Red Hat Enterprise Linux Automation with Ansible

Guided Exercise: Modifying and Copying Files to Hosts ☆

In this exercise, you will use standard Ansible modules to create, install, edit, and remove files on managed hosts and manage the permissions, ownership, and SELinux contexts of those files.

Outcomes

You should be able to:

- Retrieve files from managed hosts, by host name, and store them locally.
- Create playbooks that use common file management modules such as `copy`, `file`, `lineinfile`, and `blockinfile`.

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

On `workstation`, run the `lab file-manage start` command. The script creates the `file-manage` project directory, and downloads the Ansible configuration file and the host inventory file needed for the exercise.

```
[student@workstation ~]$ lab file-manage start
```

Procedure 5.1. Instructions

1. As the `student` user on `workstation`, change to the `/home/student/file-manage` working directory. Create a playbook called `secure_log_backups.yml` in the current working directory. Configure the playbook to use the `fetch` module to retrieve the `/var/log/secure` log file from each of the managed hosts and store them on the control node. The playbook should create the `secure-backups` directory with subdirectories named after the host name of each managed host. Store the backup files in their respective subdirectories.

   1.1. Navigate to the `/home/student/file-manage` working directory.

   ```
   [student@workstation ~]$ cd ~/file-manage
   [student@workstation file-manage]$
   ```

   1.2. Create the `secure_log_backups.yml` playbook with initial content:

   ```
   ---
   - name: Use the fetch module to retrieve secure log files
     hosts: all
     remote_user: root
   ```

1.3. Add a task to the `secure_log_backups.yml` playbook that retrieves the `/var/log/secure` log file from the managed hosts and stores it in the `~/file-manage/secure-backups` directory. The `fetch` module creates the `~/file-manage/secure-backups` directory if it does not exist. Use the `flat: no` parameter to ensure the default behavior of appending the host name, path, and file name to the destination:

```
  tasks:
    - name: Fetch the /var/log/secure log file from managed hosts
      fetch:
        src: /var/log/secure
        dest: secure-backups
        flat: no
```

1.4. Before running the playbook, run the `ansible-playbook --syntax-check secure_log_backups.yml` command to verify its syntax. Correct any errors before moving to the next step.

```
[student@workstation file-manage]$ ansible-playbook --syntax-check \
> secure_log_backups.yml

playbook: secure_log_backups.yml
```

1.5. Run `ansible-playbook secure_log_backups.yml` to execute the playbook:

```
[student@workstation file-manage]$ ansible-playbook secure_log_backups.yml
PLAY [Use the fetch module to retrieve secure log files] ******************

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

TASK [Fetch the /var/log/secure file from managed hosts] ******************
changed: [serverb.lab.example.com]
changed: [servera.lab.example.com]

PLAY RECAP ****************************************************************
servera.lab.example.com    : ok=2    changed=1    unreachable=0    failed=0
serverb.lab.example.com    : ok=2    changed=1    unreachable=0    failed=0
```

1.6. Verify the playbook results:

```
[student@workstation file-manage]$ tree -F secure-backups
secure-backups
├── servera.lab.example.com/
│   └── var/
│       └── log/
│           └── secure
└── serverb.lab.example.com/
    └── var/
        └── log/
            └── secure
```

2. Create the `copy_file.yml` playbook in the current working directory. Configure the playbook to copy the `/home/student/file-manage/files/users.txt` file to all managed hosts as the root user.

2.1. Add the following initial content to the `copy_file.yml` playbook:

```
---
- name: Using the copy module
  hosts: all
  remote_user: root
```

2.2. Add a task to use the `copy` module to copy the `/home/student/file-manage/files/users.txt` file to all managed hosts. Use the `copy` module to set the following parameters for the `users.txt` file:

Parameter	Values
src	files/users.txt
dest	/home/devops/users.txt
owner	devops
group	devops
mode	u+rw,g-wx,o-rwx
setype	samba_share_t

```
  tasks:
    - name: Copy a file to managed hosts and set attributes
      copy:
        src: files/users.txt
        dest: /home/devops/users.txt
        owner: devops
        group: devops
        mode: u+rw,g-wx,o-rwx
        setype: samba_share_t
```

2.3. Use the `ansible-playbook --syntax-check copy_file.yml` command to verify the syntax of the `copy_file.yml` playbook.

```
[student@workstation file-manage]$ ansible-playbook --syntax-check copy_file.yml

playbook: copy_file.yml
```

