IMPLEMENTING ROLES

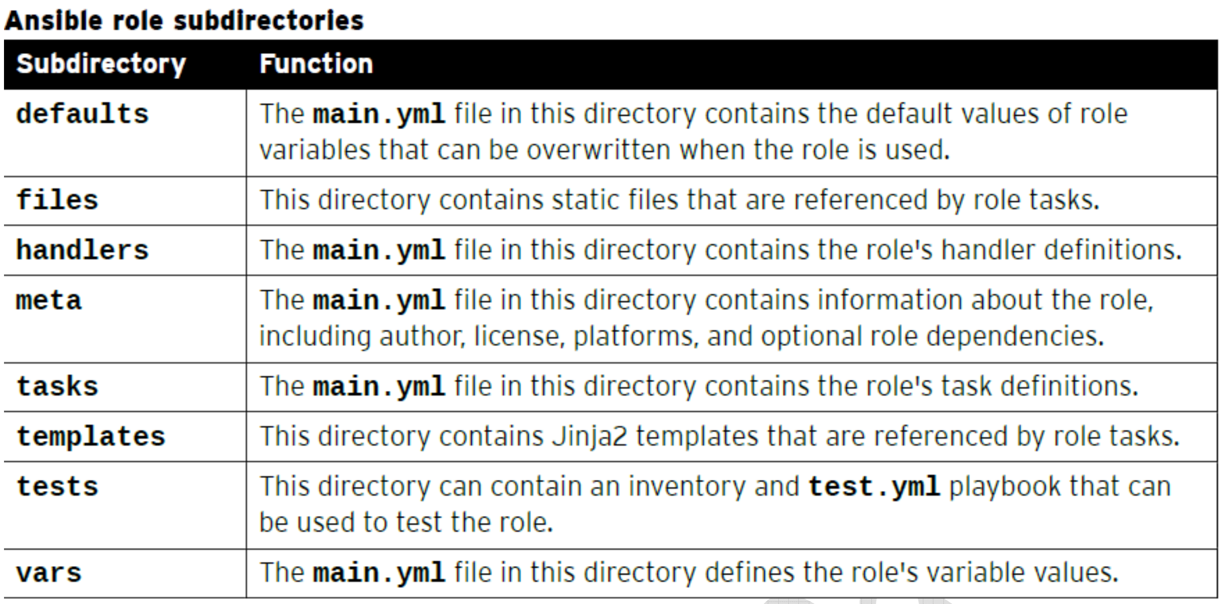
Overview	
Goal	Create and manage roles
Objectives	• Describe the structure and behavior of a role • Create a role • Deploy roles with Ansible Galaxy
Sections	• Describing Role Structure (and Quiz) • Creating Roles (and Guided Exercise) • Deploying Roles with Ansible Galaxy (and Guided Exercise)
Lab	• Implementing Roles

Describing Role Structure

Structuring Ansible playbooks with roles

Datacenters have a variety of different types of hosts. Some serve as web servers, others as database servers, and others can have software development tools installed and configured on them. An Ansible playbook, with tasks and handlers to handle all of these cases, would become large and complex over time. Ansible *roles* allow administrators to organize their playbooks into separate, smaller playbooks and files.
Roles provide Ansible with a way to load tasks, handlers, and variables from external files. Static files and templates can also be associated and referenced by a role. The files that define a role have specific names and are organized in a rigid directory structure, which will be discussed later. Roles can be written so they are general purpose and can be reused.

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, git repository, or other purpose
• Roles make larger projects more manageable
• Roles can be developed in parallel by different administrators


Ansible role subdirectories

Subdirectory	Function
`defaults`	The **`main.yml`** file in this directory contains the default values of role variables that can be overwritten when the role is used.
`files`	This directory contains static files that are referenced by role tasks.
`handlers`	The **`main.yml`** file in this directory contains the role's handler definitions.
`meta`	The **`main.yml`** file in this directory contains information about the role, including author, license, platforms, and optional role dependencies.
`tasks`	The **`main.yml`** file in this directory contains the role's task definitions.
`templates`	This directory contains Jinja2 templates that are referenced by role tasks.
`tests`	This directory can contain an inventory and **`test.yml`** playbook that can be used to test the role.
`vars`	The **`main.yml`** file in this directory defines the role's variable values.

