Table of Contents

Enable Ansible logging

- By default, Ansible is not configured to log its output anywhere. To change this behavior by setting the **log_path** configuration setting in your Ansible configuration file (`ansible.cfg`) to allow Ansible to log its output to a specific destination.
- When you add below property in `ansible.cfg` file under `defaults` section

```
[defaults]
log_path = playbooks.log
```

- This would enable Ansible playbooks and ad-hoc commands to log its output to a file named `playbooks.log` in your project directory.
- Now when you execute a playbook the output that will be either successful or error execution will be written in log file.

Ansible Roles

- An Ansible role is a collection of **files, tasks, templates, variables, and handlers** that together serve a certain purpose like installing/configuring a service.
- An Ansible Role is a directory structure contains directories: `defaults, vars, tasks, files, templates, meta, handlers`.
- Each directory must contain a main.yml file which contains relevant content. Let's look little closer to each directory.
  - `tasks`: `main.yml` file contains the role's task definitions
  - `defaults`: `main.yml` file contains the default values of role. Variables in default have the lowest priority so they are easy to override.
  - `vars`: `main.yml` file defines the role's variable values.Variables in vars have higher priority than variables in defaults directory.
  - `files`: This directory contains static files that are referenced by role tasks.
  - `templates`: This directory contains static files that are referenced by role tasks.
  - `meta`: `main.yml` file contains metadata of role like an author, support platforms, dependencies.
  - `handlers`: The main.yml file contains handlers which can be invoked by "notify" directives and are associated with service.

--

- Structuring Ansible playbooks with roles:
  - Use of Ansible roles has the following benefits:
    - Roles group content, allowing easy sharing of code with others
    - Roles can be written that define the essential elements of a system type: web server, database server, File Server, or other purpose.
    - Roles make larger projects more manageable. Roles can be developed in parallel by different administrators.
- Create a roles in project directory and Create a directory structure using **ansible-galaxy** command.

```
sudo yum install tree -y
mkdir roles
ansible-galaxy init roles/webserver
# This will empty files awith a specific directory structure
tree roles/webserver
# Output as below
roles/webserver/
├── defaults
│   └── main.yml
├── files
├── handlers
│   └── main.yml
├── meta
│   └── main.yml
├── README.md
├── tasks
│   └── main.yml
├── templates
├── tests
│   ├── inventory
│   └── test.yml
└── vars
    └── main.yml

8 directories, 8 files
```

--

- Create below files under **roles/webserver**
- **tasks/main.yml**

```
---
# tasks file for roles/webserver
- import_tasks: install.yml
- import_tasks: configure.yml
```

- **tasks/install.yml**

```
---
- name: Install httpd Package
  yum: name=httpd state=latest
```

- **tasks/configure.yml**

```
---
- name: Copy index.j2 template to destination
  template: src=templates/index.j2 dest=/var/www/html/index.html
  notify:
  - restart-webserver
```

- **handlers/main.yml**

```
---
- name: restart-webserver
  service: name=httpd state=restarted
```

--

- **templates/index.j2**

```
<html>
<head><title>My Page</title></head>
<body>
<h1>
Welcome to {{ inventory_hostname }}.
</h1>
<h2>A new feature added.</h2>
</body>
</html>
```

--

- To Execute the Ansible Role, make sure you are inside the main project directory, create below file as:
- **execute_role.yml**

```
---
- hosts: dev
  pre_tasks:
    - debug:
        msg: "Task before any role is applied"
  roles:
   - webserver
```

```
    post_tasks:
      - debug:
          msg: "Task after all role is completed"
```

