How to Configure Ansible Managed Nodes and Run ad-hoc Commands – Part 3

Ravi Saive | October 25, 2019 | Read Time: 7 mins | Ansible | 1 Comment

In the previous two articles of this Ansible Series, we've explained Core Components of Ansible and Setting Up Ansible Control Node. In this part 3, we will demonstrate how you can configure Ansible managed nodes to run ad-hoc commands on remote hosts.

Setup Passwordless SSH Authentication to Ansible Managed Nodes

As a recap on our last topic, managing remote hosts with Ansible requires setting up of Passwordless SSH authentication between the Ansible control node and the managed hosts. This involves the generation of a key pair (Public and Private SSH key pair) on the Ansible Control node and copying the Public key to all of the remote hosts. This will be a crucial step going forward and will make your work much easier.

Configure Privilege Escalation on Managed Nodes

When logged in as a regular user, you may be required to perform certain tasks on managed nodes that require elevated privileges or root privileges. These tasks include package management, adding new users & groups, and modifying system configurations to mention just but a few. To achieve this, you need to invoke certain directives in the playbook to run the tasks as a privileged user on the remote hosts.

become

Ansible allows you to '**become**' another user on the managed node different from the one currently logged in. The `become:yes` directive elevates your privileges and allows you to perform tasks that require root privileges such as installing and updating packages and rebooting the system.

Consider a playbook `httpd.yml` that installs and starts Apache webserver as shown:

```
---
- name: install and start Apache webserver
  hosts: webservers

  tasks:
       - name: install httpd
         yum: name=httpd  state=latest
         become: yes
       - name: check httpd status
         service: name=httpd state=started
```

The `become: yes` the directive allows you to execute commands as a root user on the remote host.

become_user

Another directive that you can use to become another user is the **become_user**. This allows you to switch to a sudo user on the remote host upon logging in and not the user you log in as.

For example, to run a command as **tecmint** user on the remote use the directive as shown.

```
- name: Run a command as the apache user
  command: somecommand
  become: yes
  become_user: tecmint
```

become_method

This directive will override the default method set in **ansible.cfg** file which is usually set to sudo.

become_flags

These are used at play or task level, for instance when you need to switch to a user when the shell is set to nologin.

For example,

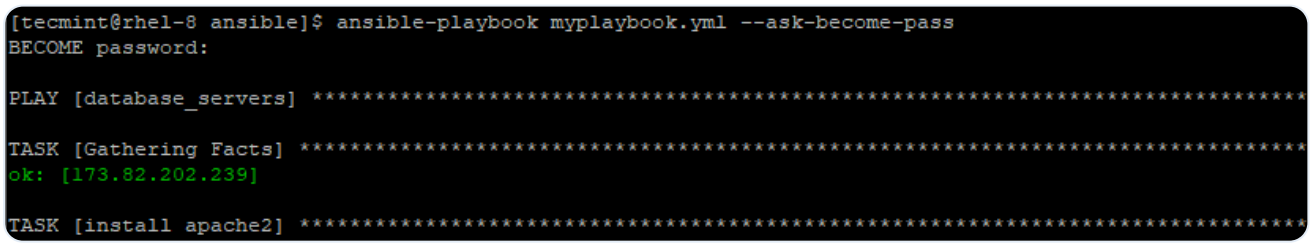
```
- name: Run a command as nobody
  command: somecommand
  become: true
  become_method: su
  become_user: nobody
  become_flags: '-s /bin/sh'
```

Command-line Options in Privilege Escalation

Let's take a look at some of the command-line options that you can use to elevate your privileges when running commands:

- `--ask-become-pass`, `-K` – This prompts you for the password of the sudo user on the remote system that you are trying to connect.

```
$ ansible-playbook myplaybook.yml --ask-become-pass
```



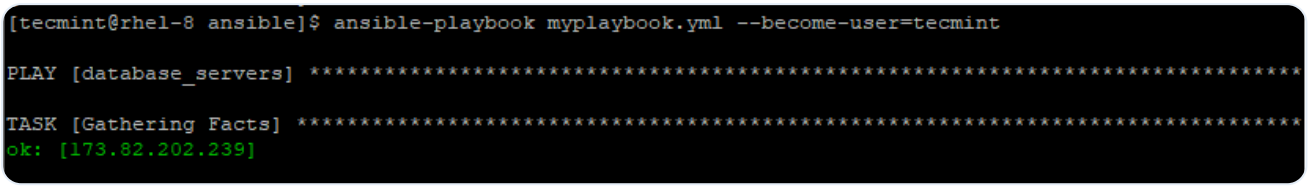
Ansible Become Pass

- `--become`, `-b` – This allows you to run the task as a root user without prompting for a password.

```
$ ansible-playbook myplaybook.yml --become
```

