DUMPSARENA

Red Hat Certified Specialist in Advanced Automation: Ansible Best Practices

RedHat EX447

Version Demo

Total Demo Questions: 5

Total Premium Questions: 56

Buy Premium PDF

https://dumpsarena.co

sales@dumpsarena.co

sales@dumpsarena.co
dumpsarena.co

Topic Break Down

Topic	No. of Questions
Topic 1, Manage playbook execution	6
Topic 2, Manage inventories	5
Topic 3, Manage task execution	30
Topic 4, Manage complex plays and playbooks	9
Topic 5, Manage Ansible content collections	5
Topic 6, Mix Questions	1
Total	56

QUESTION NO: 1 - (SIMULATION)

Create a playbook called issue.yml in /home/sandy/ansible which changes the file /etc/issue on all managed nodes: If host is a member of (lev then write "Development" If host is a member of test then write "Test" If host is a member of prod then write "Production"

ANSWER: See the explanation for the answer

Explanation:

Create `/home/sandy/ansible/issue.yml` with a play that uses each host's inventory group membership to select the required content:

```
- name: Configure /etc/issue according to environment
  hosts: all
  become: true
  gather_facts: false

  tasks:
    - name: Set development issue
      ansible.builtin.copy:
        dest: /etc/issue
        content: "Development\n"
        owner: root
        group: root
        mode: '0644'
      when: "'dev' in group_names"

    - name: Set test issue
      ansible.builtin.copy:
        dest: /etc/issue
        content: "Test\n"
        owner: root
        group: root
        mode: '0644'
      when: "'test' in group_names"

    - name: Set production issue
      ansible.builtin.copy:
        dest: /etc/issue
        content: "Production\n"
        owner: root
        group: root
        mode: '0644'
      when: "'prod' in group_names"
```

Run it from the Ansible project directory:

```
cd /home/sandy/ansible
ansible-playbook issue.yml
```

The `when` conditions inspect `group_names`, so each managed host receives the appropriate `/etc/issue` text based on membership in `dev`, `test`, or `prod`.

QUESTION NO: 2 - (SIMULATION)

Create a file called adhoc.sh in /home/sandy/ansible which will use adhoc commands to set up a new repository. The name of the repo will be 'EPEL' the description 'RHEL8' the baseurl is 'https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rmp' there is no gpgcheck, but you should enable the repo.

* You should be able to use an bash script using adhoc commands to enable repos. Depending on your lab setup, you may need to make this repo "state=absent" after you pass this task.

ANSWER: See the explanation for the answer

Explanation:

Create `/home/sandy/ansible/adhoc.sh` with an Ansible ad-hoc `yum_repository` command:

```
#!/bin/bash
ansible all -m yum_repository -a 'name=EPEL description=RHEL8
baseurl=https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rmp gpgcheck=no
enabled=yes state=present'
```

Make the script executable and run it from the Ansible project directory:

```
cd /home/sandy/ansible
chmod 755 adhoc.sh
./adhoc.sh
```

`name=EPEL` creates the repository identifier.
`description=RHEL8` sets its description.
`gpgcheck=no` disables GPG checking.
`enabled=yes` enables the repository.

The base URL above follows the URL exactly as supplied in the simulation question.

QUESTION NO: 3 - (SIMULATION)

Create a playbook **that** changes the default target on **all** nodes to **multi-user** tarqet. Do this in playbook file called target. yml in /home/sandy/ansible

ANSWER: See the explanation for the answer

Explanation:

Create `/home/sandy/ansible/target.yml` with the following content:

```
- name: Set the default systemd target
  hosts: all
  become: true
  tasks:
    - name: Configure multi-user.target as the default target
      ansible.builtin.file:
        src: /usr/lib/systemd/system/multi-user.target
        dest: /etc/systemd/system/default.target
        state: link
        force: true
```

The default systemd target is controlled by the `/etc/systemd/system/default.target` symbolic link. This play creates or replaces that link on every managed node so that it points to `multi-user.target`.

QUESTION NO: 4 - (SIMULATION)

Using the Simulation Program, perform the following tasks:

1. Use an ansible ad-hoc command, check the connectivity of your servers.

2. Use an ad-hoc ansible command, find the free space of your servers.

3. Use an ad-hoc ansible command, find out the memory usage of your servers.

4. Do an ls -l on the targets /var/log/messages file.

5. Tail the contents of the targets /var/log/messages file.

ANSWER: See the explanation for the answer

Explanation:

Run the following ad-hoc commands from the Ansible control node (using the inventory configured for the simulation):

```
ansible all -m ping
ansible all -m command -a 'df -h'
ansible all -m command -a 'free -h'
ansible all -m command -a 'ls -l /var/log/messages'
ansible all -b -m command -a 'tail /var/log/messages'
```

`ping` verifies Ansible connectivity to every managed host.
`df -h` displays filesystem capacity and available/free space in human-readable units.
`free -h` displays memory and swap usage in human-readable units.
`ls -l /var/log/messages` shows the file's detailed permissions, ownership, size, and timestamp.
`-b` on the final command uses privilege escalation, which may be required because `/var/log/messages` is commonly readable only by root. The command tails the log on all target hosts.

QUESTION NO: 5 - (SIMULATION)

Create the users in the file usersjist.yml file provided. Do this in a playbook called users.yml located at /home/sandy/ansible. The passwords for these users should be set using the lock.yml file from TASK7. When running the playbook, the lock.yml file should be unlocked with secret.txt file from TASK 7.

All users with the job of 'developer' should be created on the dev hosts, add them to the group devops, their password should be set using the pw_dev variable. Likewise create users with the job of 'manager' on the proxy host and add the users to the group 'managers', their password should be set using the pw_mgr variable.

users_list.yml

```
users:
  - username: bill
    job: developer
  - username: chris
    job: manager
  - username: dave
    job: test
  - username: ethan
    job: developer
```

ANSWER: See the explanation for the answer

Explanation:

Create `/home/sandy/ansible/users.yml` as follows (the supplied list file shown is `users_list.yml`):

```
---
- name: Create developer users
  hosts: dev
  become: true
  vars_files:
    - users_list.yml
    - lock.yml
  tasks:
    - name: Create developer accounts
      ansible.builtin.user:
```

```
        name: "{{ item.username }}"
        groups: devops
        append: true
        password: "{{ pw_dev }}"
      loop: "{{ users }}"
      when: item.job == 'developer'

- name: Create manager users
  hosts: proxy
  become: true
  vars_files:
    - users_list.yml
    - lock.yml
  tasks:
    - name: Create manager accounts
      ansible.builtin.user:
        name: "{{ item.username }}"
        groups: managers
        append: true
        password: "{{ pw_mgr }}"
      loop: "{{ users }}"
      when: item.job == 'manager'
```

Run it from `/home/sandy/ansible`, supplying the vault password file to decrypt `lock.yml`:

```
cd /home/sandy/ansible
ansible-playbook users.yml --vault-password-file=secret.txt
```

This creates `bill` and `ethan` only on the `dev` hosts, adding them to `devops` with `pw_dev`. It creates `chris` only on the `proxy` host, adding that user to `managers` with `pw_mgr`. The `test` user is intentionally not created.