Imtiaz Rahman
SBAC Bank Limited
writeimtiaz@gmail.com
https://imtiazrahman.com

NETWORK AUTOMATION (NetDevOps) WITH ANSIBLE

SANOG32
bdNOC9
02-10 August, 2018
Dhaka, Bangladesh
• **Session 1:**
  - 14:30 PM – 16:00 PM (Theory with example)

• **Session 2:**
  - 16:30 PM – 18:00 PM (Configuration and hands on LAB)
1. Devops/NetDevOps ?
2. Why automation ?
3. Tools for automation
4. Why Ansible ?
5. Ansible introduction
6. Ansible Language Basics
7. Ansible encryption decryption
8. How to run
9. Demo
10. Configuration & Hands on LAB
DevOps

>devops ≠
DevOps integrates developers and operations teams to improve collaboration and productivity by automating infrastructure, workflows and continuously measuring application performance.

Dev + Ops = DevOps
NetDevOps = Networking + DevOps

infrastructure as code
Why automation?

- Avoid repeated task
- Avoid typographical error (Typos)
- Faster deployment
- Identical configuration
Tools for automation

ANSIBLE
CF Engine
SALTSTACK

GitLab
CHEF
puppet labs
What is ANSIBLE?

- Open source IT automation tool
- Red hat Enterprise Linux, CentOS, Debian, OS X, Ubuntu etc.
- Need python
Why ANSIBLE?

• Simple
• Push model
• Agentless
Why ANSIBLE?

Puppet

Puppet master

SSL

Puppet Client/agent

Ansible

Controller node with ansible

Agentless

SSH

Managed node’s
How it works

1. Laptop/Desktop/Server
2. Copy python module
3. Run Module on device
4. Delete Module from device
5. Return result
What can be done??

- Configuration Management
- Provisioning VMs or IaaS instances
- Software Testing
- Continuous Integration/Continuous Deployment (CI/CD)
- Configure hardware switches, routers, firewall etc.
- Other (Ansible can do all of that and much more)
Ansible Container

- Build container images from ansible playbook
- No more Dockerfile
- Create container the same way you deploy to servers
- Deploy to container orchestration platform
- Currently support Docker, OpenShift and Kubernetes
Dockerfile

RUN apt-get update && apt-get install -y \
    aufs-tools \
    automake \
    build-essential \
    curl \
    dpkg-sig \
    libcap-dev \
    libsdl2-dev \
    mercurial \
    reprepro \
    ruby1.9.1 \
    ruby1.9.1-dev \
    s3cmd=1.1.* \
    && rm -rf /var/lib/apt/lists/*

Ansible task

- name: Install Packages
  package:
    name: "{{ packages }}"
    state: present
ANSIBLE terms

- YAML
- Jinja2
- Playbooks
- Facts
- Inventory
- Roles
- Task
- Hosts
ANSIBLE Introduction

Real world

Build a house

Master Plan
(small plan)

work

tools

Ansible world

Configure a device

playbook
(play, play)

tasks

modules

---

- name: load new acl
  ios_config:
    lines:
      name: Add banner
      ios_config:
        lines:

ios_config
ios_command
ANSIBLE Introduction

YAML

- Start with - - -

- File extension .yml/.yaml

- Easy for a human to read

---

- hosts: ios-routers
gather_facts: no
connection: local
tasks:
  - name: Save Configuration
    ios_command:
      commands:
        - write memory
    host: "{{ ansible_host }}"
Playbook

• Tell Ansible what to do

• Send commands to remote devices

• Plain text YAML file

• Each playbook contains one or more plays
---

- name: PLAY START
  hosts: ios-routers
  gather_facts: no
  connection: local

tasks:

- name: LOGIN INFORMATION
  include_vars: secrets.yml

- name: ADD BANNER
  ios_config:
    provider: "{ provider }"
    lines:
      - banner motd ^Welcome to SANOG 32^
Module

