We adapt. You succeed.

ansible@galaxy.suse.com

Nürnberg, 13.04.2021

What is Ansible?

Ansible is a tool used for configuration management, orchestration and application deployment.

Installation

- To install ansible:
  sudo zypper install ansible

(one must have opensuse repo-oss repository activated for that)

- To prepare the (client) machines to be used with ansible:
  Export the ssh-key from your runner machine
  ssh-copy-id <remote_user:>@ip or hostname (for example ssh-copy-id root@192.168.112.208)

- To prepare the (client) machines to be used with ‘zypper’ module:
  sudo zypper install python-xml

ansible cli examples:

Syntax: ansible <target*> <module> <args/options>
*target - could be a single or a group of machines

```
ansible 192.168.122.208 -m ping -i inventory -u root
ansible all -raw “ls -alh” -i inventory -u root -k
ansible workers -raw “zypper lr -puU” -i inventory -u root

#useful to make mistakes
ansible all -m ping -a “blah” -i inventory -u root

#run a script on remote machine(s) [given that you have copied the script on all remote machines]
ansible 192.168.122.208 -m script -a “script.sh” -u root -i inventory
```

Config files

/etc/ansible/hosts
/etc/ansible/ansible.cfg

inventory (file)
playbook.yaml
module.py

ansible playbooks

Syntax-related key takeaways:

- hosts: [ip, group tag] **=** remote host(s) on which the orchestration will be executed

gather_facts: True/False **=** gathers remote host(s) details; full facts json is huge and contains lots of info regarding the remote machine;

remote_user: [user] = user@<remote.machine> in the ssh command

become: True/False = if the playbook will be ran as another user

become_user: [postgres] = if become: True, then that is equal to sudo -u postgres <command>

tasks:

- name: hello world task

ansible playbooks

Syntax-related key takeaways:

tasks:	
- name: hello world task	= the name of your task
shell/raw/command: zypper ref	= raw/shell code followed by the string
when: ansible_facts[‘os_family’] == “Debian”	= execute the command only on machines that return in the facts json file os_family: “Suse”/“Debian”/“Centos”
register: zypper_ref_output	- we just created a variable related to this command we run
- debug:	
var: zypper_ref_output.stdout_lines	- this way we check the stdout (on each remote machine) of the cmd/code we ran

Zypper module for ansible

- installation:
  (* on runner machine: ansible-galaxy collection install community.general)

  * on target (client) machines: zypper install python-xml

tasks:	
- zypper:	= the name of the module we're using
name: nmap	= the name of the package/pattern/patch
state: present/latest/absent/dist-upgrade	= the state of the package
disable_recommends: no	
type: package/patch/pattern/product	
allow_vendor_change: True/False	

Additional places to check:

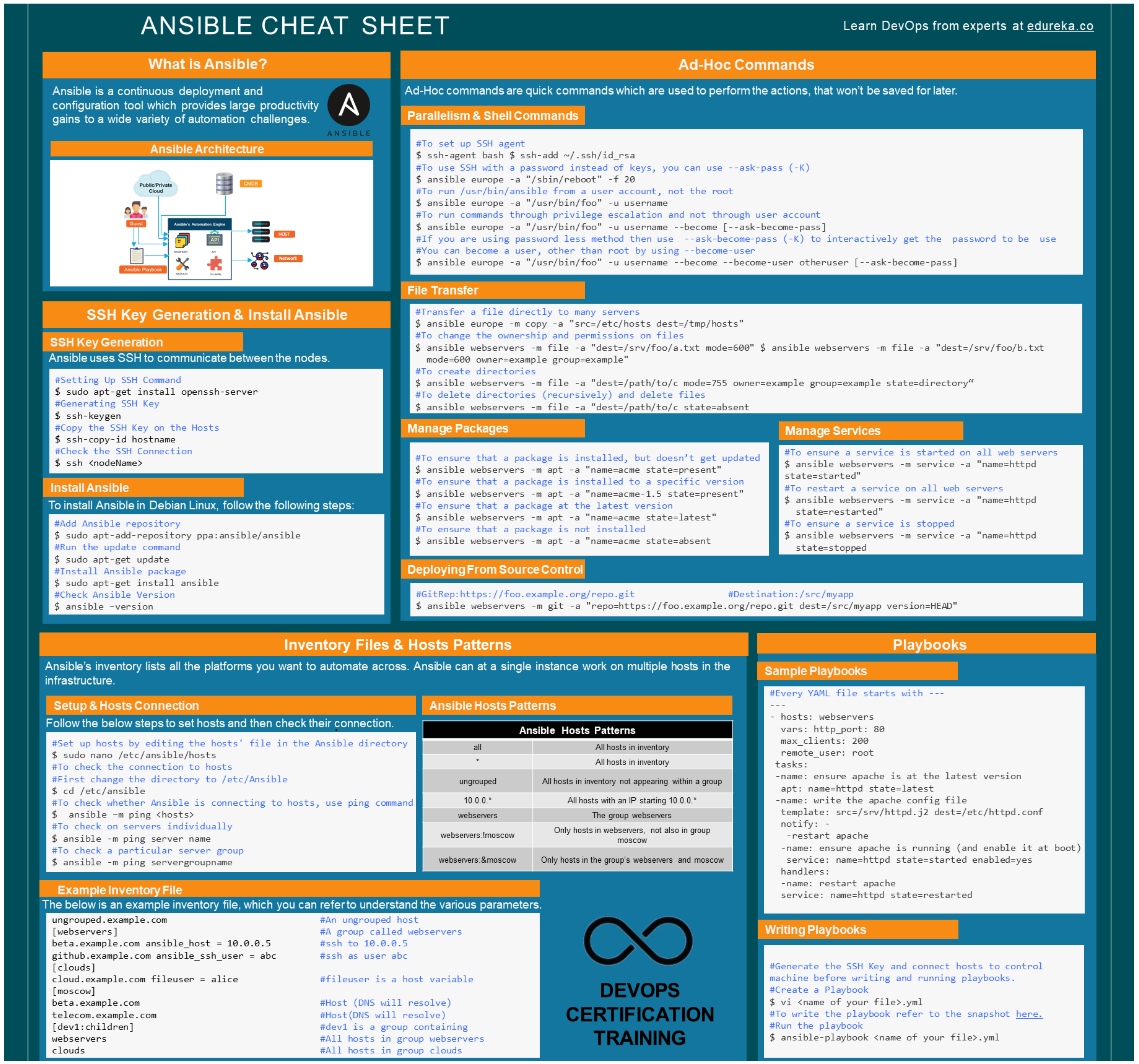
https://github.com/ansible/ansible

https://docs.ansible.com/ansible/latest/user_guide/intro_getting_started.html

https://www.tutorialspoint.com/ansible/index.htm

(what tutorial I've learned)
https://www.udemy.com/course/learning-path-automation-with-ansible-puppet-and-salt

Ansible cheat sheet



ANSIBLE CHEAT SHEET

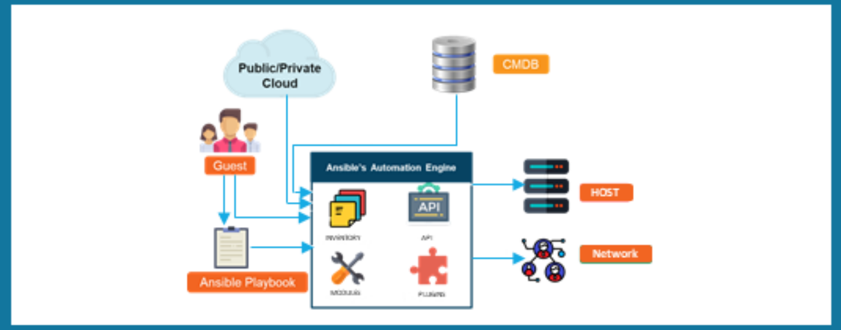
Learn DevOps from experts at edureka.co

What is Ansible?

Ansible is a continuous deployment and configuration tool which provides large productivity gains to a wide variety of automation challenges.

ANSIBLE

Ansible Architecture



SSH Key Generation & Install Ansible

SSH Key Generation

Ansible uses SSH to communicate between the nodes.

```
#Setting Up SSH Command
$ sudo apt-get install openssh-server
#Generating SSH Key
$ ssh-keygen
#Copy the SSH Key on the Hosts
$ ssh-copy-id hostname
#Check the SSH Connection
$ ssh <nodeName>
```

Install Ansible

To install Ansible in Debian Linux, follow the following steps:

```
#Add Ansible repository
$ sudo apt-add-repository ppa:ansible/ansible
#Run the update command
$ sudo apt-get update
#Install Ansible package
$ sudo apt-get install ansible
#Check Ansible Version
$ ansible –version
```

Ad-Hoc Commands

Ad-Hoc commands are quick commands which are used to perform the actions, that won't be saved for later.

