network-runner Documentation

Ansible Developers

Mar 14, 2022

Contents

CHAPTER 1

Contents

1.1 Network-Runner: Ansible Networking API

1.1.1 Overview

Network-Runner is a python library that abstracts management and interaction with switching hardware to Ansible Networking. This library is not tested with all the modules included with Ansible Networking. In theory it should work with any switch that has compatible modules included with Ansible Networking if the provider tasks are added to this library's Ansible role. See the contributor documentation for more information about adding support for an Ansible Networking module to this library.

- Free software: Apache license
- Documentation: https://network-runner.readthedocs.io/en/latest/
- Source: https://github.com/ansible-network/network-runner/

1.1.2 Components

The Network-Runner library consists of the following components:

`Python API` Imported directly by python.

`Ansible Role` Used by Ansible during Ansible invocation.

1.1.3 Use Cases

`Python API`

Any python application could need the ability to communicate with a switch to perform a task that network-runner is able to complete. The interaction with ansible is designed in a library style that will allow direct import and invocation in python independent of a running OpenStack deployment.

1.1.4 API Features

The following matrix indicates which features have been implmented.

	openvswitch	junos	nxos	eos	enos	cumulus	dellos10	fos
Create VLAN	N/A	Y	Y	Y	Y	Y	Y	Y
Delete VLAN	N/A	Y	Y	Y	Y	Y	Y	Y
Delete Port	Y	Y	Y	Y	Y	Y	Y	Y
Config Access Port	Y	Y	Y	Y	Y	Y	Y	Y
Config Trunk Port	N	Y	N	Y	Y	Y	N	Y
List VLANs	N	N	N	N	N	N	N	N
Add Trunk VLAN	N	N	N	N	N	N	N	Y
Delete Trunk VLAN	N	N	N	N	N	N	N	Y
Get Port Conf	N	N	N	N	N	N	N	Y

1.2 Install and configure

This section describes how to install and configure the network-runner python library.

1.2.1 Prerequisites

To successfully install and configure the Networking-Ansible library, you will need Switch credentials that allow configuration changes to the ports and VLANS planned to be managed.

For security purposes it is important that you do not provide administrator access to the switch for network-runner. A user should be created and granted access for the permissions needed for network-runner.

Network-Runner will need permissions to execute the actions you plan to use. This could include creating and deleting VLANS, configuring ports in access mode and trunk mode and deleting port configurations.

Collecting this information and ensuring the proper permissions will successful installation and operation.

1.2.2 Install

This section describes how to install and configure the Network-Runner library.

Note: Note that installation and configuration may vary by distribution.

Manual Installation

This section describes how to install the Network-Runner library in the case that a package is not available for your distribution. Network-Runner can be installed from PyPI using pip.

Install components

1. Install the packages:

```
# pip install networking-ansible
```

1.2.3 Next steps

If you have installed and configured Network-Runner, your environment now includes the network-runner library.

Network-runner can now be imported and instantiated. See user.

Note: Installation and configuration may vary by distribution.

1.3 User guide

Network-Runner can be called directly via python API. This section will show an example of end users could import the network-runner API and execute switch level network configuration.

1. In a python environment import the network-runner api module.

   ```
   from network_runner import api
   ```

2. Create a Host object and create an Inventory object adding the host object to the inventory object. Then instantiate the NetworkRunner class.

   ```
   from network_runner.models.inventory import Inventory
   from network_runner.models.inventory.hosts import Host

   host = Host(name='testhost',
               ansible_host='192.168.121.37',
               ansible_user='root',
               ansible_ssh_pass='password',
               ansible_network_os='junos')

   inventory = Inventory()
   inventory.hosts.add(host)

   net_runr = api.NetworkRunner(inventory)
   ```

3. Call functions that configure the switch.

   ```
   hostname = 'testhost'
   vlan = 37
   port = 'xe-0/0/7'
   t_vlans = [3,7,73]

   # Create a vlan
   net_runr.create_vlan(hostname, vlan)
   # delete a vlan
   net_runr.delete_vlan(hostname, vlan)

   # Configure an access port
   net_runr.conf_access_port(hostname, port, vlan)
   # configure a trunk port
   net_runr.conf_trunk_port(hostname, port, vlan, t_vlans)
   ```

(continues on next page)

(continued from previous page)

```
# delete a port
net_runr.delete_port(hostname, port)
```

1.4 Contributor Documentation

1.4.1 Contributing

1.4.2 Provider Support

Network-Runner contains platform specific functionality in an Ansible role that enables specific hardware devices per provider switch added to this role. The network-runner role is stored in etc/ansible/roles directory. A platform is refered to as a provider in the network-runner context.

Adding a provider

To add a provider to networki-runners's capabilities the provider must be added to the Ansible role. To add the provider to the Ansible role a new directory using the Ansible Networking module name [1] must be added. That directory will contains files that will define the Ansible tasks nessessary to perform the respective configuration action.

Inside the directory for the provider, a file for default values, plus a file for each supported action is required. The currently required files for basic support are listed below. Each file other than the defaults.yaml file must be added with atleast a noop task in each.

- defaults.yaml

  Defines default values for VLAN name and ID. For example some switch vendors use the name "default" and VLAN ID 1 as a default VLAN to assign switchports to. Open vSwitch expects no VLAN ID and VLAN name in the case that port is not assigned to a specific VLAN.

- create_vlan.yaml

  Defines the Ansible tasks to create a VLAN on a switch.

- delete_vlan.yaml

  Defines the Ansible tasks to delete a VLAN on a switch.

- update_access_port.yaml

  Defines the Ansible tasks to assign a VLAN to a switchport in access mode.

- conf_trunk_port.yaml

  Defines the Ansible tasks to configure a switchport in trunk mode.

- delete_port.yaml

  Defines the Ansible tasks to remove configuration from a switchport.

[1] https://docs.ansible.com/ansible/2.6/modules/list_of_network_modules.html

1.5 References

References of network-runner.

1.6 Indices and tables

- genindex
- modindex
- search