- `--become-user=BECOME_USER` – It allows you to run tasks as another user.

```
$ ansible-playbook myplaybook.yml --become-user=tecmint
```



Ansible Become User

Validate a Working Configuration using Ad-Hoc Ansible Commands

Sometimes, you may want to perform quick and simple tasks on remote hosts or servers in Ansible without necessarily having to create a playbook. In that case, you would require to run an **ad-hoc** command.

What is an Adhoc Command?

An ansible **ad-hoc command** is a one-line command that helps you execute simple tasks in a simple yet efficient manner without the need of creating playbooks. Such tasks include copying files between hosts, rebooting servers, adding & removing users and installing a single package.

In this tutorial, we explore various applications of Ansible Ad-Hoc commands. We are going to use the **inventory file** below for a demonstration.

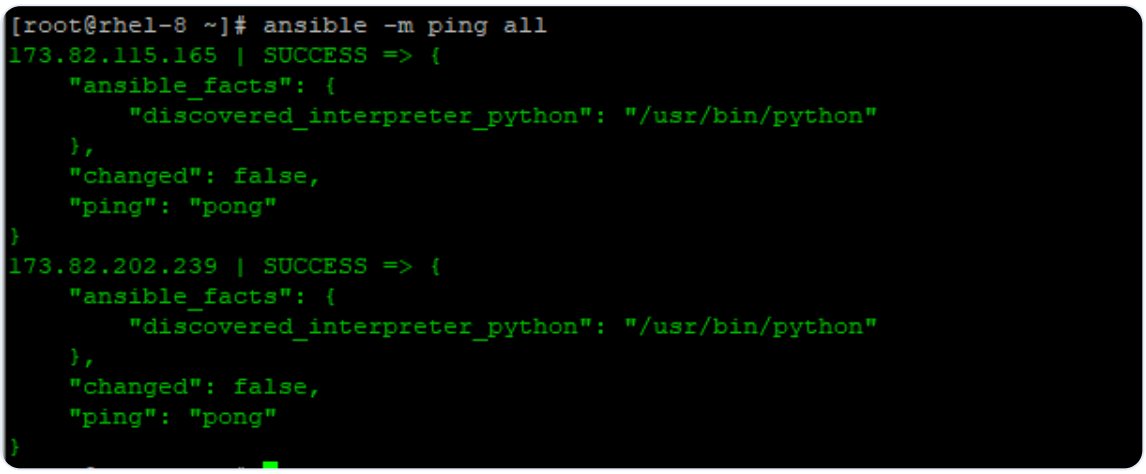
```
[webservers]
173.82.115.165

[database_servers]
173.82.202.239
```

Basic Usage of Adhoc Commands

The most basic usage of Ansible-Adhoc commands is pinging a host or a group of hosts.

```
# ansible -m ping all
```

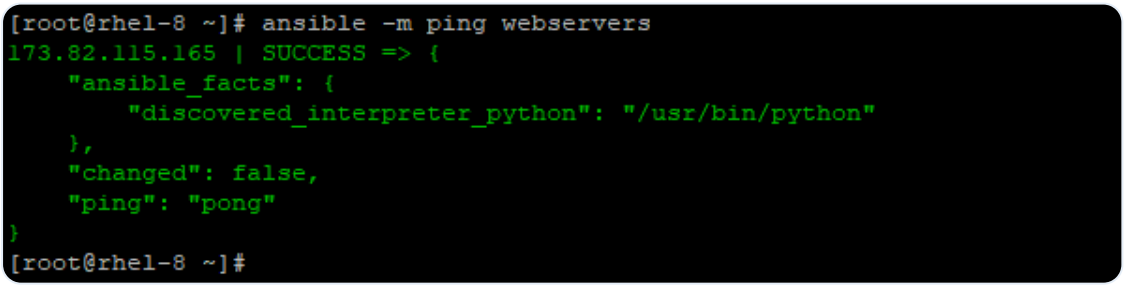
In the above command, the `-m` parameter is the module option. Ping is the adhoc command and the second parameter all represents all hosts in the inventory file. The output of the command is shown below:



