Linux Academy
Study Guide

Ansible Cheat Sheet

Contents

Prerequisites

Linux

- Packages:
  - » `git`
  - » `python`
  - » `python-devel`
  - » `python-pip`
  - » `openssl`
  - » `ansible`

Installation and Configuration

Linux

Be sure to install *epel-release* first and then update your caches (if CentOS). On Ubuntu/Debian distributions, you may install from the default repositories. Assuming CentOS, as in our course, do the following:

```
sudo yum install git python python-devel python-pip openssl ansible
```

User Accounts

Create a user called *ansible* (example) on the server you intend to use Ansible to run playbooks from AND each of the Ansible nodes you intend to run playbooks on. Set the user as a `sudo`-capable user and include the `NOPASSWD: ALL` directive in */etc/sudoers*.

Create an SSH key with *ssh-keygen* on the Ansible server. Exchange that key using *ssh-copy-id* on each of the nodes you are running playbooks on. This allows the playbook to run with escalated privileges as needed.

Configuration Files

- /etc/ansible/ansible.cfg
  - » Primary Ansible configuration file (agentless, daemon-less configuration, read on each ansible command run)
  - » Uncomment "inventory" field

    - Uncomment "become user" field
- /etc/ansible/hosts
    - Copy original to */etc/ansible/hosts.original*
    - Create one or more sections with group names, sample below

```
[local]

localhost

[web]

webserver1

webserver2

[db]

dbserver1

dbserver2
```

Running Arbitrary Commands

Format

Arbitrary commands can be run against hosts or groups of hosts at the command line, one at a time. In order to list the contents of the home directory for the ansible user, for example:

```
ansible GROUPNAME -a "ls -al /home/ansible"
```

Running a command that requires `sudo` privileges should not be run with the `sudo` command, but rather the `sudo` parameter in the ansible command itself, like so:

```
ansible GROUPNAME -s -a "ls -al /var/log/messages"
```

You can also execute a single module against one or more hosts at the command line by using the module parameter. As an example:

```
ansible GROUPNAME -s -m yum -a "name=httpd state=latest"
```

Test if all machines in your inventory respond to a ping request:

```
ansible all -m ping
```

YAML Structure for Playbooks

Sample Playbook with Major Sections

```
--- # COMMENT ABOUT PLAYBOOK
- hosts: hostsToRunAgainst
  remote_user: ansible
  become: yes
  become_method: sudo
  connection: ssh
  gather_facts: no
  vars:
    var1: value
    var2: value
  tasks:
  - name: Some description of what we are doing
    yum:
      name: httpd
      state: latest
    notify:
    - startservice
  handlers:
  - name: startservice
    service:
      name: httpd
      state: restarted
```

Quick Notes

ansible-playbook

- **Calling a playbook** • `ansible-playbook /path/to/playbook.yaml`

Inventory

- */etc/ansible/hosts* • Defines nodes/groups of nodes to operate against
- *$ANSIBLE_HOSTS* • Shell variable containing one or more ansible hosts