BATCH : BATCH 85

LESSON : Ansible

DATE : 24.10.2022

SUBJECT : Ansible Introduction

techproeducation

techproeducation

techproeducation

techproeducation

techproedu

What is Ansible?

- Ansible is a tool written in Python, and it uses the declarative markup language YAML to describe the desired state of devices and configuration.

What is Ansible?

- Real Life Use Cases

  Provisioning, Configuration Management, Application Deployment, Continuous Deployment, Automation, and Orchestration
- Ansible with Docker
- Ansible vs Alternative Tools (Puppet, Chef)
- Ansible Components
  - Modules & Playbook

What is Ansible?

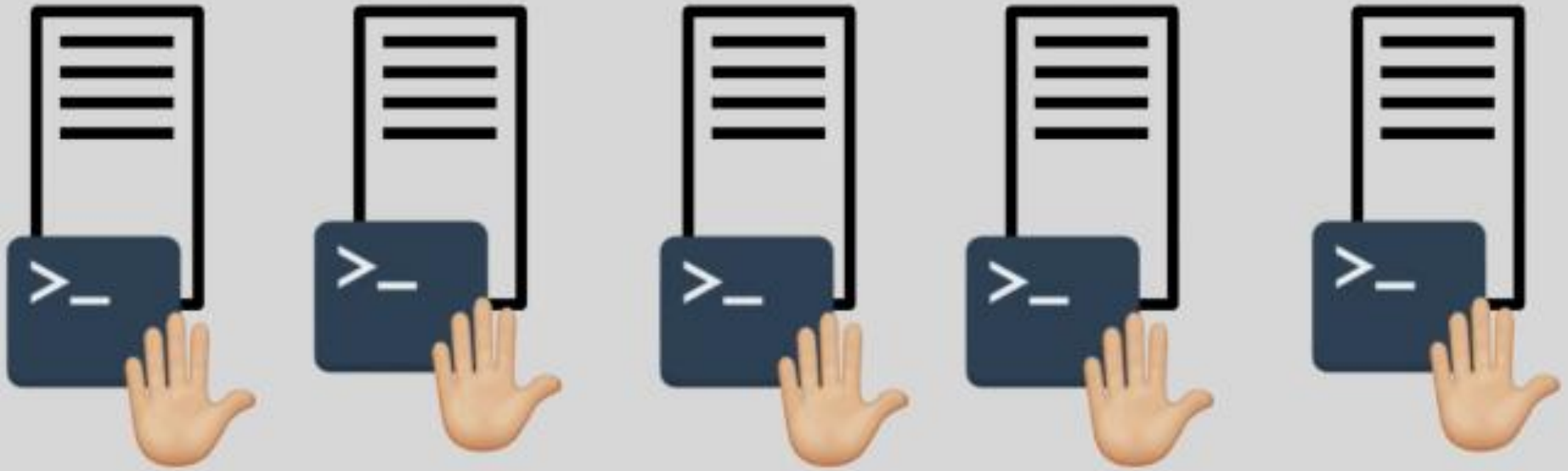
- Tool to automate IT tasks.
- Agentless
- Easy to use
- No need for learning a specific language for Ansible
- Ansible uses YAML

Why use Ansible?

Automation

Update root passwords of 200 machines every 3 months

Manual task takes approximately 45 seconds for each.

