**How to Install and Configure ‘Ansible’ Automation Tool for IT Management – Part 1**

Ansible is an open source, powerful automation software for configuring, managing and deploying software applications on the nodes without any downtime just by using SSH. Today, most of the IT Automation tools runs as a agent in remote host, but ansible just need a SSH connection and Python (2.4 or later) to be installed on the remote nodes to perform it’s action.

#### How Ansible Works?

There are many similar automation tools available like Puppet, Capistrano, Chef, Salt, Space Walk etc, but Ansible categorize into two types of server: controlling machines and nodes.

The controlling machine, where Ansible is installed and Nodes are managed by this controlling machine over SSH. The location of nodes are specified by controlling machine through its inventory.

The controlling machine (Ansible) deploys modules to nodes using SSH protocol and these modules are stored temporarily on remote nodes and communicate with the Ansible machine through a JSON connection over the standard output.

Ansible is agent-less, that means no need of any agent installation on remote nodes, so it means there are no any background daemons or programs are executing for Ansible, when it’s not managing any nodes.

Ansible can handle 100’s of nodes from a single system over SSH connection and the entire operation can be handled and executed by one single command ‘ansible’. But, in some cases, where you required to execute multiple commands for a deployment, here we can build playbooks.

Playbooks are bunch of commands which can perform multiple tasks and each playbooks are in YAML file format.

#### What’s the Use of Ansible

**Ansible** can be used in IT infrastructure to manage and deploy software applications to remote nodes. For example, let’s say you need to deploy a single software or multiple software to 100’s of nodes by a single command, here ansible comes into picture, with the help of Ansible you can deploy as many as applications to many nodes with one single command, but you must have a little programming knowledge for understanding the ansible scripts.

We’ve compiled a series on Ansible, title ‘**Preparation for the Deployment of your IT Infrastructure with Ansible IT Automation Tool**‘, through parts 1-4 and covers the following topics.

In this article, we will show you how to install ‘Ansible’ on RHEL/CentOS 7/6, Fedora 21-19, Ubuntu 14.10-13.04 and Debian 7/6 systems and also we will go through some basics on how how to manage a server by installing packages, applying updates and much more from basic to pro.

#### Prerequisites

#### Operating System: RHEL/CentOS/Fedora and Ubuntu/Debian/Linux Mint

1. **Ji**nja2: A modern, fast and easy to use stand-alone template engine for Python.
2. PyYAML: A YAML parser and emitter for the Python programming language.
3. parmiko: A native Python SSHv2 channel library.
4. httplib2: A comprehensive HTTP client library.
5. sshpass: A non-interactive ssh password authentication.

#### My Environment Setup

##### **Controlling Machine – Ansible**

Operating System : Linux Mint 17.1 Rebecca

IP Address : 192.168.0.254

Host-name : tecmint.instrcutor.com

User : tecmint

##### **Remote Nodes**

Node 1: 192.168.0.112

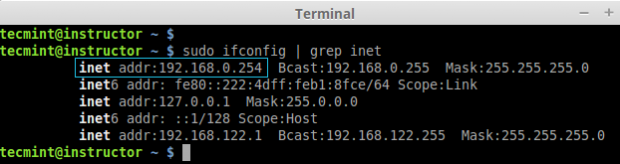
Node 2: 192.168.0.113

Node 3: 192.168.0.114

### Step 1: Installing Controlling Machine – Ansible

**1.** Before installing ‘**Ansible**‘ on the server, let’s first verify the details of the server like hostname and IP Address. Login into server as a root user and execute the below command to confirm system settings that we’re going to use for this setup.

# sudo ifconfig | grep inet

[](https://www.tecmint.com/wp-content/uploads/2015/01/Verify-System-Details.png)

*Verify System Details*

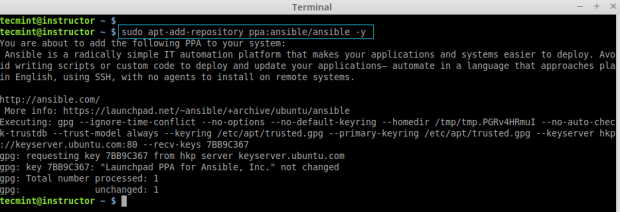
**2.** Once you confirm your system settings, it’s time to install ‘Ansible’ software on the system.

##### **On Ubuntu/Debian/Linux Mint**

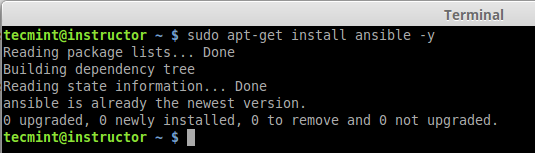
Here we are going to use official Ansible PPA repository on the system, just run the below commands to add the repository.

$ sudo apt-add-repository ppa:ansible/ansible -y

$ sudo apt-get update && sudo apt-get install ansible -y

[](https://www.tecmint.com/wp-content/uploads/2015/01/Add-Ansible-PPA3.png)

*Add Ansible PPA*

[](https://www.tecmint.com/wp-content/uploads/2015/01/Install-Ansible-in-Ubuntu1.png)

*Install Ansible in Ubuntu*

##### **On RHEL/CentOS/Fedora**

Unfortunately, there are no official Ansible repository for RedHat based clones, but we can install Ansible by enabling epel repository under RHEL/CentOS 6, 7 and currently supported fedora distributions.

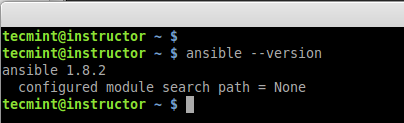
Fedora users can directly install Ansible through default repository, but if you are using RHEL/CentOS 6, 7, you have to [enable EPEL repo](https://www.tecmint.com/how-to-enable-epel-repository-for-rhel-centos-6-5/).

After configuring epel repository, you can install Ansible using following command.

$ sudo yum install ansible -y

After installed successfully, you can verify the version by executing below command.

# ansible --version

[](https://www.tecmint.com/wp-content/uploads/2015/01/Verify-Ansible-Version1.png)

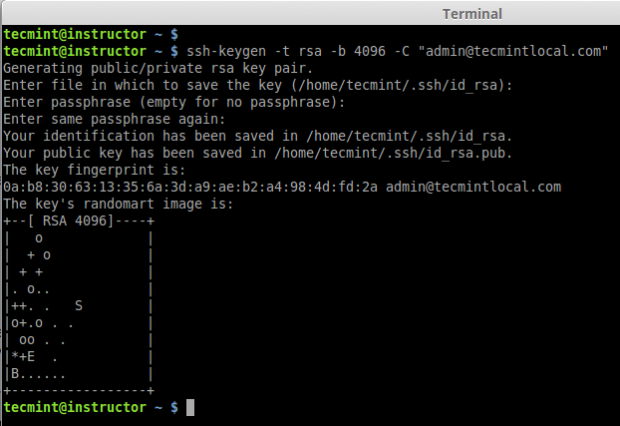
*Verify Ansible Version*

### Step 2: Preparing SSH Keys to Remote Hosts

**4.** To perform any deployment or management from the localhost to remote host first we need to create and copy the ssh keys to the remote host. In every remote host there will be a user account **tecmint** (in your case may be different user).

First let we create a SSH key using below command and copy the key to remote hosts.

# ssh-keygen -t rsa -b 4096 -C "admin@tecmintlocal.com"

[](https://www.tecmint.com/wp-content/uploads/2015/01/Create-SSH-Key1.png)

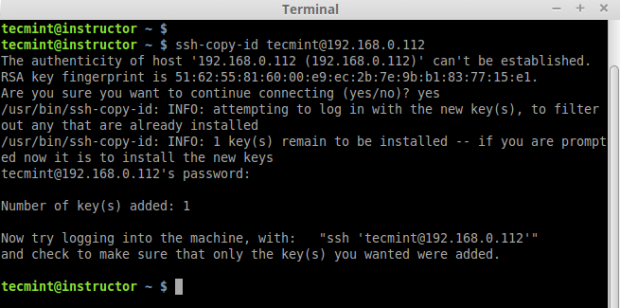
*Create SSH Key*

**5.** After creating SSH Key successfully, now copy the created key to all three remote server’s.

# ssh-copy-id tecmint@192.168.0.112

# ssh-copy-id tecmint@192.168.0.113

# ssh-copy-id tecmint@192.168.0.114

[](https://www.tecmint.com/wp-content/uploads/2015/01/Copy-SSH-Key1.png)

### Step 3: Creating Inventory File for Remote Hosts

Inventory file, This file hold the host information’s like which host we need to get connect from local to remote. Default inventory file will be under /etc/ansible/hosts.

**7.** Now let’s add these three hosts to inventory file. Open and edit file using your favourite editor, Here I use vim.

# sudo vim /etc/ansible/hosts

Add the following three hosts IP address..

[web-servers]

192.168.0.112

192.168.0.113

192.168.0.114

**Note**: The ‘**web-servers**‘ in the brackets indicates as group names, it is used in classifying systems and deciding which systems you are going to controlling at what times and for what reason.

*Create Ansible Inventory File*

**8.** Now time to check our all 3 server by just doing a ping from my localhost. To perform the action we need to use the command ‘ansible‘ with options ‘**-m**‘ (module) and ‘**-all**‘ (group of servers).

