



ANSIBLE

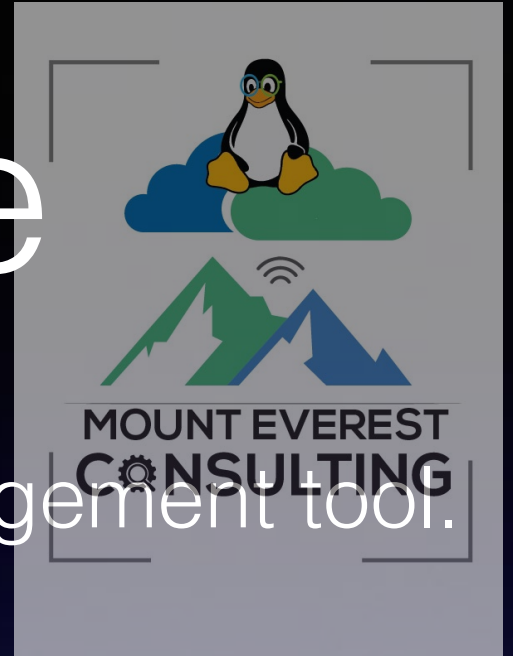


# What is covered

- What is Ansible
- How Ansible works
- Adhoc commands
- Playbooks
- Variable loops and handlers
- Roles



# What is Ansible

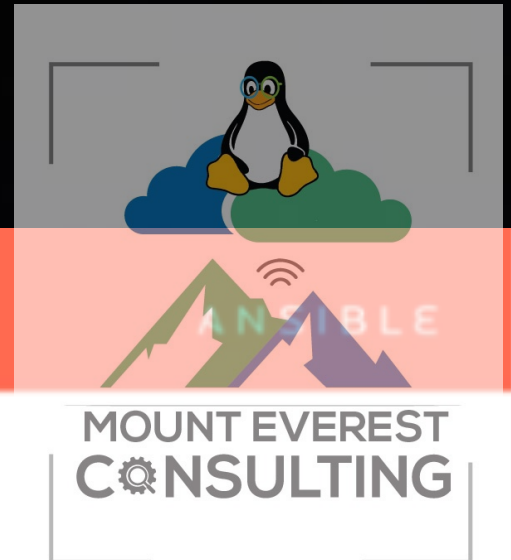
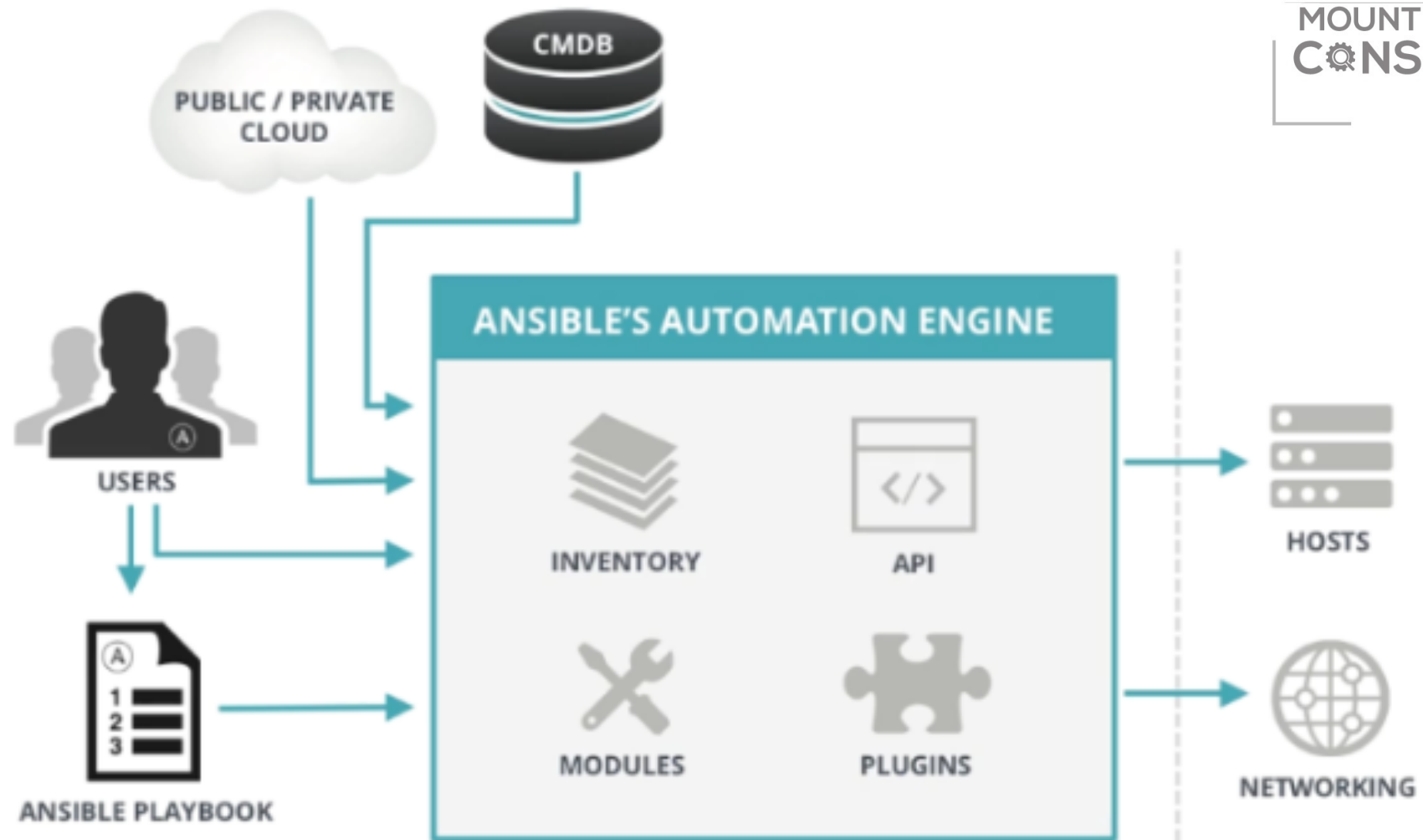


