**✅ Step-by-Step: Install Ansible on Ubuntu**

**🔹 Step 1: Update the System**

sudo apt update && sudo apt upgrade -y

**🔹 Step 2: Install Required Dependencies**

sudo apt install software-properties-common -y

**🔹 Step 3: Add the Ansible PPA**

Ansible provides an official maintained PPA (for latest versions):

sudo add-apt-repository --yes --update ppa:ansible/ansible

**🔹 Step 4: Install Ansible**

sudo apt install ansible -y

**🔹 Step 5: Verify the Installation**

ansible --version

**✅ STEP 2: Inject Public Key into EC2s**

You now have 2 options:

**✅ Manual Way: Use .pem to Add Ansible Master’s Public Key**

Let’s assume:

* You already have access to the EC2 instances using a .pem file
* Your Ansible master has a public key at: ~/.ssh/id\_rsa.pub

**🔹 Step 1: Copy Ansible Public Key Locally**

On your Ansible control node (master):

cat ~/.ssh/id\_rsa.pub

Copy the full output (starts with ssh-rsa ...)

**🔹 Step 2: SSH into Target EC2 Using Existing .pem**

chmod 400 DevOps-Shack.pem

ssh -i mykey.pem ubuntu@<TARGET\_IP>

**🔹 Step 3: Manually Add Ansible Public Key**

Once inside the EC2 instance:

mkdir -p ~/.ssh

echo "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDb..." >> ~/.ssh/authorized\_keys

chmod 600 ~/.ssh/authorized\_keys

chmod 700 ~/.ssh

✅ Now your Ansible master can connect without .pem, just using:

ssh ubuntu@<TARGET\_IP>

**If Servers Already Exist**

Create a bash script (inject-ansible-key.sh):

#!/bin/bash

# inject-ansible-key.sh

ANSIBLE\_PUB\_KEY=$(cat ~/.ssh/id\_rsa.pub)

PEM\_FILE="DevOps-Shack.pem"

for ip in $(cat ec2-ip-list.txt); do

echo "Injecting key into $ip"

ssh -o StrictHostKeyChecking=no -i $PEM\_FILE ubuntu@$ip "

mkdir -p ~/.ssh

echo '$ANSIBLE\_PUB\_KEY' >> ~/.ssh/authorized\_keys

chmod 700 ~/.ssh

chmod 600 ~/.ssh/authorized\_keys

"

Done

Run it:

chmod +x inject-ansible-key.sh

./inject-ansible-key.sh

**Script to Auto-Tag All Matching Instances**

#!/bin/bash

# Get all instance IDs with Name=web-ansible

INSTANCE\_IDS=$(aws ec2 describe-instances \

--region ap-south-1 \

--filters "Name=tag:Name,Values=web-ansible" \

--query "Reservations[].Instances[].InstanceId" \

--output text)

echo "Found Instances:"

echo "$INSTANCE\_IDS"

INDEX=1

for instance\_id in $INSTANCE\_IDS; do

NAME\_TAG="web-ansible-$(printf "%02d" $INDEX)"

echo "Tagging $instance\_id as $NAME\_TAG..."

aws ec2 create-tags \

--region ap-south-1 \

--resources $instance\_id \

--tags Key=Name,Value=$NAME\_TAG Key=Role,Value=web Key=Environment,Value=dev Key=Index,Value=$(printf "%02d" $INDEX)

INDEX=$((INDEX+1))

done

**🔹 1. Ping All Hosts**

ansible all -i inventory.ini -m ping

📌 Checks connectivity via Ansible's ping module (uses SSH, not ICMP)

**🔹 2. Run Shell Command on All Hosts**

ansible all -i inventory.ini -m shell -a "uptime"

**🔹 3. Check Disk Space**

ansible all -i inventory.ini -m shell -a "df -h"

**🔹 4. Reboot All Servers**

ansible all -i inventory.ini -m reboot --become

**🔹 5. Install Package (e.g., htop) – Debian/Ubuntu**

ansible all -i inventory.ini -m apt -a "name=htop state=present" --become

**🔹 6. Install Package – RHEL/CentOS**

ansible all -i inventory.ini -m apt -a "update\_cache=yes" --become

ansible all -i inventory.ini -m apt -a "name=maven state=present" --become

**🔹 7. Create a User**

ansible all -i inventory.ini -m user -a "name=devops state=present" --become

**🔹 8. Change File Permissions**

ansible all -i inventory.ini -m file -a "path=/opt/myscript.sh mode=0755" --become

**🔹 9. Copy a File to Remote Hosts**

ansible all -i inventory.ini -m copy -a "src=./a.txt dest=/home/ubuntu/a.txt" --become

**🔹 10. Remove a Package**

ansible all -i inventory.ini -m apt -a "name=maven state=absent" --become

**🔹 11. Start a Service**

ansible all -i inventory.ini -m apt -a "name=nginx state=present" --become

ansible all -i inventory.ini -m service -a "name=nginx state=started" --become

**🔹 12. Stop a Service**

ansible all -i inventory.ini -m service -a "name=nginx state=stopped" --become

**🔹 13. Create a Directory**

ansible all -i inventory.ini -m file -a "path=/opt/mydir state=directory mode=0755" --become

