Using Ansible to Automate Environement Configuration, Application (NSO) Deployment, and Beyond

Hands On Guide

Use case

Requirements

- Use case: Cisco NSO to manage DNS servers and invoke an action of synchronization from DNS master to DNS targets.
- Application deployment: Install Cisco NSO and service packages, device inventory on host N. NSO host manages DNS hosts, M, T1, and T2.
- Security compliance -1 , the communication between NSO to hosts it manages (M, T1 and T2) is limited to no-login, key based ssh.
- Security compliance -2, the transport among DNS hosts is limited to non-interactive, no-login, key based.
- Users:
  - dvans: owns and runs ansible play books
  - dvnso: owns and runs NSO
  - cl00254: device (DNS servers)
  - cl94644: performs synchronization from master to targets

Lab setup

The set up is composed of five VM's: Ansible controller (A), NSO(N), DNS master server (M), and two DNS targets (T1 and T2).

Role of each host

- Ansbile controller (Host A):
    - Ansible playbook host.
- Cisco NSO application VM (Host N):
    - Cisco NSO application host manages DNS master server M.
    - Hosts M, T1, and T2 are also known to NSO.
    - DNS synchronization operation from M to T1 and T2 are initiated from NSO as the service action.
- DNS master M:
    - DNS master managed by NSO. NSO end users can pick and choose the DNS configration portion to synchronize to targets.
- DNS targets T1 and T2:
    - DNS targets in the network

Ansible Playbook Design

- Ansible inventory (hosts) with three groups:
    - nso
    - master
    - targets
- Roles:
    - se
        - tasks: pre-fetch ssh public key files
    - master
        - tasks: sync script install, update authorized keys to allow cl00254. Update sudoers.
    - target
        - tasks: update authorized keys to allow cl00254 and cl94644. Update sudoers.
    - nso
        - tasks: NSO and packages installation and testing

Access your setup

Steps:

- RDP to Jump start server
- putty to your assigned ansible host (A)

Jump start server and VM Assignment

Create and Test Ansible Playbooks

1. Inspect pre-defined directories for the lab.

   Log on to your Ansible controller vm (A), refer Jump start server and VM Assignment for VM assignment and credentials.

   - Check home directory, expect to see ansibleproject, image and package files, and helper scripts are pre-loaded for you.

     Sample output:

     ```
     [dvans@cl-lab-212 ~]$ ls
     ansibleproject      ncs-4.5.0.1-unix- bind-2.0.0.tar.gz      solution
     dns-manager.tar.gz nso-4.5.0.1.linux.x86_64.installer.bin  inventory.tar.gz
       scripts
     ```

   - Inspect `/home/asnibleproject`, expect to see `group_vars`, `hosts`, `roles`, and `vars`.

     Sample output:

     ```
     [dvans@cl-lab-212 ~]$ ls ansibleproject/
     group_vars hosts  roles  vars
     ```

2. Inspect pre-populated Ansible inventory file `/home/dvans/home/ansibleproject/hosts`.

   `hosts` contains the group ip address for hosts (N, M, T1, and T2). Make sure the ip address of NSO matches to Jump start server and VM Assignment

   Sample contents of hosts: hosts.

3. Create roles using ansible-galaxy. `ansible-galaxy init` creates directories roles skeleton directories.

Sample output:

```
[dvans@cl90 ~]$ cd ansibleproject/roles
[dvans@cl90 roles]$ ansible-galaxy init se
- se was created successfully
[dvans@cl90 roles]$ ansible-galaxy init master
- master was created successfully
[dvans@cl90 roles]$ ansible-galaxy init target
- target was created successfully
[dvans@cl90 roles]$ ansible-galaxy init nso
- nso was created successfully
[dvans@cl90 roles]$ ls
master  nso  se  target
```

4. Create playbook to invoke role based tasks.

   `/home/dvans/ansibleproject/cl-playbook.yml` is the playbook calls out all the roles we created in previous step; the associated main.yml play book for each role are executed in the order defined in `cl-playbook.yml` .

   Sample file: cl-playbook.yml

5. Create tasks for role "se". We add this role to ease the key exchange for nso host N, dns master M and dns targets T1/T2. The task of this role is to pre fetch public rsa key files from M, T1 and T2 to ansible controller A. The fetched publick key files are then distributed to proper user's authorized keys files. We define the task in `/home/dvans/ansibleproject/roles/se/tasks/main.yml` .

