



# Network Automation with Ansible

April 13, 2018

# Agenda

## Part 1/2 [2 hr.]

- Ansible Concepts
- Lab
- Basic Playbooks
- Lab

## Lab [90 min. home-work]

## Part 2/2 [30 min.]

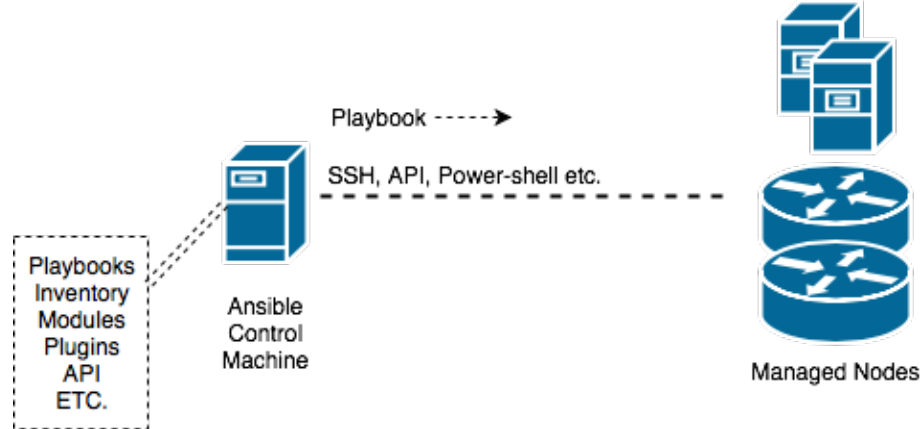
- Roles
- Lab

## Reference

## Acknowledgement

# What is Ansible

- Platform that can automate:
  - Software provisioning
  - Configuration management
  - Application deployment.
- Automation model: 2 components
  - Ansible Control Machine: Linux server with Ansible SW
  - Managed devices: Devices that are being automated by Ansible.
- Ansible Control Machine talks to devices over SSH (and other transports)
- Only requirement on on network devices = enable SSH



# Ansible Concepts: Config File

- Ansible config file can be edited for customization
- To find your Ansible config file, do:
  - `$ ansible --version`
- Other methods to customize:
  - Multiple config files
  - Environmental settings
  - Command line options
  - Roles
- In this session, we will configure the below (uncomment):
  - `inventory = /etc/ansible/hosts`
  - `host_key_checking = False`
  - `timeout = 10`
  - `retry_files_enabled = False`

# Ansible Concepts: Inventory File

- Managed device info is saved in inventory file
- Inventory file path should be uncommented in the config file
- Device info can be listed as individual hosts or groups
- Variables can be assigned to hosts or groups
- Default groups:
  - all
  - ungrouped

## Example:

```
$ cat /etc/ansible/hosts
```

```
[IOS]
172.16.101.91 ansible_user=cisco ansible_ssh_pass=cisco

[XR]
172.16.101.92 ansible_user=cisco ansible_ssh_pass=cisco

[ALL:children]
IOS
XR
```

# Ansible Concepts: Modules

- Modules are the nuts and bolts of Ansible automation tasks
- Playbooks use Modules to execute tasks on the managed devices
- Modules are Operating System specific.
- Example modules:
  - raw
  - ios\_command

```
$ ansible IOS -m raw -a "show ip route sum"
```

```
$ cat ios_sh_ip_route_sum.yml
```

```
---
```

```
- name: route summary from IOS devices
```

```
hosts: IOS
```

```
gather_facts: false
```

```
tasks:
```

```
- raw: sho ip route summary
```

# Ansible Concepts: YAML

- YAML = YAML Ain't Markup Language
- YAML is a data serialization language
- Ansible playbooks are written in YAML
- YAML is intuitive, human readable
- Space indentation is important
- Tab invalid. Use “space” key.
- Check out Youtube links in the reference section

