****

Install and Configure Ansible on Ubuntu 14.04

****

Configuration management systems are designed to make controlling large numbers of servers easy for administrators and operations teams. They allow you to control many different systems in an automated way from one central location.

While there are many popular configuration management systems available for Linux systems, such as Chef and Puppet, these are often more complex than many people want or need. **Ansible** is a great alternative to these options because it has a much smaller overhead to get started.

It communicates over normal SSH channels in order to retrieve information from remote machines, issue commands, and copy files. Because of this, an Ansible system does not require any additional software to be installed on the client computers.

Any computer that you can administer through SSH, you can also administer through Ansible.

Configuration files are mainly written in the YAML data serialization format due to its expressive nature and its similarity to popular markup languages. Ansible can interact with clients through either command line tools or through its configuration scripts called Playbooks.

****

Install Ansible

****

To do this effectively, we need to install the software-properties-common package, which will give us the ability to work with PPAs easily. (This package was called python-software-properties on older versions of Ubuntu.)

 # sudo apt-get update

 # sudo apt-get install software-properties-common

Once the package is installed, we can add the Ansible PPA by typing the following command:

 # sudo apt-add-repository ppa:ansible/ansible

Press ENTER to accept the PPA addition.

Next, we need to refresh our system's package index so that it is aware of the packages available in the PPA. Afterwards, we can install the software:

 # sudo apt-get update

 # sudo apt-get install ansible

We now have all of the software required to administer our servers through Ansible.
Edit /etc/hosts

 # nano /etc/hosts

172.17.0.2 ansible-master
172.17.0.3 ansible-agent1
172.17.0.4 ansible-agent2
172.17.0.5 ansible-agent3

add the all clients-hostname in here with ip
also add ansible-master hostname and ip in each client's /etc/hosts

 # nano /etc/hosts

172.17.0.2 ansible-master

****

Set Up SSH Keys

****

As we mentioned above, Ansible primarily communicates with client computers through SSH. While it certainly has the ability to handle password-based SSH authentication, SSH keys help keep things simple.
Create a user (create the same user for all machines so that make work more easy)

 # adduser ansible

 # usermod -aG sudo ansible

 # visudo

ansible ALL=(ALL) NOPASSWD: ALL

 # sudo su ansible

 # cd /home/ansible/

 # ssh-keygen

 # ssh-copy-id username@client-hostname
the "pub_key" saved as "autherized_keys" in .ssh directory,
if there are same user in both machine, we can use
 # ssh-copy-id client-hostname

 # ssh ansible@ansible-master # it will not ask password

same steps done in master for other client-hosts and also in each client-hosts for communicating client to master.

****

Configuring Ansible Hosts

****

Ansible keeps track of all of the servers that it knows about through a "hosts" file. We need to set up this file first before we can begin to communicate with our other computers.

