The Path to a Programmable Network

_Open Telemetry and Model-Driven Configuration_

Vincent Boon
12-Jul-2019
SGNOG-7, Singapore
Overview

• Motivation
• Exciting news from the world of software development
• Strategies for automation
• Standards
How it starts
What have we wrought?
Trouble with configuring via the CLI

• Configs applied and maintained by hand
• Vendor-specific syntax, ever-changing
• Inconsistent configurations, unexplained special cases
• Configurations are forgotten; until... reactive break-fix model
• Workarounds: brute force, diligence, hard work, RANCID
• Fragile scrapers break on upgrades

• Human-oriented interfaces are for humans
Manual control

Controlled by software
Combining the reliability of software with the low cost of network changes
Exciting news from the world of software

- Agile practices
  - Lighter but more intense teamwork
- git + pull requests
  - collaborative text editing, focused review
- CI/CD + tests + sandboxes
  - safety nets are safer
- DevOps
  - the best tools and practices adopted by operations
program

configure

1:100
Strategies
Pre-conditions for automation

- **Inventory**: do you have good detail of all devices?
- **Requirements**: business’s expectation of function level; now & plans
- **Standards**: which model/vocabulary to use in automation
- **Telemetry**: feedback channels
- **Automation**: control channels; tooling
- **Trust**: get experience, confidently predict automation’s behaviour
- **other**: Budget, Stakeholders, Suppliers
<table>
<thead>
<tr>
<th></th>
<th>Provisioning</th>
<th>Monitoring</th>
<th>Security</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>Automation</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>Telemetry</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>Inventory</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>Standards</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>Suppliers</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>Partners</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>Budget</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>Trust</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>
Standards
IETF NETCONF

- 2006 RPC-based protocol for configuring network devices
- "SNMP done right"
- Replaces CLI-based programmatic interfaces (perl/expect over SSH)
- Installs, manipulates configuration
- Can validate config before activation
- Atomic commit/transaction across multiple devices
- Structured message and errors (XML/JSON)
NETCONF: XML over SSH

$ ssh -oHostKeyAlgorithms=+ssh-dss root@ios-xe-mgmt.cisco.com -p 10000 -s netconf

S: <hello> <capabilities> ... </capabilities> </hello>

C: <hello> <capabilities> ... </capabilities> </hello>
NETCONF: XML over SSH (continued)

C: <rpc>
    <get-config>
        <source><running/></source>
    </get-config>
</rpc>

S: <rpc-reply>
    <data>
        <interfaces><interface> <name>eth0</name> <enabled>true</enabled>...</interface>
    </data>
</rpc-reply>
NETCONF landscape

- **YANG** - the modeling language for NETCONF
  - like MIBs' ASN.1 notation
  - IETF provides some basic models, eg RFC8343 "ietf-interfaces"

- **NETCONF** - the client-server protocol
  - NETCONF: sends XML over SSH
  - RESTCONF: sends XML or JSON over HTTP

- **OpenConfig** - the community
  - a group of carriers and vendors sharing their YANG models
  - more than just "vendor MIBs" (called native models)
Where to get YANG models

- yangcatalog.org
  - https://github.com/YangModels/yang (yangcatalog.org)
- openconfig.org
  - https://github.com/openconfig/public
- https://github.com/Juniper/yang
Summary
Suggested approach:

• Get familiar with the OpenConfig models, eg BGP. Many examples available showing YANG/NX-OS CLI

• Try simple templated deployment (interfaces)

• Try same using YDK or NCClient, in Python
Additional thoughts

• The divide-and-conquer NETCONF strategy reflects what we’ve seen at network operators and in large scale networks

• The “surgical” approach to configuration updates is
  – Faster and less disruptive
  – Allows for shared control of network devices (multi-tenant)

• Alternate approach is to distribute "complete" configuration files via tools like Ansible, possibly generated from YANG models
  – Reliable, whole-device 'restore'

• Ultimate end game is an Intent Based Network?