Provisioning – allocating resources to the container or a virtual machine

It is Transitioning from one system state to another system state.

The configuration of the resources after provisioning, used when you build up a server to perform a specific function.

Similar to virtual machine technology of cloning but that Ansible's provisioning process is actually installing and configuring the software each time, rather than just copying that image.

Example: you have a server and you make it to an FTP server, an Email server or a DB server



**Automation**

Define tasks to be executed automatically

Ordered tasks

Make decisions

Ad-hoc tasks

**Orchestration**

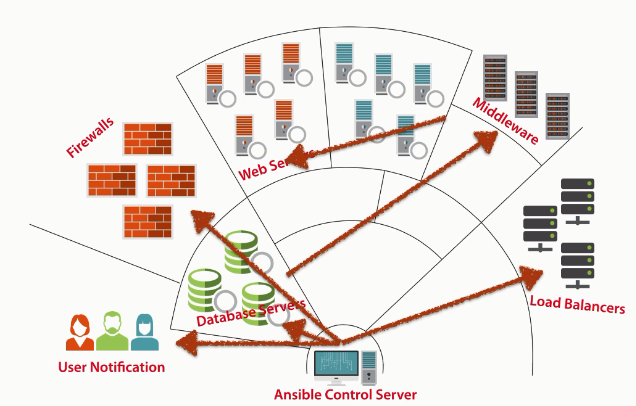
Automating tasks and dependencies across different system

**Why Ansible?**

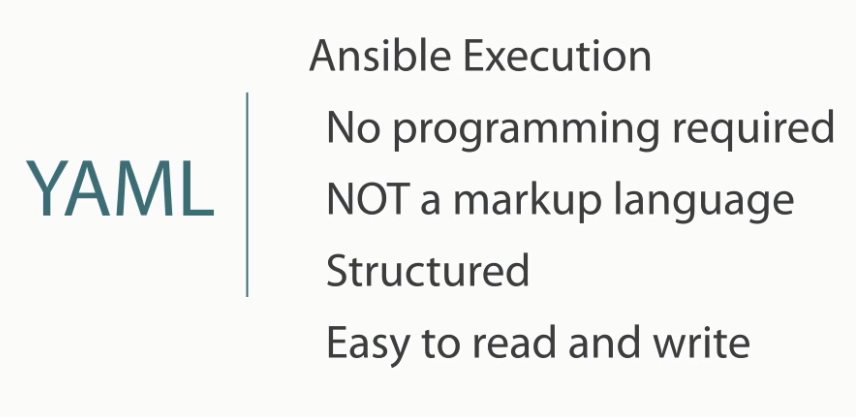
Its Clean

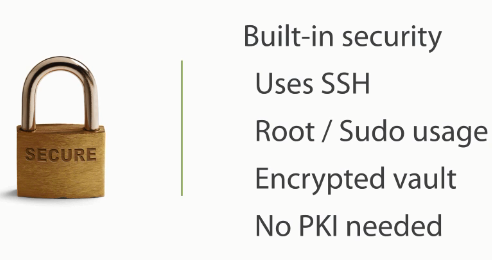
* Server management system
* Less dependencies and more stable, also upgrading ansible is easy
* Fast to implement
* No agents need to be installed on your remote systems (time saving as no need to make agent up and running)
* No database – no index, backups or performance management when dealing with software updates
* Runs playbooks on the remote system

(An Agent is a program that must be installed on the remote machine in order for your system to work with it)



**YAML**





**Architecture and process flow**

Installation

Python based Unix/Linux required

Windows not supported

Need SSH installed

Architecture

Inventory

Modules

Playbook

Ansible configuration file

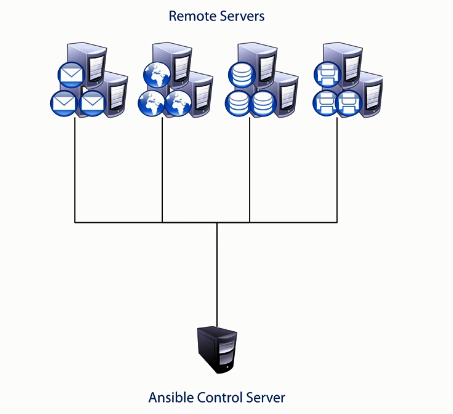
**Python**

Helps in building variables

* Host variables – use variables defined in inventory per host or group
* Facts – use data gathered from the remote managed host
* Dynamic variables