2.4. Run the playbook:

```
[student@workstation file-manage]$ ansible-playbook copy_file.yml
PLAY [Using the copy module] ***************************************

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

TASK [Copy a file to managed hosts and set attributes] *************
changed: [servera.lab.example.com]
changed: [serverb.lab.example.com]

PLAY RECAP *********************************************************
servera.lab.example.com : ok=2    changed=1    unreachable=0    failed=0
serverb.lab.example.com : ok=2    changed=1    unreachable=0    failed=0
```

2.5. Use an ad hoc command to execute the `ls -Z` command as user `devops` to verify the attributes of the `users.txt` file on the managed hosts.

```
[student@workstation file-manage]$ ansible all -m command -a 'ls -Z' -u devops
servera.lab.example.com | CHANGED | rc=0 >>
unconfined_u:object_r:samba_share_t:s0 users.txt

serverb.lab.example.com | CHANGED | rc=0 >>
unconfined_u:object_r:samba_share_t:s0 users.txt
```

3. In a previous step, the `samba_share_t` SELinux type field was set for the `users.txt` file. However, it is now determined that default values should be set for the SELinux file context.

   Create a playbook called `selinux_defaults.yml` in the current working directory. Configure the playbook to use the `file` module to ensure the default SELinux context for `user`, `role`, `type`, and `level` fields.

> **NOTE**
>
> In the real world you would also edit `copy_file.yml` and remove the `setype` keyword.

3.1. Create the `selinux_defaults.yml` playbook:

```
---
- name: Using the file module to ensure SELinux file context
  hosts: all
  remote_user: root
  tasks:
    - name:  SELinux file context is set to defaults
      file:
        path: /home/devops/users.txt
        seuser: _default
        serole: _default
        setype: _default
        selevel: _default
```

3.2. Use the `ansible-playbook --syntax-check selinux_defaults.yml` command to verify the syntax of the `selinux_defaults.yml` playbook.

```
[student@workstation file-manage]$ ansible-playbook --syntax-check \
> selinux_defaults.yml

playbook: selinux_defaults.yml
```

3.3. Run the playbook:

```
[student@workstation file-manage]$ ansible-playbook selinux_defaults.yml
PLAY [Using the file module to ensure SELinux file context] ************

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

TASK [SELinux file context is set to defaults] *************************
changed: [serverb.lab.example.com]
changed: [servera.lab.example.com]

PLAY RECAP *************************************************************
servera.lab.example.com : ok=2    changed=1    unreachable=0    failed=0
serverb.lab.example.com : ok=2    changed=1    unreachable=0    failed=0
```

3.4. Use an ad hoc command to execute the `ls -Z` command as user `devops` to verify the default file attributes of `unconfined_u:object_r:user_home_t:s0`.

```
[student@workstation file-manage]$ ansible all -m command -a 'ls -Z' -u devops
servera.lab.example.com | CHANGED | rc=0 >>
unconfined_u:object_r:user_home_t:s0 users.txt

serverb.lab.example.com | CHANGED | rc=0 >>
unconfined_u:object_r:user_home_t:s0 users.txt
```

4. Create a playbook called `add_line.yml` in the current working directory. Configure the playbook to use the `lineinfile` module to append the line `This line was added by the lineinfile module.` to the `/home/devops/users.txt` file on all managed hosts.

4.1. Create the `add_line.yml` playbook:

```
---
- name: Add text to an existing file
  hosts: all
  remote_user: devops
  tasks:
    - name: Add a single line of text to a file
      lineinfile:
        path: /home/devops/users.txt
        line: This line was added by the lineinfile module.
        state: present
```

4.2. Use `ansible-playbook --syntax-check add_line.yml` command to verify the syntax of the `add_line.yml` playbook.

```
[student@workstation file-manage]$ ansible-playbook --syntax-check add_line.yml

playbook: add_line.yml
```

4.3. Run the playbook:

```
[student@workstation file-manage]$ ansible-playbook add_line.yml
PLAY [Add text to an existing file] **********************************

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

TASK [Add a single line of text to a file] ***************************
changed: [servera.lab.example.com]
changed: [serverb.lab.example.com]

PLAY RECAP ***********************************************************
servera.lab.example.com : ok=2    changed=1    unreachable=0    failed=0
serverb.lab.example.com : ok=2    changed=1    unreachable=0    failed=0
```