Defining variables and defaults

Role variables are defined by creating a **vars/main.yml** file with key: value pairs in the role directory hierarchy. They are referenced in the role YAML file like any other variable: **{{ VAR_NAME }}.** These variables have a high priority and can not be overridden by inventory variables.
Default variables allow default values to be set for variables of included or dependent roles. They are defined by creating a **defaults/main.yml** file with key: value pairs in the role directory hierarchy. Default variables have the lowest priority of any variables available. They can be easily overridden by any other variable, including inventory variables.
Define a specific variable in either **vars/main.yml** or **defaults/main.yml**, but not in both places. Default variables should be used when it is intended that their values will be overridden.

Using Ansible roles in a playbook

Using roles in a playbook are straightforward. The following example shows how to use Ansible roles.

```
---
- hosts: remote.example.com
  roles:
    - role1
    - role2
```

For each role specified, the role tasks, role handlers, role variables, and role dependencies will be included in the playbook, in that order. Any **copy, script, template, or include tasks** in the role can reference the relevant files, templates, or tasks without absolute or relative path names. Ansible will look for them in the role's files, templates, or tasks respectively, based on requirement.

References

Playbook Roles and Include Statements — Ansible Documentation
http://docs.ansible.com/ansible/playbooks_roles.html

Guided Exercise: Creating Roles

#cd /home/ansible/playbook/dev-roles

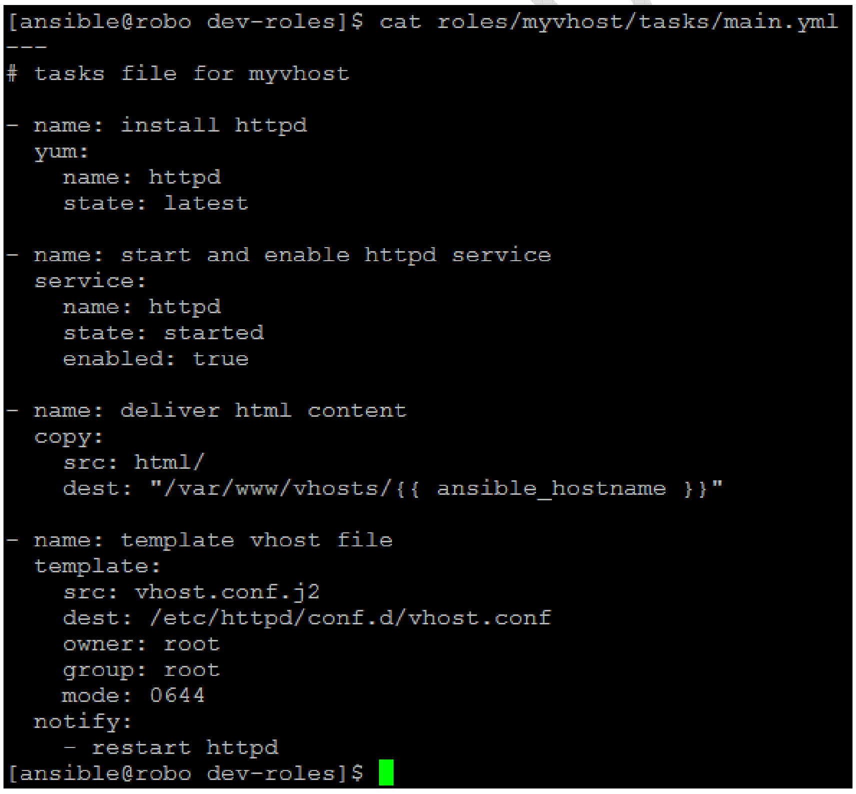
Create the directory structure for a role called **myvhost**. The role will include fixed files, templates, tasks, and handlers. A dependency will be created later, so it should have a meta subdirectory.