# ansible -m ping web-servers

OR

# ansible -m ping -all

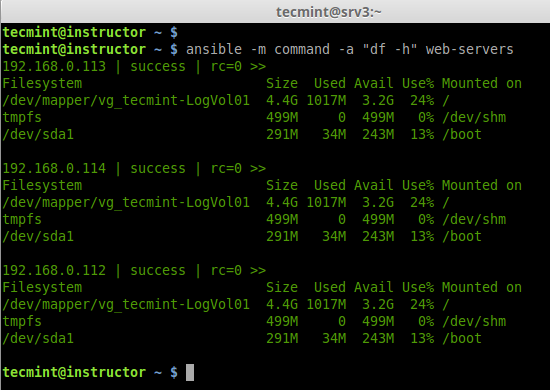
*Ping Remote Hosts*

In the above example, we’ve used ping module with Ansible command to ping all remote hosts at ones, the same way there are various modules can be used with Ansible, you can find available modules from ansible Official site [here](https://docs.ansible.com/list_of_all_modules.html" \t "_blank).

**9.** Now, here we are using another module called ‘**command**‘, which is used to execute list of commands (like, df, free, uptim, etc.) on all selected remote hosts at one go, for example watch out few examples shown below.

**a.** To check the partitions on all remote hosts

# ansible -m command -a "df -h" web-servers

[](https://www.tecmint.com/wp-content/uploads/2015/01/Check-Disk-Space-on-all-Hosts1.png)

*Check Disk Space on all Hosts*

**b.** Check memory usage on all remote hosts.

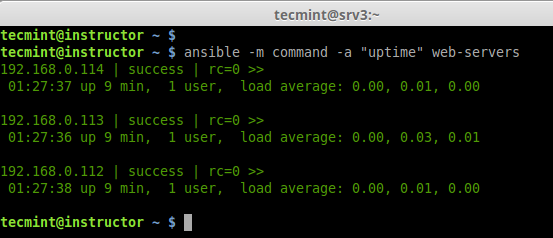
# ansible -m command -a "free -mt" web-servers

[](https://www.tecmint.com/wp-content/uploads/2015/01/Check-Memory-on-all-Hosts1.png)

*Check Memory on all Hosts*

**c.** Checking Uptime for all 3 servers.

# ansible -m command -a "uptime" web-servers

[](https://www.tecmint.com/wp-content/uploads/2015/01/Check-uptime-on-all-Hosts1.png)

*Check uptime on all Hosts*

**d.** Check for hostname and Architecture.

# ansible -m command -a "arch" web-servers

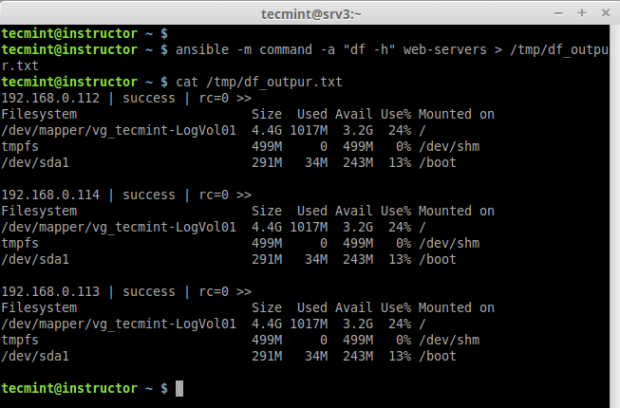
# ansible -m shell -a "hostname" web-servers

[](https://www.tecmint.com/wp-content/uploads/2015/01/Check-hostname-on-all-Hosts1.png)

*Check hostname on all Hosts*

**e.** If we need the output to any file we can redirect as below.

# ansible -m command -a "df -h" web-servers > /tmp/df\_outpur.txt

[](https://www.tecmint.com/wp-content/uploads/2015/01/Redirect-Output-to-File1.png)

*Redirect Output to File*

Like this way, we can run many shell commands using ansible as what we have run the above steps.

### Conclusion

Okay, We can see how to in next article.

Ansible is a Powerful IT automation tool which is must every sysadmins for deploying applications and managing server’s at one go. Among any other automation tool such as puppet, Capistrano, salt, Ansible is quit very interesting and very easy to setup for production environment. Capistrano oh no i feel headache please leave me alone :p this what i used to say.

Ansible use only SSH as there agent. We don’t have to install and run any agent in the remote servers. Hope this article will be interesting one for you too. In our next article, I will show you how to setup the directory structure for Ansible deployment and creating playbooks and working with it.

Till then keep on tracking us to get updated articles and don’t forget to tell us your opinions on the Ansible and also tell us do you use any other automation tool which is more powerful than Ansible….

### Reference Links

[http://www.ansible.com/get-started](https://www.ansible.com/get-started" \t "_blank)  
[http://docs.ansible.com/](https://docs.ansible.com/" \t "_blank)

6. After copying all SSH Keys to remote host, now perform a ssh key authentication on all remote hosts to check whether authentication working or not.

$ ssh tecmint@192.168.0.112

$ ssh tecmint@192.168.0.113

$ ssh tecmint@192.168.0.114

*SSH Key Authentication*