**Task 2: Ad-Hoc Commands**

Generate an SSH keypair on the control node. You can perform this step manually.

Write a script /home/automation/plays/adhoc that uses Ansible ad-hoc commands to achieve the following:

* User **automation** is created on all inventory hosts (not the control node).
* SSH key (that you generated) is copied to all inventory hosts for the **automation** user and stored in /home/automation/.ssh/authorized\_keys.
* The **automation** user is allowed to elevate privileges on all inventory hosts without having to provide a password.

After running the adhoc script on the control node as the **automation** user, you should be able to SSH into all inventory hosts using the **automation** user without password, as well as a run all privileged commands.

#!/bin/bash

# create user automation

ansible all -m user -a "name=automation state=present"

# create .ssh directory

ansible all -m file -a "path=/home/automation/.ssh state=directory owner=automation group=automation mode=0700"

# copy id\_rsa.pub

ansible all -m copy -a "src=/home/automation/.ssh/id\_rsa.pub dest=/home/automation/.ssh/authorized\_keys owner=automation group=automation mode=0600"

# add sudo permission

ansible all -m copy -a "content='automation ALL=(ALL) NOPASSWD: ALL' dest=/etc/sudoers.d/automation owner=root group=root mode=0600"

**Task 3: File Content**

Create a playbook /home/automation/plays/motd.yml that runs on all inventory hosts and does the following:

* The playbook should replace any existing content of /etc/motd with text. Text depends on the host group.
* On hosts in the **proxy** host group the line should be “Welcome to HAProxy server”.
* On hosts in the **webservers** host group the line should be “Welcome to Apache server”.
* On hosts in the **database** host group the line should be “Welcome to MySQL server”.

I used the [magic variable](https://docs.ansible.com/ansible/latest/reference_appendices/special_variables.html) inventory\_hostname to match the conditions:

---

- name: task 3

hosts: all

become: yes

tasks:

- name: copy content to HAProxy

copy:

content: "Welcome to HAProxy server"

dest: /etc/motd

when: inventory\_hostname in groups["proxy"]

- name: copy content to webservers

copy:

content: "Welcome to Apache server"

dest: /etc/motd

when: inventory\_hostname in groups["webservers"]

- name: copy content to Database

copy:

content: "Welcome to MySQL server"

dest: /etc/motd

when: inventory\_hostname in groups["database"]

**Task 4: Configure SSH Server**

Create a playbook /home/automation/plays/sshd.yml that runs on all inventory hosts and configures SSHD daemon as follows:

* banner is set to /etc/motd
* X11Forwarding is disabled
* MaxAuthTries is set to 3

---

- name: task 4

hosts: all

become: yes

tasks:

- name: configure sshd daemon

lineinfile:

path: /etc/ssh/sshd\_config

regexp: "^Banner"

line: Banner /etc/motd

- name: disable X11Forwarding

lineinfile:

path: /etc/ssh/sshd\_config

regexp: "^X11Forwarding"

line: X11Forwarding no

- name: set MaxAuthTries = 3

lineinfile:

path: /etc/ssh/sshd\_config

regexp: "^MaxAuthTries"

line: MaxAuthTries 3

- name: restart ssh server

service:

name: sshd

state: restarted

enabled: yes

**Task 5: Ansible Vault**

Create Ansible vault file /home/automation/plays/secret.yml. Encryption/decryption password is **devops**.

Add the following variables to the vault:

* **user\_password** with value of **devops**
* **database\_password** with value of **devops**

Store Ansible vault password in the file /home/automation/plays/vault\_key.

Ansible vault commands are needed here. For example: ansible-vault view secret.yml --vault-password-file vault\_key. At the end of the exercise you must have two files:

* secret.yml: encrypted file with credentials
* vault\_key: the encryption key

**Task 6: Users and Groups**

You have been provided with the list of users below.

Use /home/automation/plays/vars/user\_list.yml file to save this content.

---

users:

- username: alice

uid: 1201

- username: vincent

uid: 1202

- username: sandy

uid: 2201

- username: patrick

uid: 2202

Create a playbook /home/automation/plays/users.yml that uses the vault file /home/automation/plays/secret.yml to achieve the following:

* Users whose user ID starts with 1 should be created on servers in the **webservers** host group. User password should be used from the **user\_password** variable.
* Users whose user ID starts with 2 should be created on servers in the **database** host group. User password should be used from the **user\_password** variable.
* All users should be members of a supplementary group **wheel**.
* Shell should be set to /bin/bash for all users.
* Account passwords should use the SHA512 hash format.
* Each user should have an SSH key uploaded (use the SSH key that you created previously, see task #2).
* We will have to loop through a list of users:
* ---
* users:
* - username: alice
* uid: 1201
* - username: vincent
* uid: 1202
* - username: sandy
* uid: 2201
* - username: patrick
* uid: 2202
* This is my playbook:
* ---
* - name: task 6
* hosts: all
* become: yes
* vars\_files:
* - ./vars/user\_list.yml
* - ./secret.yml
* tasks:
* - name: ensure group wheel exists
* group:
* name: wheel
* state: present
* - name: create users in webservers group
* loop: "{{ users }}"
* user:
* name: "{{ item.username }}"
* password: "{{ user\_password | password\_hash('sha512')}}"
* update\_password: on\_create
* groups: wheel
* shell: /bin/bash
* when:
* - inventory\_hostname in groups['webservers']
* - item.uid | string | first == "1"
* - name: create users in database group
* loop: "{{ users }}"
* user:
* name: "{{ item.username }}"
* password: "{{ user\_password | password\_hash('sha512')}}"
* update\_password: on\_create
* groups: wheel
* shell: /bin/bash
* when:
* - inventory\_hostname in groups['database']
* - item.uid | string | first == "2"
* - name: create .ssh directory (webservers)
* loop: "{{ users }}"
* file:
* name: "/home/{{ item.username }}/.ssh"
* state: directory
* owner: "{{ item.username }}"
* group: "{{ item.username }}"
* mode: 0700
* when:
* - inventory\_hostname in groups['webservers']
* - item.uid | string | first == "1"
* - name: create .ssh directory (database)
* loop: "{{ users }}"
* file:
* name: "/home/{{ item.username }}/.ssh"
* state: directory
* owner: "{{ item.username }}"
* group: "{{ item.username }}"
* mode: 0700
* when:
* - inventory\_hostname in groups['database']
* - item.uid | string | first == "2"
* - name: copy ssh authorized key (webservers)
* loop: "{{ users }}"
* copy:
* src: "/home/automation/.ssh/id\_rsa.pub"
* dest: "/home/{{ item.username }}/.ssh/authorized\_keys"
* owner: "{{ item.username }}"
* group: "{{ item.username }}"
* mode: 0600
* when:
* - inventory\_hostname in groups['webservers']
* - item.uid|string|first == "1"
* - name: copy ssh authorized key (database)
* loop: "{{ users }}"
* copy:
* src: "/home/automation/.ssh/id\_rsa.pub"
* dest: "/home/{{ item.username }}/.ssh/authorized\_keys"
* owner: "{{ item.username }}"
* group: "{{ item.username }}"
* mode: 0600
* when:
* - inventory\_hostname in groups['database']
* - item.uid|string|first == "2"

**Task 7: Scheduled Tasks**

Create a playbook /home/automation/plays/regular\_tasks.yml that runs on servers in the **proxy** host group and does the following:

* A root crontab record is created that runs every hour.
* The cron job appends the file /var/log/time.log with the output from the **date** command.
* I chose to use a md5 hash for the job name because it looks fancy to me (and I wanted to learn how to do it). But it is not a requirement and a descriptive name could be more appropriate depending on the environment.
* ---
* - name: task 7
* hosts: proxy
* become: yes
* tasks:
* - name: create a cron job with a fancy name on proxy hosts
* cron:
* name: "{{ 'proxy: append date to time.log' | hash('md5') }}"
* minute: "0"
* job: "date >> /var/log/time.log"