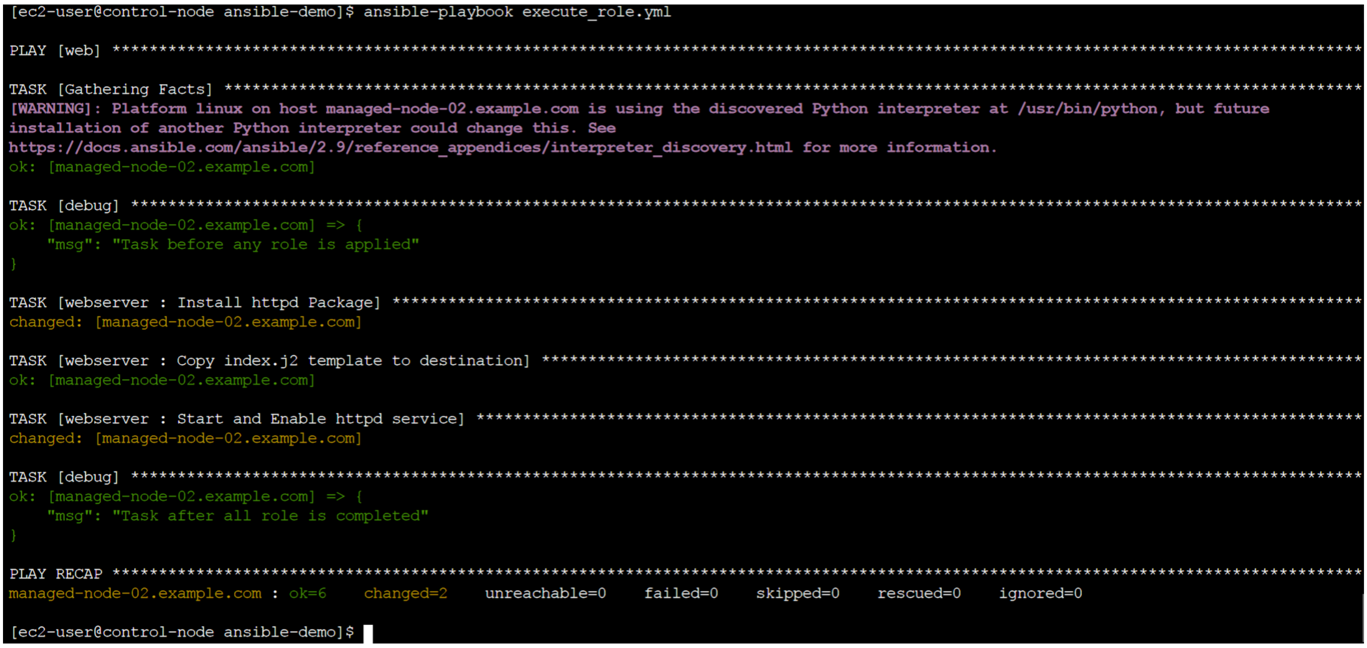
- By default, Ansible looks for roles in two locations:
  - in a directory called **roles/**, relative to the playbook file (current directory)
  - in **/etc/ansible/roles**, global roles path
- Verify the directory structure
- Execute the role

```
ansible-playbook execute_role.yml
```