# Ansible Concepts: YAML

- Key-value-pair is represented as:
  - `<key><colon><space><value>`
- List: “ordered data”, represented as:
  - `<dash><space><data>`
- Dictionary: “non-ordered data”
  - Bunch of key-value pairs
- There can be lists of dictionaries and dictionary of lists

- Key-value pair

```
platform: ASR9K
```

- List

```
- show ip int brief
- show ip route summ
```

- Dictionary

```
name: Verify Router OS
hosts: IOS
gather_facts: false
connection: local
```



# Ansible Concepts: Playbooks

- Playbooks are the main means of Ansible automation.
- Playbook is a collection of plays
- Each play is a collection tasks
- Each task is performed by modules

Example structure:

```
- name: play1
  hosts: group1
  tasks:
    - module1: parameters
    - module2: parameters

- name: play2
  hosts: group2
  tasks:
    - module1: parameters
    - module2: parameters
```

# Ansible Concepts: Playbooks

## Example:

- Capture the below data from IOS and XR devices
  - Interface list
  - Route summary
- Playbook:
  - play1
  - play2

```
- name: play1
hosts: IOS
tasks:
  - raw: show ip int br
  - raw: sho ip route summ
```

```
- name: play2
hosts: XR
tasks:
  - raw: show ipv4 int br
  - raw: sho route summ
```

# Basic Playbooks

# Basic Playbooks: ios & xr command module

- Module Names: ios\_command & iosxr\_command
- Module sends exec command to remote devices and returns the results
- Both modules require local connection execution method
- Required Parameters for ios & xr command module:
  - commands option : specify router command to retrieve data

# ios\_command

---

- name: IOS Module Router Config
- hosts: IOS
- gather\_facts: false
- connection: local

tasks:

- name: Collect Router Version and Config

ios\_command:

authorize: yes

commands:

- show version
- show run

register: value

- debug: var=value.stdout\_lines



# iosxr\_command

---

- name: XR Module Router Config
- hosts: XR
- gather\_facts: false
- connection: local

tasks:

- name: Collect Router Version and Config

iosxr\_command:

commands:

- show version
- show ip int bri

register: value

- debug: var=value.stdout\_lines

# Basic Playbooks: Register & Debug

- Basic Playbooks contain register and debug commands.
- Register
  - The “register” statement is used to capture the output of a task into a variable.
  - In previous example, we are saving the output of the show commands to the variable value.
  - Refer: [http://docs.ansible.com/ansible/latest/playbooks\\_conditionals.html#register-variables](http://docs.ansible.com/ansible/latest/playbooks_conditionals.html#register-variables)
- Debug
  - The “debug” module prints statements during playbook execution.
  - The “debug” module takes in a var parameter, which is the variable you want to print.
  - Refer: [http://docs.ansible.com/ansible/latest/debug\\_module.html](http://docs.ansible.com/ansible/latest/debug_module.html)

# Basic Playbooks: ios & xr config module

- Module Names: `ios_config` & `iosxr_config`
- The config modules are used to configure the cisco routers.
- The modules uses parent and line options to structure the configuration in a hierarchical way.
- Both modules require local connection execution method.

# ios\_config

---

- name: IOS Module Router Config
- hosts: IOS
- gather\_facts: false
- connection: local

tasks:

- name: Configure Interface Setting

ios\_config:

parents: "interface Ethernet1"

lines:

- "description test"
- "ip address 172.31.1.1 255.255.255.0"

# iosxr\_config

---

- name: XR Module Router Config
- hosts: XR
- gather\_facts: false
- connection: local

tasks:

- name: Configure Interface Setting

iosxr\_config:

parents: "interface GigabitEthernet0/0/0/0"

lines:

- "description test"
- "ip address 172.31.1.1 255.255.255.0"



# Basic Playbooks: Variables

- Ansible variables are used to store information that will change with each host.

- Variable can be defined:

- inventory file (ansible\_host)
- created directly in the playbook
- created in a separate file and included within the playbook.

