Model Driven Network Programmability

October 15, 2018

Charles Eckel, Cisco DevNet
eckelcu@cisco.com, @neckelcu
Agenda

• What is SDN
• Network programmability
• Hands-on exercises
• 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
SDN - A Broader Definition

- Application Developer Environment
- Management and Orchestration
- Analysis and Monitoring, Performance and Security
- Network Services
- Control Plane
- Forwarding Plane
- Network Elements and Abstraction
- Transport

Harvest Network Intelligence

Program for Optimized Experience

Generic feedback/control/policy loop between apps and the network
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
Network programmability
Why Network Programmability Matters

Network Expenses

![Bar chart showing 33% CAPEX and 67% OPEX. Source: Forrester]

Deployment Speed

![Bar chart showing deployment time for computing and networking. Source: Open Compute Project]

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

NETCONF, RESTCONF and YANG

SNMP Experience

CLI Best Practices

Operations Requirements
YANG
YANG
Data Modeling Language for Networking

- Modeling language, defined in RFC 6020
- Models configuration and state data, RPCs, 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 network-topology.yang

• Container 'network-topology' with list of 'topology' items

• List items (leafs) have a ‘topology-id' which is also the key for the list

```yang
container network-topology {
  list topology {
    description "This is the model of an abstract topology. A topology contains nodes and links. Each topology MUST be identified by unique topology-id for reason that a network could contain many topologies."
    key "topology-id";
    leaf topology-id {
      type topology-id;
      description "It is presumed that a datastore will contain many topologies. To distinguish between topologies it is vital to have UNIQUE topology identifiers."
    }
    leaf server-provided {
      type boolean;
      config false;
      description "Indicates whether the topology is configurable by clients, or whether it is provided by the server. This leaf is populated by the server implementing the model. It is set to false for topologies that are created by a client; it is set to true otherwise. If it is set to true, any attempt to edit the topology MUST be rejected."
    }
  }
}
```
Tools to work with YANG Models

• **pyang** - An extensible YANG validator and converter
  • Command line tool
  • 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 and RPC Builder
  • Web Based GUI
  • [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)
Display a YANG Module

$ pyang -f tree <yang-file>
pyang Tip – JavaScript Tree Output

- Use `pyang -f jstree -p <model.yang> -o <output.html>`
- Produces collapsible Tree / HTML
Building a Plugin/Application with YANG tools

1. Generate APIs
   - Yang Tools
     - Yang Model
     - Generated API Definition

2. Create API Bundle
   - Maven Build Tools
   - “API” OSGI Bundle

3. Create Plugin Bundle
   - Maven Build Tools
   - “Plugin” OSGI Bundle

4. Deploy
   - Controller
   - Module
   - Module
   - Plugin source code
NETCONF
NETCONF

IETF network management protocol

• Defined in RFC 4741 (2006), updated by RFC 6241 (2011)
• Connection oriented, with transport via SSH/TSL
• Data defined by YANG models, encoded in XML
• Distinguishes between configuration and state data
• Multiple configuration datastores (candidate, running, startup)
• Change validation, transactions, filtering, and notifications

In Summary:
NETCONF provides fundamental programming features for convenient and robust automation of network services
NETCONF Sessions

• NETCONF is connection-oriented
  • SSH, TLS as underlying transport
  • XML for payload

• NETCONF client establishes session with server

• Session establishment: <hello> exchange
  • Announce capabilities, modules, features

• Session termination
  • <close-session>, <kill-session>
NETCONF Commands

- get : to retrieve operational data
- get-config : to retrieve configuration data
- edit-config : to edit a device configuration
- copy-config : to copy a configuration to another data store (e.g. non-volatile memory)
- delete-config : to delete a configuration in a data store
RESTCONF
RESTCONF

