**ICOS Provisioning with Ansible**

Created by:

Ron Wilhelmson

[ron.wilhelmson@deltaww.com](mailto:ron.wilhelmson@deltaww.com)

Version:

Draft 1

# Introduction

This guide will outline the utilization of Ansible Playbooks to automate the provisioning of Delta Networks switches with ICOS for layer 2 and 3 switching and routing applications.

**Objective**

The objective of this guide is to document the basic steps required to provision a switch that has been configured with ICOS. Complementary information for basic setup of the ICOS switch can be referenced here [ICOS NOS installation with Ansible](https://github.com/DeltaProducts/SolutionCenter/blob/master/ICOS%20NOS%20installation%20with%20Ansible.pdf) and Ansible user guide can be referenced at [Getting Started with Ansible](https://github.com/DeltaProducts/SolutionCenter/blob/master/Getting%20Started%20with%20Ansible.pdf).

**Pre-install Connectivity and** **Setup**

Network and Systems required

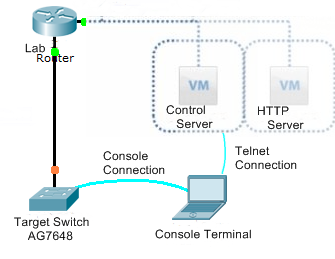
The basic systems required for building and running Ansible playbooks consist of the following:

Control Server – Linux server that runs Ansible and contains playbooks.

Target Switch – Switch that is to be configured running ONIE and may be many.

Web Server – HTTP location where update, license, and install files are located.

Console Terminal – PC with console connection to switch and telnet to control server.

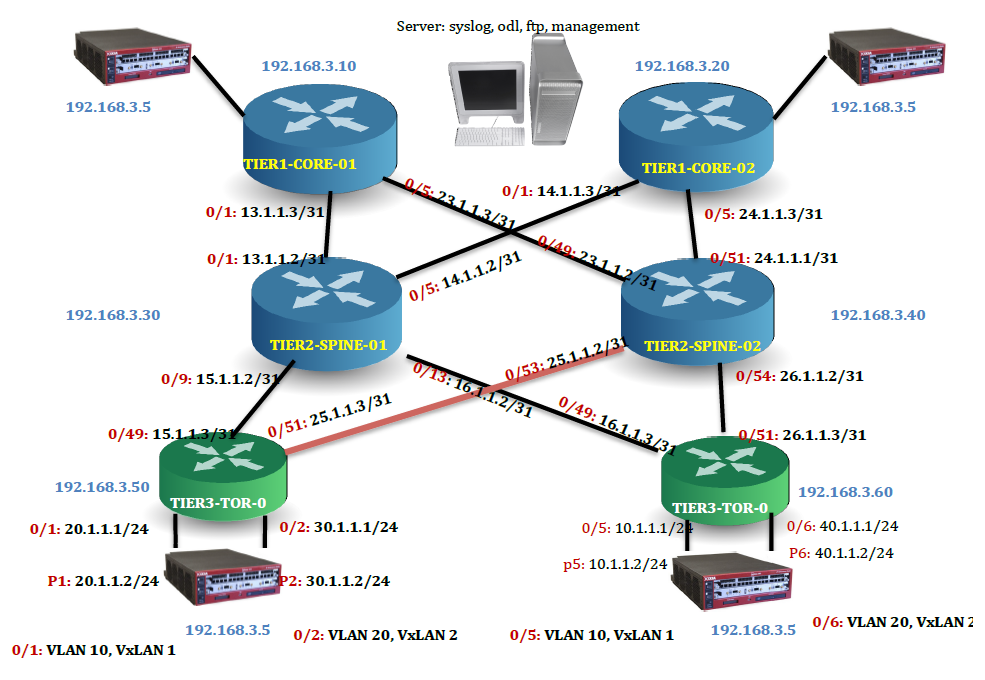
**LAB** **Network Diagram**

This network topology shows the general configuration for provisioning a single switch. In a general lab topology there may be multiple switch roles and configurations as shown below and in referenced examples. In these examples this topology is used for each role one at a time.

Note: Please reference [Getting Started with Ansible](https://github.com/DeltaProducts/SolutionCenter/blob/master/Getting%20Started%20with%20Ansible.pdf) for additional details on system, switch, and server setups.

**Lab topology (simulating a cluster or POD)**

A general lab topology may look more like the following topology with Core, Spine, and TOR configurations.



