type 'system:shutdown' or 'logout' to shutdown OpenDaylight.

opendaylight-user@root>
Install Features using Karaf

- OpenDaylight distro comes without any features enabled by default
- All features are available for you to install
  - `feature:list` list all features available
  - `feature:list -i` list all features installed
  - `feature:list -r` list all features required
  - `feature:install <feature>` install the `<feature>` feature
  - `feature:install <feature-1> <feature-2> … <feature-n>` install list of features
  - `feature:uninstall <feature>` uninstalls the `<feature>` feature
Install DLUX, NETCONF, and RESTCONF

```bash
opendaylight_user@root> feature:install odl-dlux-core
opendaylight_user@root> feature:install odl-dluxapps-yangui
opendaylight_user@root> feature:install odl-restconf-all
opendaylight_user@root> feature:install odl-netconf-all
opendaylight_user@root> feature:install odl-netconf-topology
opendaylight_user@root> feature:install odl-netconf-connector-ssh
```
Honeycomb/VPP using NETCONF

- VPP is a high-performance, open source, software forwarder
  - [http://www.fd.io](http://www.fd.io)
- Honeycomb provides NETCONF and RESTCONF interfaces to VPP
Honeycomb/VPP Using NETCONF

Step by Step

1. Create VM for Honeycomb and VPP
2. Install VPP and Honeycomb on VM
3. Start VPP and Honeycomb
4. Connect to VPP using CLI
5. Add interface(s) to VPP
6. Connect to VPP using Honeycomb/NETCONF
7. Connect to VPP using OpenDaylight
Honeycomb/VPP Using NETCONF

7. Connect to VPP Using OpenDaylight

- Enable NETCONF interface on OpenDaylight
  - feature:install odl-restconf-all odl-netconf-all odl-netconf-topology odl-netconf-connector-ssh

- Add VPP to OpenDaylight using Postman
  - PUT
    http://{odl_address}:8181/restconf/config/network-topology:network-topology/topology/topology-netconf/node/vpp1
  - Postman collection

- Interact with VPP using OpenDaylight DLUX
OpenDaylight as a Platform for Network Programmability

GET NETCONF Topology


Authorization: Basic YWRtaW46YWRYwYrtaW5=
Content-Type: application/xml

Body:

```
{
  "topology": [
    {
      "topology-id": "topology-netconf",
      "node": [
        {
          "node-id": "controller-config",
          "netconf-node-topology:available-capabilities": {
            "available-capability": [
              {
                "capability": "urn:ietf:params:netconf:capability:candidate:1.0",
```
OpenDaylight as a Platform for Network Programmability
Conclusions
Key Takeaways

• SDN is more than just OpenFlow
• Network programmability is key benefit of SDN
• OpenDaylight provides a platform for network applications and programmable network infrastructure via YANG, NETCONF, RESTCONF
Additional resources
Open Source Dev Center

Your Source for Open Source at Cisco
https://developer.cisco.com/opensource

• Contributions to open source
• Use in products/solutions
• Community forums, blogs
• Developer Events
  • IETF Hackathons and MEF LSO Hackathons featuring open source implementations of open standards
OpenDaylight Microsite
https://developer.cisco.com/opendaylight
Building Applications on Top of OpenDaylight

- **AUTODEV**: Visualize and manage IoT sensors embedded in motor vehicles.
- **BGP and PCEP Pathman**: Visualize topologies and program MPLS traffic engineering (TE) paths.
- **BIERMAN**: Visualize and manage BIER network topologies within ODL.
- **DevNet Sample Apps**: Learn how to use ODL and create your own apps that run on top of it.
- **OpenFlow Manager**: Visualize OpenFlow (OF) topologies, program OF paths and gather OF stats.
- **PCE-OpenFlow**: Apply policy-based path computation traffic engineering to OpenFlow networks.
- **YANG Explorer**: Yang browser and RPC builder application to experiment with YANG models.
- **In-band OAM (OAM)**: Add operational info to packet as it traverses a path in network.
- **VPP vBridge Manager**: Define VPP-based virtual bridge domain(s) for L2 connectivity.
- **YANGMAN**: Dynamically generated UI forms and native JSON representation based on RESTCONF APIs.
- **OneM2M Plugins**: Extend the functionality of the oneM2M datastore. Protocol conversion, oneM2M data export are examples.
- **OneM2M TSDR Plugin**: Export oneM2M data to the OpenDaylight Time Series Data Repository.
- **Pathman SR**: Visualize topologies and program Segment Routing (SR) paths.
- **Service Function Chaining**: Create and deploy service chains using the NSH protocol as defined in draft-ietf-sfc-nsh
- **netACL**: Program and manage Access Control Lists (ACLs) on routers in multi-vendor network.
Tutorials and Sandboxes

OpenDaylight Nitrogen SR1 with Apps with 8 Nodes v1

Overview

OpenDaylight (ODL) is a collaborative, open-source project used to advance software-defined networking (SDN). OpenDaylight is a community-led, industry-supported framework consisting of code and blueprints. Using this framework, you can accelerate process adoption, foster innovation, reduce risk, and create a more transparent approach to SDN. OpenDaylight can be a core component within any SDN architecture. Building on open-source SDN and NFV controllers enables users to reduce operational complexity, extend the life of their existing infrastructure hardware, and enable new services and capabilities only available with SDN.

Scenarios

- Scenario 1: Explore ODL Features
- Scenario 2: Explore DLUX
- Scenario 3: Install BGP Pathman Application
- Scenario 4: Enable OpenFlow in Karaf
- Scenario 5: Install OpenFlow Manager Application
- Scenario 6: Explore Pathman Segment Routing
- Scenario 7: Explore netACL Application
- Scenario 8: Explore Yangman
Continue Your Education

• Become a DevNet Member:
  • https://developer.cisco.com/join

• Access OpenDaylight resources
  • https://developer.cisco.com/site/opendaylight/

• Visit our Open Source Dev Center:
  https://developer.cisco.com/site/opensource/
Thank you!