--

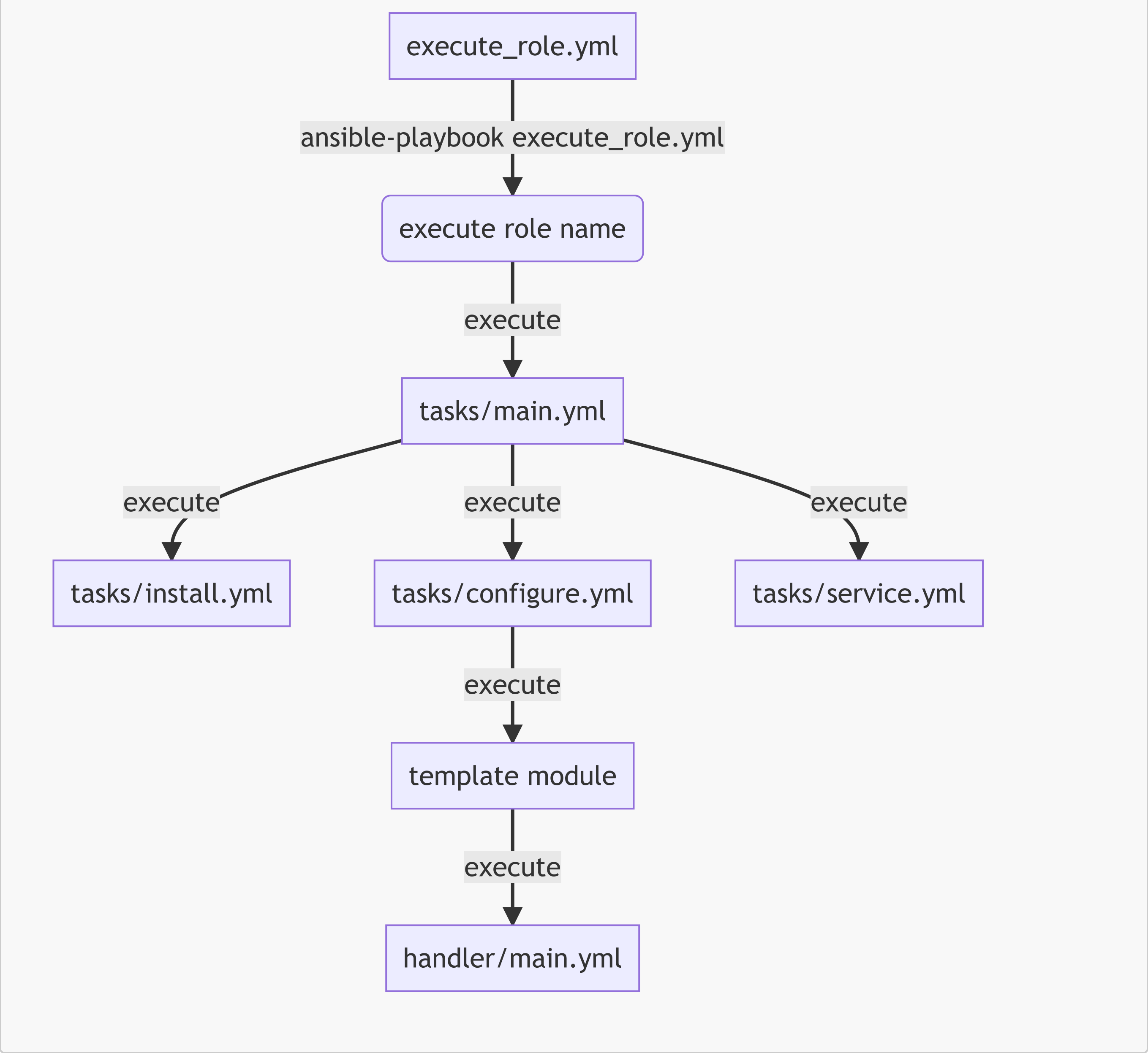
- In summary, Ansible executes your playbook in the following order:
  - `pre_tasks` will run first.
  - statically imported roles listed under roles will run.
  - tasks listed under the **tasks** section.
  - handlers triggered by **roles** or `tasks`.
  - `post_tasks` will run last.

--



```
[ec2-user@control-node ansible-demo]$ ansible-playbook execute_role.yml

PLAY [web] ******************************************************************************************************

TASK [Gathering Facts] ******************************************************************************************
[WARNING]: Platform linux on host managed-node-02.example.com is using the discovered Python interpreter at /usr/bin/python, but future
installation of another Python interpreter could change this. See
https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more information.
ok: [managed-node-02.example.com]

TASK [debug] ****************************************************************************************************
ok: [managed-node-02.example.com] => {
    "msg": "Task before any role is applied"
}

TASK [webserver : Install httpd Package] ************************************************************************
changed: [managed-node-02.example.com]

TASK [webserver : Copy index.j2 template to destination] ********************************************************
ok: [managed-node-02.example.com]

TASK [webserver : Start and Enable httpd service] ***************************************************************
changed: [managed-node-02.example.com]

TASK [debug] ****************************************************************************************************
ok: [managed-node-02.example.com] => {
    "msg": "Task after all role is completed"
}

PLAY RECAP ******************************************************************************************************
managed-node-02.example.com : ok=6    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[ec2-user@control-node ansible-demo]$
```

--



Ansible Cloud Modules

- Ansible supports multiple cloud modules:
  - `ansible-doc -l | grep -i 'aws'`
  - `ansible-doc iam_role`

```
- name: Create a role with description
  iam_role:
    name: mynewrole
    assume_role_policy_document: "{{ lookup('file','policy.json') }}"
    description: This is My New Role
```

--

EC2 instance using Ansible Playbook

- To launch a new ec2 instance using ansible-playbook.
- Ansible provides multiple cloud modules for multiple services.
- Attach EC2-Role to the control node, so that it can launch ec2 instance using ansible

```
sudo yum install python-pip -y
sudo pip install boto
```

- Executing EC2 Ansible Playbook to create and start or stop instance with `ec2` module.

> Delete the AWS Resources that are created for testing.