 # sudo nano /etc/ansible/hosts

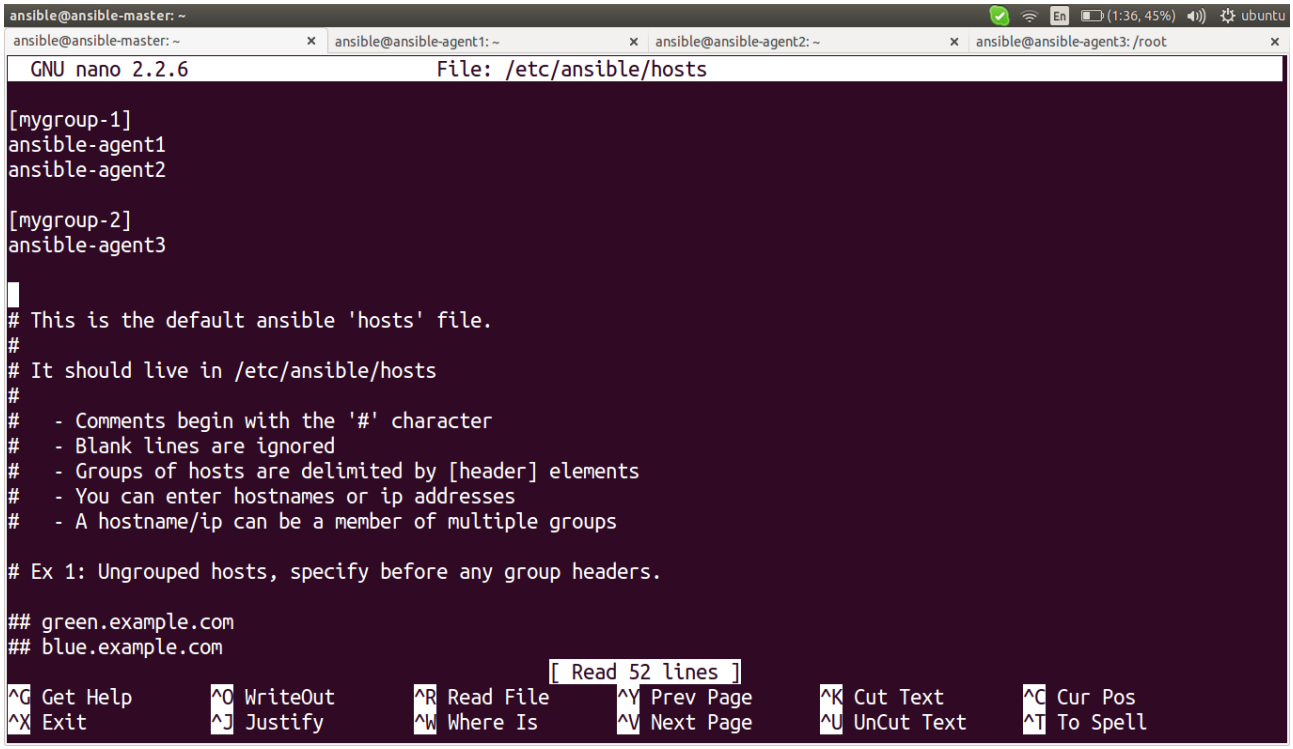
The hosts file is fairly flexible and can be configured in a few different ways. The syntax we are going to use though looks something like this:

[group_name]
hostname or ip

eg:

```
[mygroup-1]          # group
ansible-agent1
ansible-agent2

[mygroup-2]          # group
ansible-agent3
```



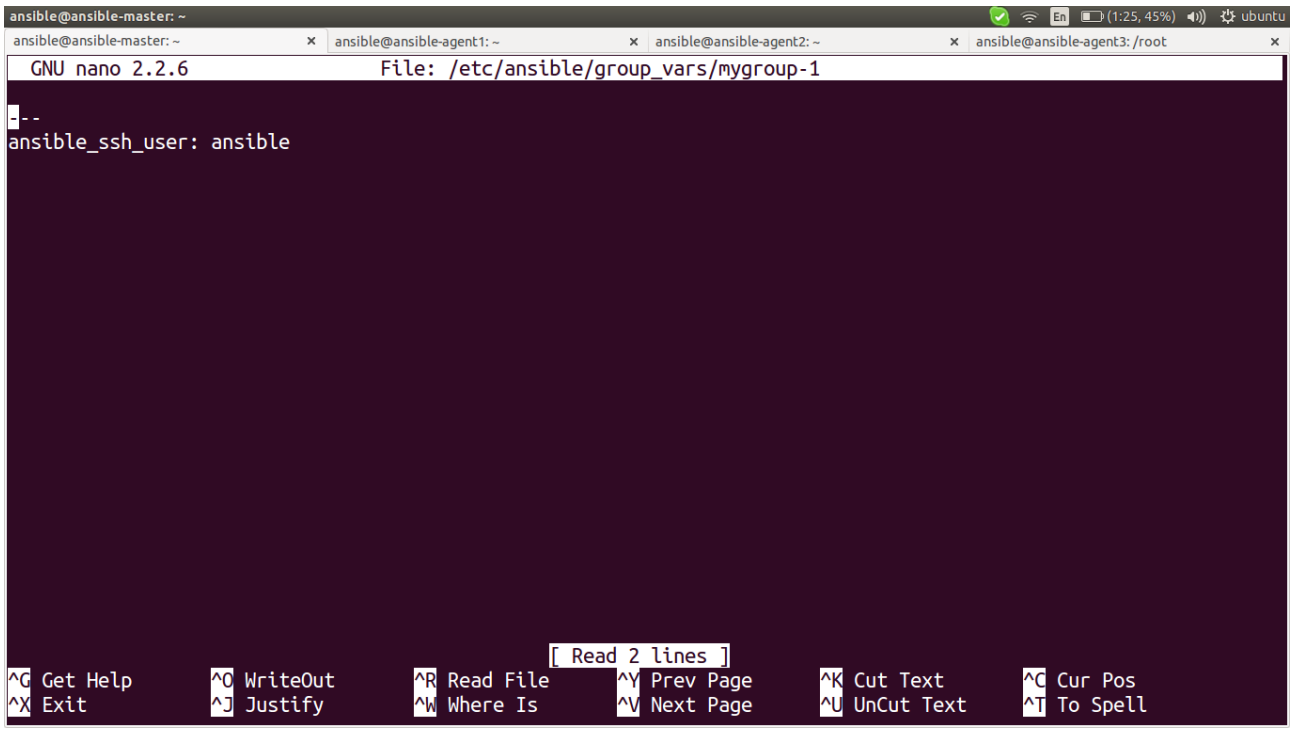
We can create a file that tells all of the servers in the particular group to connect using the particular user.

