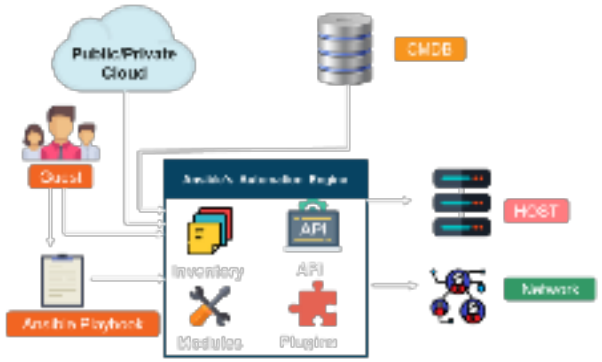
Ansible Cheat Sheet

What is Ansible?

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

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"
```

Ansible Cheat Sheet

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
```

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
```

Example Inventory File

```
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
```

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