- Variables are defined in playbooks

- Using “{{ }}” the single/double quotes around double curly brackets
- Using {{ }} the double curly brackets if its part of a sentence/string

```
---  
- name: Play 1  
  hosts: IOS  
  gather_facts: false  
  vars:  
    host: "{{ ansible_host }}"  
    username: "This variable is {{ ansible_user }}"  
    password: "{{ ansible_ssh_pass }}"
```

# Basic Playbooks: Loops

- Ansible loops are used when repeatedly performing the same task with a set of different items.
- Ansible with\_items loop is a combination of with\_ and lookup().

## With\_items before Ansible Ver 2.5

```
tasks:
  - name: Collect Rtr Ver and Cfg
    ios_command:
      authorize: yes
      commands: "{{ item }}"
```

```
with_items:
  - show version
  - show run
```

## Updated to loop in Ansible Ver 2.5

```
tasks:
  - name: Collect Rtr Ver and Cfg
    ios_command:
      authorize: yes
      commands: "{{ item }}"
```

```
loop:
  - show version
  - show run
```

# Basic Playbooks: Conditionals

- Ansible conditionals are used in a statement to decide whether to run the task or not.
- Ansible uses a **when** clause to dictate a conditional which needs to be true in order for the task to be performed.

```
tasks:
  - name: Collect Router Version
    ios_command:
      authorize: yes
    commands:
      - show ip int bri
    when: ansible_user == "cisco"
```

# Automating Network Operations Tasks

# Network Automation Exercises

- **Exercise 1 – Configure OSPF on all routers**
  - Create Ansible playbook to configure OSPF on both IOS and XR router
  - Setup pre and post checks to ensure OSPF is working correctly
- **Exercise 2 – Automatically backup router's config to server daily**
  - Create Ansible playbook to capture router's running config from all Cisco device types.
  - Setup cron job to execute playbook daily and backup router's config on server.
- **Exercise 3 – Create a playbook to compare two files to find the differences**
  - Create Ansible playbook to find differences between two files.
  - Ex: comparing pre-upgrade and post-upgrade router captures.
- **Exercise 4 – Ansible Vault**
  - Use ansible-vault feature to encrypt sensitive data in inventory files.
  - Ex: comparing pre-upgrade and post-upgrade router captures.

# Conclusion

- Ansible is an open-source, agentless automation tool that can be leveraged for networks configuration management functions.
- Ansible-playbooks provides capabilities to automate daily operations tasks.
- Automating repetitive tasks with Ansible can reduce OPEX costs and improve efficiency.
- With increasing support of modules, it is possible to automate even more network functions through Ansible.

# Reference

# Reference

- Ansible user guide ([http://docs.ansible.com/ansible/latest/user\\_guide/index.html](http://docs.ansible.com/ansible/latest/user_guide/index.html) )
- Ansible installation ([http://docs.ansible.com/ansible/latest/installation\\_guide/intro\\_installation.html](http://docs.ansible.com/ansible/latest/installation_guide/intro_installation.html) )
- YAML resources
  - <http://docs.ansible.com/ansible/latest/YAMLSyntax.html>
  - <http://www.yaml.org>
  - <https://www.youtube.com/watch?v=cdLNKUoMc6c>
  - [https://www.youtube.com/watch?v=U9\\_gfT0n\\_5Q](https://www.youtube.com/watch?v=U9_gfT0n_5Q)
- Ansible Training
  - Ansible for the Absolute Beginner ( <https://www.udemy.com/learn-ansible/learn/v4/overview> )
  - Ansible for Network Engineers (<https://www.udemy.com/ansible-for-network-engineers-cisco-quick-start-gns3-ansible/learn/v4/overview> )
  - Kirk Byers Ansible training (<https://cisco.jiveon.com/docs/DOC-1829500> )
- Dcloud lab (<https://dcloud2-rtp.cisco.com/content/demo/396089?returnPathTitleKey=content-view&isLoggingIn=true> )



# Acknowledgement

# Acknowledgements

- Some material in this session are sourced from Ansible docs
  - <http://docs.ansible.com/ansible/latest/index.html>



*TOMORROW starts here.*