Parallelism & Shell Commands

```
#To set up SSH agent
$ ssh-agent bash $ ssh-add ~/.ssh/id_rsa
#To use SSH with a password instead of keys, you can use --ask-pass (-K)
$ ansible europe -a "/sbin/reboot" -f 20
#To run /usr/bin/ansible from a user account, not the root
$ ansible europe -a "/usr/bin/foo" -u username
#To run commands through privilege escalation and not through user account
$ ansible europe -a "/usr/bin/foo" -u username --become [--ask-become-pass]
#If you are using password less method then use  --ask-become-pass (-K) to interactively get the  password to be  use
#You can become a user, other than root by using --become-user
$ ansible europe -a "/usr/bin/foo" -u username --become --become-user otheruser [--ask-become-pass]
```

File Transfer

```
#Transfer a file directly to many servers
$ ansible europe -m copy -a "src=/etc/hosts dest=/tmp/hosts"
#To change the ownership and permissions on files
$ ansible webservers -m file -a "dest=/srv/foo/a.txt mode=600" $ ansible webservers -m file -a "dest=/srv/foo/b.txt
  mode=600 owner=example group=example"
#To create directories
$ ansible webservers -m file -a "dest=/path/to/c mode=755 owner=example group=example state=directory“
#To delete directories (recursively) and delete files
$ ansible webservers -m file -a "dest=/path/to/c state=absent
```

Manage Packages

```
#To ensure that a package is installed, but doesn't get updated
$ ansible webservers -m apt -a "name=acme state=present"
#To ensure that a package is installed to a specific version
$ ansible webservers -m apt -a "name=acme-1.5 state=present"
#To ensure that a package at the latest version
$ ansible webservers -m apt -a "name=acme state=latest"
#To ensure that a package is not installed
$ ansible webservers -m apt -a "name=acme state=absent
```

Manage Services

```
#To ensure a service is started on all web servers
$ ansible webservers -m service -a "name=httpd
state=started"
#To restart a service on all web servers
$ ansible webservers -m service -a "name=httpd
  state=restarted"
#To ensure a service is stopped
$ ansible webservers -m service -a "name=httpd
  state=stopped
```

Deploying From Source Control

```
#GitRep:https://foo.example.org/repo.git                   #Destination:/src/myapp
$ ansible webservers -m git -a "repo=https://foo.example.org/repo.git dest=/src/myapp version=HEAD"
```

Inventory Files & Hosts Patterns

Ansible's inventory lists all the platforms you want to automate across. Ansible can at a single instance work on multiple hosts in the infrastructure.

Setup & Hosts Connection

Follow the below steps to set hosts and then check their connection.

```
#Set up hosts by editing the hosts' file in the Ansible directory
$ sudo nano /etc/ansible/hosts
#To check the connection to hosts
#First change the directory to /etc/Ansible
$ cd /etc/ansible
#To check whether Ansible is connecting to hosts, use ping command
$  ansible –m ping <hosts>
#To check on servers individually
$ ansible -m ping server name
#To check a particular server group
$ ansible -m ping servergroupname
```

Ansible Hosts Patterns

Ansible Hosts Patterns	
all	All hosts in inventory
*	All hosts in inventory
ungrouped	All hosts in inventory not appearing within a group
10.0.0.*	All hosts with an IP starting 10.0.0.*
webservers	The group webservers
webservers:!moscow	Only hosts in webservers, not also in group moscow
webservers:&moscow	Only hosts in the group's webservers and moscow

Example Inventory File

The below is an example inventory file, which you can refer to understand the various parameters.

```
ungrouped.example.com                          #An ungrouped host
[webservers]                                   #A group called webservers
beta.example.com ansible_host = 10.0.0.5       #ssh to 10.0.0.5
github.example.com ansible_ssh_user = abc      #ssh as user abc
[clouds]
cloud.example.com fileuser = alice             #fileuser is a host variable
[moscow]
beta.example.com                               #Host (DNS will resolve)
telecom.example.com                            #Host(DNS will resolve)
[dev1:children]                                #dev1 is a group containing
webservers                                     #All hosts in group webservers
clouds                                         #All hosts in group clouds
```

DEVOPS CERTIFICATION TRAINING

Playbooks

Sample Playbooks

```
#Every YAML file starts with ---
---
- hosts: webservers
  vars: http_port: 80
  max_clients: 200
  remote_user: root
 tasks:
 -name: ensure apache is at the latest version
  apt: name=httpd state=latest
 -name: write the apache config file
  template: src=/srv/httpd.j2 dest=/etc/httpd.conf
  notify: -
   -restart apache
  -name: ensure apache is running (and enable it at boot)
   service: name=httpd state=started enabled=yes
  handlers:
  -name: restart apache
  service: name=httpd state=restarted
```

Writing Playbooks

```
#Generate the SSH Key and connect hosts to control
machine before writing and running playbooks.
#Create a Playbook
$ vi <name of your file>.yml
#To write the playbook refer to the snapshot here.
#Run the playbook
$ ansible-playbook <name of your file>.yml
```