Ansible Ping All Hosts

To ping, a particular group of hosts, replace 'all' parameter with the group name. In the example below, we are testing connectivity with hosts under the webservers group.

```
# ansible -m ping webservers
```



Ansible Ping Group of Hosts

Additionally, you can use the `-a` attribute to specify regular Linux commands in double quotation marks. For example, to check system uptime of remote systems, run:

```
# ansible -a "uptime" all
```

```
[root@rhel-8 ~]# ansible -a "uptime" all
173.82.202.239 | CHANGED | rc=0 >>
 12:44:18 up  1:02,  1 user,  load average: 0.01, 0.04, 0.00

173.82.115.165 | CHANGED | rc=0 >>
 12:48:39 up 1 day,  1:14,  1 user,  load average: 0.00, 0.01, 0.05
```

Ansible Check Uptime of Remote Host

To check disk usage of remote hosts run.

```
# ansible -a "df -Th" all
```

```
[root@rhel-8 ~]# ansible -a "df -Th" all
173.82.115.165 | CHANGED | rc=0 >>
Filesystem     Type      Size  Used Avail Use% Mounted on
/dev/vda1      ext4       20G  2.3G   17G  13% /
devtmpfs       devtmpfs  484M     0  484M   0% /dev
tmpfs          tmpfs     495M     0  495M   0% /dev/shm
tmpfs          tmpfs     495M   44M  451M   9% /run
tmpfs          tmpfs     495M     0  495M   0% /sys/fs/cgroup
tmpfs          tmpfs      99M     0   99M   0% /run/user/0

173.82.202.239 | CHANGED | rc=0 >>
Filesystem     Type      Size  Used Avail Use% Mounted on
udev           devtmpfs  461M     0  461M   0% /dev
tmpfs          tmpfs      99M  5.4M   93M   6% /run
/dev/vda1      ext4       20G  4.3G   15G  23% /
tmpfs          tmpfs     491M     0  491M   0% /dev/shm
tmpfs          tmpfs     5.0M     0  5.0M   0% /run/lock
tmpfs          tmpfs     491M     0  491M   0% /sys/fs/cgroup
/dev/loop0     squashfs   55M   55M     0 100% /snap/core18/1192
/dev/loop1     squashfs   55M   55M     0 100% /snap/core18/1223
/dev/loop2     squashfs  8.7M  8.7M     0 100% /snap/nmap/564
tmpfs          tmpfs      99M     0   99M   0% /run/user/0
```

Ansible Check Disk Usage of Remote Hosts

There are hundreds upon hundreds of modules that you can use with Adhoc command. To view the entire list of modules with their descriptions, run the command below.