**🔹 14. Gather System Facts (e.g., RAM, CPU)**

ansible all -i inventory.ini -m setup

**🔹 16. Run Command Without Using a Module**

ansible all -i inventory.ini -a "whoami"

🔹 Default module is command if -m is omitted

**🔹 19. Check OS Distribution**

ansible all -i inventory.ini -m shell -a "cat /etc/os-release"

**🔹 20. Download File from URL**

ansible all -i inventory.ini -m get\_url -a "url=https://github.com/vmg/redcarpet/archive/refs/heads/master.zip dest=/tmp/redcarpet.zip" --become

**Bonus: Use Tags to Target Groups**

If your inventory has groups like [web], [db], etc.:

ansible web -i inventory.ini -m shell -a "ps aux | grep nginx"

**Dynamic Inventory File, Playbooks, Projects**

**Tagging Script:**

#!/bin/bash

# Fetch instance IDs that match Environment=dev and Role=web

instance\_ids=$(aws ec2 describe-instances \

--filters "Name=tag:Environment,Values=dev" "Name=tag:Role,Values=web" "Name=instance-state-name,Values=running" \

--query 'Reservations[\*].Instances[\*].InstanceId' \

--output text)

# Sort instance IDs deterministically

sorted\_ids=($(echo "$instance\_ids" | tr '\t' '\n' | sort))

# Rename instances sequentially

counter=1

for id in "${sorted\_ids[@]}"; do

name="web-$(printf "%02d" $counter)"

echo "Tagging $id as $name"

aws ec2 create-tags --resources "$id" \

--tags Key=Name,Value="$name"

((counter++))

done

**ansible.cfg**

[defaults]

inventory = ./inventory/aws\_ec2.yaml

host\_key\_checking = False

[ssh\_connection]

ssh\_args = -o StrictHostKeyChecking=no -o UserKnownHostsFile=/dev/null

**Dynamic Inventory**

inventory/aws\_ec2.yaml

plugin: amazon.aws.aws\_ec2

regions:

- ap-south-1

filters:

tag:Environment: dev

tag:Role: web

instance-state-name: running

compose:

ansible\_host: public\_ip\_address

keyed\_groups:

- key: tags.Name

prefix: name

- key: tags.Environment

prefix: env

# Step 1: Install venv module if not already present

sudo apt install python3-venv -y

# Step 2: Create a virtual environment

python3 -m venv ansible-env

# Step 3: Activate it

source ansible-env/bin/activate

# Step 4: Install required Python packages

pip install boto3 botocore docker

ansible-galaxy collection install amazon.aws

ansible-inventory -i inventory/aws\_ec2.yaml --graph

**Copy Pub Key**

sudo apt update

sudo apt install jq -y

#!/bin/bash

# Define vars

PEM\_FILE="DevOps-Shack.pem"

PUB\_KEY=$(cat ~/.ssh/id\_rsa.pub)

USER="ubuntu" # or ec2-user

INVENTORY\_FILE="inventory/aws\_ec2.yaml"

# Extract hostnames/IPs from dynamic inventory

HOSTS=$(ansible-inventory -i $INVENTORY\_FILE --list | jq -r '.\_meta.hostvars | keys[]')

for HOST in $HOSTS; do

echo "Injecting key into $HOST"

ssh -o StrictHostKeyChecking=no -i $PEM\_FILE $USER@$HOST "

mkdir -p ~/.ssh && \

echo \"$PUB\_KEY\" >> ~/.ssh/authorized\_keys && \

chmod 700 ~/.ssh && \

chmod 600 ~/.ssh/authorized\_keys

"

done

**Run a playbook**

ansible-playbook -i inventory/aws\_ec2.yaml install.yaml -l env\_dev

**✅ FORMAT REMINDER: Basic Structure of an Ansible Playbook**

- name: <Name of the Play>

hosts: <Target hosts>

become: <yes/no>

tasks:

- name: <Task description>

<module>:

<module-options>

**🔰 1. Install NGINX on Ubuntu**

- name: Install NGINX Web Server

hosts: env\_dev

become: yes

tasks:

- name: Install nginx

apt:

name: nginx

state: present

update\_cache: yes

- name: Start nginx service

service:

name: nginx

state: started

enabled: yes

**🧑‍💻 2. Create a User and Add to sudo Group**

- name: Create admin user

hosts: all

become: yes

tasks:

- name: Add user 'devops'

user:

name: devops

state: present

groups: sudo

shell: /bin/bash

**3. Copy File to Remote Hosts**

- name: Deploy Configuration File

hosts: app

become: yes

tasks:

- name: Copy app config

copy:

src: ./1.txt

dest: /home/ubuntu/1.txt

owner: root

group: root

mode: '0644'

**4. Set File Permissions**

- name: Fix permissions on script

hosts: all

become: yes

tasks:

- name: Make script executable

file:

path: /usr/local/bin/deploy.sh

mode: '0755'

**5. Install Multiple Packages**

- name: Install common tools

hosts: all

become: yes

tasks:

- name: Install packages

apt:

name:

- curl

- unzip

- htop

state: present

update\_cache: yes

**Variables**

**✅ 1. Inline Variables in Playbook**

- name: Install a configurable package

hosts: all

become: yes

vars:

package\_name: nginx

tasks:

- name: Install "{{ package\_name }}"

apt:

name: "{{ package\_name }}"

state: present

**✅ 2. External Variable File**

**🔹 vars/web\_vars.yml**

package\_name: apache2

**🔹 Playbook**

- name: Use external vars

hosts: web

become: yes

vars\_files:

- vars/web\_vars.yml

tasks:

- name: Install "{{ package\_name }}"

apt:

name: "{{ package\_name }}"

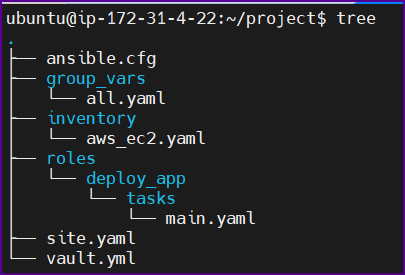
state: present

**✅ 3. Command Line Variables (extra-vars)**

ansible-playbook -i inventory.ini webserver.yml -e "package\_name=httpd"

🔹 This **overrides any other definition** of package\_name.

**Ansible Project**



ansible-vault create vault.yaml

**ansible.cfg**

[defaults]

inventory = ./inventory/aws\_ec2.yaml

host\_key\_checking = False

[ssh\_connection]

ssh\_args = -o StrictHostKeyChecking=no -o UserKnownHostsFile=/dev/null

**group\_vars/all.yaml**

git\_repo\_url: "https://github.com/jaiswaladi246/Boardgame.git"

repo\_dir: "/home/ubuntu/Boardgame"

docker\_image\_name: "adijaiswal/board"

docker\_tag: "latest"

**inventory/aws\_ec2.yaml**

plugin: amazon.aws.aws\_ec2

regions:

- ap-south-1

filters:

tag:Environment: dev

tag:Role: web

instance-state-name: running

compose:

ansible\_host: public\_ip\_address

keyed\_groups:

- key: tags.Name

prefix: name

- key: tags.Environment

prefix: env

**site.yaml**

- name: Complete Maven-Docker Deployment

hosts: env\_dev

become: yes

vars\_files:

- group\_vars/all.yaml

- vault.yaml

roles:

- deploy\_app

**ansible-vault create vault.yml**

docker\_username: your\_dockerhub\_username

docker\_password: your\_dockerhub\_password

**roles/deploy\_app/tasks/main.yaml**

---

- name: Install OpenJDK 17 and basic tools

apt:

name:

- openjdk-17-jdk

- git

- curl

state: present

update\_cache: yes

- name: Install Maven 3.8.7 (for JDK 17)

block:

- name: Download & extract Maven 3.8.7

unarchive:

src: https://dlcdn.apache.org/maven/maven-3/3.8.7/binaries/apache-maven-3.8.7-bin.tar.gz

dest: /opt/

remote\_src: yes

- name: Set Maven environment

copy:

dest: /etc/profile.d/maven.sh

content: |

export JAVA\_HOME=/usr/lib/jvm/java-17-openjdk-amd64

export M2\_HOME=/opt/apache-maven-3.8.7

export PATH=$JAVA\_HOME/bin:$M2\_HOME/bin:$PATH

mode: "0755"

- name: Add ubuntu to docker group

user:

name: ubuntu

groups: docker

append: yes

tags: docker

- name: Ensure Docker service is running

service:

name: docker

state: started

enabled: true

tags: docker

- name: Clone or update Git repository

git:

repo: "{{ git\_repo\_url }}"

dest: "{{ repo\_dir }}"

version: main

update: yes

become: false

tags: git

- name: Build Maven project

shell: mvn clean package

args:

chdir: "{{ repo\_dir }}"

become: false

register: maven\_result

failed\_when: maven\_result.rc != 0

tags: maven

- name: Print Maven stdout if failed

debug:

var: maven\_result.stdout\_lines

when: maven\_result.rc != 0

tags: maven

- name: Print Maven stderr if failed

debug:

var: maven\_result.stderr\_lines

when: maven\_result.rc != 0

tags: maven

- name: Build Docker image

community.docker.docker\_image:

name: "{{ docker\_image\_name }}"

tag: "{{ docker\_tag }}"

source: build

build:

path: "{{ repo\_dir }}"

tags: docker

- name: Log in to Docker Hub

community.docker.docker\_login:

username: "{{ docker\_username }}"

password: "{{ docker\_password }}"

tags: docker

- name: Push Docker image

community.docker.docker\_image:

name: "{{ docker\_image\_name }}"

tag: "{{ docker\_tag }}"

push: true

source: local

tags: docker

ansible-playbook site.yaml -u ubuntu --ask-vault-pass