- Ansible is an open source configuration management tool.
- Automates software provisioning, configuration management, and application deployment
- Redhat acquired Ansible in 2015.
- Playbooks are written in simple human readable yaml files. No special coding skills needed.
- Agent-less architecture. Uses open SSH for connection.
- Can also orchestrate resources.

- Tasks executed in order.
- Playbooks can be version controlled.
- Idempotent
- It plays well with others. Lot of other tools supports ansible. ex. AWS SSM, Jenkins, packer etc.
- Modular architecture. You can use standard modules or create your own modules using python.



# ANSIBLE ARCHITECTURE



# Modules

- Modules are libraries that perform actual work.
- They are called using ad-hoc command mode or through playbooks in form of tasks.
- They take certain inputs and return in JSON format.
- Typically they return following info:
  - Changed
  - failed
  - msg
  - skipped
  - ansible\_facts
  - exception

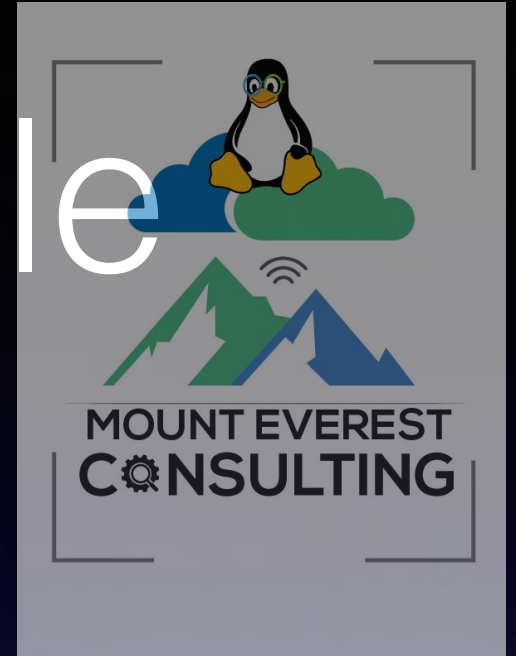


# Who Manages Modules



- Core modules are maintained by the Ansible Engineering team. These modules are integral to the basic foundations of the Ansible distribution
- Network modules are maintained by the Ansible Network Team. Please note there are additional networking modules that are categorized as Certified or Community not maintained by Ansible
- Certified modules are part of a future planned program currently in development
- Community modules are submitted and maintained by the Ansible community. These modules are not maintained by Ansible, and are included as a convenience