To do this, we will create a directory in the Ansible configuration structure called group_vars. Within this folder, we can create YAML-formatted files for each group we want to configure:

 # sudo mkdir /etc/ansible/group_vars

 # sudo nano /etc/ansible/group_vars/mygroup-1

---
ansible_ssh_user: ansible



We can put our configuration in here. YAML files start with "---", so make sure you don't forget that part.

If you want to specify configuration details for every server, regardless of group association, you can put those details in a file at /etc/ansible/group_vars/all. Individual hosts can be configured by creating files under a directory at /etc/ansible/host_vars.

****

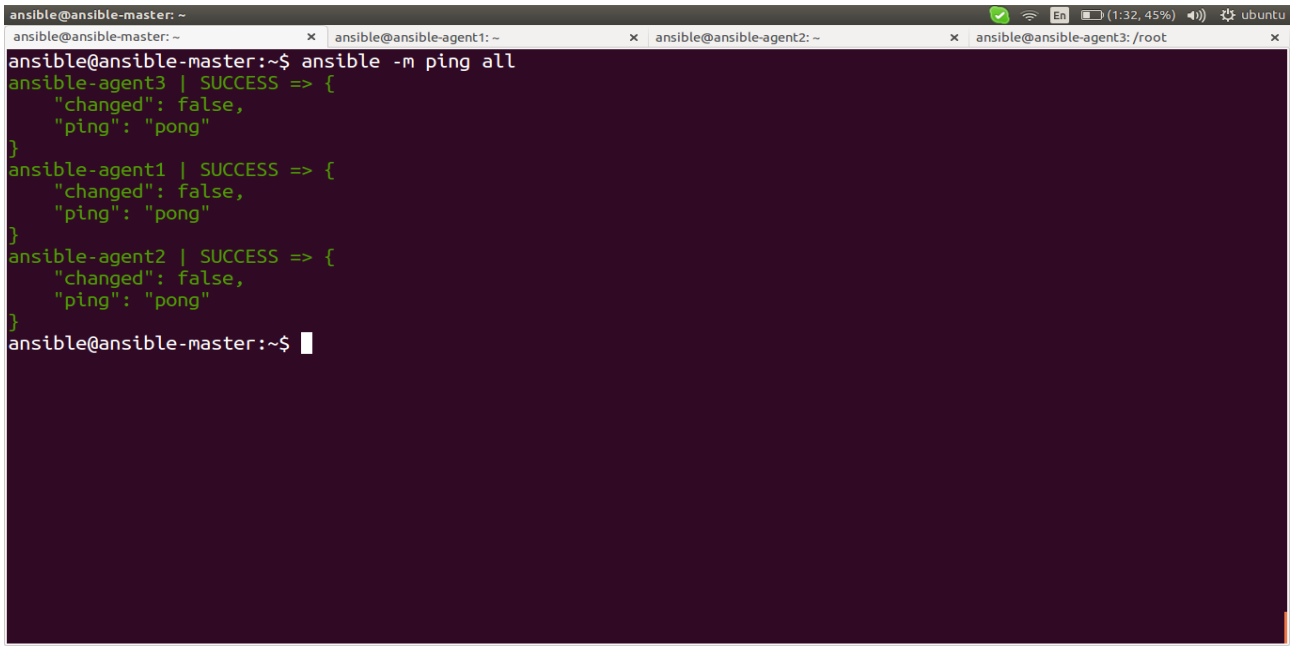
Using Simple Ansible Commands

****

Now that we have our hosts set up and enough configuration details to allow us to successfully connect to our hosts, we can try out our very first command.

