Create directories and files

- We can use the **file** module to create files and directories (like touch), manage permissions and ownership on files and directories, modify SELinux properties, and create symlinks.

#To Create a directory:

ansible -m file -a "dest=/tmp/test mode=644 state=directory" all

#To check the folder

ansible all -a "ls /tmp“

#To fetch the information about the folder

ansible all -m stat -a "path=/tmp/test"

Cont.

#To create a empty file

ansible -m file -a "path=/tmp/test/t1.txt mode=644 state=touch" all

#To verify the file

ansible all -a "ls /tmp/test“

#Add content to the file

ansible all -m copy -a "content='This is a sample text file.' dest=/tmp/test/t1.txt“

#To verify the content

ansible all -a "cat /tmp/test/t1.txt"

Cont.

#To create a symlink [**state=link**]

ansible -m file -a "path=/tmp/test/t1.txt mode=644 state=touch state=link" all

#To change file owner

owner=root

#To change file group

group=root

#To change file mode

mode = 644

mode = "u=rw,g=r,o=r"

Cont.

Delete directories and files

#To delete file

ansible all -m file -a "dest=/tmp/test/t1.txt state=absent“

#To verify the file presence

ansible all -a "ls /tmp/test“

#To delete the folder

ansible all -m file -a "dest=/tmp/test/ state=absent"

#To verify the presence

ansible all -a "ls /tmp"

Discovered Facts

- Facts are bits of information derived from examining a host systems that are stored as variables for later use in a play.

Collect and display the discovered facts for the localhost

ansible localhost -m setup

```
[root@ip-172-31-21-214 ~]# ansible localhost -m setup
localhost | SUCCESS => {
    "ansible_facts": {
        "ansible_all_ipv4_addresses": [
            "172.31.21.214"
        ],
        "ansible_all_ipv6_addresses": [
            "fe80::88:76ff:fe3d:cd0"
        ],
        "ansible_apparmor": {
            "status": "disabled"
        },
        "ansible_architecture": "x86_64",
        "ansible_bios_date": "08/24/2006",
        "ansible_bios_version": "4.2.amazon",
```

Extract Individual Fact information

- Display facts from all hosts and store them indexed by hostname at (/tmp/facts).

  ansible all -m setup --tree /tmp/facts

- Display only facts regarding memory found by ansible on all hosts and output them.

  ansible -m setup devservers -a 'filter=ansible_*_mb'

- Display only facts about certain interfaces.

  ansible all -m setup -a 'filter=ansible_eth[0-2]'

Cont.

```
[root@ip-172-31-21-214 DevOps]# ansible -m setup devservers -a 'filter=ansible_uptime_seconds'
172.31.20.197 | SUCCESS => {
    "ansible_facts": {
        "ansible_uptime_seconds": 4783
    },
    "changed": false
}

[root@ip-172-31-21-214 DevOps]# ansible -m setup devservers -a 'filter=ansible_processor_cores'
172.31.20.197 | SUCCESS => {
    "ansible_facts": {
        "ansible_processor_cores": 1
    },
    "changed": false
}
```

Plays & Playbooks

- Plays are ordered sets of tasks to execute against host selections from inventory.
- Ansible performs automation and orchestration of IT environments via Playbooks.
- Playbooks are a YAML definition of automation tasks that describe how a particular piece of automation should be done.
- Playbooks are written in YAML, a simple human-readable syntax popular for defining configuration.

Cont.

- Ansible Playbooks consist of series of ‘plays’ that define automation across a set of hosts, known as the ‘inventory’.
- Each ‘play’ consists of multiple ‘tasks,’ that can target one, many, or all of the hosts in the inventory.
- Each task is a call to an Ansible module - a small piece of code for doing a specific task.
- These tasks can be simple, such as placing a configuration file on a target machine, or installing a software package.
- Ansible includes hundreds of modules, ranging from simple configuration management, to managing network devices, to modules for maintaining infrastructure on every major cloud provider.

Cont.

- Ansible are written in a manner to allow for easy configuration of desired state - they check that the task that is specified actually needs to be done before executing it.
- For example, if an Ansible task is defined to start a webserver, configuration is only done if the webserver is not already started.
- This desired state configuration, sometimes referred to as 'idempotency,' ensures that configuration can be applied repeatedly without side effects, and that configuration runs quickly and efficiently when it has already been applied.
- **Idempotence** is the ability to run an operation which produces the same result whether run once or multiple times

Cont.

- A playbook will generally consist of the following elements:

Tasks — This can be used to include smaller files or provide further instructions.

Handlers — This can be used to do things like restart a service or carry out other tasks.

Templates — You can vary certain things in this to produce dynamic configuration files.

Files — This one is simple. It's a static file which probably doesn't need to be different across servers.

YAML

- Ansible uses a simple syntax (YAML) and simple command-line tools for all its powerful abilities.

- Syntax: (playbook.yml)

```
---
# This is the beginning of a YAML file.
```