Chapter: 10 — Automating linux Administration tasks

Managing software and subscriptions.

yum module.

```
- yum:
    name: package_name
    state: present/      # installs
           absent/       # removes
           latest        # installs latest version.
```

if '*' i.e. all mentioned instead name section, it will consider it as 'all'.

if '@Development Tools' mentioned in name section.
yum group install Development Tools.
$ yum group list => To see packages in group.

Installing multiple packages.

```
- yum:
    name:
      - abc
      - def
      - ijk
    state: present.
```

package_facts (sets ansible_facts.packages variable)

collects installed packages details on M.N.

```
- package_facts:
    manager: auto
- debug:
    var: ansible_facts.packages
```

yum module => RHEL
~~DNF~~ dnf module => Fedora
apt module => Debian & ubuntu
win_package => Windows

tasks:
- yum:
  name: httpd
  state: present
  when: "ansible_distribution == 'RedHat'"

package module => Generic / common module.

Registering the system

$ subscription-manager register --username=xyz
--password=pqrs.
→ ~~subs~~ register the new system.

$ subscription-manager attach --pool=poolID
→ attach an available subscription.

$ subscription-manager list --available
→ list available pools in your account.

redhat_subscription module.

- name: Registering the system.
  redhat_subscription:
    username: xyz
    password: pqrs
    pool_ids: poolID
    state: present

To enable Red Hat software repositories on the new system.

```
$ subscription-manager repos \
> --enable "rhel-8-for-x86_64-baseos-rpms" \
> --enable "rhel-8-for-x86_64-baseos-debug-rpms"
```

To list the available repos.

```
$ subscription_manager repos --list
```

rhsm_repository module

```
- name: Enable Red Hat repositories
  rhsm_repository:
    name:
      - rhel-8-for-x86_64-baseos-rpms
      - rhel-8-for-x86-64-baseos-debug-rpms
    state: present.
```

Configuring a yum repository

```
tasks:
- name: Creating new repo
  yum_repository:
    file: /etc/yum.repos.d/xyz   # .repo extension is automatically added
    name: BaseOS  # [BaseOS]
    description: BaseOS  # name:
    baseurl: xyz                 also no need to give whole path.
    enabled: yes
    gpg check key: yes
```

If gpgcheck ⇒ yes ⇒ need to install GPG key before yum_repository module.

tasks:

- name: Deploy the GPG key.
  rpm_key:
    key: xyz
    state: present.
- yum_repository:
  ⋮

Managing users and authentication.

- Ansible user module lets you manage user accounts on a remote hosts.

- name: adding user.
  user:
    name: name_of_user
    shell: /bin/bash
    groups: sys_admin, developer.
    append: yes (add the groups)
    (otherwise overwrite them)

    generate_ssh_key: yes (does not overwrite
    ssh_key_bits: 2048 existing SSH key)
    ssh_key_file: .ssh/id_my_rsa

    comment: # description of user account
    group: # sets primary group
    groups: # sets secondary group
    home: # sets users home directory

create_home : # yes/no for home dir. creation
system : # makes system account
uid : # sets UID of user.

- name : Group module (Group module allows
group : to manage group on
name : auditors the managed hosts)
state : present

gid : (sets GID)
local : (forces use of 'local' command)
name : (Name of group)
state : (present/absent)
system : (creates system group)

known_hosts module
lets you add or remove host keys from the known_hosts file on managed host.

- name : Copy host keys to remote servers.
known_hosts :
path : /etc/ssh/ssh_known_hosts
name : user1
key : "{{ lookup ('file', 'pubkeys/user1') }}"

- lookup plugin allows ansible to access data from outside sources.

ø authorized_key module
add or remove SSH authorized key per user accounts.

```
- name: set authorized key.
  authorized_key:
    user: user1
    state: present
    key: "{{ lookup ('file', '/home/user1/.ssh/
                      id_rsa.pub') }}"
```

Managing the Boot process and scheduled processes

ø at module
Scheduling one-time job

```
- name: remove tempuser
  at:
    command: userdel -r tempuser      # job
    count: 20                         # count of units
    units: minutes                    # min/hr/days/weeks
    unique: yes                       # yes/no if a
                                      job is already running, it
                                      will not be executed again
    state:                            # present/absent.
    script_file:                      # An existing script file
                                      to be executed on the future.
```

cron module
job scheduling task

```
- cron:
    name: "Flushing"
    user: "root"
    minute: 45
    how: 11
    job: "php ./app/mut cache:clear"
```

```
special_time :      # reboot, yearly, annually, monthly,
                      weekly, daily, hourly
state:              # present/absent
cron_file:          # pre-written crontab file
backup:             # yes/no; Backup the file before editing
```

systemd & service module
managing services or reloading daemons
service => start/stop/restart/enable
systemd => daemon-reload, more config. options.

```
- service:
    name: httpd
    state : started
```

```
- systemd:
    name: apache2
    State: reloaded
    daemon-reload: yes
```

Reboot module
will shut down the M.N, then wait until it is back up again prior to carrying on with the play.

```
- reboot:
  reboot_timeout: 180
```

```
- reboot:
```

Command module is considered more secure ~~but~~ because it is not affected by the users environment

To sanitize any variable;
"{{ var | quote }}" is preferred over "{{ var }}"

all the environmental variables => ansible_env
you can isolate by lookup plugin;
msg: "{{ lookup ('env', 'user', 'HOME', 'SHELL') }}"

Managing storage.