#mkdir -p roles/myvhost/{files,handlers}
#mkdir roles/myvhost/{meta,tasks,templates}

Create the **main.yml** file in the **tasks** subdirectory of the role. The role should perform four tasks:
• Install the *httpd* package.
• Start and enable the **httpd** service.
• Download the HTML content into the virtual host **DocumentRoot** directory.
• Install the template configuration file that configures the webserver.

Use a text editor to create a file called **roles/myvhost/tasks/main.yml**. Include code to use the **yum** module to install the *httpd* package.

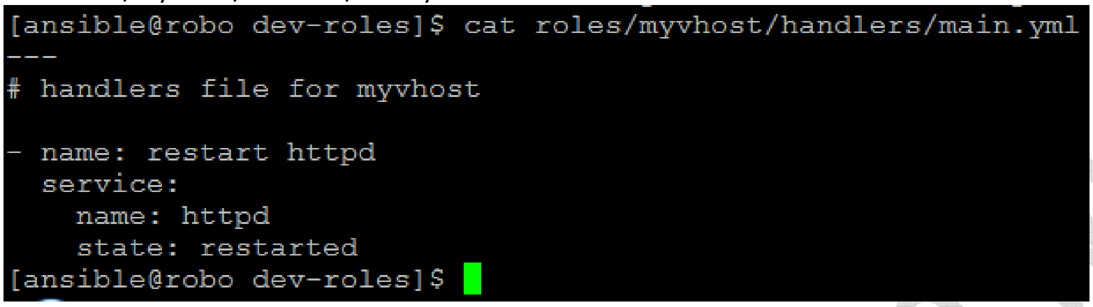
# cat roles/myvhost/tasks/main.yml



```
[ansible@robo dev-roles]$ cat roles/myvhost/tasks/main.yml
---
# tasks file for myvhost

- name: install httpd
  yum:
    name: httpd
    state: latest

- name: start and enable httpd service
  service:
    name: httpd
    state: started
    enabled: true

- name: deliver html content
  copy:
    src: html/
    dest: "/var/www/vhosts/{{ ansible_hostname }}"

- name: template vhost file
  template:
    src: vhost.conf.j2
    dest: /etc/httpd/conf.d/vhost.conf
    owner: root
    group: root
    mode: 0644
  notify:
    - restart httpd
[ansible@robo dev-roles]$
```

Create the handler for restarting the **httpd** service. Use a text editor to create a file called **roles/myvhost/handlers/main.yml**. Include code to use the service module. The file contents should look like the following:

# cat roles/myvhost/handlers/main.yml



```
[ansible@robo dev-roles]$ cat roles/myvhost/handlers/main.yml
---
# handlers file for myvhost

- name: restart httpd
  service:
    name: httpd
    state: restarted
[ansible@robo dev-roles]$
```

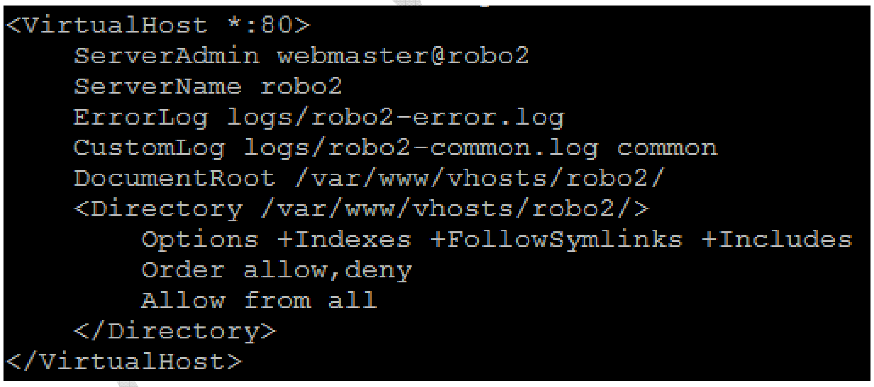
Create the HTML content that will be served by the webserver and Create an index.html file below that directory with the contents: “simple index”. Be sure to use this string verbatim because the grading script looks for it.

```
# mkdir -p roles/myvhost/files/html
# echo 'simple index' > roles/myvhost/files/html/index.html
```