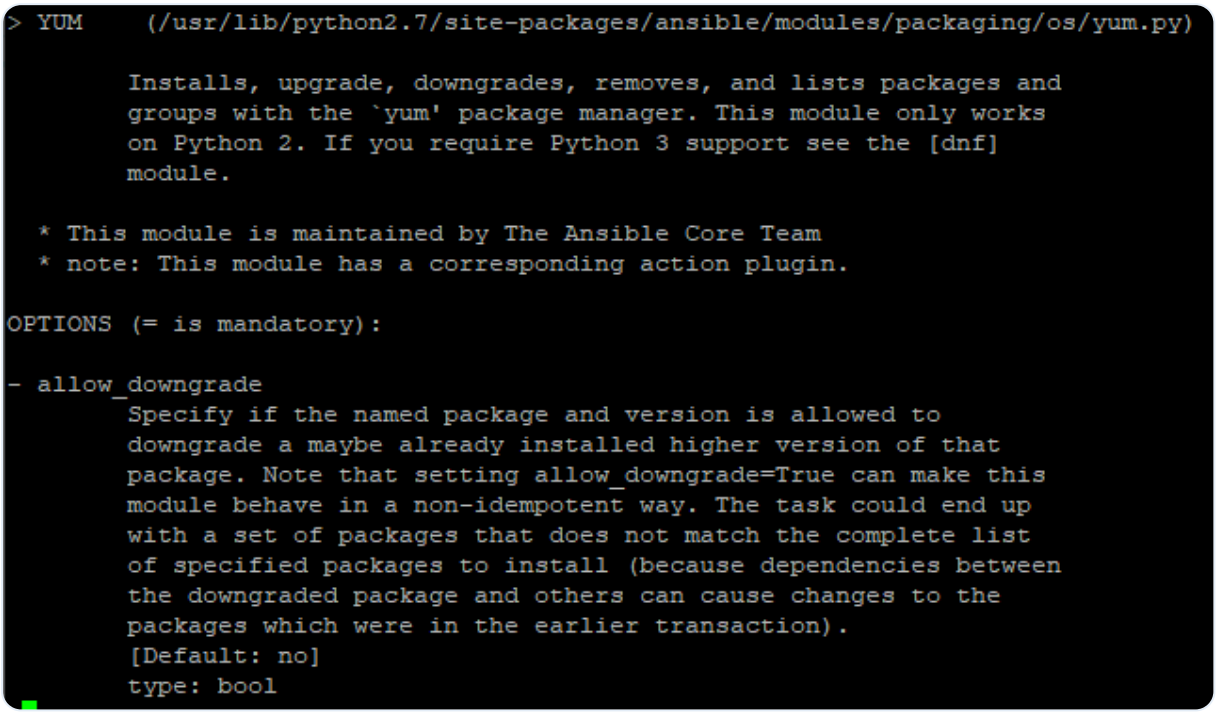
```
# ansible-doc -l
```

To view detailed information about a particular module, run the command.

```
# ansible-doc module_name
```

For example, to search for more details about the yum module run:

```
# ansible-doc yum
```



Ansible Check Yum Module

Managing Packages / Services with Ansible

Ansible adhoc commands can be used for the installation and removal of packages using yum and apt package managers.

To install Apache web server on the CentOS 7 host under webservers group in the inventory file run the command:

```
# ansible webservers -m yum -a "name=httpd  state=present"
```

```
[root@rhel-8 ~]# ansible webservers -m yum -a "name=httpd  state=present"
173.82.115.165 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": true,
    "changes": {
        "installed": [
            "httpd"
        ]
    },
    "msg": "",
    "rc": 0,
    "results": [
        "Loaded plugins: fastestmirror\nLoading mirror speeds from cached hostfile\n * base:
```

Ansible Install Apache on Remote Hosts

To verify the installation of the Apache web server, log in to the remote client and run.

```
# rpm -qa | grep httpd
```

```
[root@centos-7 ~]#
[root@centos-7 ~]# rpm -qa | grep httpd
httpd-tools-2.4.6-90.el7.centos.x86_64
httpd-2.4.6-90.el7.centos.x86_64
[root@centos-7 ~]#
```

Confirm Apache Installation

To uninstall Apache, simple change the state from present to absent.

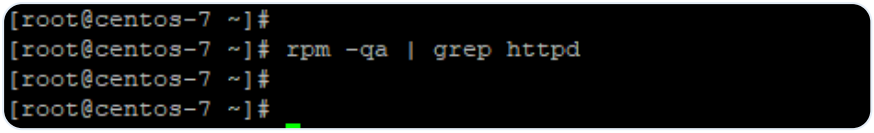
```
# ansible webservers -m yum -a "name=httpd  state=absent"
```

```
[root@rhel-8 ~]# ansible webservers -m yum -a "name=httpd  state=absent"
173.82.115.165 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": true,
    "changes": {
        "removed": [
            "httpd"
        ]
    },
    "msg": "",
    "rc": 0,
```

