**Vagrant Commands:**

vagrant status

Vagrant up

Vagrant up <vmname>

Vagrant ssh acs

**Vagrant file:**

Vagrant.configure(2) do |config|

  config.vm.define "acs" do |acs|

    acs.vm.box ="ubuntu/trusty64"

    acs.vm.hostname="acs"

    acs.vm.network "private\_network", ip: "192.168.33.10"

 end

  config.vm.define "web" do |web|

    web.vm.box ="ubuntu/trusty64"

    web.vm.hostname="web"

    web.vm.network "private\_network", ip: "192.168.33.20"

    web.vm.network "forwarded\_port", guest: 80, host:8083

 end

  config.vm.define "db" do |db|

    db.vm.box ="ubuntu/trusty64"

    db.vm.hostname="db"

    db.vm.network "private\_network", ip: "192.168.33.30"

 end

end

**Virtual box commands**

vboxmanage list runningvms

**Ansible command:**

Ubuntu ansible instalation:

sudo apt-get install ansible

centOS- ansible installation:

sudo yum install epel-release

sudo yum install ansible

**Update the library:**

sudo yum install gcc

sudo yum install python-setuptools

sudo easy\_install pip

sudo yum install python-devel

**Running ping module in web and db servers**

ansible 192.168.33.20 -i inventory -u vagrant -m ping -k

ansible 192.168.33.30 -i inventory -u vagrant -m ping -k

ansible all -i inventory -u vagrant -m command -a "/usr/sbin/yum update -y" (to run in all servers)

while I m running ping module it will ask for the password of the servers and the default password is vagrant once we type that it will through error saying

vagrant@acs:~$ cd exercise1/

vagrant@acs:~/exercise1$ ansible 192.168.33.30 -i inventory -u vagrant -m ping -k

SSH password:

192.168.33.30 | FAILED => SSH encountered an unknown error during the connection. We recommend you re-run the command using -vvvv, which will enable SSH debugging output to help diagnose the issue

Hostkey checking is enable by default

This is because remote system is not connected with the acs before so for that we need to add fingerprint on to the host so for that

We need to add remote into acs

Ssh [vagrant@192.168.33.20](mailto:vagrant@192.168.33.20)

**Give vagrant as password**

**Result:**

ssh vagrant@192.168.33.30

The authenticity of host '192.168.33.30 (192.168.33.30)' can't be established.

ECDSA key fingerprint is 63:6b:b2:f9:2d:12:6f:28:93:8b:79:94:e5:11:f5:19.

Are you sure you want to continue connecting (yes/no)? yes

Warning: Permanently added '192.168.33.30' (ECDSA) to the list of known hosts.