SSH

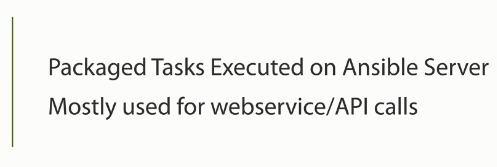
Process in order to talk to the remote server



**Execution types**

Remote execution of plays

Local – when remote box is not executing plays



**Ansible Architecture**

Inventory maps hosts

Configuration sets ansible parameters

Modules define actions

Playbooks to coordinate multiple tasks

Python to build the execution

SSH to deliver the tasks

**Creating Environment**

**Components**

**Vagrant** – to start server up and running – virtual machine controller, define VM’s to startup and initial configs (IP, hostname, etc.)

**Virtual box** – Provide environment to run virtual machines

**Ansible** – Automation/provisioning, application to push configuration and automation to remote systems

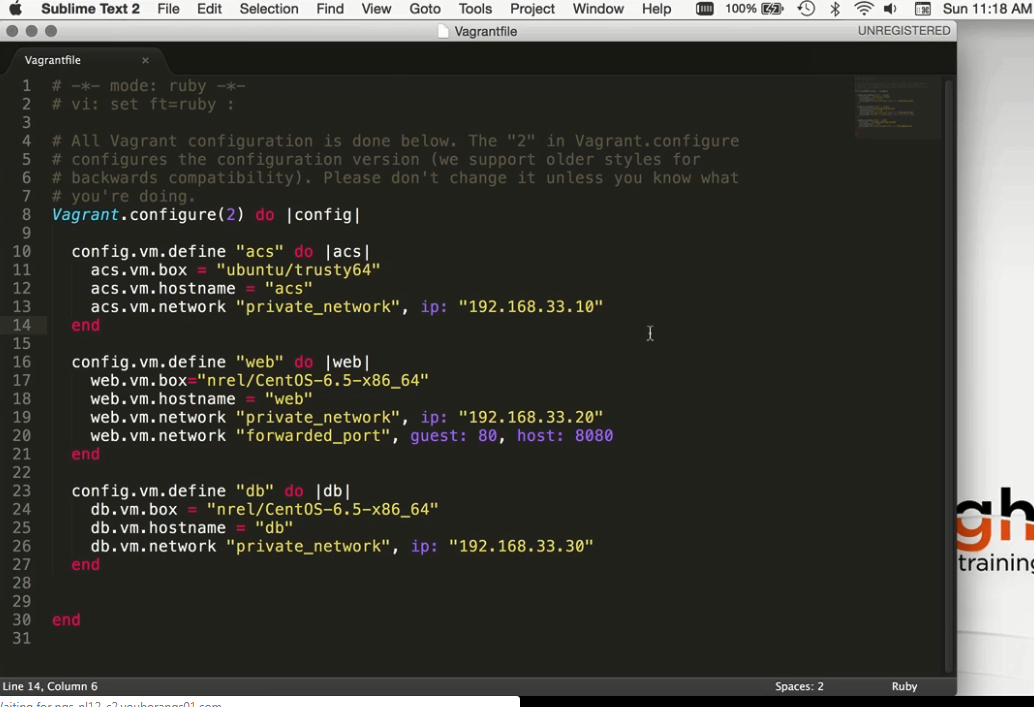
Download from Vagrantup.com and install

Download virtual box which is our hypervisor

After vagrant installation, the main work is in the terminal

Steps

* Type vagrant – to check the vagrant installed
* Vboxmanage – to check virtual box installed
* Mkdir hands-on-ansible
* Vagrant init - Create a vagrant file with default configuration, open and edit in any kind of text editor



Configure vagrant

Use commands:

vagrant up

vboxmanage list runningvms – tells us which files are running

vagrant ssh acs

sudo apt-get install ansible

Using the web box

sudo yum install epel-release

vagrant ssh db

sudo yum install gcc

sudo easy\_install pip

sudo yum install python-devel

sudo pip install ansible

ansible – to verify that the installation is done successfully

exit the db server and go to acs server

vagrant ssh acs

mkdir exercise1

cd exercise1

ls

create a basic inventory file

vi inventory

Put the IP addresses in it

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

SSH password:

ssh vagrant@192.168.33.20