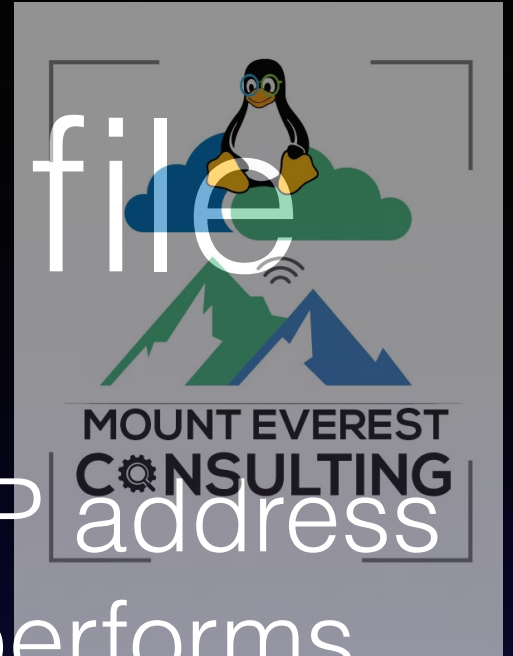
# Modules example



- service
- file
- copy
- cron
- mount
- timezone
- In absence of module you can use run command module. There is no concept of desired state and idempotent.
  - Command
  - Shell
  - Script



# Ansible inventory file



- Generally consists of Server name or IP address of hosts where ansible connects and performs tasks.

- It is in form of hosts and groups

```
mail.example.com
```

```
[webservers]  
foo.example.com  
bar.example.com
```

```
[dbservers]  
one.example.com  
two.example.com  
three.example.com
```

# Ansible

```
ansible web -m command -a "uptime" -o
```

```
ansible web -m yum -a "name=httpd state=present" -b
```

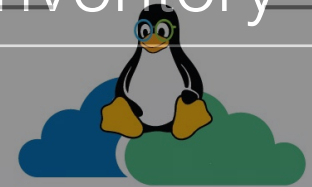
```
ansible web -m service -a "name=httpd state=started" -b
```

```
---  
- name: Install and start apache web server  
  hosts: web
```

```
  tasks:
```

- name: install apache  
 yum:  
 name: httpd  
 state: present
- name: start apache  
 service:  
 name: httpd  
 state: started

## Inventory



MOUNT EVEREST  
CONSULTING

[mail@example.com](mailto:mail@example.com)

[web]

[foo.example.com](http://foo.example.com)

[bar.example.com](http://bar.example.com)

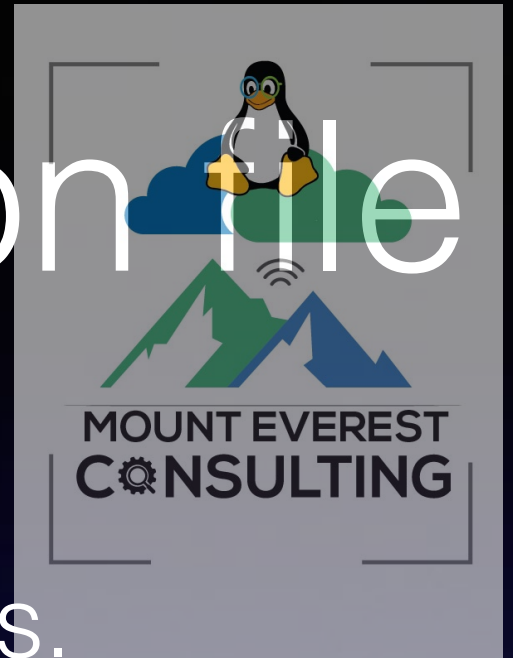
[dbservers]

[one.example.com](http://one.example.com)

[two.example.com](http://two.example.com)