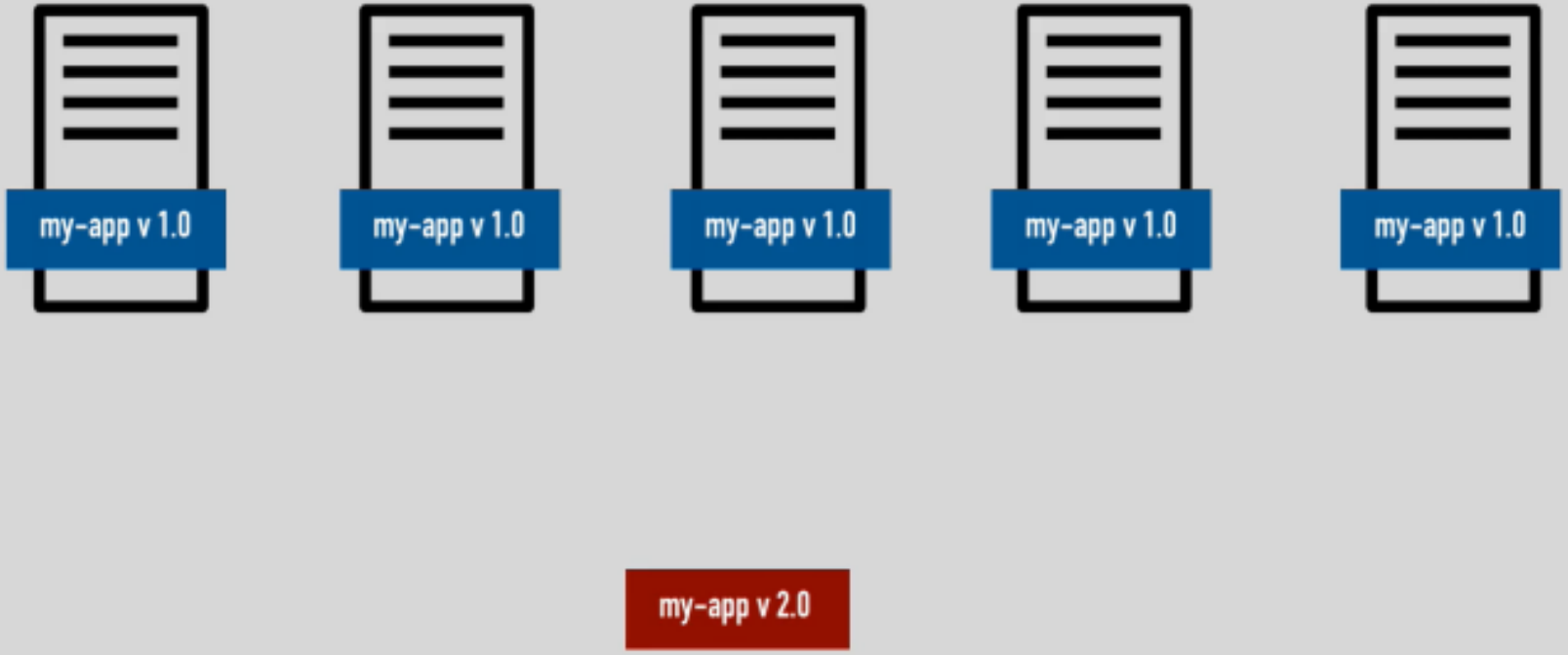


Why use Ansible?