Ansible Remove Apache

Again, to confirm the removal of httpd run.

```
# rpm -qa | grep httpd
```



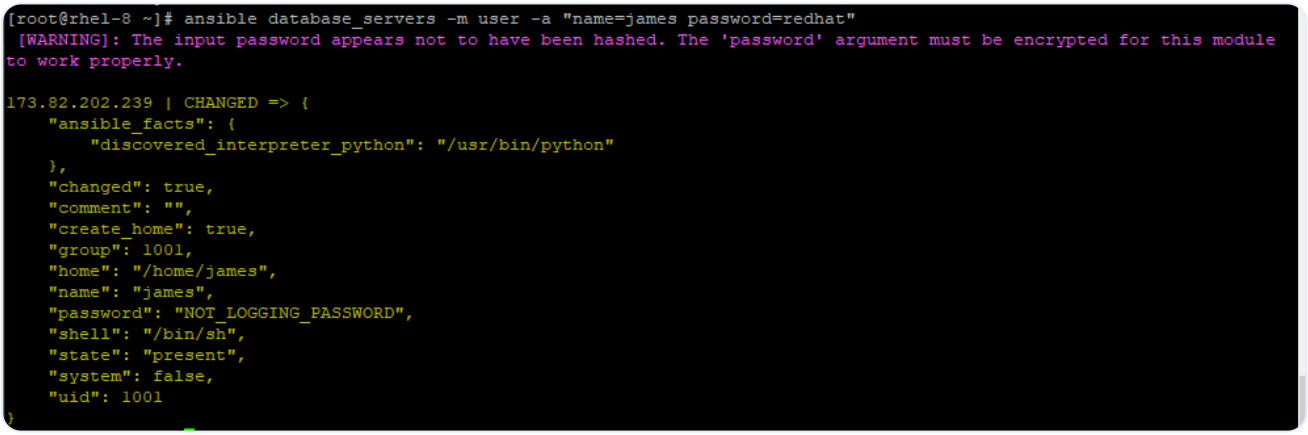
Confirm Removal of Apache

As observed, Apache web server packages have been purged.

Creating Users and Groups Using Ansible

When creating users, the '**user**' module comes in handy. To create a new user **james** with password **redhat** on the client system database_server, issue the command.

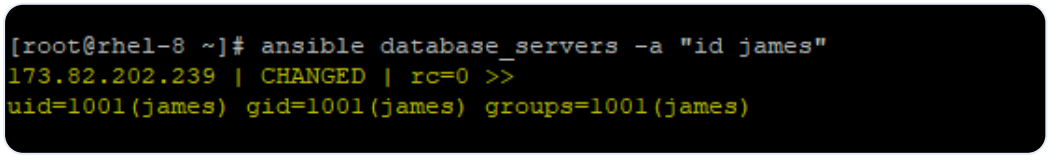
```
# ansible database_server -m user -a "name=james password=redhat"
```



Ansible Create User on Remote Hosts

To confirm the creation of the new user, run the command:

```
# ansible database_servers -a "id james"
```



Ansible Confirm User Creation

To remove the user, run the command:

```
# ansible database_servers -m user -a "name=james state=absent"
```

```
[root@rhel-8 ~]# ansible database_servers -m user -a "name=james state=absent"
173.82.202.239 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": true,
    "force": false,
    "name": "james",
    "remove": false,
    "state": "absent"
}
```

Ansible Remove User

Privilege Escalation

If you are running Ansible as a regular user, Ansible provides privilege escalation in remote hosts using the `--become` option to acquire root privileges and `-k` to prompt for the password.

For example, to run the Ansible adhoc command 'netstat -pnltu' with the privileged option – `-become` and option `-K` to prompt for the root user's password to run the command.

```
$ ansible webservers -m shell -a 'netstat -pnltu' --become -K
```

```
[tecmint@rhel-8 ~]$ ansible webservers -m shell -a 'netstat -pnltu' --become -K
BECOME password:
173.82.115.165 | CHANGED | rc=0 >>
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 173.82.115.165:3306     0.0.0.0:*               LISTEN      710/mysqld
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      654/sshd
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      808/master
tcp6       0      0 :::22                   :::*                    LISTEN      654/sshd
udp        0      0 173.82.115.165:123      0.0.0.0:*                           420/ntpd
udp        0      0 127.0.0.1:123           0.0.0.0:*                           420/ntpd
udp        0      0 0.0.0.0:123             0.0.0.0:*                           420/ntpd
udp6       0      0 fe80::216:3eff:fe91:123 :::*                                420/ntpd
udp6       0      0 ::1:123                 :::*                                420/ntpd
udp6       0      0 :::123                  :::*                                420/ntpd
```

Ansible Privilege Escalation

To become another user other than root, use the `--become-user` attribute.