Create a file vhost.conf.j2 and add below lines. (refer the attachment)

vhost.conf.j2

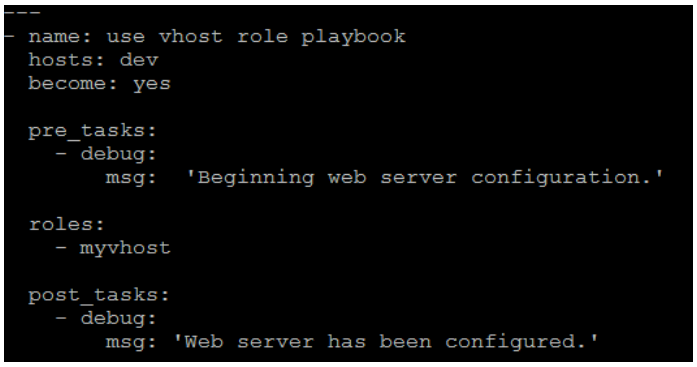
# cat vhost.conf.j2



```
<VirtualHost *:80>
    ServerAdmin webmaster@robo2
    ServerName robo2
    ErrorLog logs/robo2-error.log
    CustomLog logs/robo2-common.log common
    DocumentRoot /var/www/vhosts/robo2/
    <Directory /var/www/vhosts/robo2/>
        Options +Indexes +FollowSymlinks +Includes
        Order allow,deny
        Allow from all
    </Directory>
</VirtualHost>
```

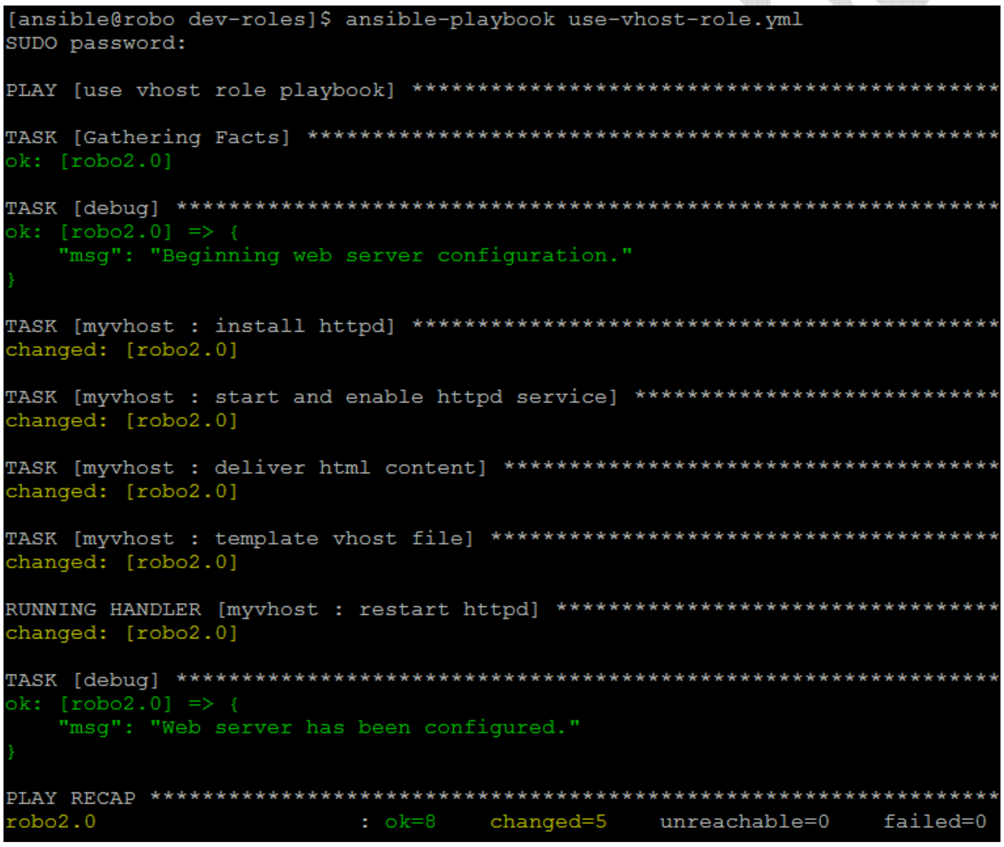
# mv vhost.conf.j2 roles/myvhost/templates/