- Modules control system resources, packages, files.
- Can be executed directly on remote hosts or through Playbooks
- Over 450 ships with Ansible
- User can also write their own modules
ANSEBLE Introduction (Network modules)

- `asa_acl` - Manage access-lists on a Cisco ASA
- `asa_command` - Run arbitrary commands on Cisco ASA devices
- `eos_banner` - Manage multiline banners on Arista EOS devices
- `eos_config` - Manage Arista EOS configuration sections
- `bigip_command` - Run arbitrary command on F5 devices.
- `bigip_hostname` - Manage the hostname of a BIG-IP.
- `ios_banner` - Manage multiline banners on Cisco IOS devices
- `ios_command` - Run commands on remote devices running Cisco IOS
- `ios_config` - Manage Cisco IOS configuration sections
- `iosxr_command` - Run commands on remote devices running Cisco IOS XR
- `iosxr_config` - Manage Cisco IOS XR configuration sections
- `junos_command` - Run arbitrary commands on an Juniper JUNOS device
- `junos_config` - Manage configuration on devices running Juniper JUNOS

http://docs.ansible.com/ansible/list_of_network_modules.html
Task

- At a basic level, a task is nothing more than a call to an ansible module
- Task run sequentially
- **name**: configure interface settings
  
  ```
  ios_config:
  lines:
    - description test interface
    - ip address 172.31.1.1 255.255.255.0
  parents: interface Ethernet1
  ```

- **name**: load new acl into device
  
  ```
  ios_config:
  lines:
    - 10 permit ip host 1.1.1.1 any log
    - 20 permit ip host 2.2.2.2 any log
  parents: ip access-list extended test
  before: no ip access-list extended test
  match: exact
  ```
ANSIBLE Introduction

Diagram:
- Playbook
  - Play
    - Task
      - Module
  - Play
    - Task
      - Module
  - Play
    - Task
      - Module
---
- hosts: all-ios
  gather_facts: no
  connection: local

tasks:

- name: OBTAIN LOGIN INFORMATION
  include_vars: secrets.yml

- name: DEFINE PROVIDER
  set_fact:
    provider:
      host: "{{ ansible_host }}"
      username: "{{ creds['username'] }}"
      password: "{{ creds['password'] }}"
      auth_pass: "{{ creds['auth_pass'] }}"

- name: ADD BANNER
  ios_config:
    provider: "{{ provider }}"
    authorize: yes
    lines:
      - banner motd ^Welcom to BDNOG9^
ANSIBLE Introduction

Hosts

- List of devices or group of devices where ansible push configuration
- Name and variable assign
- Default location /etc/ansible/hosts
- Can make your own
**Ansible Introduction**

**Hosts file sample**

**INI-like (one of Ansible defaults)**

```
[ios-routers]  
R_2691 ansible_host=192.168.45.3  
R_3745 ansible_host=192.168.45.4

[v6-router]  
R_7200 ansible_host=2001:db8::1001::1
```
ANSIBLE Introduction

Inventory

• **Collections of files or directories inside a directory**

• `ansible-playbook -i <directory-name> playbook.yml`

• **Can have (not mandatory)**
  
  • hosts (file)
  • host_vars (dir)
  • group_vars (dir)

• **Can be accessed across multiple roles**
Roles

- Ansible roles are a special kind of playbook that are fully self-contained with tasks, variables, configuration templates and other supporting files

- Has it’s own directory structure
ANSIBLE Introduction

roles sample

- router_config
  - inventory
    - hosts
  - output
    - SOUTH.cfg
  - playbook.yml
  - roles
    - router
      - tasks
        - main.yml
      - templates
        - router_config.j2
      - vars
        - main.yml
Jinja2