For example to run 'df -Th' as tecmint user on the remote hosts and prompt for the password run:

```
$ ansible all -m shell -a 'df -Th' --become-user tecmint -K
```

```
[tecmint@rhel-8 ~]$  ansible all -m shell -a 'df -Th' --become-user tecmint -K
BECOME password:
173.82.115.165 | CHANGED | rc=0 >>
Filesystem     Type      Size  Used Avail Use% Mounted on
/dev/vda1      ext4       20G  2.3G   17G  12% /
devtmpfs       devtmpfs  484M     0  484M   0% /dev
tmpfs          tmpfs     495M     0  495M   0% /dev/shm
tmpfs          tmpfs     495M   57M  439M  12% /run
tmpfs          tmpfs     495M     0  495M   0% /sys/fs/cgroup
tmpfs          tmpfs      99M     0   99M   0% /run/user/1002

173.82.202.239 | CHANGED | rc=0 >>
Filesystem     Type      Size  Used Avail Use% Mounted on
udev           devtmpfs  461M     0  461M   0% /dev
tmpfs          tmpfs      99M  5.4M   93M   6% /run
/dev/vda1      ext4       20G  4.5G   15G  24% /
tmpfs          tmpfs     491M     0  491M   0% /dev/shm
tmpfs          tmpfs     5.0M     0  5.0M   0% /run/lock
tmpfs          tmpfs     491M     0  491M   0% /sys/fs/cgroup
/dev/loop0     squashfs   55M   55M     0 100% /snap/core18/1192
/dev/loop1     squashfs   55M   55M     0 100% /snap/core18/1223
/dev/loop2     squashfs  8.7M  8.7M     0 100% /snap/nmap/564
tmpfs          tmpfs      99M     0   99M   0% /run/user/1000
```

Ansible Become Another User

Gathering Facts about Host Systems

Facts refer to detailed information about a system. This includes information about the IP address, system architecture, memory, and CPU to mention a few.

To retrieve information about remote hosts, run the command:

```
$ ansible all -m setup
```

```
[tecmint@rhel-8 ~]$ ansible all -m setup
173.82.115.165 | SUCCESS => {
    "ansible_facts": {
        "ansible_all_ipv4_addresses": [
            "173.82.115.165"
        ],
        "ansible_all_ipv6_addresses": [
            "fe80::216:3eff:fe91:b7d2"
        ],
        "ansible_apparmor": {
            "status": "disabled"
        },
        "ansible_architecture": "x86_64",
        "ansible_bios_date": "04/01/2014",
        "ansible_bios_version": "1.11.0-2.el7",
        "ansible_cmdline": {
            "BOOT_IMAGE": "/boot/vmlinuz-3.10.0-862.2.3.el7.centos.plus.x86_64",
            "LANG": "en_US.UTF-8",
            "biosdevname": "0",
            "console": "tty0",
            "net.ifnames": "0",
            "nomodeset": true,
            "ro": true,
            "root": "/dev/vda1",
            "vconsole.font": "latarcyrheb-sun16",
            "vconsole.keymap": "us"
```

Ansible Gather System Facts

File Transfer / Copy Files

Ansible uses the module **copy** to securely copy files from the Ansible control to multiple remote hosts.

Below is an example of a copy operation:

```
# ansible webservers -m copy -a "src=/var/log/secure dest=/tmp/"
```

```
[root@rhel-8 ~]# ansible webservers -m copy -a "src=/var/log/secure dest=/tmp/"
173.82.115.165 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": true,
    "checksum": "57b9ed284e384e3ef9e2b64b9d8092258feefe21",
    "dest": "/tmp/secure",
    "gid": 0,
    "group": "root",
    "md5sum": "1f01d6eae26ef292ea643b94c3f33366",
    "mode": "0644",
    "owner": "root",
    "size": 7957579,
    "src": "/root/.ansible/tmp/ansible-tmp-1571348281.29-70422240384185/source",
    "state": "file",
    "uid": 0
}
```

Ansible Copy Files to Remote Host

The command copies the /var/log/secure file in the Ansible Control node to remote hosts in the webservers group in the /tmp destination.

You can use the file module to change permissions and file ownership.

```
# ansible webservers -m file -a "dest=/tmp/secure mode=600"
```

```
[root@rhel-8 /]# ansible webservers -m file -a "dest=/tmp/secure mode=600"
173.82.115.165 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": true,
    "gid": 0,
    "group": "root",
    "mode": "0600",
    "owner": "root",
    "path": "/tmp/secure",
    "size": 7957579,
    "state": "file",
    "uid": 0
}
```