Write a playbook that uses the role, called **use-vhost-role.yml**. It should have the following content:
# cat use-vhost-role.yml



```
---
- name: use vhost role playbook
  hosts: dev
  become: yes

  pre_tasks:
    - debug:
        msg:  'Beginning web server configuration.'

  roles:
    - myvhost

  post_tasks:
    - debug:
        msg: 'Web server has been configured.'
```

#ansible-playbook –syntax-check use-vhost-role.yml
#ansible-playbook use-vhost-role.yml



```
[ansible@robo dev-roles]$ ansible-playbook use-vhost-role.yml
SUDO password:

PLAY [use vhost role playbook] ********************************************

TASK [Gathering Facts] ****************************************************
ok: [robo2.0]

TASK [debug] **************************************************************
ok: [robo2.0] => {
    "msg": "Beginning web server configuration."
}

TASK [myvhost : install httpd] ********************************************
changed: [robo2.0]

TASK [myvhost : start and enable httpd service] ***************************
changed: [robo2.0]

TASK [myvhost : deliver html content] *************************************
changed: [robo2.0]

TASK [myvhost : template vhost file] **************************************
changed: [robo2.0]

RUNNING HANDLER [myvhost : restart httpd] *********************************
changed: [robo2.0]

TASK [debug] **************************************************************
ok: [robo2.0] => {
    "msg": "Web server has been configured."
}

PLAY RECAP ****************************************************************
robo2.0                    : ok=8    changed=5    unreachable=0    failed=0
```

<u>Output: -</u>
#curl -S 192.168.56.5
simple index

Note: - kindly uninstall the package once the playbook executed successfully.

Deploying Roles with Ansible Galaxy

Ansible Galaxy

Ansible Galaxy [https://galaxy.ansible.com] is a public library of Ansible roles written by a variety of Ansible administrators and users. It is an archive that contains thousands of Ansible roles and it has a searchable database that helps Ansible users identify roles that might help them accomplish an administrative task. Ansible Galaxy includes links to documentation and videos for new Ansible users and role developers.

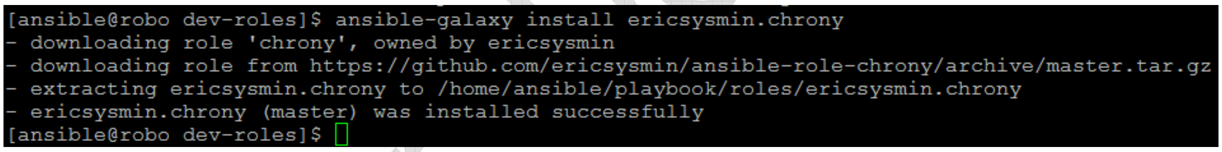
References

Ansible Galaxy — Ansible Documentation
http://docs.ansible.com/ansible/galaxy.html

