Network Fabric Configuration for SONiC

1. Base CLI Framework
2. CLI framework streamlining is WIP with-in community -
3. Python Utilities
4. Minigraph
5. Config_db.json

I want better
- Single unified interface for management and orchestration
- No additional software
- Language agnostic
- Follows microservices model
- Futuristic

Let us build one before OCP 2019!!
Unified Management Service for SONiC

- Unified REST APIs for all* SONiC features
- Designed to be scalable for large deployments
- Flexibility to integrate SONiC in any infra software
- Abstraction for seamless upgrades

AIMS Container

HTTP Server

AIMS Request handler

AIMS Validation

AIMS Response handler

Existing SONiC Containers

- SNMP
- SWSS
- BGP
- LLDP
- DB

SONiC NOS on a switch

AIMS SDK

OK, How about I give you an SDK to develop custom scripts and well tested playbooks for most SONiC deployments!!

<table>
<thead>
<tr>
<th></th>
<th>Image/Config Upgrade</th>
<th>VLANS/IP/LAGs</th>
<th>BGP Fabric</th>
<th>Monitoring/ Troubleshooting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ansible</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Chef</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Custom Scripts</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Okay, but what in its for me – How do I run SONiC in DEVOPS centric environment?
How does this work - Demo

STEP 1: Bring up 2xSpine and 2xLeaf

STEP 2: Generate Fabric Configuration
- Create SONIC VS and connect using Linux bridge configuration.
- Setup management IP address/Gateway configuration for reachability.
- Generate Ansible "vars" for fabric configuration including interface IP, BGP AS numbers, Loopback interfaces, BGP neighbor etc.

STEP 3: Run Playbook to provision fabric and verify fabric configuration
- Run Playbook using AIMS APIs as backend configure the fabric.
- Validate the configuration using BGP summary and pings across switches.

---

<table>
<thead>
<tr>
<th>Role</th>
<th>Name</th>
<th>MGMTIP</th>
<th>Interfaces</th>
<th>ASN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spine</td>
<td>Spine01</td>
<td>192.168.0.50</td>
<td>Eth0-Eth60</td>
<td>62001</td>
</tr>
<tr>
<td>Spine</td>
<td>Spine02</td>
<td>192.168.0.51</td>
<td>Eth0-Eth60</td>
<td>62002</td>
</tr>
<tr>
<td>Leaf</td>
<td>Leaf01</td>
<td>192.168.0.54</td>
<td>Eth0-Eth60</td>
<td>63001</td>
</tr>
<tr>
<td>Leaf</td>
<td>Leaf02</td>
<td>192.168.0.55</td>
<td>Eth0-Eth60</td>
<td>63002</td>
</tr>
</tbody>
</table>
Looking Forward

• Provide SDK for all protocol
• Provide cookie cutter “ready to deploy” Open Source Solutions
• Integration with Cloud Infra software
• Open Source after scale and stress test