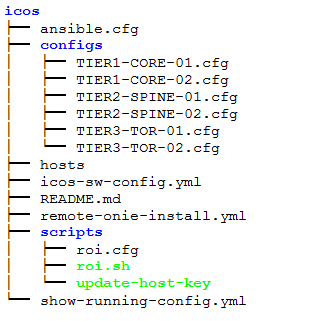
The configuration files referenced below are used to provision each of the switches in this topology. See references for configuration files for these nodes. Note that IP references may be different (in blue) in a specific lab or production configuration. Additional details can be found in the following document:

<https://github.com/DeltaProducts/SolutionCenter/blob/master/DC%20Use%20Cases.pdf>

**Ansible Files**

The following describes the primary files you will need to get the provisioning playbook running. The configuration files are examples of running configurations that can be modified and copied to the download directory on your local HTTP server.

Ansible Directory and Files



**ansible.cfg**

The ansible.cfg file contains all of the configuration variables that can be set and Ansible will read this configuration file when it is initiated. For these playbooks the standard ansible.cfg file is being used with one exception. In order to get a more readable output when the playbook is running the following line can be modified:

stdout\_callback = yaml

**hosts**

The hosts file is a reference file used to define the switches and systems that Ansible will be executing tasks on.

Example hosts file:

# This is the default ansible 'hosts' file.

#

# It should live in /etc/ansible/hosts

#

# - Comments begin with the '#' character

# - Blank lines are ignored

# - Groups of hosts are delimited by [header] elements

# - You can enter hostnames or ip addresses

# - A hostname/ip can be a member of multiple groups

#

# DPR Lab Agema Switch AG7648

#

[switches]

10.62.10.34

[http\_server]

10.62.10.22

[control\_server]

10.62.10.22

**playbook**

Ansible playbooks are written in the YAML language. For ICOS provisioning playbook we will have the following sections:

hosts – defines the systems to pass commands to. In this case the target switches.

vars – variables defined for use with in task executions

tasks – commands that will be executed by the playbook

For this playbook we are also using the remote\_user: admin and become:: yes as required for connecting to the target switch and executing as sudo for provisioning commands.

Note: Please see the references at the end of this document for additional details on commands and modules.

# Ansible Playbook ICOS Provisioning

The following YAML format playbook is used to remotely provision an ICOS switch by referencing configuration files for layer 2 and 3 applications.

**ICOS Switch Configuration Playbook**

The following playbook along with associated configuration files are utilized to initiate and run the ICOS switch configuration playbook on the target switch running ICOS. Edits to the configuration files may be required for your specific network details and switch application. The playbook straight forward and executes six tasks as follows:

1. Get Switch Configuration File
2. Create startup-config from Configuration File
3. Configure ICOS Switch
4. Restart Switch
5. Wait for switch to come back up
6. Show ICOS running configuration

The playbook is shown below.

ICOS Provisioning Playbook: icos-sw-config.yml

---

- hosts: switches

remote\_user: admin

become: yes

gather\_facts: yes

vars:

http\_server: 10.62.10.22

config\_file: TIER1-CORE-01.cfg

install\_file: http://{{ http\_server }}/configs/{{ config\_file }}

tasks:

- name: Get Switch Configuration File

get\_url: url={{ install\_file }} dest=/home/admin/{{ config\_file }} mode=0755

tags: get url cfg

- name: Create startup-config from Configuation File

command: 'cp {{ config\_file }} /mnt/fastpath/startup-config'

tags: copy cfg

- name: Configure ICOS Switch

shell: 'icos-cfg -a {{ config\_file }}'

register: icos\_cfg

failed\_when: icos\_cfg.rc != 230

tags: sudo icos-cfg

- name: Restart Switch

shell: 'shutdown -r now "Ansible updates triggered"'

async: 0

poll: 0

ignore\_errors: true

become: true

- name: Wait for Switch to come Back Up

local\_action:

module: wait\_for

host={{ inventory\_hostname }}

connect\_timeout=10

port=22

delay=25

timeout=2500

state=started

- name: Show ICOS running configuration

shell: 'icos-show running-config'

register: running-config

tags: show running-config

Ansible utilizes SSH to connect to the target switch to execute the commands defined in the playbook tasks. It does this by creating a series of tasks and then uses sftp, by default, to put the Ansible created commands in the .ansible/tmp directory on the target switch. Once the command file is on the target system Ansible issues an SSH command to execute the temporary command file and then cleans up by deleting the temporary file.