Ansible Change File Permissions

Additionally, you can append the owner and group arguments as shown:

```
# ansible webservers -m file -a "dest=/tmp/secure mode=600 owner=tecmint gr
```

```
[root@rhel-8 /]# ansible webservers -m file -a "dest=/tmp/secure mode=600 owner=tecmint group=tecmint"
173.82.115.165 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": true,
    "gid": 1002,
    "group": "tecmint",
    "mode": "0600",
    "owner": "tecmint",
    "path": "/tmp/secure",
    "size": 7957579,
    "state": "file",
    "uid": 1002
}
```

Ansible Append User and Group Attributes

You can also create directories, in a similar manner to `mkdir -p` as shown.

```
$ ansible webservers -m file -a "dest=/path/to/directory mode=755 owner=tec
```

For example,

```
$ ansible webservers -m file -a "dest=/home/tecmint/data mode=755 owner=tec
```

```
[root@rhel-8 /]# ansible webservers -m file -a "dest=/home/tecmint/data mode=755 owner=tecmint group=tecmint state=directory"
173.82.115.165 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": true,
    "gid": 1002,
    "group": "tecmint",
    "mode": "0755",
    "owner": "tecmint",
    "path": "/home/tecmint/data",
    "size": 4096,
    "state": "directory",
    "uid": 1002
```

Ansible Create a Directory

Conclusion

In this article, we shed light on how you can configure managed nodes to run Ansible ad-hoc commands to manage remote hosts. We do hope you found it useful. Give it a shot and let us know how it went.

ad-hoc command examples, Ansible Exam Guide, Ansible Tips

Hey TecMint readers,

Exciting news! Every month, our top blog commenters will have the chance to win fantastic rewards, like free Linux eBooks such as RHCE, RHCSA, LFCS, Learn Linux, and Awk, each worth $20!

Learn more about the contest and stand a chance to win by sharing your thoughts below!

PREVIOUS ARTICLE:

How to Manage Networking with NetworkManager in RHEL/CentOS 8

NEXT ARTICLE:

How to Install Stratis to Manage Layered Local Storage on RHEL 8

Ravi Saive

I am an experienced GNU/Linux expert and a full-stack software developer with over a decade in the field of Linux and Open Source technologies

Each tutorial at TecMint is created by a team of experienced Linux system administrators so that it meets our high-quality *standards*.

Related Posts

```
aaronk@tecmint:~$ ansible prod_servers -a "systemctl status firewalld" -u root
[redacted].235 | FAILED! => {
    "changed": false,
    "module_stderr": "Shared connection to [redacted].235 closed.\r\n",
    "module_stdout": "/bin/sh: /usr/bin/python: No such file or directory\r\n",
    "msg": "MODULE FAILURE",
    "rc": 127
}
[redacted]80 | FAILED! => {
    "changed": false,
    "module_stderr": "Shared connection to [redacted]80 closed.\r\n",
    "module_stdout": "/bin/sh: /usr/bin/python: No such file or directory\r\n",
    "msg": "MODULE FAILURE",
    "rc": 127
}
aaronk@tecmint:~$
```

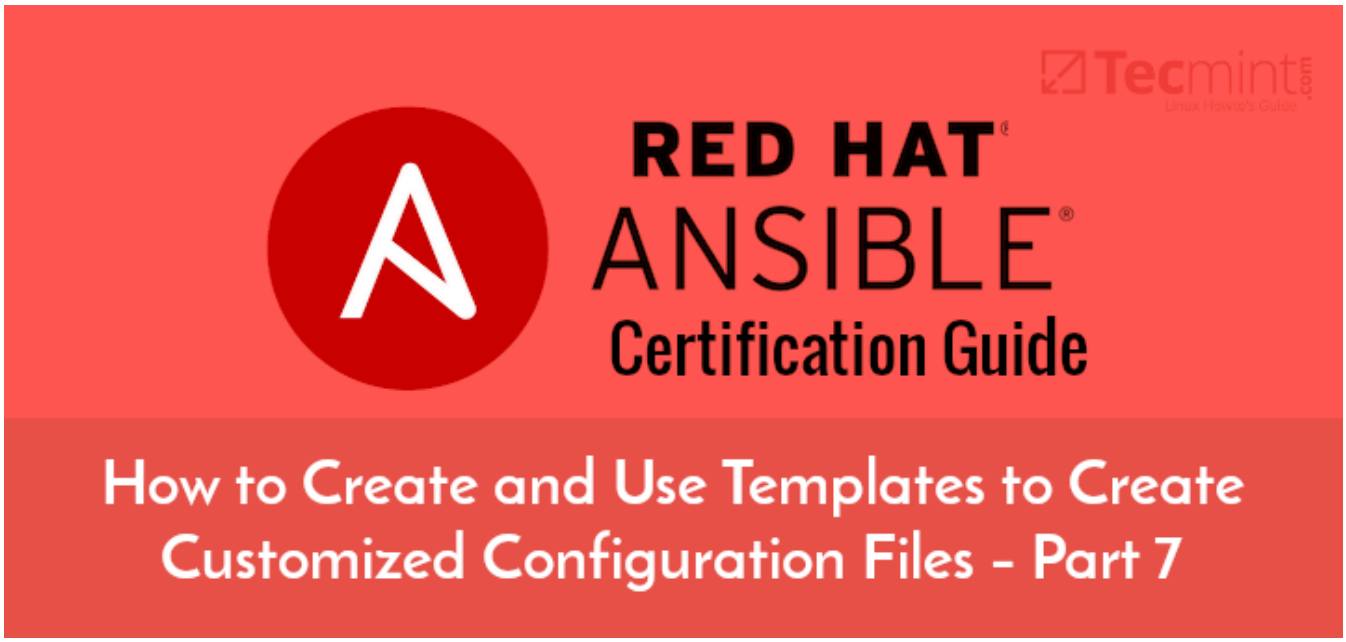
How to Fix “Shared connection to x.x.xx closed” Ansible Error

