| **Name:Denzel A. Dinglasan** | **Date Performed: 02/10/2023** |
| --- | --- |
| **Course/Section:CPE31S6** | **Date Submitted: 09/10/2023** |
| **Instructor: Dr. Jonathan Vidal Taylar** | **Semester and SY: 1st Sem 2023-2024** |
| **Activity 7: Managing Files and Creating Roles in Ansible** | |
| 1. **Objectives:**    1. Manage files in remote servers    2. Implement roles in ansible | |
| 1. **Discussion**:   In this activity, we look at the concept of copying a file to a server. We are going to create a file into our git repository and use Ansible to grab that file and put it into a particular place so that we could do things like customize a default website, or maybe install a default configuration file. We will also implement roles to consolidate plays. | |
| **Task 1: Create a file and copy it to remote servers**   1. Using the previous directory we created, create a directory, and named it “*files*.” Create a file inside that directory and name it “*default\_site.html*.” Edit the file and put basic HTML syntax. Any content will do, as long as it will display text later. Save the file and exit.        1. Edit the *site.yml* file and just below the *web\_servers* play, create a new file to copy the default html file for site:  * name: copy default html file for site   tags: apache, apache2, httpd  copy:  src: default\_site.html  dest: /var/www/html/index.html  owner: root  group: root  mode: 0644     1. Run the playbook *site.yml*. Describe the changes.        1. Go to the remote servers (*web\_servers*) listed in your inventory. Use cat command to check if the index.html is the same as the local repository file (*default\_site.html*). Do both for Ubuntu and CentOS servers. On the CentOS server, go to the browser and type its IP address. Describe the output.            1. Sync your local repository with GitHub and describe the changes. | |
| **Task 2: Download a file and extract it to a remote server**   1. Edit the site.yml. Just before the web\_servers play, create a new play:  * hosts: workstations   become: true  tasks:  - name: install unzip  package:  name: unzip  - name: install terraform  unarchive:  src: <https://releases.hashicorp.com/terraform/0.12.28/terraform_0.12.28_linux_amd64.zip>  dest: /usr/local/bin  remote\_src: yes  mode: 0755  owner: root  group: root     1. Edit the inventory file and add workstations group. Add any Ubuntu remote server. Make sure to remember the IP address.      1. Run the playbook. Describe the output.        1. On the Ubuntu remote workstation, type terraform to verify installation of terraform. Describe the output. | |
| **Task 3: Create roles**   1. Edit the site.yml. Configure roles as follows: (make sure to create a copy of the old site.yml file because you will be copying the specific plays for all groups)     Save the file and exit.       1. Under the same directory, create a new directory and name it roles. Enter the roles directory and create new directories: base, web\_servers, file\_servers, db\_servers and workstations. For each directory, create a directory and name it tasks.          1. Go to tasks for all directory and create a file. Name it main.yml. In each of the tasks for all directories, copy and paste the code from the old site.yml file. Show all contents of main.yml files for all tasks.                1. Run the site.yml playbook and describe the output.             **https://github.com/ddinglasan/HOA7.1\_Dinglasan.git** | |
| **Reflections:**  Answer the following:   * 1. What is the importance of creating roles?   To make it easier to manage and organize automated tasks in an Ansible playbook. Setting roles hides and reuse sets of tasks, variables, and files.   * 1. What is the importance of managing files?   It is crucial for organizing, securing, and accessing data efficiently.  **Conclusion:**  In this activity, I learned how to properly manage files in a remote server and create roles of tasks for certain hosts. It makes the tasks manageable and encapsulated. | |