Ansible Roles

This document introduces Ansible roles, which offer an elegant way of organizing playbooks.

This content is **supplemental**. You are under no obligation to learn this material. However, students with a particular interest in infrastructure will benefit from using them in the project.

Tasks and Playbooks

Tasks are Ansible's equivalent of commands. They allow you to specify some configuration you want Ansible to perform on the target machine.

```
name: Create user
user:
  name: user
  comment: Silly test user
  group: admin
```

Several tasks that perform a related function are called a **playbook**.

```
# Create user and enable sudo
name: Create admin user
user:
  name: admin
  password: $6$883282938423usrdl;ghj2o348uysjdf;
  group: admin sudo

name: Update /etc/sudoers
copy:
  src: sudoers
  dest: /etc/sudoers
```

Ideally, a playbook should do just one thing, instead of several. For example, you should use one playbook to install Filebeat, and a different playbook to install Metricbeat.

This allows you to more easily maintain and reuse each playbook on different VMs.

However, Ansible doesn't provide a way to run many playbooks at once. Instead, you must use a feature called **roles**.

Roles: Directory Structure

A **role** is a playbook dedicated to a single task, such as installing Filebeat. A role might be responsible for performing the following tasks: - Downloading and unarching a file. - Installing software. - Copying configuration files into place. - Starting services.

The example role that installs Filebeat must do all of these things.

Writing a role is only slightly different from writing a playbook. The one difference is in how you organize your directory structure.

Instead of having a single playbook called `main.yml`, your directory structure will look like:

```
/etc/
  ansible/
    inventory
    hosts
    main.yml
    roles/
      install-filebeat/
      install-metricbeat/
```

In other words, you will: - Add a `roles` directory under `ansible`. - Create a new directory for each role you want to create.

Note that there is still a `main.yml`, but instead of including a long list of tasks, it will contain a list of the roles you want to run:

```
- hosts: elkservers
- become: True
- roles:
  - install-elk

- hosts: dvwa
- become: True
- roles:
  - install-filebeat
  - install-metricbeat
```

This example will run `install-filebeat` and `install-metricbeat` on the DVWA VMs, and `install-elk` on `elkservers`.

Each role directory will have its own contents, as well.

Roles: Basic Files

Inside of each role directory, you will have the following files and folders:

```
install-filebeat/
  files/
    example.config
  tasks/
    main.yml
```

This directory can contain many other files and folders, but these are the only ones you will need for the project.

Note the following: - `main.yml` is the same as the playbooks you have been writing. It contains the tasks required to complete the role. - `files` contains files you need to upload to the target VM. - This is particularly relevant to the `copy` module. - When you use the `copy` module in `tasks/main.yml`, you must specify a path to the source file, and a path to the target VM, as in: `bash name: Copy configuration to target VM copy: src: example.config dest: /etc/example.config` - If you pass a simple file name to `src`, Ansible will look for the file in the `files` directory.

- This means you should always put files you want a role to upload in the role's `files` directory. *Never* pass an absolute path outside of the role folder.

Roles in the Project If you are new to Ansible, feel free to implement your playbook in a single file.

If you want an additional challenge, create the following directory structure on your Ansible VM:

```
/etc/
  ansible/
    roles/
      install-elk
        files/
        tasks/
          main.yml
      install-filebeat
        files/
          filebeat.yml
        tasks/
          main.yml
      install-metricbeat/
        files/
          metricbeat.yml
        tasks/
          main.yml
```

The `filebeat.yml` and `metricbeat.yml` are provided in the Resources directory. You will need to edit them lightly before they work, as instructed in the project activity file.

After creating this structure, you can put the Ansible tasks specific to each challenge in the appropriate `tasks/main.yml`. This will make for a more organized project that is much easier to test.

Again, this work should be done as a bonus, not a requirement. But be encouraged to give it a try. You know everything you need to know to use roles, and they are considered the gold standard for configuration within the infrastructure community.

If you're up for the challenge, they're a powerful boost to your resume, and a valuable skill.