4.4. Use the `command` module with the `cat` option, as the `devops` user, to verify the content of the `users.txt` file on the managed hosts.

```
[student@workstation file-manage]$ ansible all -m command \
> -a 'cat users.txt' -u devops
serverb.lab.example.com | CHANGED | rc=0 >>
This line was added by the lineinfile module.

servera.lab.example.com | CHANGED | rc=0 >>
This line was added by the lineinfile module.
```

5. Create a playbook called `add_block.yml` in the current working directory. Configure the playbook to use the `blockinfile` module to append the following block of text to the `/home/devops/users.txt` file on all managed hosts.

```
This block of text consists of two lines.
They have been added by the blockinfile module.
```

5.1. Create the `add_block.yml` playbook:

```
---
- name: Add block of text to a file
  hosts: all
  remote_user: devops
  tasks:
    - name: Add a block of text to an existing file
      blockinfile:
        path: /home/devops/users.txt
        block: |
          This block of text consists of two lines.
          They have been added by the blockinfile module.
        state: present
```

5.2. Use the `ansible-playbook --syntax-check add_block.yml` command to verify the syntax of the `add_block.yml` playbook.

```
[student@workstation file-manage]$ ansible-playbook --syntax-check add_block.yml

playbook: add_block.yml
```

5.3. Run the playbook:

```
[student@workstation file-manage]$ ansible-playbook add_block.yml

PLAY [Add block of text to a file] ************************************

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

TASK [Add a block of text to an existing file] ****************************
changed: [servera.lab.example.com]
changed: [serverb.lab.example.com]

PLAY RECAP ************************************************************
servera.lab.example.com    : ok=2    changed=1    unreachable=0    failed=0
serverb.lab.example.com    : ok=2    changed=1    unreachable=0    failed=0
```

5.4. Use the `command` module with the `cat` command to verify the correct content of the `/home/devops/users.txt` file on the managed host.

```
[student@workstation file-manage]$ ansible all -m command \
> -a 'cat users.txt' -u devops
serverb.lab.example.com | CHANGED | rc=0 >>
This line was added by the lineinfile module.
# BEGIN ANSIBLE MANAGED BLOCK
This block of text consists of two lines.
They have been added by the blockinfile module.
# END ANSIBLE MANAGED BLOCK

servera.lab.example.com | CHANGED | rc=0 >>
This line was added by the lineinfile module.
# BEGIN ANSIBLE MANAGED BLOCK
This block of text consists of two lines.
They have been added by the blockinfile module.
# END ANSIBLE MANAGED BLOCK
```

6. Create a playbook called `remove_file.yml` in the current working directory. Configure the playbook to use the `file` module to remove the `/home/devops/users.txt` file from all managed hosts.

6.1. Create the `remove_file.yml` playbook:

```
---
- name: Use the file module to remove a file
  hosts: all
  remote_user: devops
  tasks:
    - name: Remove a file from managed hosts
      file:
        path: /home/devops/users.txt
        state: absent
```

6.2. Use the `ansible-playbook --syntax-check remove_file.yml` command to verify the syntax of the `remove_file.yml` playbook.

```
[student@workstation file-manage]$ ansible-playbook --syntax-check remove_file.yml

playbook: remove_file.yml
```

6.3. Run the playbook:

```
[student@workstation file-manage]$ ansible-playbook remove_file.yml
PLAY [Use the file module to remove a file] ******************************

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

TASK [Remove a file from managed hosts] **********************************
changed: [serverb.lab.example.com]
changed: [servera.lab.example.com]

PLAY RECAP ***************************************************************
servera.lab.example.com    : ok=2    changed=1    unreachable=0    failed=0
serverb.lab.example.com    : ok=2    changed=1    unreachable=0    failed=0
```

6.4. Use an ad hoc command to execute the `ls -l` command to confirm that the `users.txt` file no longer exists on the managed hosts.

```
[student@workstation file-manage]$ ansible all -m command -a 'ls -l' -u devops
serverb.lab.example.com | CHANGED | rc=0 >>
total 0

servera.lab.example.com | CHANGED | rc=0 >>
total 0
```

Finish

On `workstation`, run the `lab file-manage finish` script to clean up this exercise.

```
[student@workstation ~]$ lab file-manage finish
```

This concludes the guided exercise.

← PREVIOUS (/ROL/APP/COURSES/RH294-8.4/PAGES/CH05) → NEXT (/ROL/APP/COURSES/RH294-8.4/PAGES/CH05S03)

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

Cookie Preferences