Guided Exercise: Deploying Roles with Ansible Galaxy

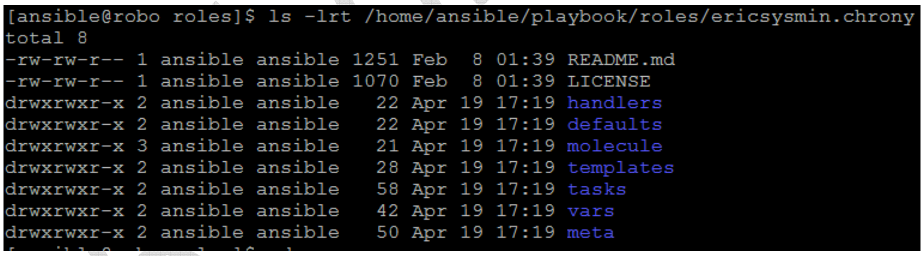
How to download and use the Ansible role from galaxy.
use the **ansible-galaxy** command to utilize the requirements file you just created to download and install the ericsysmin.chrony role.

#cd /home/ansible/playbook/
#ansible-galaxy install ericsysmin.chrony



```
[ansible@robo dev-roles]$ ansible-galaxy install ericsysmin.chrony
- downloading role 'chrony', owned by ericsysmin
- downloading role from https://github.com/ericsysmin/ansible-role-chrony/archive/master.tar.gz
- extracting ericsysmin.chrony to /home/ansible/playbook/roles/ericsysmin.chrony
- ericsysmin.chrony (master) was installed successfully
[ansible@robo dev-roles]$
```

#ls -lrt /home/ansible/playbook/roles/ericsysmin.chrony

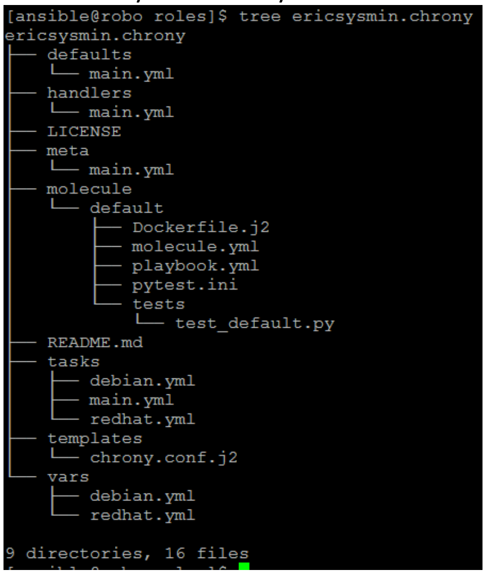


```
[ansible@robo roles]$ ls -lrt /home/ansible/playbook/roles/ericsysmin.chrony
total 8
-rw-rw-r-- 1 ansible ansible 1251 Feb  8 01:39 README.md
-rw-rw-r-- 1 ansible ansible 1070 Feb  8 01:39 LICENSE
drwxrwxr-x 2 ansible ansible   22 Apr 19 17:19 handlers
drwxrwxr-x 2 ansible ansible   22 Apr 19 17:19 defaults
drwxrwxr-x 3 ansible ansible   21 Apr 19 17:19 molecule
drwxrwxr-x 2 ansible ansible   28 Apr 19 17:19 templates
drwxrwxr-x 2 ansible ansible   58 Apr 19 17:19 tasks
drwxrwxr-x 2 ansible ansible   42 Apr 19 17:19 vars
drwxrwxr-x 2 ansible ansible   50 Apr 19 17:19 meta
```

Verify the Downloaded role in your role directory.