Restful API for YANG data models

- IETF RFC 8040
- Configuration and state data exposed as resources
- Access data using REST verbs (GET / PUT / POST …)
- Construct URIs, based on structure of YANG model, to access data
- HTTP instead of SSH for transport
- JSON in addition to XML for data encoding

In Summary:
RESTCONF provides light weight interface to network datastores leveraging well known combination of REST and JSON
RESTCONF URI & JSON Example

```
<node xmlns="urn:TBD:params:xml:ns:yang:network-topology">
  <node-id>vpp1</node-id>
  <host xmlns="urn:opendaylight:netconf-node-topology">{{vpp1_address}}</host>
  <port xmlns="urn:opendaylight:netconf-node-topology">2831</port>
  <username xmlns="urn:opendaylight:netconf-node-topology">admin</username>
  <password xmlns="urn:opendaylight:netconf-node-topology">admin</password>
  <tcp-only xmlns="urn:opendaylight:netconf-node-topology">false</tcp-only>
  <keepalive-delay xmlns="urn:opendaylight:netconf-node-topology">0</keepalive-delay>
</node>
```
High Level Manageability Architecture

- Application
  - ANY (C, Java, Python)
    - NETCONF client
  - ANY (Java, Python, Perl, PHP)
    - RESTCONF client

- Transport
  - YANG-based XML
  - SSH / TLS
  - YANG-based XML/JSON
  - HTTPS

- Network Device
  - Manageability Infra
    - Config DB
      - BGP
      - QoS
      - VXLAN
  - RESTCONF server
  - NETCONF server
  - YANG-based XML
  - ANY (Java, Python, Perl, PHP)
Hands-on exercises
Sign up for DevNet

• Free and easy using event specific URL
• https://developer.cisco.com/join/iitrtc18
https://learninglabs.cisco.com/modules/fundamentals
Introduction to Model Driven Programmability (ex: NETCONF/YANG)

Explore the reasons behind the move to Model Driven Programmability from traditional interfaces such as OSLINAP. Learn about the interaction between YANG data models and the new standard transport protocols of NETCONF and RESTCONF. Discover how to leverage NETCONF/RESTCONF to query and configure network devices.

Choose a learning lab to start learning

What and Why of Model Driven Programmability

What is "Model Driven Programmability" and why was it developed? What purpose do the new protocols and standards of YANG, NETCONF, and RESTCONF provide? Get the answers to these questions in this lab.

Last Completed on 10/05/2018 @ 10:02 AM

Introducing YANG Data Modeling for the Network

What's YANG got to do with it? In this lab you'll find out all about it! Learn about the YANG modeling language, checkout some of the available model options, and even see what network data looks like when it's parsed into these models.

Last Completed on 10/05/2018 @ 10:03 AM

Exploring IOS XE YANG Data Models with NETCONF

Learn the ins and outs to working with NETCONF to access the YANG modeled configuration and operational data on your network devices. Get hands-on by initiating NETCONF connections, retrieving data, and sending configurations to the network.

Last Completed on 10/05/2018 @ 10:03 AM

Exploring IOS XE YANG Data Models with RESTCONF

So you want a REST API for the network? Well RESTCONF is your tool then. Checkout how YANG models become URLs with RESTCONF! Learn all there is to know about CRUD! You'll explore RESTCONF with basic API calls and with OAuth!
Accessing DevNet Sandbox to Reserve Your Own Setup
https://devnetsandbox.cisco.com/RM/Topology
Reserve Same Setup as Used in Learning Lab
IOS XE Programmability
Conclusions
Key Takeaways

• SDN is more than just OpenFlow
• Network programmability is key benefit of SDN
• You don’t need a computer science degree to program the network
Helpful Links

- Join DevNet: https://developer.cisco.com/join/iitrtc18
- Cisco DevNet: https://developer.cisco.com
- Learning Labs: https://learninglabs.cisco.com
- Sandboxes: https://developer.cisco.com/site/sandbox/#
- @eckelcu: https://twitter.com/eckelcu
Thank you!