• template engine for the Python programming language
• File extension .j2
• Support conditions, loops
• Variable declaration
{% for interface in cisco_1921_interfaces %}
interface {{ interface }}
    {% if interface == 'GigabitEthernet0/0' %}
description {{ item.int_descp }}
ip address {{ item.ipv4_addp }} {{ item.ipv4_mus }}
    {% elif interface == 'GigabitEthernet0/1' %}
description {{ item.int_descs }}
ip address {{ item.ipv4_adds }} {{ item.ipv4_mus }}
    {% endif %}
    no shutdown
    exit
{% endfor %}

ip route {{ item.static_route1 }} {{ item.static_gw1 }}
ip route {{ item.static_route2 }} {{ item.static_gw1 }}
Ansible Language Basics
Variable
Introduction to ansible variable

- Variable names should be letters, numbers, and underscores.
- Variables should always start with a letter.
- `ispl`, `ISP1`, `isp_dcl`, `ispdc` is valid
- `1ISP_DC`, `10`, `ISP DC` is not valid
Ansible Language Basics: Variable

Variable declaration and assignment

Variables
isp1_dc: 10.x.x.2

Lists
isp :
- isp1_dc: 10.x.x.2
- isp2_dc: 20.x.x.6

Dictionaries
isp :
- isp_dc: 10.x.x.2
  subnet: 255.255.255.252
- isp_dc: 20.x.x.6
  subnet: 255.255.255.248
Ansible Language Basics: Variable

Accessing Variable

Variables

```
{{ isp1_dc }}
```

Lists (looping)

```
{{ item }}
```

Dictionaries (looping)

```
{{ item isp_dc }}
{{ item subnet }}
```
Ansible Language Basics: Variable

host_vars and group_vars

- apricot
  - inventory
    - group_vars
      - all.yml
      - ipv4_router.yml
      - host_vars
        - south_router.yml
        - hosts
    - playbook.yml
    - roles
Ansible Language Basics: Variable

host_vars

Host-specific variables

host_vars/south_router.yml

Variable to be used by south_router host
Ansible Language Basics: Variable

group_vars

Host group-specific variables

group_vars/ipv4_router.yml

Variable to be used by any host in ipv4_router group
Ansible Language Basics

LOOPS
Introduction to Loops

- A loop is an instruction that repeats until a specified condition is reached
- Used for doing the same thing for multiple times
Ansible Language Basics: loops

Types of Loops

- Standard
- Nested
- Do-Until
- for
Ansible Language Basics: loops

cat vars/main.yml

interface_address:
  - INTERFACE: "GigabitEthernet0/0"
    DESC: "ISP1"
    DC_IP: "10.X.X.1"
    MASK: "255.255.255.252"
  - INTERFACE: "GigabitEthernet0/1"
    DESC: "ISP2"
    DC_IP: "172.X.X.5"
    MASK: "255.255.255.252"

cat templates/interface.j2

{% for i in interface_address %}
  interface {{ iINTERFACE }}
    description ->{{ i.DESC }}
    ip address {{ i.DC_IP }} {{ i.MASK }}
    no shutdown
{% endfor %}

Output

templates

roles

interface GigabitEthernet0/0
  description ->>> ISP1
  ip address 10.x.x.1 255.255.255.252

interface GigabitEthernet0/1
  description ->>> ISP2
  ip address 172.x.x.5 255.255.255.252
Comments
Comments in ansible

```
# #
{
#   #
}
```
Conditionals
Ansible Language Basics: conditionals

The `when` statement

---

```
- name: SET IP ADDRESS TO SOUTH ROUTER
  ios_config:
    provider: "{{ provider }}"
    authorize: yes
    parents: "interface FastEthernet0/1"
    lines:
      - description SOUTH-CUSTOMER
      - ip address 10.10.20.1 255.255.255.248
      - ipv6 address 2001:db8:2001::9/64
      after: "no shutdown"
    when: ansible_host == "2001:db8::20"
```
Filters
Introduction to filters

Filters are from jinja 2

used for transforming data inside a template expression

Filters are separated from the variable by a pipe symbol (|)
Ansible Language Basics: filters

**jinja2 filters**

{{ list1 | min }}

replace(s, old, new, count=None)

