SmartFabric Director: Data Center Automation, Monitoring and Operational Management

As modern, open and software-driven networks change how cloud providers and enterprises approach the data center, the need to simplify management and increase efficiency across virtual and physical network environments has never been greater.

The Dell EMC SmartFabric Director enables data center operators to build, operate and monitor an open network underlay fabric based on Dell EMC Open Networking PowerSwitch Series switches. SmartFabric Director automates and simplifies the provisioning and monitoring of the fabric using Openconfig based models and protocols. Tight integration with VMware vSphere and NSX-T allows SmartFabric Director to dramatically simplify fabric provisioning for dynamic virtualized workloads and overlays.

SmartFabric Director – a Joint collaboration effort between Dell Technologies and VMWare – had the following goals in mind, as we embarked on the project:

- Simplicity: Reduce the steps to deploy a fabric while providing a single point for fabric lifecycle management.
- Openness: Support for open standards to maximize flexibility, interoperability and technology investment.
- Consistency: Apply a consistent policy and automation framework across physical and virtual environments to reduce complexity while increasing efficiency.

We will look at these goals in some detail and describe the design choices made and how they help achieve the desired outcomes for customers

Openness

API-First Design

Most Enterprises have a variety of applications and tools to aid the Data Center Networks. It was important for SFD to fit into the Customers' Software Architecture, which meant supporting programmatic interfaces and not just GUI. SFD has an API-First mindset – anything support through the Graphical User Interface (GUI) is also supported through an API – from Day One. SFD supports a REST based Northbound API to enable orchestration systems to programmatically use the product.

gNMI

gNMI (gRPC **N**etwork **M**anagement **I**nterface) is a protocol that provides the mechanism to manipulate (create, update, delete) the configuration of network devices, and state retrieval. The content provided can be modeled with, but not limited to, YANG objects using a path consisting of elements names and map attributes. gNMI uses vendor neutral Openconfig YANG Objects to describe the elements and attributes. gNMI is built on top of gRPC.

gRPC

gRPC an open source framework developed by Google and managed by CNCF (Cloud Native Compute Foundation). The RPC framework built on top of HTTP/2. The framework allows for Unary, server streaming, client streaming and bi-directional streaming RPCs. gNMI allows for Multiplexing of RPCs over a single channel provided by library.

Here are some of the RPC calls used in SFD:

- Set
    - Manipulate the (writeable) state of a target (Switch). This includes update and delete.
    - Each unary Set RPC is a transaction.
- Subscribe
    - Streaming RPC for Switch to send state to client (SFD).
    - Immutable subscriptions with an overall mode:
        - STREAM - "streaming telemetry" - long-lived push from device.
        - POLL – client (SFD)-requested streaming.
        - ONCE – target (Switch) advertises entire dataset and closes RPC.

  To ensure data fidelity, state is always time stamped at target(Switch)

- Get
    - Snapshot of path state at a particular time.
    - Typically used for configuration state retrieval.
- Capabilities
    - Used to understand encodings and models that are supported by a target.

ProtoBuf

Data exchanged between SFD and Switches is encoded using Google Protocol Buffers. Protocol Buffers (a.k.a., protobuf) are language-neutral, platform-neutral, extensible mechanisms for serializing structured data.

OpenConfig

OpenConfig started as an informal working group of network operators with the goal of moving networks toward a more dynamic, programmable infrastructure by adopting software-defined networking principles such as declarative configuration and model-driven management and operations. Initial focus of OpenConfig is on compiling a consistent set of vendor-neutral data models (written in YANG) based on operational needs from use cases and requirements from multiple network operators. Openconfig also enables Streaming telemetry for network monitoring in which data is streamed from devices continuously with efficient, incremental updates. Operators can subscribe

to the specific data items they need, using OpenConfig data models as the common interface.

Example: Establishing a Connection and Retrieving Data

In this example, the LLDP neighbors are obtained from the switch. The SFD establishes a gNMI channel with the Dell OS10 switch with Call credentials and issues the gNMI GET request for the following Openconfig path:

path: elem { name: "lldp" } elem { name: "interfaces" }

Along with the GET request, SFD also requests the switch to encode the return data using protobuf. (Note: The protobufs used between the switch and SFD are assumed to be compatible).

The switch would send the LLDP neighbors on each interface. The protobuf will be deserialized by SFD and converted to its native data-model.

Example: Changing the MTU on an interface

In this example, the interface MTU on an ethernet interface is changed. Also, the interface is enabled.

path: elem { name: "interfaces" } elem { name: "interface" key { key: "name" value: "ethernet1/1/1" } }

data: config { enabled { value: true } name { value: "ethernet1/1/1" } mtu { value: 9100 } }

This results in the following configuration on the switch

sc2-t5-s4148-l2# show running-configuration interface ethernet 1/1/1

!

interface ethernet1/1/1

no shutdown

switchport access vlan 1

mtu 9100

flowcontrol receive on

sc2-t5-s4148-l2#