Tecmint’s Guide to RedHat Ansible Automation Exam Preparation Guide

How to Use Ansible Vault in Playbooks to Protect Sensitive Data – Part 10

How to Create and Download Roles on Ansible Galaxy and Use Them – Part 9

How to Work with Ansible Variables and Facts – Part 8



How to Create Templates in Ansible to Create Configurations On Managed Nodes – Part 7

1 Comment

Leave a Reply

Quick Comment

May 11, 2022 at 7:30 pm

I think you need both `--become` and `--become-user` for `--become-user` to work.
See: https://stackoverflow.com/a/38290243/13203944

Reply

Got Something to Say? Join the Discussion...

Thank you for taking the time to share your thoughts with us. We appreciate your decision to leave a comment and value your contribution to the discussion. It's important to note that we moderate all comments in accordance with our comment policy to ensure a respectful and constructive conversation.

Rest assured that your email address will remain private and will not be published or shared with anyone. We prioritize the privacy and security of our users.

Name *

Email *

Save my name, email, and website in this browser for the next time I comment.

Post Comment

Search...

Do You Enjoy My Blog?

Support from readers like YOU keeps this blog running. Buying me a cup of coffee is a simple and affordable way to show your appreciation and help keep the posts coming!

Buy Me a Coffee

Linux Commands and Tools

The Power of Linux "History Command" in Bash Shell

7 'dmesg' Commands for Troubleshooting and Collecting Information of Linux Systems

How to Test Website Loading Speed in Linux Terminal

Tips to Create ISO from CD, Watch User Activity and Check Memory Usages of Browser

How to Keep 'sudo' Password Timeout Session Longer in Linux

How to Create Multiple User Accounts in Linux

Linux Server Monitoring Tools

Inxi – A Powerful Feature-Rich Commandline System Information Tool for Linux

Smem – Reports Memory Consumption Per-Process and Per-User in Linux

How to Use Nmap Script Engine (NSE) Scripts in Linux

Cpustat – Monitors CPU Utilization by Running Processes in Linux

Sysstat – All-in-One System Performance and Usage Activity Monitoring Tool For Linux

How to Add Hosts in OpenNMS Monitoring Server

Learn Linux Tricks & Tips

Display Command Output or File Contents in Column Format

How to Monitor Progress of (Copy/Backup/Compress) Data using 'pv' Command

4 Ways to Find Server Public IP Address in Linux Terminal

How to Change UUID of Partition in Linux Filesystem

4 Ways to Batch Convert Your PNG to JPG and Vice-Versa

How to Transfer Files Between Two Computers using nc and pv Commands

Best Linux Tools

11 Best Screen Recorders For Linux in 2024

The 27 Best IDEs and Code Editors for Linux

5 Best Microsoft Word Alternatives for Linux in 2024

3 Essential Linux Disk Scanning Tools (GUI and Terminal Based)

15 Best Free and Open Source Software (FOSS) Programs for Linux

5 Best PDF to Word Converters for Linux