**Playbook Execution**

The Ansible command to run this playbook configuration is as follows:

ansible-playbook icos-sw-config.yml --ask-pass --ask-become-pass -v

Note: The password that will be requested, with the -–ask-pass option, is for the default ICOS administration account which is “admin”. When the playbook requests ssh\_password: the admin password is “broadcom” and when it asks for SUDO password[defaults to SSH password]: a simple return will be needed to default it to the admin password as the same password is used for “sudo” execution when needed. The –v option increases the verbosity of the execution and provides a more detailed output.

The following will be run and output shown on a terminal connection from your PC or directly if working on a workstation.

Example Playbook Output: icos-sw-config.yml

root@DPR-LABVM-01:/root/ansible# ansible-playbook icos-sw-config.yml --ask-pass --ask-become-pass -v

Using /root/ansible/ansible.cfg as config file

SSH password:

SUDO password[defaults to SSH password]:

PLAY [switches] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

TASK [Gathering Facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [10.62.10.34]

TASK [Get Switch Configuration File] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [10.62.10.34] => {

"changed": false,

"dest": "/home/admin/TIER1-CORE-01.cfg",

"gid": 0,

"group": "root",

"mode": "0755",

"owner": "root",

"size": 1167,

"state": "file",

"uid": 0,

"url": "http://10.62.10.22/configs/TIER1-CORE-01.cfg"

}

MSG:

file already exists

TASK [Create startup-config from Configuation File] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [10.62.10.34] => {

"changed": true,

"cmd": [

"cp",

"TIER1-CORE-01.cfg",

"/mnt/fastpath/startup-config"

],

"delta": "0:00:00.004892",

"end": "2018-05-17 21:31:29.648816",

"rc": 0,

"start": "2018-05-17 21:31:29.643924"

}

TASK [ICOS/Linux Configure Switch - sudo icos-cfg] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [10.62.10.34] => {

"changed": true,

"cmd": "icos-cfg -a TIER1-CORE-01.cfg",

"delta": "0:00:00.888642",

"end": "2018-05-17 21:31:31.148768",

"failed\_when\_result": false,

"rc": 230,

"start": "2018-05-17 21:31:30.260126"

}

STDOUT:

Completed apply operation.

MSG:

non-zero return code

TASK [restart switch] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [10.62.10.34] => {

"changed": true,

"cmd": "shutdown -r now \"Ansible updates triggered\"",

"delta": "0:00:00.015116",

"end": "2018-05-17 21:31:31.761950",

"rc": 0,

"start": "2018-05-17 21:31:31.746834"

}

TASK [wait for switch to come back up] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [10.62.10.34 -> localhost] => {

"changed": false,

"elapsed": 54,

"path": null,

"port": 22,

"search\_regex": null,

"state": "started"

}

TASK [Show ICOS running configuration] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [10.62.10.34] => {

"changed": true,

"cmd": "icos-show running-config",

"delta": "0:00:13.890291",

"end": "2018-05-17 21:32:41.991775",

"rc": 0,

"start": "2018-05-17 21:32:28.101484"

}

STDOUT:

Waiting for service to be operational.

!Current Configuration:

!

!System Description "Broadcom Trident2 56854 AG7648 System - 48 10G SFP+ and 6 40G QSFP+, 3.2.2.6, Linux 3.16.0-29-generic, 201412130048"

!System Software Version "3.2.2.6"

!System Up Time "0 days 0 hrs 0 mins 26 secs"

!Cut-through mode is configured as disabled

!Additional Packages BGP-4,QOS,Multicast,IPv6,Routing,Data Center

!Current System Time: May 17 21:32:41 2018

!

vlan database

exit

configure

ip routing

line console

exit

line telnet

exit

line ssh

exit

ip vrf "management"

exit

!