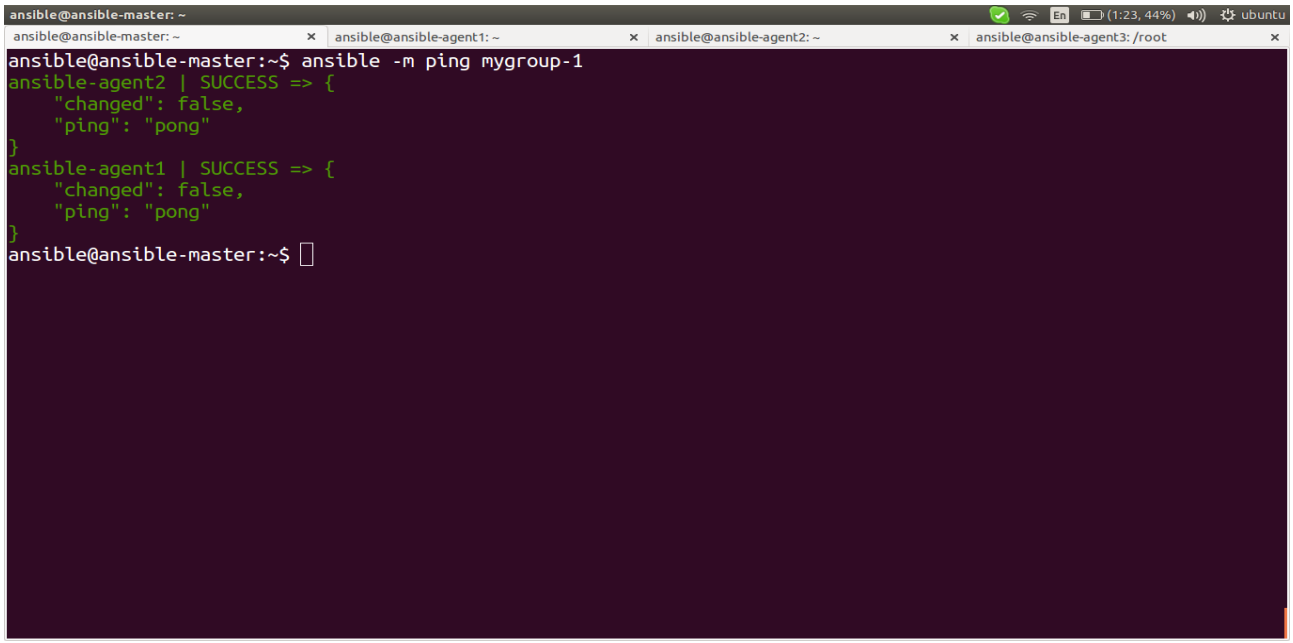
Ping all of the servers you configured by typing:

 # ansible -m ping all



to ping to particular group

 # ansible -m ping mygroup-1



The "shell" module lets us send a terminal command to the remote host and retrieve the results. For instance, to find out the memory usage on our host1 machine, we could use:

 # ansible -m shell -a 'free -m' mygroup-1

```
ansible@ansible-master:~$ ansible -m shell -a 'free -m' mygroup-1
ansible-agent1 | SUCCESS | rc=0 >>
             total       used       free     shared    buffers     cached
Mem:          3853       2427       1425         30         24        382
-/+ buffers/cache:       2021       1831
Swap:         7627         75       7552

ansible-agent2 | SUCCESS | rc=0 >>
             total       used       free     shared    buffers     cached
Mem:          3853       2422       1431         30         24        382
-/+ buffers/cache:       2015       1837
Swap:         7627         75       7552

ansible@ansible-master:~$
```

links::

installation and configuration>>
https://www.digitalocean.com/community/tutorials/how-to-install-and-configure-ansible-on-ubuntu-14-04

