Red Hat Enterprise Linux Automation with Ansible

FEEDBACK TRANSLATIONS CERTIFICATE OF ATTENDANCE

Guided Exercise: Deploying Roles with Ansible Galaxy ☆

In this exercise, you will use Ansible Galaxy to download and install an Ansible role.

Outcomes

You should be able to:

- create a roles file to specify role dependencies for a playbook
- install roles specified in a roles file
- list roles using the `ansible-galaxy` command

Scenario Overview

Your organization places custom files in the `/etc/skel` directory on all hosts. As a result, new user accounts are configured with a standardized organization-specific Bash environment.

You will test the development version of the Ansible role responsible for deploying Bash environment skeleton files.

Log in to `workstation` as `student` using `student` as the password.

On `workstation`, run the `lab role-galaxy start` command. This creates the working directory, `/home/student/role-galaxy`, and populates it with an Ansible configuration file and host inventory.

```
[student@workstation ~]$ lab role-galaxy start
```

Procedure 7.3. Instructions

1. Change to the `role-galaxy` working directory.

   ```
   [student@workstation ~]$ cd ~/role-galaxy
   [student@workstation role-galaxy]$
   ```

2. To test the Ansible role that configures skeleton files, add the role specification to a roles file.

   Launch your favorite text editor and create a file called `requirements.yml` in the `roles` subdirectory. The URL of the role's Git repository is: `git@workstation.lab.example.com:student/bash_env`. To see how the role affects the behavior of production hosts, use the `master` branch of the repository. Set the local name of the role to `student.bash_env`.

   The `roles/requirements.yml` now contains the following content:

```
---
# requirements.yml

- src: git@workstation.lab.example.com:student/bash_env
  scm: git
  version: master
  name: student.bash_env
```

3. Use the `ansible-galaxy` command to process the roles file you just created and install the `student.bash_env` role.

   3.1. For comparison, display the contents of the `roles` subdirectory before the role is installed.

   ```
   [student@workstation role-galaxy]$ ls roles/
   requirements.yml
   ```

   3.2. Use Ansible Galaxy to download and install the roles listed in the `roles/requirements.yml` file. Be sure that any downloaded roles are stored in the `roles` subdirectory.

   ```
   [student@workstation role-galaxy]$ ansible-galaxy install -r \
   > roles/requirements.yml -p roles
   - extracting student.bash_env to /home/student/role-galaxy/roles/student.bash_env
   - student.bash_env (master) was installed successfully
   ```

   3.3. Display the `roles` subdirectory after the role has been installed. Confirm that it has a new subdirectory called `student.bash_env`, matching the `name` value specified in the YAML file.

   ```
   [student@workstation role-galaxy]$ ls roles/
   requirements.yml  student.bash_env
   ```

   3.4. Try using the `ansible-galaxy` command, without any options, to list the project roles:

   ```
   [student@workstation role-galaxy]$ ansible-galaxy list
   # /usr/share/ansible/roles
   ...output omitted...
   - rhel-system-roles.postfix, (unknown version)
   - rhel-system-roles.selinux, (unknown version)
   - rhel-system-roles.timesync, (unknown version)
   # /etc/ansible/roles
    [WARNING]: - the configured path /home/student/.ansible/roles does not exist.
   ```

   Because you used the `-p` option with the `ansible-galaxy install` command, the `student.bash_env` role was not installed in the default location. Use the `-p` option with the `ansible-galaxy list` command to list the downloaded roles:

   ```
   [student@workstation role-galaxy]$ ansible-galaxy list -p roles
   # /home/student/role-galaxy/roles
   - student.bash_env, master
   ...output omitted...
   [WARNING]: - the configured path /home/student/.ansible/roles does not exist.
   ```

   > **NOTE**
   >
   > The `/home/student/.ansible/roles` directory is in your default `roles_path`, but since you have not attempted to install a role without using the `-p` option, `ansible-galaxy` has not yet created the directory.

4. Create a playbook, called `use-bash_env-role.yml`, that uses the `student.bash_env` role. The contents of the playbook should match the following:

```
---
- name: use student.bash_env role playbook
  hosts: devservers
  vars:
    default_prompt: '[\u on \h in \W dir]\$ '
  pre_tasks:
    - name: Ensure test user does not exist
      user:
        name: student2
        state: absent
        force: yes
        remove: yes

  roles:
    - student.bash_env

  post_tasks:
    - name: Create the test user
      user:
        name: student2
        state: present
        password: "{{ 'redhat' | password_hash('sha512', 'mysecretsalt') }}"
```

To see the effects of the configuration change, a new user account must be created. The `pre_tasks` and `post_tasks` section of the playbook ensure that the `student2` user account is created each time the playbook is executed. After playbook execution, the `student2` account is accessed with a password of `redhat`.

> **NOTE**
>
> The `user2` password is generated using a filter. Filters take data and modify it; here, the string `redhat` is modified by passing it to the `password_hash` module. Filters are an advanced topic not covered in this course.

5. Run the playbook. The `student.bash_env` role creates standard template configuration files in `/etc/skel` on the managed host. The files it creates include `.bashrc`, `.bash_profile`, and `.vimrc`.

```
[student@workstation role-galaxy]$ ansible-playbook use-bash_env-role.yml

PLAY [use student.bash_env role playbook] ***********************************

TASK [Gathering Facts] ******************************************************
ok: [servera.lab.example.com]

TASK [Ensure test user does not exist] **************************************
ok: [servera.lab.example.com]

TASK [student.bash_env : put away .bashrc] **********************************
changed: [servera.lab.example.com]

TASK [student.bash_env : put away .bash_profile] ****************************
ok: [servera.lab.example.com]

TASK [student.bash_env : put away .vimrc] ***********************************
changed: [servera.lab.example.com]

TASK [Create the test user] *************************************************
changed: [servera.lab.example.com]

PLAY RECAP ******************************************************************
servera.lab.example.com    : ok=6    changed=4    unreachable=0    failed=0
```

6. Connect to `servera` as the `student2` user using SSH. Observe the custom prompt for the `student2` user, and then disconnect from `servera`.

```
[student@workstation role-galaxy]$ ssh student2@servera
Activate the web console with: systemctl enable --now cockpit.socket

[student2 on servera in ~ dir]$ exit
logout
Connection to servera closed.
[student@workstation role-galaxy]$
```

7. Execute the playbook using the development version of the `student.bash_env` role.

   The development version of the role is located in the `dev` branch of the Git repository. The development version of the role uses a new variable, `prompt_color`. Before executing the playbook, add the `prompt_color` variable to the `vars` section of the playbook and set its value to `blue`.

   7.1. Update the `roles/requirements.yml` file, and set the `version` value to `dev`. The `roles/requirements.yml` file now contains:

   ```
   ---
   # requirements.yml

   - src: git@workstation.lab.example.com:student/bash_env
     scm: git
     version: dev
     name: student.bash_env
   ```

   7.2. Use the `ansible-galaxy install` command to install the role using the updated roles file. Use the `--force` option to overwrite the existing `master` version of the role with the `dev` version of the role.

   ```
   [student@workstation role-galaxy]$ ansible-galaxy install \
   > -r roles/requirements.yml --force -p roles
   - changing role student.bash_env from master to dev
   - extracting student.bash_env to /home/student/role-galaxy/roles/student.bash_env
   - student.bash_env (dev) was installed successfully
   ```

   7.3. Edit the `use-bash_env-role.yml` file. Add the `prompt_color` variable with a value of `blue` to the `vars` section of the playbook. The file now contains:

   ```
   ---
   - name: use student.bash_env role playbook
     hosts: devservers
     vars:
       prompt_color: blue
       default_prompt: '[\u on \h in \W dir]\$ '
     pre_tasks:
   ...output omitted...
   ```