interface loopback 0

no shutdown

ip address 10.0.0.10 255.255.255.255

exit

interface 0/1

no shutdown

routing

ip address 13.1.1.3 255.255.255.254

exit

interface 0/5

no shutdown

routing

ip address 23.1.1.3 255.255.255.254

exit

router ospf

exit

router ospf vrf "management"

exit

ipv6 router ospf

exit

router bgp 65534

bgp router-id 10.0.0.10

maximum-paths 24

neighbor 13.1.1.2 remote-as 64601

neighbor 13.1.1.2 fall-over bfd

neighbor 23.1.1.2 remote-as 64601

neighbor 23.1.1.2 fall-over bfd

redistribute connected

address-family ipv4 vrf "management"

exit

address-family vpnv4 unicast

exit

address-family ipv6

exit

exit

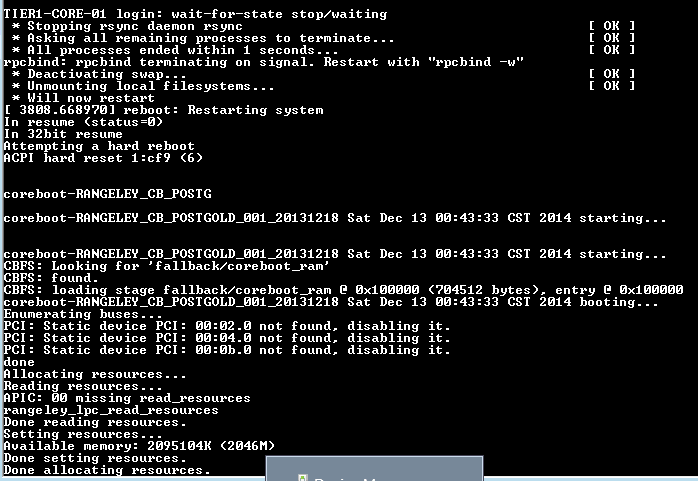
exit

PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

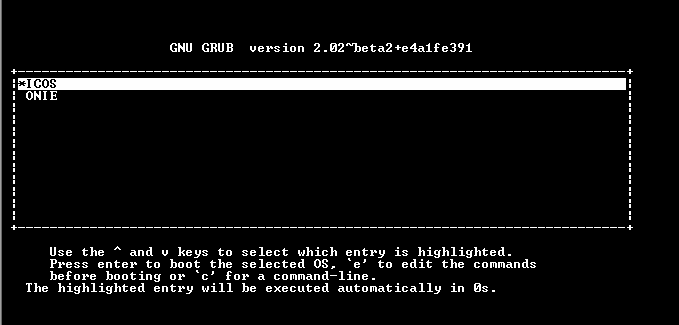
10.62.10.34 : ok=7 changed=4 unreachable=0 failed=0

root@DPR-LABVM-01:/root/ansible#

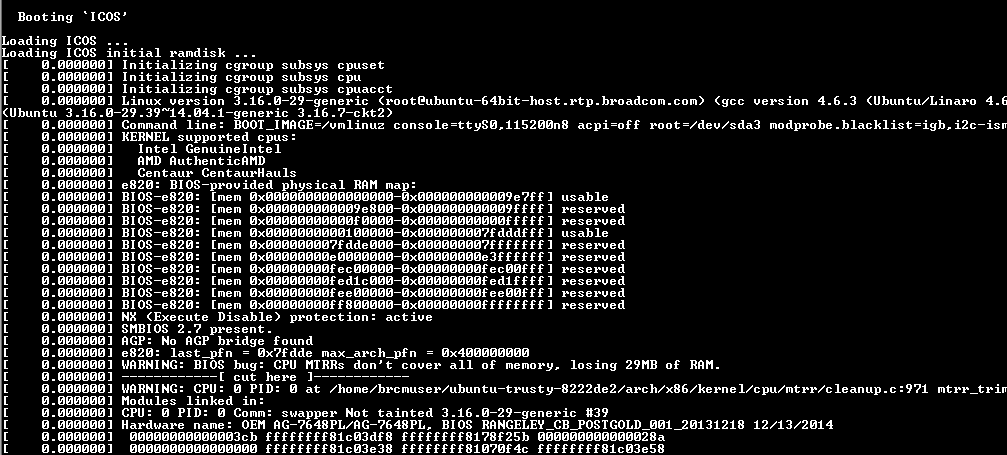
The console connection to the target switch will display the following screens when the playbook is executed and initiates the reboot of the switch:



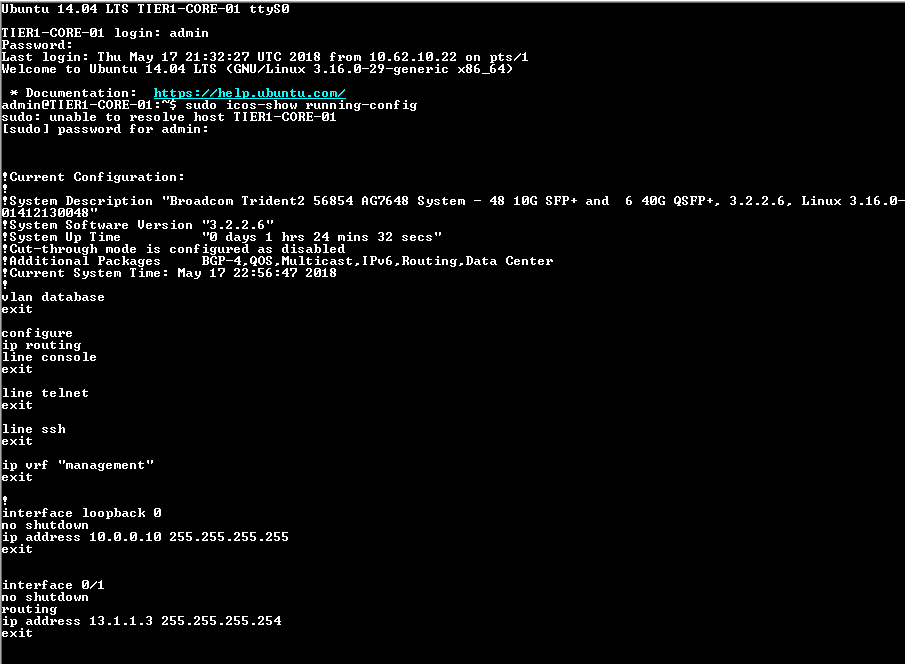
ICOS will default to the current ICOS release that was running on the switch:

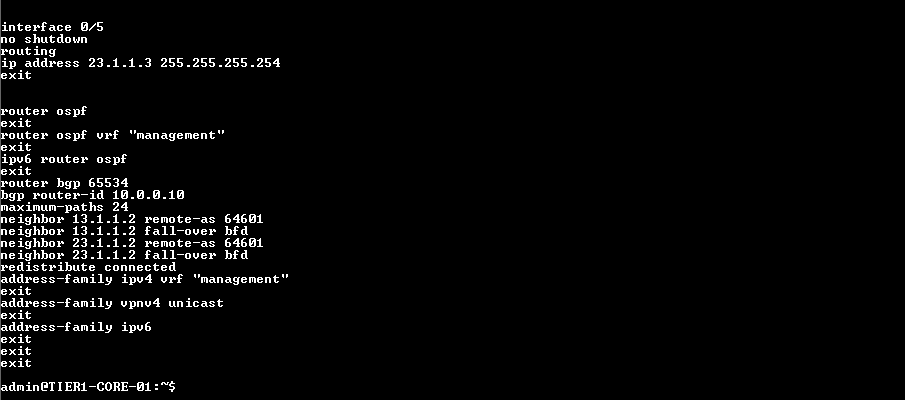


When the ICOS software completes loading the switch will boot into ICOS:



The install completes when the following Ubuntu login screen is displayed. At this prompt you can access the ICOS environment with user ID **admin** and password **broadcom**:





The command icos-show is used, as shown above, with the sudo password ”boradcom” and running-config option to verify the current running config on the switch.

**Example Playbooks and Projects:**

The following git repository has example playbooks created for this guide and other applications that may be useful:

<https://github.com/DeltaProducts/Getting-Started-with-Ansible>

**Example Configuration Files:**

Configuration files for the referenced LAB Topology can be found here:

<https://github.com/DeltaProducts/Getting-Started-with-Ansible/tree/master/configs>

**References:**

Ansible User Guide

<http://docs.ansible.com/ansible/devel/user_guide/intro_getting_started.html>

Ansible Installation Guide

<http://docs.ansible.com/ansible/latest/installation_guide/intro_installation.html>

Ansible ansible-playbook Guide:

<https://docs.ansible.com/ansible/2.4/ansible-playbook.html>

Ansible Commands

<http://docs.ansible.com/ansible/latest/modules/list_of_commands_modules.html>

Ansible Modules

<http://docs.ansible.com/ansible/latest/modules/list_of_all_modules.html>

Tera Term Guide

<https://ttssh2.osdn.jp/index.html.en>

Open Network Install Environment (ONIE) Installation Guide

<https://github.com/DeltaProducts/SolutionCenter/blob/master/ONIE%20recovery%20from%20bootable%20USB.pdf>

Apache Web Server Setup Guide

<https://www.digitalocean.com/community/tutorials/how-to-install-the-apache-web-server-on-ubuntu-16-04>