#cd /home/ansible/playbook/roles

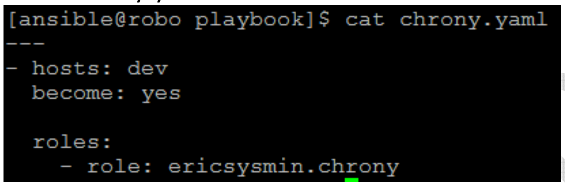
# tree ericsysmin.chrony



```
[ansible@robo roles]$ tree ericsysmin.chrony
ericsysmin.chrony
├── defaults
│   └── main.yml
├── handlers
│   └── main.yml
├── LICENSE
├── meta
│   └── main.yml
├── molecule
│   └── default
│       ├── Dockerfile.j2
│       ├── molecule.yml
│       ├── playbook.yml
│       ├── pytest.ini
│       └── tests
│           └── test_default.py
├── README.md
├── tasks
│   ├── debian.yml
│   ├── main.yml
│   └── redhat.yml
├── templates
│   └── chrony.conf.j2
└── vars
    ├── debian.yml
    └── redhat.yml

9 directories, 16 files
```

Create a playbook to call crony roles to configure on a remote server.

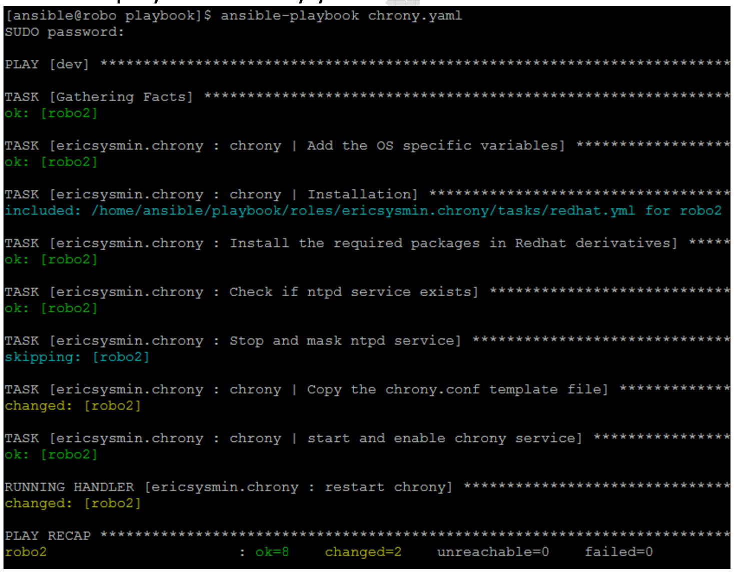
# cat chrony.yaml



```
[ansible@robo playbook]$ cat chrony.yaml
---
- hosts: dev
  become: yes

  roles:
    - role: ericsysmin.chrony
```

# ansible-playbook --syntax-check chrony.yaml

# ansible-playbook chrony.yaml



```
[ansible@robo playbook]$ ansible-playbook chrony.yaml
SUDO password:

PLAY [dev] ***********************************************************************

TASK [Gathering Facts] ***********************************************************
ok: [robo2]

TASK [ericsysmin.chrony : chrony | Add the OS specific variables] ****************
ok: [robo2]

TASK [ericsysmin.chrony : chrony | Installation] *********************************
included: /home/ansible/playbook/roles/ericsysmin.chrony/tasks/redhat.yml for robo2

TASK [ericsysmin.chrony : Install the required packages in Redhat derivatives] ****
ok: [robo2]

TASK [ericsysmin.chrony : Check if ntpd service exists] **************************
ok: [robo2]

TASK [ericsysmin.chrony : Stop and mask ntpd service] ****************************
skipping: [robo2]

TASK [ericsysmin.chrony : chrony | Copy the chrony.conf template file] ***********
changed: [robo2]

TASK [ericsysmin.chrony : chrony | start and enable chrony service] **************
ok: [robo2]

RUNNING HANDLER [ericsysmin.chrony : restart chrony] *****************************
changed: [robo2]

PLAY RECAP ***********************************************************************
robo2                      : ok=8    changed=2    unreachable=0    failed=0
```

To Verify the status of service on remote machine.

#ansible dev -a 'systemctl status chronyd.service' -b