Writing Loops and Conditional Tasks

In this Lab you will write a playbook containing tasks that have conditionals and loops.

Introduction:

Task Iteration with Loops

Using loops saves administrators from the need to write multiple tasks that use the same module. For example, instead of writing five tasks to ensure five users exist, you can write one task that iterates over a list of five users to ensure they all exist. Ansible supports iterating a task over a set of items using the loop keyword. You can configure loops to repeat a task using each item in a list, the contents of each of the files in a list, a generated sequence of numbers, or using more complicated structures. This section covers simple loops that iterate over a list of items. Consult the documentation for more advanced looping scenarios.

1. Log in as **root** with password as **linux**

1.1 Let us create the directory called data-loop directory

```
# mkdir data-loop
# cd data-loop
```

1.2 Let us create a Playbook manifest that adds a new user on the target system using the user module as shown:

```
# wget
https://raw.githubusercontent.com/EyesOnCloud/ansible/mai
n/create_users.yaml
```

Output:

```
[root@ansi-master data-loop]# wget https://raw.githubusercontent.com/EyesOnCloud/ansible/main/create_users.yaml
--2021-12-11 15:32:31--  https://raw.githubusercontent.com/EyesOnCloud/ansible/main/create_users.yaml
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.109.133, 185.199.110.133, 185.199.111.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.109.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 197 [text/plain]
Saving to: 'create_users.yaml'

create_users.yaml                   100%[=================================================================>]     197

2021-12-11 15:32:31 (9.24 MB/s) - 'create_users.yaml' saved [197/197]
```

Lets view the manifest

```
# cat -n create_users.yaml
```

Output:

```
[root@ansi-master data-loop]# cat -n create_users.yaml
     1  ---
     2  - hosts: ansi-node1
     3    become: yes
     4    tasks:
     5      - name: Create new users
     6        user:
     7          name: '{{ item }}'
     8          state: present
     9        loop:
    10          - john
    11          - mike
    12          - andrew
```

1.3 Verify its syntax is correct by running ansible-playbook –syntax-check

```
# ansible-playbook --syntax-check create_users.yaml
```

Output:

```
[root@ansi-master ~]# ansible-playbook --syntax-check create_users.yaml

playbook: create_users.yaml
```

1.4 Let us run the play book

```
# ansible-playbook -v create_users.yaml
```

Output:

```
[root@ansi-master data-loop]# ansible-playbook -v create_users.yaml
Using /etc/ansible/ansible.cfg as config file

PLAY [ansi-node1] ***************************************************************************************************

TASK [Gathering Facts] **********************************************************************************************
ok: [ansi-node1]

TASK [Create new users] *********************************************************************************************
ok: [ansi-node1] => (item=john) => {"ansible_loop_var": "item", "append": false, "changed": false, "comment": "", "
hn", "item": "john", "move_home": false, "name": "john", "shell": "/bin/bash", "state": "present", "uid": 1001}
ok: [ansi-node1] => (item=mike) => {"ansible_loop_var": "item", "append": false, "changed": false, "comment": "", "
ke", "item": "mike", "move_home": false, "name": "mike", "shell": "/bin/bash", "state": "present", "uid": 1002}
ok: [ansi-node1] => (item=andrew) => {"ansible_loop_var": "item", "append": false, "changed": false, "comment": ""
andrew", "item": "andrew", "move_home": false, "name": "andrew", "shell": "/bin/bash", "state": "present", "uid":

PLAY RECAP **********************************************************************************************************
ansi-node1                 : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

1.5 Let's verify the loop task on the ansi-node1 host

```
# ssh ansi-node1 getent passwd | cut -d: -f1 | tail 5
```

Output: