Lesson 03 Demo 01

Executing Ad Hoc Commands

Objective: To demonstrate ad hoc commands for quickly executing tasks on remote servers without writing full playbooks

Tools required: Ansible, Ubuntu OS

Prerequisites: None

Steps to be followed:

1. Generate SSH key pair on the main node
2. Copy the SSH key on the other two nodes
3. Update the host file with the host IP address
4. Establish connectivity between specified hosts and the Ansible server
5. Gather System Information Using Ad-Hoc Commands

Step 1: Generate SSH key pair on the main node

1.1 Use the following command to generate the SSH key on the main Ansible server:

ssh-keygen

```
ravitulsianisim@ip-172-31-67-38:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ravitulsianisim/.ssh/id_rsa):
Created directory '/home/ravitulsianisim/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ravitulsianisim/.ssh/id_rsa
Your public key has been saved in /home/ravitulsianisim/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:FYogbFj/k84qd2BKdzntH2/WiIjRK3Qe9v4TcGF1UzU ravitulsianisim@ip-172-31-67-38
The key's randomart image is:
+---[RSA 3072]----+
| +o .     .  ..E*|
|. oo . . . .o  .o|
| .  . . . .. .   |
|     . . .. .    |
|      ++S  o     |
|  . +o*.*   .    |
| . + +oO =.. +   |
|  o ..+ = oo= .  |
|   o.. . oo+o.   |
+----[SHA256]-----+
ravitulsianisim@ip-172-31-67-38:~$
```

Step 2: Copy the SSH key on the other two nodes

2.1 Use the following command to copy the public key in a file named **authorized_keys** in localhost:

cat .ssh/id_rsa.pub >> .ssh/authorized_keys

```
ravitulsianisim@ip-172-31-67-38:~$ cat .ssh/id_rsa.pub >> .ssh/authorized_keys
ravitulsianisim@ip-172-31-67-38:~$
```

2.2 Use the following command to check the SSH connection with localhost:

ssh localhost -p 42006

```
ravitulsianisim@ip-172-31-67-38:~$ ssh localhost -p 42006
The authenticity of host '[localhost]:42006 ([127.0.0.1]:42006)' can't be established.
ED25519 key fingerprint is SHA256:DNlvr4Nn9ktOnAt7NGC+DXM5kwdnEGLJS7ur90h65H0.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[localhost]:42006' (ED25519) to the list of known hosts.
Enter passphrase for key '/home/ravitulsianisim/.ssh/id_rsa':
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 6.2.0-1018-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

  System information as of Wed Jul  3 10:31:00 UTC 2024

  System load:  0.1005859375       Processes:               249
  Usage of /:   23.2% of 48.39GB   Users logged in:         0
  Memory usage: 27%                IPv4 address for docker0: 172.17.0.1
  Swap usage:   0%                 IPv4 address for ens5:    172.31.67.38

 * Ubuntu Pro delivers the most comprehensive open source security and
   compliance features.

   https://ubuntu.com/aws/pro
```

2.3 Now, use the following command to exit from the localhost:

exit

```
ravitulsianisim@ip-172-31-67-38:~$ exit
logout
Connection to localhost closed.
ravitulsianisim@ip-172-31-67-38:~$
```

2.4 Run the following command to go to the **.ssh** directory of the Ansible server:

cd .ssh

```
ravitulsianisim@ip-172-31-67-38:~$ cd  .ssh
ravitulsianisim@ip-172-31-67-38:~/.ssh$
```

2.5 Run the following command to copy the public key to another node that will connect to the main Ansible server:

ssh-copy-id username@ip -p 22

```
ravitulsianisim@ip-172-31-67-38:~/.ssh$ ssh-copy-id username@ip -p 22
/usr/bin/ssh-copy-id: ERROR: Too many arguments.  Expecting a target hostname, got:

Usage: /usr/bin/ssh-copy-id [-h|-?|-f|-n|-s] [-i [identity_file]] [-p port] [-F alternative ssh_config file] [[-o <ssh -o options>] ...] [user@]hostn
ame
        -f: force mode -- copy keys without trying to check if they are already installed
        -n: dry run    -- no keys are actually copied
        -s: use sftp   -- use sftp instead of executing remote-commands. Can be useful if the remote only allows sftp
        -h|-?: print this help
ravitulsianisim@ip-172-31-67-38:~/.ssh$
```

2.6 Execute the following command to come out of the **.ssh** directory of the Ansible server:

cd

```
ravitulsianisim@ip-172-31-67-38:~/.ssh$ cd
```

Step 3: Update the host file with the host IP address

3.1 Use the following command to open the Ansible inventory file and add the host localhost to it:

sudo vi /etc/ansible/hosts

```
ravitulsianisim@ip-172-31-67-38:~$ sudo vi /etc/ansible/hosts
ravitulsianisim@ip-172-31-67-38:~$
```

3.2 When the file opens, add the below two lines of the code at the end of the file:

[webservers]

localhost:42006

40.86.1.9:42006

```
## [webservers]
## alpha.example.org
## beta.example.org
## 192.168.1.100
## 192.168.1.110

# If you have multiple hosts following a pattern, you can specify
# them like this:

## www[001:006].example.com

# Ex 3: A collection of database servers in the 'dbservers' group:

## [dbservers]
##
## db01.intranet.mydomain.net
## db02.intranet.mydomain.net
## 10.25.1.56
## 10.25.1.57

# Here's another example of host ranges, this time there are no
# leading 0s:

## db-[99:101]-node.example.com
[webservers]
localhost:42006
40.86.1.9:42006
```

Note: Press **esc**, then write **:wq** and press **enter** to save the file.

Step 4: Establish connectivity between specified hosts and the Ansible server

4.1 Run the following command to verify connectivity to all servers listed under the **webservers** group in your Ansible hosts file:

ansible -m ping webservers

```
ravitulsianisim@ip-172-31-67-38:~$ ansible -m ping webservers
Enter passphrase for key '/home/ravitulsianisim/.ssh/id_rsa':
localhost | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

4.2 Use the following command to check the number of hosts in the host file:
ansible all --list-hosts

```
ravitulsianisim@ip-172-31-67-38:~$ ansible all --list-hosts
  hosts (2):
    localhost
    40.86.1.9
ravitulsianisim@ip-172-31-67-38:~$
```

Step 5: Gather System Information Using Ad Hoc Commands

5.1 Run the following command to obtain the uptime from all managed hosts using an ad hoc command:
ansible all -m shell -a uptime

```
ravitulsianisim@ip-172-31-67-38:~$ ansible all -m shell -a uptime
Enter passphrase for key '/home/ravitulsianisim/.ssh/id_rsa':
localhost | CHANGED | rc=0 >>
 11:50:29 up  1:36,  1 user,  load average: 0.04, 0.03, 0.00
```

5.2 Similarly, execute the below command to obtain detailed information about memory usage on all hosts:
ansible all -m shell -a "free -m"

```
ravitulsianisim@ip-172-31-67-38:~$ ansible all -m shell -a "free -m"
Enter passphrase for key '/home/ravitulsianisim/.ssh/id_rsa':
localhost | CHANGED | rc=0 >>
              total        used        free      shared  buff/cache   available
Mem:          15689        4372        8670           7        2646       11009
Swap:             0           0           0
```

You will see that Ansible logs in to each machine in turn and runs the uptime command, returning the current uptime output.

By following these steps, you have successfully demonstrated how to use ad hoc commands for quickly executing tasks on remote servers without the need for full playbooks.