   Sample file: main.yml

6. Create tasks for role "master". As mentioned in the requirements, dns master M is managed by NSO. To meet the security compliance, the communication between NSO host N and the device M is limited to non-login, non-interactive, key based ssh. One of the tasks is to add rsa public key of N to M. In addition , we define a task to limit sudoers to perform only the allowed operations.

   The DNS synchronization from master to targets is performed by python application tool syncdns from DNS master. Thus,we also need to define a task to install syncdns package onto dns master M.

   For this play, we define tasks in `/home/dvans/ansibleproject/roles/master/tasks/main.yml` .

   Sample file: main.yml

7. Create tasks for role "target". DNS master synchronize end user selected directory to targets. To comply

with the company's security requirements, the communication between master (M) to targets (T1,T2) is no-login, non-interaction, key based ssh. The tasks defined for this role is to add rsa public key to T1 and T2 for peer user, and limit sudoers to perform only the allowed operations. Similar to that for "master", we define tasks in `/home/dvans/ansibleproject/roles/target/tasks/main.yml`.

Sample file: main.yml

8. Create the following tasks for role "nso".

    - Copy images to NSO host.
    - Install NSO.
    - Install packages (ned, service package, and inventory package)
    - Start NSO.
    - Load devices
    - Post check.

    Note, the nso task yml files should be at directory `/home/dvans/ansibleproject/roles/nso/tasks/`.

    All the above nso tasks are implemented with seperate yml files. They are included to main task yml file `main.yml`.

    - `main.yml`, include all the task yml files.

      Sample file: main.yml

    - `nso_copy_images.yml` This yml file uses ansible copy and synchroize modules. Varialbes such as nso_binary, nso_image_path, and etc, are defined under `group_vars/nso`, in previous step.

      Sample file: nso_copy_images.yml

    - `nso_install.yml` This yml file defines play to install NSO and set nso environment.
      Sample file: nso_install.yml

    - `nso_install_packages.yml`, this yml file is to install unix-bind ned, dns manager service package, and inventory package. In this play book, we use block and looping.

      Sample file: nso_install_packages.yml

    - `nso_start.yml` defines a play to start NSO application.

      Sample file: nso_start.yml

    - `nso_add_devices.yml`. This yml file creates devices and service inventory instances for NSO.

We use xml based config files to load merge to NSO's cdb. In this play book, we use templates. The template files, `device.j2` and `inventory.j2` are covered at later step.

Sample file: nso_add_devices.yml

- `nso_postcheck.yml`. In this play book, we pick two actions to make sure the installation is sucessful, rsa keys are exchanged among N,M,T1,T2 to allow required secure communication, and sudoers are set properly.

  Sample file: nso_postcheck.yml

9. Create template files for role "nso"

   Templates are used to create device instances and inventory instances in NSO in `nso*add*devices.yml'.

   Note, below two template files, `device.j2` and `inventory.j2` should be created at `/home/dvans/ansibleproject/roles/nso/templates/` directory.

   - `device.j2`, the xml format device config file with two variables.

     Sample file: device.j2

   - `inventory.j2`, the xml format inventory template file to create inventory model in NSO's cdb. There is no veriable in this template.

     Sample file: inventory.j2

10. Create variables.

    Create inventory group variables. Those variables are used in tasks and templates in later steps. They are defined at `group_vars` directory, with file name same as the group name.

    - Variables for inventory group "nso" is defined in `/home/dvans/ansibleproject/group_vars/nso`.

      Sample file: nso

    - We also defined a variable to be used for install syncdns package. It is pre-defined at `/home/dvans/ansibleproject/vars/labuser`. The sample file below shows the variable for lab user 17.

      Sample file: labuser

11. Testing

Now we are ready to test the top level play book cl-playbook.yml. To execute, we invoke ansible-playbook command `ansible-playbook -i hosts cl-playbook.yml`
We expect it runs through successfully.

Sample output: sample_output

References

1. Asnible Best Practice
2. Cisco Network Services Orchestrator Capabilities
3. Cisco Network Services Orchestrtor