**Task 12: Security**

Create a playbook /home/automation/plays/selinux.yml that runs on hosts in the **webservers** host group and does the following:

* Uses the selinux **RHEL system role**.
* Enables **httpd\_can\_network\_connect** SELinux boolean.
* The change must survive system reboot.
* Install roles:
* yum install rhel-system-roles
* Since the installed roles are not in the path, the ansible.cfg has to be modified:
* [defaults]
* roles\_path = /home/automation/plays/roles:/usr/share/ansible/roles
* [...]
* The playbook:
* ---
* - name: task 12
* hosts: webservers
* become: true
* vars:
* - selinux\_booleans:
* - name: httpd\_can\_network\_connect
* state: on
* persistent: true
* roles:
* - rhel-system-roles.selinux

**Task 13: Use Conditionals to Control Play Execution**

Create a playbook /home/automation/plays/sysctl.yml that runs on all inventory hosts and does the following:

* If a server has more than 2048MB of RAM, then parameter **vm.swappiness** is set to 10.
* If a server has less than 2048MB of RAM, then the following error message is displayed:

**Server memory less than 2048MB**

To find the fact:

$ ansible ansible2.hl.local -m setup | grep -A10 memory

[...]

"ansible\_memory\_mb": {

"nocache": {

"free": 1447,

"used": 343

},

"real": {

"free": 961,

"total": 1790,

"used": 829

},

"swap": {

[...]

So the variable to use is ansible\_memory\_mb.real.total. And the playbook:

---

- name: task 13

hosts: all

become: true

tasks:

- name: set vm.swappiness to 10 if server has 2GB memory

sysctl:

name: vm.swappiness

value: 10

state: present

when:

- ansible\_memory\_mb.real.total >= 2048

- name: report not enough total memory

debug:

msg: "Server memory less than 2048MB ({{ ansible\_memory\_mb.real.total }}MB)"

when:

- ansible\_memory\_mb.real.total < 2048

**Task 14: Use Archiving**

Create a playbook /home/automation/plays/archive.yml that runs on hosts in the **database** host group and does the following:

* A file /mnt/mysql\_backups/database\_list.txt is created that contains the following line: dev,test,qa,prod.
* A gzip archive of the file /mnt/mysql\_backups/database\_list.txt is created and stored in /mnt/mysql\_backups/archive.gz.

---

- name: task 14

hosts: database

become: true

tasks:

- name: create database list file

copy:

content: "dev,test,qa,prod"

dest: /mnt/mysql\_backups/database\_list.txt

- name: archive file

archive:

path: /mnt/mysql\_backups/database\_list.txt

dest: /mnt/mysql\_backups/archive.gz

format: "gz"

Task 15: Work with Ansible Facts

Create a playbook /home/automation/plays/facts.yml that runs on hosts in the **database** host group and does the following:

1. A custom Ansible fact **server\_role=mysql** is created that can be retrieved from **ansible\_local.custom.sample\_exam** when using Ansible setup module.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28 | [automation@ansible-control plays]$ **cat** facts.yml  ---  - hosts: database    become: yes    # gather\_facts: no      tasks:      - name: create a directory  **file**:        path: /etc/ansible/facts.d        state: directory      - name: **touch** a **file**  **file**:        path: /etc/ansible/facts.d/custom.fact        state: **touch**      - name: blockinfile      blockinfile:        path: /etc/ansible/facts.d/custom.fact        block: |          [sample\_exam]          server\_role=mysql      - name: debug      debug:        msg: "{{ ansible\_local.custom.sample\_exam }}" |

Run the playbook:

Shell

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | [automation@ansible-control plays]$ ansible-playbook facts.yml    PLAY [database] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    TASK [Gathering Facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ok: [ansible5.hl.local]    TASK [create a directory] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ok: [ansible5.hl.local]    TASK [**touch** a **file**] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  changed: [ansible5.hl.local]    TASK [blockinfile] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ok: [ansible5.hl.local]    TASK [debug] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ok: [ansible5.hl.local] => {  "msg": {  "server\_role": "mysql"  }  }    PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ansible5.hl.local : ok=5 changed=1 unreachable=0 failed=0 |

Check if facts exists:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | [automation@ansible-control plays]$ ansible database -a 'cat /etc/ansible/facts.d/custom.fact'  ansible5.hl.local | CHANGED | rc=0 >>  # BEGIN ANSIBLE MANAGED BLOCK  [sample\_exam]  server\_role=mysql  # END ANSIBLE MANAGED BLOCK    [automation@ansible-control plays]$ ansible database -m setup -a "filter=ansible\_local"  ansible5.hl.local | SUCCESS => {      "ansible\_facts": {          "ansible\_local": {              "custom": {                  "sample\_exam": {                      "server\_role": "mysql"                  }              }          }      },      "changed": **false**  } |

**Task 16: Software Packages**

Create a playbook /home/automation/plays/packages.yml that runs on all inventory hosts and does the following:

* Installs **tcpdump** and **mailx** packages on hosts in the **proxy** host groups.
* Installs **lsof** and **mailx** packages on hosts in the **database** host groups.

---

- name: task 16

hosts: all

become: true

tasks:

- name: install software in proxy group

yum:

name: "{{ item }}"

state: latest

loop:

- tcpdump

- mailx

when:

- inventory\_hostname in groups['proxy']

- name: install software in database group

yum:

name: "{{ item }}"

state: latest

loop:

- lsof

- mailx

when:

- inventory\_hostname in groups['database']

**Task 17: Services**

Create a playbook /home/automation/plays/target.yml that runs on hosts in the **webservers** host group and does the following:

* Sets the default boot target to **multi-user**.

---

- name: task 17

hosts: webservers

become: true

tasks:

- name: set default target to multi-user

file:

src: /usr/lib/systemd/system/multi-user.target

dest: /etc/systemd/system/default.target

state: link

**Task 18. Create and Use Templates to Create Customised Configuration Files**

Create a playbook /home/automation/plays/server\_list.yml that does the following:

* Playbook uses a Jinja2 template server\_list.j2 to create a file /etc/server\_list.txt on hosts in the **database** host group.
* The file /etc/server\_list.txt is owned by the **automation** user.
* File permissions are set to **0600**.
* SELinux file label should be set to **net\_conf\_t**.
* The content of the file is a list of FQDNs of all inventory hosts.

After running the playbook, the content of the file /etc/server\_list.txt should be the following:

ansible2.hl.local

ansible3.hl.local

ansible4.hl.local

ansible5.hl.local

Note: if the FQDN of any inventory host changes, re-running the playbook should update the file with the new values.

The playbook:

---

- name: task 18

hosts: database

become: true

tasks:

- name: deploy server list

template:

src: server\_list.j2

dest: /etc/server\_list.txt

owner: automation

group: automation

mode: 0600

setype: net\_conf\_t

The template:

{% for host in groups["all"] %}

{{ hostvars[host]['inventory\_hostname'] }}

{% endfor %}

- name: reate template

    lineinfile:

      path: templates/server\_list.j2

      line: '{{ ansible\_fqdn }}'

      create: yes

ANSIBLE FACTS!!!

---

- name: create ansible fats

hosts: all

tasks:

- name: creating a customs dir

file:

path: /etc/ansible/facts.d

state: directory

- name: creating a custom facts file

file:

path: /etc/ansible/facts.d/custom.fact

state: touch

- name: creating the server role

blockinfile:

path: /etc/ansible/facts.d/custom.fact

block: |

[sample\_exam]

server\_role=mysql