⌀ Parted Module.

Supports partition of block devices.
'parted' is also a command.

parameters:

align:	Configures partition alignment
device:	Block device.
flags:	Flags for the partition => Type
number:	Partition number
part_end:	Partition size from the beginning of the disk.
state:	present / absent
unit:	Size units

⌀ lvg module

The lvg takes as parameters the block devices to configure as the back end physical volumes for the volume group.

parameters:

pesize: Size of physical extent; multiple of 128 KiB ~~multiple of 2~~ power of 2

pvs: Comma-seperated device for vg.

vg: name of vg

state: present/absent.

- Lvg:
  vg: vg1
  pvs: /dev/vda1
  pesize: 32

- Lvg:
  vg: vg1
  pvs: /dev/vda1, /dev/xvbd1

Adding physical extent.

Lvol module

- Creates logical volume
- Supports resizing & shrinking
- filesystems
- Snapshots

parameters:

Lv: Name of LV

resizefs: Resize fs with LV

Shrink: Shrink LV

size: Size of LV

Snapshot: name of snapshot of LV

State: present/absent

vg: vg name for LV

- Lvol:
  vg: vg1
  lv: lv1
  size: 2g

filesystem module

creating & resizing a filesystem.

parameters:

dev: Block dev name
Fstype: filesystem type
resizefs: Grows the filesystem size to the block device

- filesystem:
  - fstype: xfs
    dev: /dev/vdb1

Mount module

mount point on /etc/fstab.

parameters:

fstype: filesystem type
opts: Mount options
src: Device to be mounted
path: Mount point
state: present - Temporary
absent - Removes
mounted - entry in /etc/fstab

- name: mount device with ID
  mount:
    path: /data
    src: UUID = xyz
    fstype: xfs
    state: present

SWAP

Redhat Ansible currently doesn't support swap memory.
To manage swap memory:
- Create new vg & lv with lvg & lvol module
- run mkswap command via command module.
- run swapon command via command module.

```
- lvg:
    vg: vgswap
    pvs: /dev/vda1
    state: present

- lvol:
    vg: vgswap
    lv: lvswap
    size: 10g

- command: mkswap /dev/vgswap/lvswap
  when: ansible_swaptotal_mb < 128

- command: swapon /dev/vgswap/lvswap
  when: ansible_swaptotal_mb < 128
```

Ansible facts for storage configuration.

$ ansible node1 -m setup
→ Collects ansible facts of node1

$ ansible node1 -m setup -a 'filter = ansible_devices'
→ filter all storage devices available information

ansible_device_links ⇒ all the links available for each storage device.

ansible_mounts ⇒ info about the current mounted devices on the M.N.

⇒ Managing Network Configuration.

linux-system-roles ⇒ symlink to ⇒ rhel-system-roles

The network role ⇒ Two variable.

network_provider: nm

network_connections:

parameters of network_connections;

name :	Connection profile name
state :	up = active ; down = not active
persistent_state:	present / absent
type:	connection type; ethernet, bridge etc.
autoconnect :	connection automatically starts.
mac:	Restrict to specific MAC address.
interface_name:	Restrict to specific interface.
zone :	Config. firewalld zone for the interface
ip :	IP config. for the connection.

```
- hosts: webservers
  vars:
    network_connections:
      :
  roles:
    - rhel-system-roles.network
```

ø nmcli module
- Alternate to the network role
- supports management of network connections
- —"— network devices.
- supports config. of teaming & bonding to N.I
- IPv4 & IPv6 addressing

parameters:
conn_name: configures the connection name
autoconnect: Auto conn. activation on boot.
dns4: config dns server for IPV4 (upto 3)
gw4: —"— Gateway —"—
ifname: interface to be bound to the conn
ip4: IPv4 address for the interface
state: Enables/Disables & present/absent
type: Type of device or network conn.

ø hostname module
sets the hostname for a managed host without modifying the /etc/hosts file.

- hostname:
  name: mangedhost1

ø firewalld module
- management of firewallD on M.N
- config of firewallD rules for service & ports
- supports zone management.

- firewalld:
  service: http
  permanent: yes
  state: enabled

parameters of firewalld:

interface : Interface name to manage with firewalld
port : Port or port range.
rich_rule : Rich rule for firewalld
service : Name of service to manage
source : source network to manage
zone : firewalld zone
state : Enable/Disable firewalld configuration.
type : Type of device or network conn.

$ ansible node1 -m setup -a "gather_subset= network filter=ansible_interfaces"

→ gather_subset = network
restrict the facts includes in network subset

ansible_interfaces
All network interfaces for M.N

ansible_NICname
Information about config for network interface.

ansible_dns ⇒ DNS servers IP addresses.
ansible_domain ⇒ subdomain for the M.N
ansible_all_ipv4_addresses ⇒ IPV4 address on M.N
ipv6 ⇒ IPv6
ansible_fqdn ⇒ Fully Qualified Domain Name of M.N.
ansible_hostname ⇒ Unqualified hostname.
⇒ String in fqDN before first period.
ansible_nodename ⇒ hostname for M.N as reported by the system.

Inventory_hostname ⇒ hostname as conf. in Ansible Inventory file.