****

playbook

****

to run playbook ::

ansible-playbook apache.yml --sudo

create playbook under {ansible-home-dir}
eg: /etc/ansible/

**********playbook**********

to copy file from local to remote ::

1) create a directory in {ansible-home-dir}
eg: /etc/ansible/files

2) add some file in “files” directory

3) done

**********playbook**********

to add some content to file ::

```
- name: copy-contents
  copy: content="hi" dest="/home/new"
```

**********playbook**********

to paste some file from local to remote ::

```
- name: new
  copy: src="files/new" dest="/home/new1"
```

**********playbook**********

to execute some commands ::

```
- name: shell-command
  action: shell /home/mysql.sh
```

**********playbook**********

to restart when made a changes ::

```
  handlers:
  - name: service
    action: service name=apache2
                    state=restarted
```

and use notify: service # add this in appropriate path

**********playbook**********

to create files ::

```
---
- hosts: mygroup-1
  tasks:
  - name: create files
    file: path=/home/new
          owner=ansible
          group=ansible
          mode=0744
          state=touch
  - name: copy
    copy: content="hi" dest="/home/new"
  - name: create
    file: path=/home/new1
          owner=ansible
          group=ansible
          mode=0744
          state=touch
  - name: new
    copy: src="files/new" dest="/home/new1"
```

**********playbook**********

to create directory ::

```
---
- hosts: mygroup-1
  tasks:
  - name: create directory
    file: path="/home/new-dir" owner=ansible group=ansible mode=777 state=directory
  - name: copy to directory
    copy: src="files/" dest="/home/new-dir"
```

**********playbook**********

to add users ::

```
---
- hosts: mygroup-1
  tasks:
  - name: group
    action: group name=biz2
                gid=1561
                state=present

  - name: "user creation"
    action: user name=biz
            password=biz
            shell=/bin/bash
            uid=1500
            groups=biz2

  - name: shell
    shell: chage -d 0 biz
```

**********playbook**********

to install application ::

```
---
- hosts: ansible-agent1
  tasks:
  - name: apache2 istallation
    action: apt pkg=apache2
                state=installed

  - name: service
    action: service name=apache2
                    state=started

  - name: index
    action: copy src="files/index.html" dest="/var/www/html/index.html"
    notify: service

  - name: permission
    action: file path=/var/www/html/index.html
            owner=www-data
            group=www-data
    notify: service

  handlers:
  - name: service
    action: service name=apache2
                    state=restarted
```

**********playbook**********

to run scripts ::

```
---
- hosts: ansible-agent1
  tasks:
  - name: file creation
    action: file path=/home/mysql.sh
                owner=root
                mode=777
                group=root
                state=touch
  - name: copy-file
    action: copy src="files/mysql.sh" dest="/home/mysql.sh"

  - name: shell-command
    action: shell /home/mysql.sh
```

**********playbook**********

to include one playbook to another playbook ::

step:1 >> add include option to a playbook which is going to run.

```
---
- hosts: mygroup-1
  tasks:
  - name: apache2 istallation
    action: apt pkg=apache2
               state=installed

  - name: service
    action: service name=apache2
                  state=started

  - name: index
    action: copy src="files/index.html" dest="/var/www/html/index.html"
    notify: service

  - name: permission
    action: file path=/var/www/html/index.html
           owner=www-data
           group=www-data
    notify: service

  handlers:
  - name: service
    action: service name=apache2
                  state=restarted

  - include: locate.yml
```

step:2 >> comment the appropriate commands

```
--
#- hosts: mygroup-1
#  tasks:
  - name: locate istallation
    action: apt pkg=locate
            state=installed
```

note >> need to commend the started lines like -hosts, tasks, handlers...etc....

**********playbook**********

links ::

basics >>
https://lowendbox.com/blog/getting-started-with-ansible/

basic-commands >>
http://docs.ansible.com/ansible/glossary.html

playbook >>
https://www.digitalocean.com/community/tutorials/how-to-create-ansible-playbooks-to-automate-system-configuration-on-ubuntu

handlers >>
http://stackoverflow.com/questions/34018862/how-to-force-handler-to-run-before-executing-a-task-in-ansible

best >>
http://docs.ansible.com/ansible/playbooks_best_practices.html