[three.example.com](http://three.example.com)

# Ansible configuration file



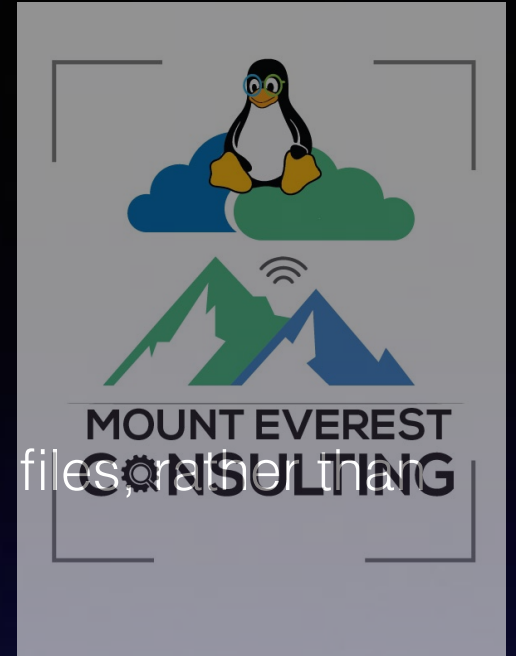
- It has all the adjustable ansible settings.
- ANSIBLE\_CONFIG (an environment variable)
- ansible.cfg (in the current directory)
- .ansible.cfg (in the home directory)
- /etc/ansible/ansible.cfg

# Tools



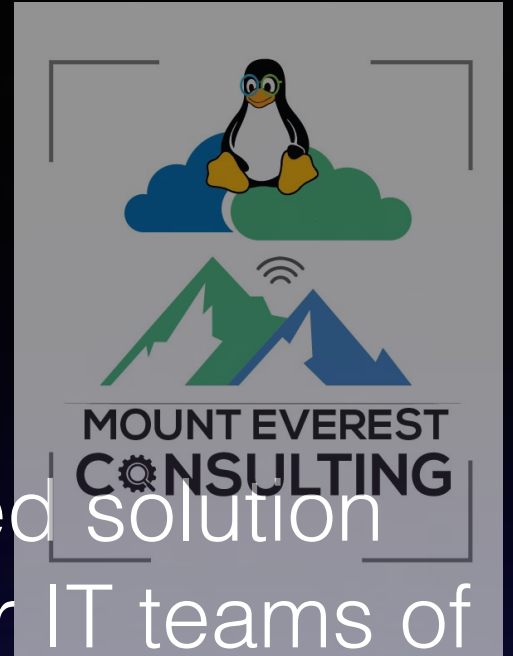
- Ansible: Runs ansible tasks in ad-hoc fashion
- Ansible-playbook: Runs a playbook. Playbooks is collection of tasks executed in sequential manner.
- ansible-valut: an encrypt any structured data file used by Ansible.
- ansible-galaxy: command to manage Ansible roles in shared repositories
- ansible-doc: Displays information on modules installed in Ansible libraries. It displays a terse listing of plugins and their short descriptions, provides a printout of thei

# Ansible Vault



- allows keeping sensitive data such as passwords or keys in encrypted files, rather than as plaintext in your playbooks or roles.
- These vault files can then be distributed or placed in source control.
- Command is `ansible-vault`. It will ask for password and encrypt using password.
- You can:
  - Create encrypt a file
  - Decrypt a file
  - Edit an encrypted file
  - Rekey an encrypted file
  - view and encrypted file

# Ansible Tower



- Ansible Tower (formerly 'AWX') is a web-based solution that makes Ansible even more easy to use for IT teams of all kinds. It's designed to be the hub for all of your automation tasks
- Tower allows you to control access to who can access what, even allowing sharing of SSH credentials without someone being able to transfer those credentials
- It logs all of your jobs, integrates well with LDAP, and has an amazing browsable REST API. Command line tools are available for easy integration with Jenkins as well

# Ad-hoc command

Single task, no desired state, for quick check etc.



```
# check all my inventory hosts are ready to be  
$ ansible all -m ping
```

```
# collect and display the discovered facts  
# for the localhost  
$ ansible localhost -m setup
```

```
# run the uptime command on all hosts in the  
# web group  
$ ansible web -m command -a "uptime"
```



# Lab 1



**# Ping module check if node is responsive.**

```
ansible web -m ping
```

**# Command module:**

```
ansible web -m command -a "uptime" -o
```

**# setup module to get information about the target node**

```
ansible web -m setup
```

**# install apache using yum module. Use -b flag to su to root user.**

```
ansible web -m yum -a "name=httpd state=present" -b
```

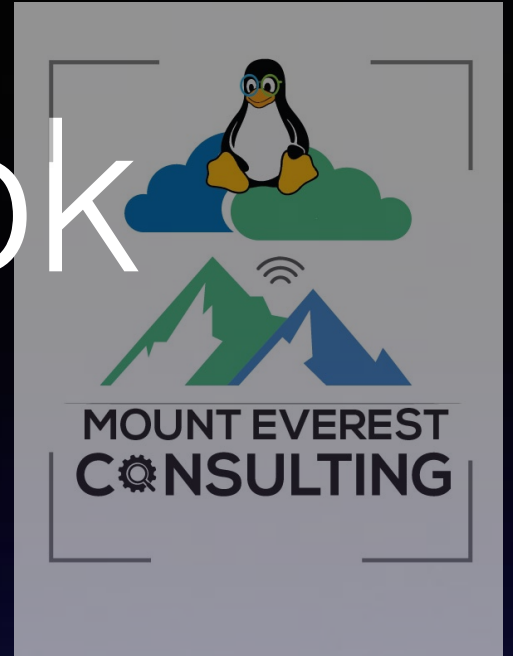
**# Start Apache server using service module**

```
ansible web -m service -a "name=httpd state=started" -b
```

**# Now Stop and remove apache server**



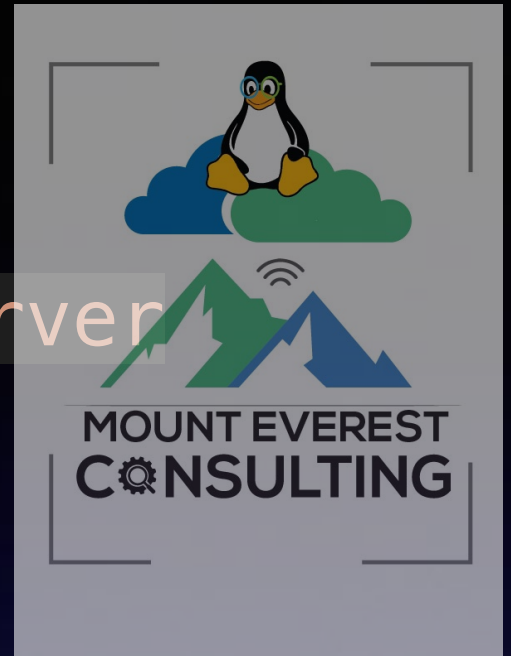
# Ansible playbook



- Written in YAML
- One or more plays
- Plays consists of tasks (call to ansible module)
- One playbook can have multiple plays

```
---  
- name: Install and start apache web server  
  hosts: web  
  become: yes  
  gather_facts: false
```

```
  tasks:  
    - name: install apache  
      yum:  
        name: httpd  
        state: present  
    - name: start apache  
      service:  
        name: httpd  
        state: started
```



```
---  
- name: Install and start apache web server  
  hosts: webservers  
  become: yes  
  gather_facts: false
```

```
tasks:
```

```
- name: ensure apache is at the latest version  
  yum:  
    name: httpd  
    state: latest  
- name: start apache  
  service:  
    name: httpd  
    state: started
```

```
- name: Update and Start Postgres database  
  hosts: databases  
  remote_user: root
```

```
tasks:
```

```
- name: ensure postgresql is at the latest version  
  yum: name=postgresql state=latest  
- name: ensure that postgresql is started  
  service: name=postgresql state=started
```



# Lab 2



- Create file `install_apache.yml`
- `ansible-playbook install_apache.yml`
- Create playbook to remove apache

# Variables



Variables can be defined in vars section

```
vars:  
  httpd_packages: httpd
```