   7.4. Execute the `use-bash_env-role.yml` playbook.

```
[student@workstation role-galaxy]$ ansible-playbook use-bash_env-role.yml

PLAY [use student.bash_env role playbook] ***********************************

TASK [Gathering Facts] ******************************************************
ok: [servera.lab.example.com]

TASK [Ensure test user does not exist] **************************************
changed: [servera.lab.example.com]

TASK [student.bash_env : put away .bashrc] **********************************
changed: [servera.lab.example.com]

TASK [student.bash_env : put away .bash_profile] ****************************
changed: [servera.lab.example.com]

TASK [student.bash_env : put away .vimrc] ***********************************
okay: [servera.lab.example.com]

TASK [Create the test user] *************************************************
changed: [servera.lab.example.com]

PLAY RECAP ******************************************************************
servera.lab.example.com    : ok=6    changed=4    unreachable=0    failed=0
```

8. Connect again to `servera` as the `student2` using SSH. Observe the error for the `student2` user, and then disconnect from `servera`.

```
[student@workstation role-galaxy]$ ssh student2@servera
Activate the web console with: systemctl enable --now cockpit.socket

-bash: [: missing `]'
[student2@servera ~]$ exit
logout
Connection to servera closed.
[student@workstation role-galaxy]$
```

A Bash error occurred while parsing the `student2` user's `.bash_profile` file.

9. Correct the error in the development version of the `student.bash_env` role, and re-execute the playbook.

9.1. Edit the `roles/student.bash_env/templates/_bash_profile.j2` file. Add the missing `]` character to line 4 and save the file. The top of the file is now:

```
# .bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
        . ~/.bashrc
fi

# User specific environment and startup programs

PATH=$PATH:$HOME/.local/bin:$HOME/bin

export PATH
```

Save the file.

9.2. Execute the `use-bash_env-role.yml` playbook.

```
[student@workstation role-galaxy]$ ansible-playbook use-bash_env-role.yml

PLAY [use student.bash_env role playbook] ***********************************

TASK [Gathering Facts] ******************************************************
ok: [servera.lab.example.com]

TASK [Ensure test user does not exist] **************************************
changed: [servera.lab.example.com]

TASK [student.bash_env : put away .bashrc] **********************************
ok: [servera.lab.example.com]

TASK [student.bash_env : put away .bash_profile] ****************************
changed: [servera.lab.example.com]

TASK [student.bash_env : put away .vimrc] ***********************************
ok: [servera.lab.example.com]

TASK [Create the test user] *************************************************
changed: [servera.lab.example.com]

PLAY RECAP ******************************************************************
servera.lab.example.com    : ok=6    changed=3    unreachable=0    failed=0
```

9.3. Connect again to `servera` as the `student2` using SSH.

```
[student@workstation role-galaxy]$ ssh student2@servera
Activate the web console with: systemctl enable --now cockpit.socket

[student2 on servera in ~ dir]$ exit
logout
Connection to servera closed.
[student@workstation role-galaxy]$
```

The error message is no longer present. The custom prompt for the `student2` user now displays with blue characters.

0. The steps above demonstrate that the development version of the `student.bash_env` role is defective. Based on testing results, developers will commit necessary fixes back to the development branch of the role. When the development branch passes required quality checks, developers merge features from the development branch into the `master` branch.

   Committing role changes to a Git repository is beyond the scope of this course.

> **IMPORTANT**
>
> When tracking the latest version of a role in a project, periodically reinstall the role to update it. This ensures that the local copy stays current with bug fixes, patches, and other features.
>
> However, if using a third-party role in production, specify the version to use in order to avoid breakage due to unexpected changes. Periodically update to the latest role version in your test environment so as to adopt improvements and changes in a controlled manner.

Finish

Run the `lab role-galaxy finish` command to clean up the managed host.

```
[student@workstation ~]$ lab role-galaxy finish
```

This concludes the guided exercise.

← PREVIOUS (/ROL/APP/COURSES/RH294-8.4/PAGES/CH07S07) → NEXT (/ROL/APP/COURSES/RH294-8.4/PAGES/CH07S09)

Red Hat Training Policies (http://s.bl-1.com/h/cZrb2DXG?url=https://www.redhat.com/en/about/red-hat-training-policies)

Terms of Use (https://www.redhat.com/en/about/terms-use)

All policies and guidelines (https://www.redhat.com/en/about/all-policies-guidelines)

Release Notes (https://learn.redhat.com/t5/Red-Hat-Learning-Subscription/Red-Hat-Learning-Subscription-Release-Notes/ba-p/22952)

Cookie Preferences