Use of Ansible Modules For System Administration Tasks

- Software packages and repositories
- Services
- Firewall rules
- File systems
- Storage devices
- File content
- Archiving
- Scheduled tasks
- Security
- Users and groups

Task. Create a playbook named ‘services.yml’ under tasks directory to perform below tasks.

- Install **httpd** service on **webservers** nodes.
- Install **mariadb** service on **prod** nodes.
- Make sure services are started and enabled.

```
---
-
 hosts: webservers
 become: True
 tasks:
        - name: Installing httpd service
          yum:
                name: httpd
                state: present
        - name: Starting and enabling httpd service
          service:
                name: httpd
                state: started
                enabled: yes
-
 hosts: prod
 become: True
 tasks:
        - name: Installing mariadb service
          yum:
                name:
                        - mariadb-server
                        - mariadb-common
                state: present
        - name: Starting and enabling mariadb service
          service:
                name: mariadb
                state: started
                enabled: yes
…
```

Task. Create a playbook ‘user.yml’ to create user on all managed hosts with below information.

- Use username as **mark**.
- Set password as **password**.
- Password must be encrypted with **Sha512.**

```
---
-
 hosts: all
 become: True
 gather_facts: False
 tasks:
              - name: Creating user
                 user:
                          name: mark
                          password: “{{ ‘password’ | password_hash(‘sha512’) }}”
                          state: present
...
```

Task. Create a playbook named ‘file.yml’ to create file ‘/root/mark_file’ on all managed nodes.

- User and group ownership must be set to **mark**.
- Configure full permissions for **user** , **read/write at group level** and **no permissions for others** on this file.
- Set **GiD** bit.

```
---
-
 hosts: all
 become: True
 gather_facts: False
 tasks:
           - name: Creating file, setting permissions and gid bit
             file:
                   path: /root/mark_file
                   owner: mark
                   group: mark
                   mode: ‘2760’
                   state: touch
…
```

Task. Using ansible ad-hoc commands, create file ‘/root/file1.txt’ on all managed nodes.

- File should contain text **This text file is created using Ansible**.
- **Remove all permissions for others** on this file

Execute Commands as ansible user:

```
ansible all -m file -a “path=/root/file1.txt mode=o-rwx state=touch” --become
ansible all -m copy -a “content=‘This text file is created using Ansible’ dest=/root/file1.txt”--become
```

Task. Using ansible playbook 'archive.yml' ,archive contents of '/etc' directory into 'etc.tar' file under '/root' directory.

- Playbook should be executed on **webservers** nodes.
- Compress the archive using **bzip2**.

```
---
-
 hosts: webservers
 become: Yes
 gather_facts: False
 tasks:
        - name: Archiving /etc directory
          archive:
                path: /etc
                dest: /root/etc.tar.bz2
                format: bz2
...
```

Task. Create a playbook ‘cronjobs.yml’ to schedule below tasks.

- Restart **rsyslog service** at **23h00** and **06h00** on **prod** nodes everyday.
- Restart **rsyslog service** at **02h00** on **webservers** nodes on every Monday.

```
---
-
 hosts: prod
 become: Yes
 gather_facts: False
 tasks:
          - name: Scheduling restart of rsyslog on prod nodes
            cron:
                        name: “Scheduling cron job on prod nodes”
                        hour: “23,6”
                        minute: “0”
                        job: /usr/bin/systemctl restart rsyslog
-
 hosts: webservers
 become: True
 gather_facts: False
 tasks:
          - name: Scheduling restart of rsyslog on webservers nodes
            cron:
                       name: “Scheduling cron job on webservers nodes”
                       hour: “2”
                       minute: “0”
                       weekday: “1”
                       job: /usr/bin/systemctl restart rsyslog
...
```

Task. Create a playbook ‘update.yml’ to update all packages on prod1 node.

```
---
-
 hosts: prod1
 become: True
 gather_facts: False
 tasks:
         - name: Update all packages on prod1 node
           yum:
                   name: ‘*’
                   state: latest
…
```

Task. Create a playbook ‘firewall.yml’ to configure firewall on all ‘webservers’ nodes.

- Inbound traffic for **http** service should be accepted.
- Setting should be persistent and reload firewall to enforce this.

```
---
-
 hosts: webservers
 become: Yes
 gather_facts: False
 tasks:
         - name: Configuring firewall on webservers nodes
           firewalld:
                   service:  http
                   state: enabled
                   permanent: yes
           notify: Reload firewall
 handlers:
         - name: Reload firewall
           service:
                   name: firewalld
                   state: reloaded
...
```

Task. Create a playbook ‘group.yml’ to perform below tasks.

- Create directory path **/web/html** on **webservers** nodes.
- Create group **testing** on **webservers** nodes and group **networks** on **prod** nodes.

```
---
-
 hosts: webservers
 become: Yes
 gather_facts: False
 tasks:
        - name: Creating directory
          file:
                  path:  /web/html
                  state: directory
        - name: Creating group
          group:
                  name: testing
                  state: present
-
 hosts: prod
 become: True
 gather_facts: False
 tasks:
        - name: Creating group
          group:
                  name: networks
                  state: present

...
```

Task. Create a playbook ‘context.yml’ to set selinux context type ‘httpd_sys_content_t’ on ‘/web/html’ directory on all webservers nodes.

- Setting should be persistent, and context should be restored.
- Verify the context type using ansible ad-hoc command.

```
---
-
 hosts: webservers
 become: Yes
 gather_facts: False
 tasks:
          - name: Setting Context type
            sefcontext:
                    target:  ‘/web/html(/.*)?’
                    setype: httpd_sys_content_t
                    state: present

          - name: Restoring context type
            command: restorecon -irv /web/html
...
```

Task. Create a playbook ‘parted.yml’ to create extended partition on all managed nodes.

- Use all remaining space for **extended partition**(container for logical partitions).
- Create one logical partition of size **200 MiB** on all managed nodes.

```
---
-
 hosts: all
 become: Yes
 gather_facts: True
 tasks:
          - name: Read device information
            parted: device=/dev/sda unit=MiB
            register: sda_info
          - name: Creating Extended partition
            parted:
                         device:  /dev/sda
                         number: “4”
                         part_type: extended
                         part_start: “{{ sda_info.partitions[2].end  + 1 }}MiB”
                         state: present
          - name: Creating logical partition
            parted:
                         device: /dev/sda
                         number: “5”
                         part_type: logical
                         part_start: “{{ sda_info.partitions[2].end  + 2 }}MiB”
                         part_end: “{{ sda_info.partitions[2].end  + 202 }}MiB”
                         state: present
…
```

Task. Create a playbook ‘mount.yml’ to format the device ‘/dev/sda5’ with ‘ext4’ filesystem.

- Mount the file system on **/mnt/partition** directory.
- Mount should be persistent.

```
---
-
 hosts: all
 become: Yes
 gather_facts: False
 tasks:
          - name: Creating filesystem
            filesystem:
                        dev: /dev/sda5
                        fstype: ext4
         - name: Mounting filesystem
           mount:
                        src: /dev/sda5
                        path: /mnt/partition
                        fstype: ext4
                        state: mounted
...
```