Can be single variable or an array or list.

```
vars:  
  httpd_packages:  
    - httpd  
    - vim  
    - wget
```

Can also be in a separate file and ref in vars\_files section

```
vars_files:  
  - external_vars.yml
```

# Lab 3a Variables



```
---
- name: Install and start apache web server
  hosts: web
  become: yes
  gather_facts: false
  vars:
    httpd_packages: httpd

tasks:
  - name: install apache
    yum:
      name: "{{ httpd_packages }}"
      state: present

  - name: start apache
    service:
      name: httpd
      state: started
```

# Loops

```
---
- name: Install and start apache web server
  hosts: web
  become: yes
  gather_facts: false
  vars:
    httpd_packages:
      - httpd
      - vim
      - wget

  tasks:
    - name: install apache
      yum:
        name: "{{ item }}"
        state: present
      with_items: "{{ httpd_packages }}"
```





# Lab 3b



# Conditionals



```
---  
- name: Install and start apache web server  
  hosts: web  
  become: yes  
  gather_facts: false  
  vars:  
    should_update_index: "yes"
```

```
tasks:  
  - name: install apache  
    yum:  
      name: httpd  
      state: present  
  - name: start apache  
    service:  
      name: httpd  
      state: started
```

```
- name: Populate index file  
  shell: echo "Hello world `date`" >> /var/www/html/index.html  
  changed_when: false  
  when: should_update_index == "yes"
```



# Lab 3c

# Handlers



```
---
- name: Install and start apache web server
  hosts: web
  become: yes
  gather_facts: false
  vars:
    httpd_packages: httpd
```

```
  tasks:
    - name: install apache
      yum:
        name: httpd
        state: present
      notify: restart apache service
```

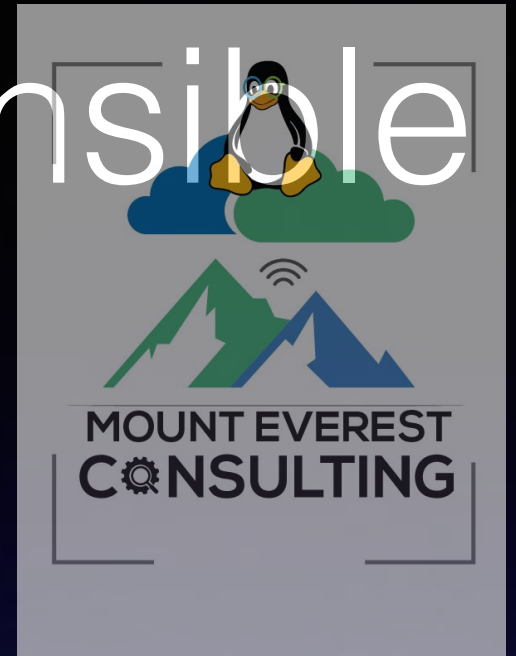
```
    - name: Populate index file
      shell: echo "Hello world `date`" > /var/www/html/index.html
      notify: restart apache service
      changed_when: false
```

```
  handlers:
    - name: restart apache service
      service:
        name: httpd
        state: restarted
```



# Lab 3d

# EC2 instance using Ansible



```
---
- name: Create AWS EC2 instance
  hosts: localhost
  connection: local
  gather_facts: False
  vars:
    instance_type: t2.micro
    image: ami-1853ac65
    keypair: iaccpk.pem
    #my_iam_aws_access_key: XXXXXXXXXXXXXXXXXXXX
    #my_iam_aws_secret_key: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
    region: us-east-1
  tasks:
    - name: Create an EC2 instance
      ec2:
        #aws_access_key: "{{ my_iam_aws_access_key }}"
        #aws_secret_key: "{{ my_iam_aws_secret_key }}"
        key_name: "{{ keypair }}"
        instance_type: "{{ instance_type }}"
        image: "{{ image }}"
        region: "{{ region }}"
        wait: yes
        count: 1
        instance_tags:
          Name: iacc-test1
      register: ec2_info
    - name: Display EC2 properties
      debug: var=ec2_info.instances
```



# Lab 5a/b