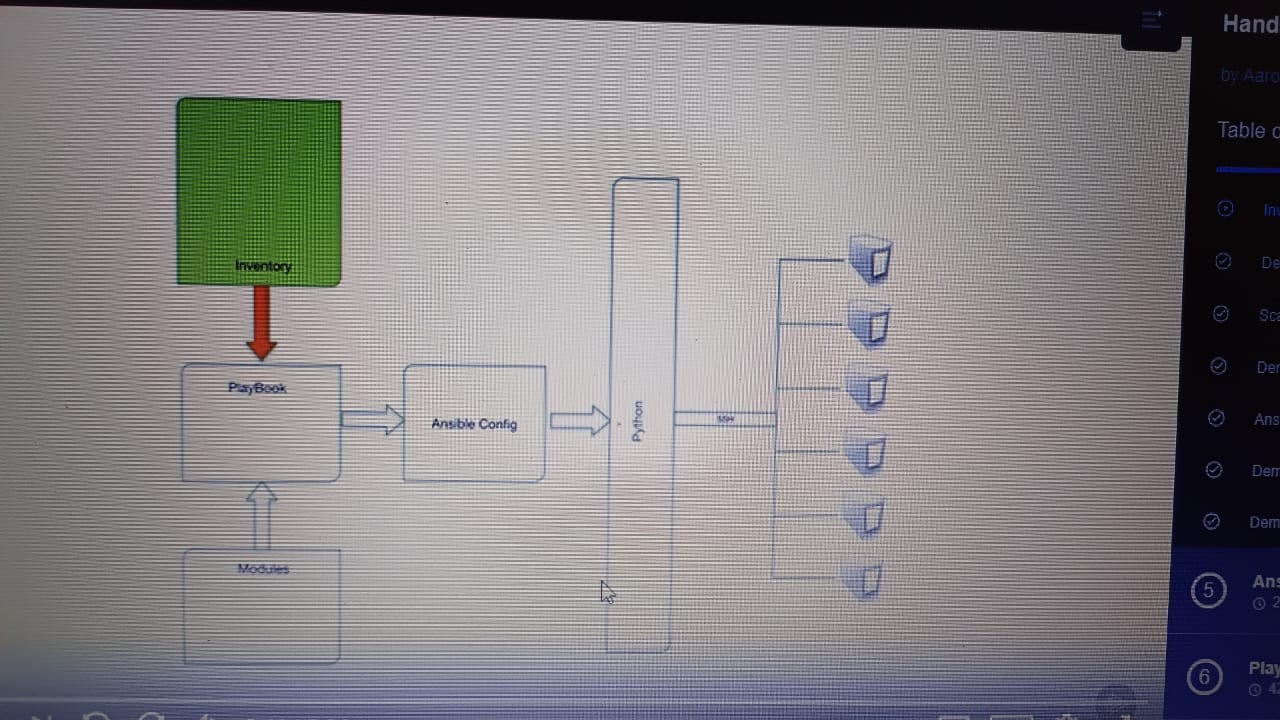
vagrant@192.168.33.30's password:

Welcome to Ubuntu 14.04.6 LTS (GNU/Linux 3.13.0-170-generic x86\_64)

After running the command if we see the known host file which is there in the cat ~/.ssh/known-host

You can able to see the key

**Architecture of the ansible:**



**Inventory features:**

1.Behavioral parameters

2. Groups

3. Groups of groups

4. Assign variables

5. Scaling out using multiple files

6. Inventory file may be Static/Dynamic

**Creating our inventory file:**

* Add behavioural parameters
* Create host-based variables
* Create a group
* Create group-based variables

#This inventory I have created in the control system and pinging modules in the remote system

#1st stage

#host

web1 ansible\_ssh\_host=192.168.33.20 ansible\_ssh\_user=vagrant ansible\_ssh\_pass=vagrant

command:

$ ansible web1 -i inventory -m ping

=====================================================================================

#2nd stage

#host

web1 ansible\_ssh\_host=192.168.33.20 ansible\_ssh\_user=vagrant ansible\_ssh\_pass=vagrant

#group

[webservers]

web1

command:

$ ansible webservers -i inventory -m ping

======================================================================================

#3rd stage

#host

web1 ansible\_ssh\_host=192.168.33.20 ansible\_ssh\_user=vagrant ansible\_ssh\_pass=vagrant

db1 ansible\_ssh\_host=192.168.33.30 ansible\_ssh\_user=vagrant ansible\_ssh\_pass=vagrant

#group

[webservers]

web1

db1

command:

$ ansible webservers -i inventory -m ping

======================================================================================

#4th stage

#host

web1 ansible\_ssh\_host=192.168.33.20 ansible\_ssh\_user=vagrant ansible\_ssh\_pass=vagrant

db1 ansible\_ssh\_host=192.168.33.30 ansible\_ssh\_user=vagrant ansible\_ssh\_pass=vagrant

#group

[webservers]

web1

[dbservers]

db1

#Parent group

[datacenters:children]

webservers

dbservers

command:

$ ansible datacenters -i inventory -m ping

======================================================================================

#5th stage

#host

web1 ansible\_ssh\_host=192.168.33.20

db1 ansible\_ssh\_host=192.168.33.30

#group

[webservers]

web1

[dbservers]

db1

#Parent group

[datacenters:children]

webservers

dbservers

#variable group

[datacenters:vars]

ansible\_ssh\_user=vagrant

ansible\_ssh\_pass=vagrant

command:

$ ansible datacenters -i inventory -m ping

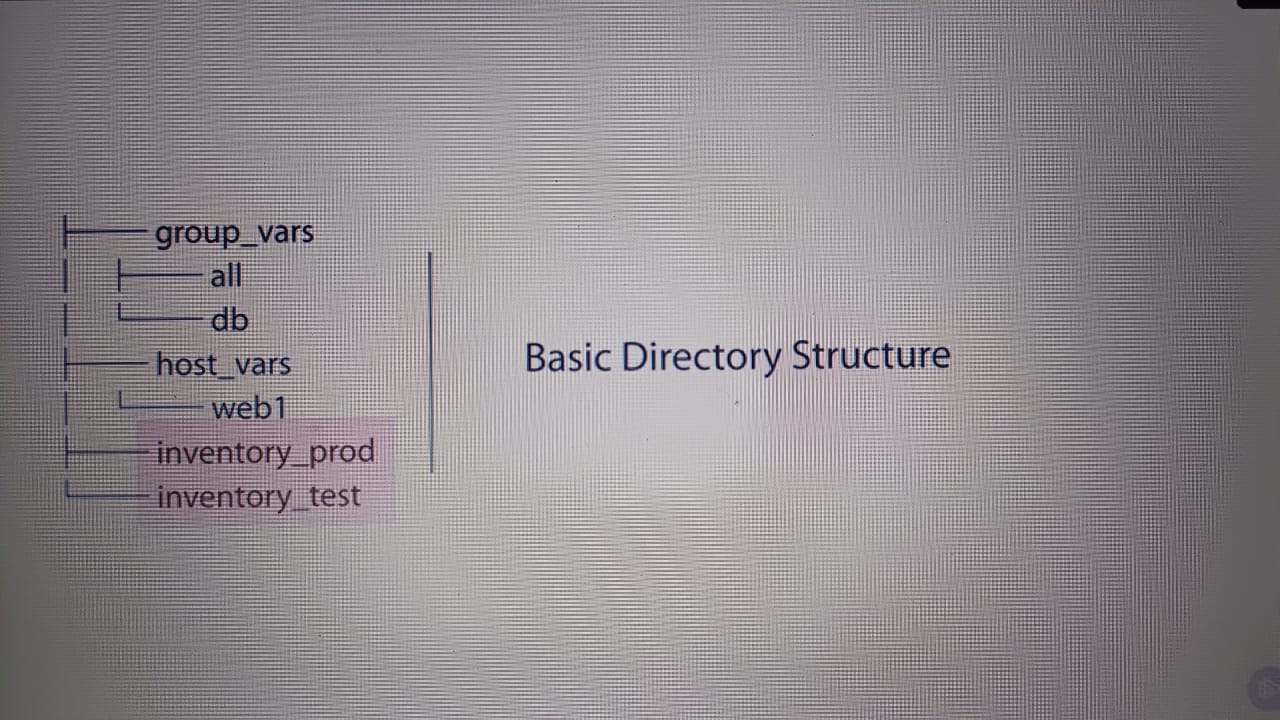
**Scaling-out inventory files:**

Using directories

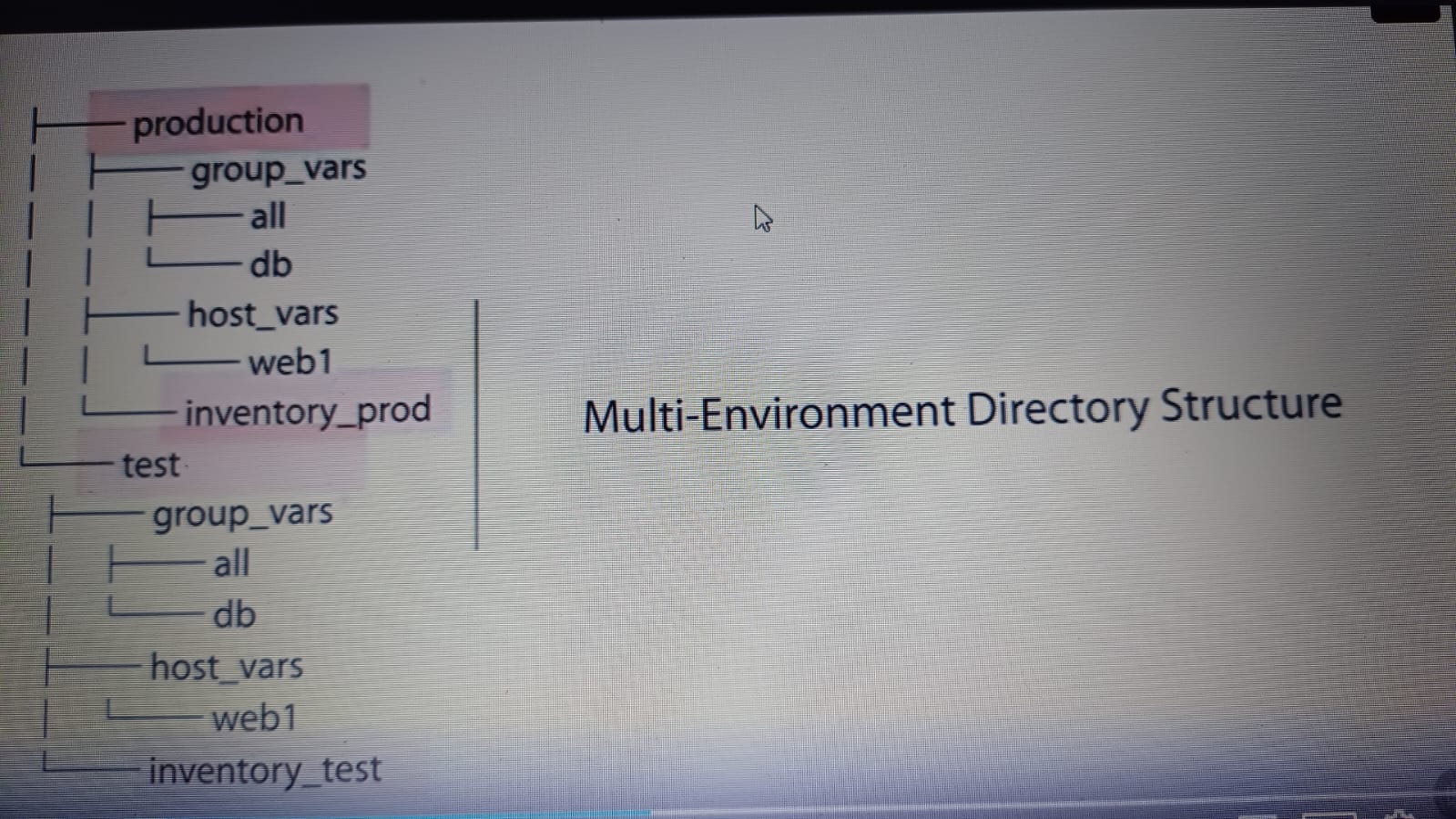
Can use to break-out long-running inventory files

Very useful when dealing with large environment

**Basic Directory structures:**



**Multi-environment directory structure:**



**Tree**

Mkdir exercise

Cd exercise

mkdir production

cd production

touch inventory\_prod

mkdir group\_vars

cd group\_vars

touch all

touch db

cd ..

mkdir host\_vars

cd host\_vars

touch web1

mkdir test

cd test

touch inventory\_test

mkdir group\_vars

cd group\_vars

touch all

touch db

cd ..

mkdir host\_vars

cd host\_vars

touch web1

**Order of operations (Precedence):**

1. (Group\_Vars) All

2. (Group\_Vars) Groupname

3. (Host\_Vars) Hostname------------------------Highest precedence

**Scaling variable files:**

Create group variable in separate file

Show order-of-precedence

There are 462 modules available in ansible

**Module lib for local:**

$ ansible-doc -l

$ ansible-doc -s <name>

$ ansible-doc <name>

**Copy module:**

Copies a file from the local to remote

Has backup capabilities

Can do validation remotely

**Fetch module:**

Pulls a file from remote host to local system

Can use md5 checksums to validate

**Apt modules:**

Manages installed applications on Debian based systems

Can install, update or delete packages

Can update entire system

**Yum modules:**

Manages installed applications on redhat based systems

Can install, update or delete packages

Can update entire system

**Service modules:**

Can stop,start or restart services

Can enable services to start on boot

**Using module to install/Start:**

Browse module doc.: