SSH Setup

Copy your Ansible Master's public key to the managed node

```
ssh-keygen  ## generate public key
ssh-copy-id <name of node>  # copy key, provide password to node
```

configure Hosts file

```
/etc/ansible/hosts
[production]
prod1.prod.local
prod2.prod.local

[dev]
devweb1.dev.local
devweb2.dev.local
```

REMOTE CMD (Ad Hoc)

Ping specific node

```
ansible -i hosts nycweb01.prod.local -m ping
```

Ping with wildcard

```
ansible -i hosts "nycweb*" -m ping
```

Ping all nodes with SSH user 'root'

```
ansible -i hosts all -m ping -u root
```

run a command

```
ansible -i hosts dev -a 'uname -a'
```

check Yum packages

```
ansible -i hosts dev -m yum
```

check if Docker rpm is installed

```
ansible -i hosts web01.nyc.local -m shell -a "rpm -qa
| grep docker"
```

Get facts about a box

```
ansible -i hosts web01.nyc.local -m setup -a
'filter=facter_*'
```

run command with sudo

```
ansible -i hosts target-host -m shell -a "cat
/etc/sudoers" --sudo
```

limit command to a certain group or server: add `--limit *.nyc`

SERVER DIAGNOSTICS

Test Connection

```
ansible -i hosts all -m ping -u root
```

Diagnostics

manage nodes via "/etc/ansible/hosts" file

Debug (debug output for playbook)

```
- debug: var=result verbosity=2
```

GALAXY

install Role (Module)

```
ansible-galaxy install geerlingguy.nginx
```

PLAYBOOKS

run playbook with sudo

```
ansible-playbook -v config-users.yaml --sudo
--sudo-user=joe --ask-sudo-pass
```

use different Hosts file

```
ansible-playbook -v -i /path/to/hosts
```

run playbook but only a specific task (tag)

```
ansible-playbook
playbooks/restore_bitbucket.yaml -i hosts --
tags rsync
```

```
or to skip: (--skip-tags tag1, tag2)
```

store output of a command as a variable

```
shell: cat /etc/network | grep eth0
register: address
debug: msg="address is {{ address.stdout }}"
```

configure multiple items with one task

```
- name: more complex items to add several
users
  user:
    name: "{{ item.name }}"
    uid: "{{ item.uid }}"
    groups: "{{ item.groups }}"
    state: present
  with_items:
     - { name: testuser1, uid: 1002,
groups: "wheel, staff" }
     - { name: testuser2, uid: 1003,
groups: staff }
```

get path location of current Playbook (pwd)

```
{{ playbook_dir }}
```

```
Set playbook to be verbose by default
- hosts: blah
  strategy: debug
```

run playbook with verbose traceback

```
ansible-playbook -i hosts myPlaybook.yaml -
vvv
```

run playbook on multiple Host groups

```
- hosts: "search_head, deployer"
```

Run playbook locally on host

```
hosts: 127.0.0.1
connection: local
```

Prompt for password during Playbook run

```
# Playbook to change user password
```

PACKAGES AND INSTALLATION

install multiple packages

```
yum: name="{{ item }}" state=present
with_items:
  - http
  - htop
  - myapp
```

JOBS AND PROCESS CONTROL

run Ansible ad hoc with 10 parallel forks

```
ansible -i hosts testnode1 -a "uname -a" -f 10
```

show human readable output
add this line to ansible.cfg

```
stdout_callback=yaml
```

CONDITIONALS

y file to n

VARIABLES

include global variables for all Roles

sample playbook

```
 splunk/
    setup_splunk_playbook.yaml
    roles/base
            /tasks/main.yaml
            /tasks/install.yaml
          search_head
            /tasks/configure.yaml
          indexer
            /tasks/configure.yaml
          some_other_role
            /tasks/some_task.yaml
    hosts
    config.yaml
```

Place your vars into config.yaml

cat splunk/config.yaml

```
 ---
 # global Splunk variables
 splunk_version: 7.0.0
```

in your playbook, include the Roles

cat setup_splunk_playbook.yaml

```
 - hosts: "search_heads"
   become_user: root
   become: true
   gather_facts: true
```

```
- name: pw change
  hosts: target
  become: true
  become_user: root
  vars_prompt:
    - name: username
      prompt: "enter username for which to
change the pw"
    - name: password
      prompt: "enter new password"
      private: yes

  tasks:
    - name: change pw
      user: "name={{ username }} password=
{{ password }} update_password=always"
```

run playbook with "dry run" / NOOP / simulate

```
ansible-playbook foo.yml --check
```

Run task on different target,

```
- name: run something on some other server
  debug: msg="running stuff"
  delegate_to: someserver
```

Delegate task to a host group

```
- name: restart web servers
  service: name=memcached state=restarted
  delegate_to: "{{ item }}"
  with_items: "{{ groups['webservers'] }}"
```

Get IP or facter of a remote host

```
- name: get IP
  debug: msg="{{ hostvars['nycweb01']
['ansible_default_ipv4']['address'] }}"
```

or

```
debug: msg="{{ hostvars[item]
['ansible_ssh_host'] }}"
with_items: "{{ groups['webservers'] }}"
```

synchronize file (copy file from Ansible host to target)

```
 - synchronize:
     src: "{{ playbook_dir
}}/files/vscode.repo"
     dest: /etc/yum.repos.d/
```

synchronize from server A to server B with a wildcard

```
    - name: copy Splunk Apps
      synchronize:
        src: "/opt/splunk/etc/apps/{{ item
}}" (server A)
        dest:
"/opt/splunk/etc/shcluster/apps/"  (server
B)
      with_items:
        - item1
        - item2
      delegate_to: server A
```

wget a file to a location

```
  - get_url:
      url:
'https://dl.google.com/go/go1.10.linux-
amd64.tar.gz'
```

```
  roles:
    - base
    - search_head
```

in your Role, include the Global Vars inside a Task

cat roles/base/tasks/main.yaml

```
---
# install Splunk Base

- name: include vars
  include_vars: "{{ playbook_dir }}/config.yaml"

- include: install.yaml
```

vars are accessible in tasks now,

cat roles/base/tasks/install.yaml

```
- name: echo version
  debug: splunk version is {{ splunk_version }}
```

Loop through a Dict variable inside a playbook

cluster:
 members:
  splunk01: 10.123.1.0
  splunk02: 10.123.1.1
  splunk03: 10.123.1.2

in the playbook,

```
- debug: msg="{{ cluster.members.values() |
map('regex_replace', '(.*)', 'https://\\1:8089') |
join(',') }}"

>> https://10.123,1.0:8089, https://10.123.1.1:8089,
etc etc
```

Use Inventory file variables inside a playbook

cat hosts

```
[apache]
nycweb01
```

playbook

```
debug: msg="IP: {{ hostvars[groups['apache'][0]]
['ansible_default_ipv4']['address'] }}"
debug: msg="Hostname: {{ hostvars[groups['apache']
[0]]['inventory_hostname'] }}"
```

register a List/Array to be used for later,

```
- name: parse all hostnames in group WebServer  and
get their IPs, place them in a list
  command: echo {{ hostvars[item]
['ansible_ssh_host'] }}"
  with_items: "{{ groups['webserver'] }}"
  register: ip_list

- name: show the IPs
  debug: msg={{ ip_list.results |
map(attribute='item') | list }}"
```

```
      dest: '/tmp'
      force: no  # dont download if file
already exists

untar tar.gz
```

USER AND GROUP MGMT

change user password for user Joe (user Fred running the cmd as sudo on the target box)

1 install passlib

```
pip install passlib
```

#2 update the pw, using a hash

```
ansible targethost -s -m user -a "name=joe
update_password=always password={{
'MyNewPassword' | password_hash('sha512')
}}" -u fred --ask-sudo-pass
```

copy public ssh key to remote authorized_keys file

```
- hosts: targetHost
  tasks:
    - name: update nessus SSH keys
      become_user: root
      become_method: sudo
      become: true
      authorized_key:
        user: nessus
        key: "{{ lookup('pipe','cat
../files/ssh_keys/nessus.pub') }}"
        state: present
```

FILES & DIRS

delete all files and hidden files in a directory

```
vars:
  app_home: /var/opt/application

tasks:
  - name: clear home dir
  - shell: "ls -la {{ app_home }}/"
    register: files_to_delete
  - file: path="{{ app_home }}/{{ item }}"
state=absent
    with_items: "{{
files_to_delete.stdout_lines }}"
```

get files from node

```
ansible node1 -s -m fetch -a "src=/etc/hosts
dest=/tmp"
```

copy file to node

```
ansible node1 -m copy -a "src=/etc/hosts
dest=/tmp/hosts"
```

remove all files matching a wildcard

```
file: path={{ item }} state=absent
with_fileglob: /tmp/*.rpm
```

export an Environment variable

```
- name: yum install
  yum: name=somepkg state=present
  environment:
    SOME_VAR: abc
```

Variables inside Inventory Hosts file

cat hosts

```
[web]
nycweb01.company.local

[web:vars]
role="super duper web server"
```

now get the "role" variable inside the playbook,

```
- hosts: web
  gather_facts: true
  tasks:
    - name: print Role var
      debug: msg={{ role }}

// super duper web server
```

MODULES

```
service: name=httpd state=[started, stopped,
restarted, reloaded] enabled=[yes,no]
user: name=joe state=[present,absent] uid=1001
groups=wheel shell=/bin/bash
group: name=splunk gid=6600 state=[present,absent]
system=[yes/no]
yum: name=apache state=[present, latest, absent,
removed]
file: path=/etc/file state=[file, link, directory,
hard, touch, absent] group=x owner=x recurse=yes
```

FACTER

get all facts from a node (ad hoc)

```
ansible -i hosts targetName -m setup -a
"filter="facter_*"
```

use fact in a playbook

```
include fact as {{ ansible_factname }}
```

add fact to Hosts file

```
[group]
host1 admin_user=jane
host2 admin_user=jack
host3

[group:vars]
admin_user=john
```

get default IPV4 address

```
ansible_default_ipv4.address
```

Local facts

place **.fact** file into /etc/ansible/facts.d on target node

```
vim /etc/ansible/facts.d/fruits.fact

[fruits]
sweet=banana, apple, grapes
bitter=grapefruit
```

get Local facts

```
ansible -i hosts mrx -m setup -a
"filter=ansible_local"
```