{{ myvar | ipaddr }}

[http://docs.ansible.com/ansible/latest/playbooks_filters.html](http://docs.ansible.com/ansible/latest/playbooks_filters.html)
ipaddr filter for static routes

```yaml
ip route {{ item.ISP_BR | ipaddr('network') }}
  (Destination network)

{{ item.ISP_BR | ipv4('netmask') }}
  (Subnet mask)

{{ item.ISP_DC | ipaddr('1') | ipaddr('address') }}
  (Gateway)
```
Facts
Ansible Language Basics: facts

Collecting facts

Is a module and called by playbook to gather useful information about remote host

gather_facts: yes/no
Ansible Language Basics

Templating (jinja2)
Ansible Language Basics: jinja2 templating

What can be used?

- Filters
- Condition
- Variable
- Loop
- Many more

Many more
**Ansible Language Basics:** jinja2

**Jinja2 template**

```jinja2
hostname {{ item.hostname }}

{% # Physical interface %}
{% for interface in cisco_1921_int %}
    interface {{ interface }}
    description -&gt; {{ cisco_1921_int[interface].dess }}
    ip address {{ cisco_1921_int[interface].addrs }}
    {{ cisco_1921_int[interface].sub }}
    no shutdown
    exit
{% endfor %}
```
Roles setup
Ansible Language Basics: roles

Roles structure and files

**tasks**
tasks/main.yml

**templates**
templates/router_config.j2

**vars**
vars/main.yml

**files**
files/myscript.sh
Debugging
Ansible Language Basics: debugging

Ansible debugging

Verbose mode `ansible -v`

`error_on_undefined_vars` in `ansible.cfg`

`fail module` with customize messages
Ansible encryption decryption
ANSIBLE Security

Ansible Vault

- It keeps sensitive data such as password, keys, variable name in encrypted format
- Need a password while encrypting, decrypting and running
- `ansible-vault` is the keyword along with encrypt, decrypt, view, etc. parameter
---
---
creds:
  username: "imtiaz"
  password: "password"
  auth_pass: "password"

$ANSIBLE_VAULT;1.1;AES256
6433646431646232666393365366
561613566303362303933343662
30653866373635386261643432

ansible-vault encrypt secretfile.yml
Installing Ansible

Python 2.6 or above for the control machine and python 2.X or later for managed node

yum, rpm, apt-get, emerge, pkg, brew, github

http://docs.ansible.com/ansible/latest/intro_installation.html
How to run

• ansible <inventory> -m

• ansible-playbook

• Ansible tower ($$)
Introduction to Ad-Hoc commands
Demo topology

Ansible host

node 1

node 2

Ubuntu

CentOS
Introduction to Ansible playbook
Demo topology

- **NEW**
  - 192.168.45.5
  - 2001:db8::50/64

- **WEST**
  - 192.168.45.4
  - 2001:db8::40/64

- **SOUTH**
  - 192.168.45.2
  - 2001:db8::20/64

- **EAST**
  - 192.168.45.3
  - 2001:db8::30/64

- **Ansible Host**
  - 192.168.45.1
  - 2001:db8::10/64
Introduction to Ansible role
Demo topology

- NEW
- WEST
  - 192.168.45.4
  - 2001:db8::40/64
- SW1
- EAST
  - 192.168.45.3
  - 2001:db8::30/64
- SOUTH
  - 192.168.45.2
  - 2001:db8::20/64
- Ansible Host
  - 192.168.45.1
  - 2001:db8::10/64
  - 192.168.45.5
  - 2001:db8::50/64
Configuration & Hands on LAB (Session 2)
Configuration and hands on LAB

1. Preparing the environment (access the lab server and router)
2. Ansible installation
3. Playing with ad-hoc command
4. How to write ansible playbook
5. Ansible deep dive with roles, templates, variable and others
6. Ansible GALAXY
Thank You

writeimtiaz@gmail.com
https://imtiazrahman.com

https://github.com/imtiazrahman/SANOG32-NETDEVOPS.git