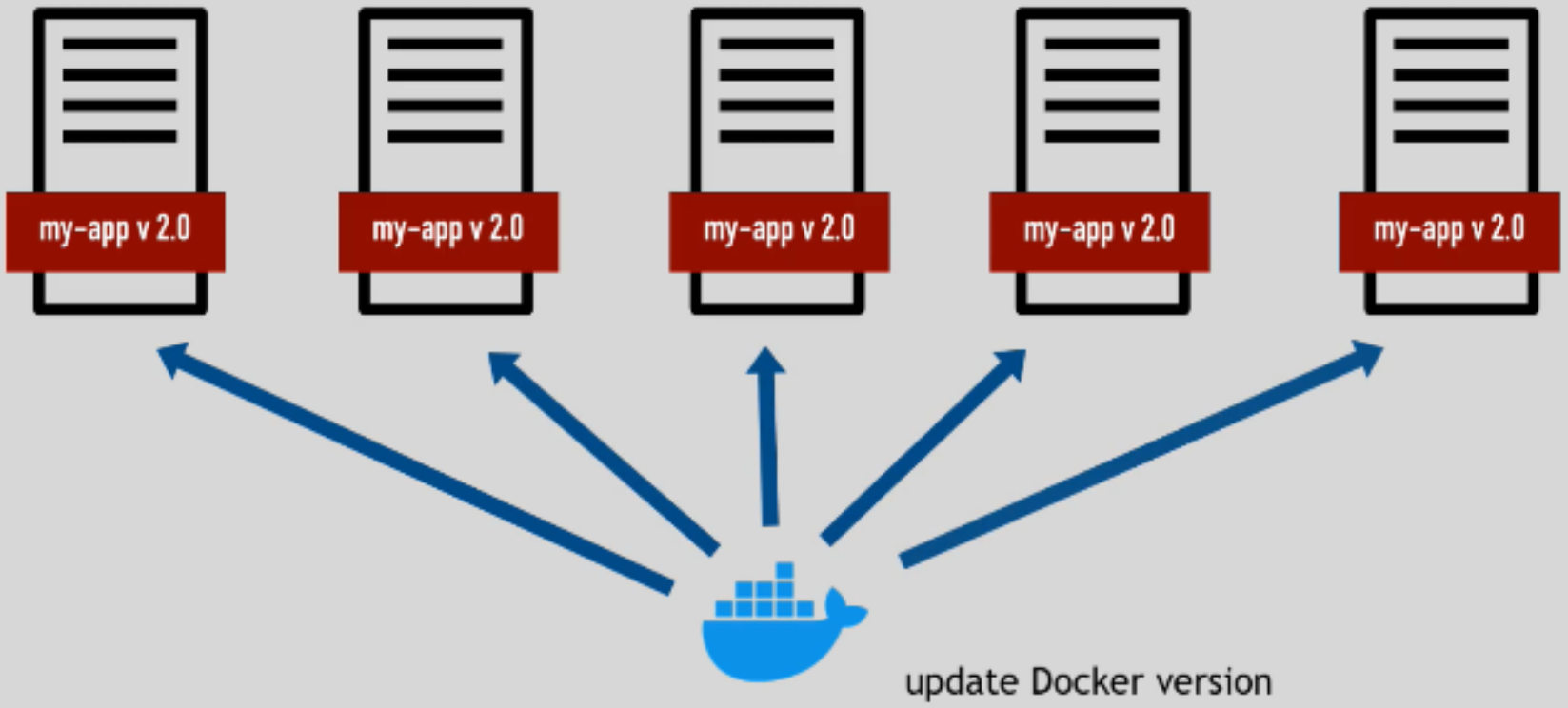
Update

Update app in 200 machines.

Manual task takes takes a lot of time.



Why use Ansible?



Why use Ansible?

Updates

Back Ups

Repetitive Tasks

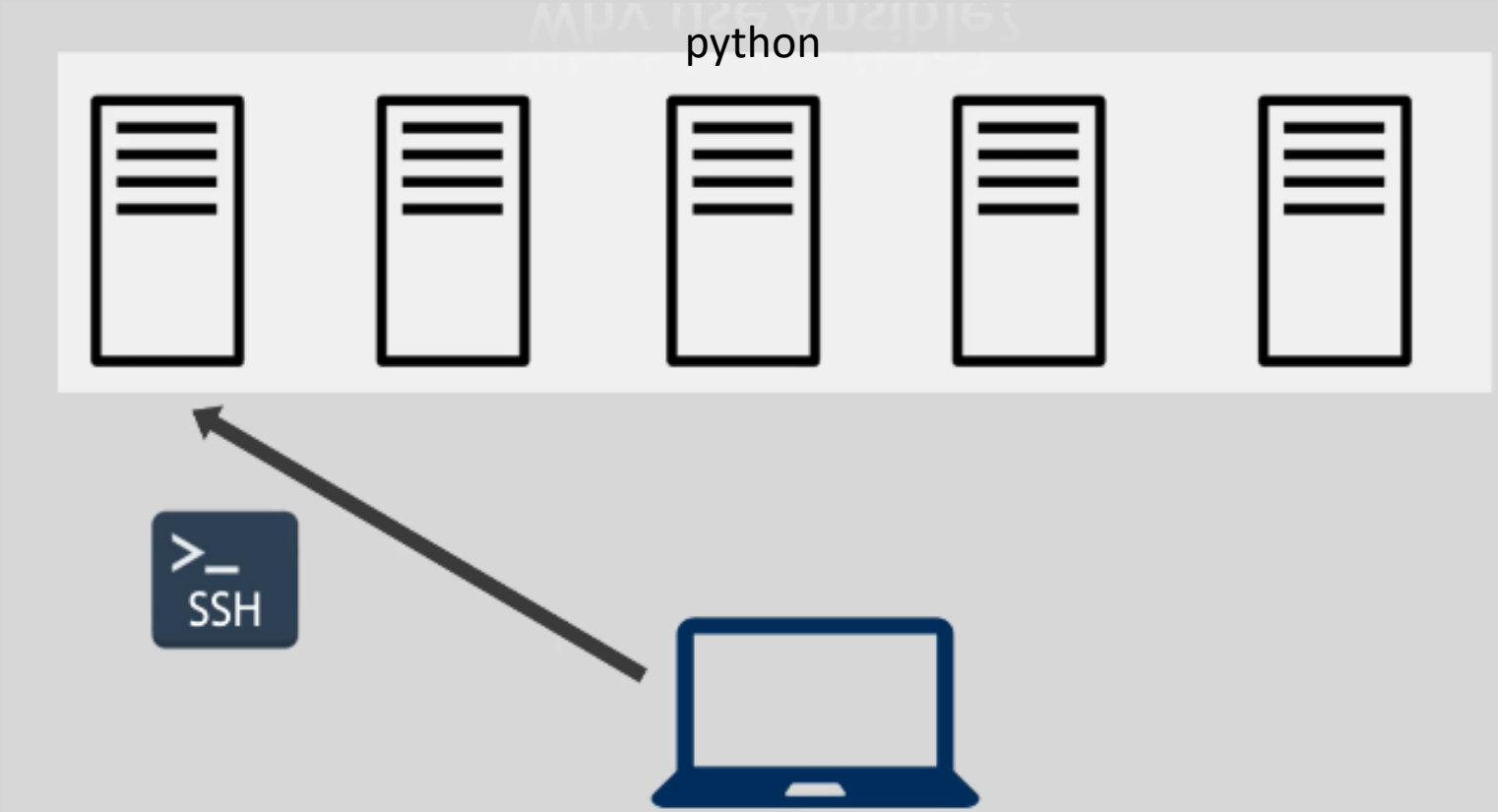
Create User

System Reboots

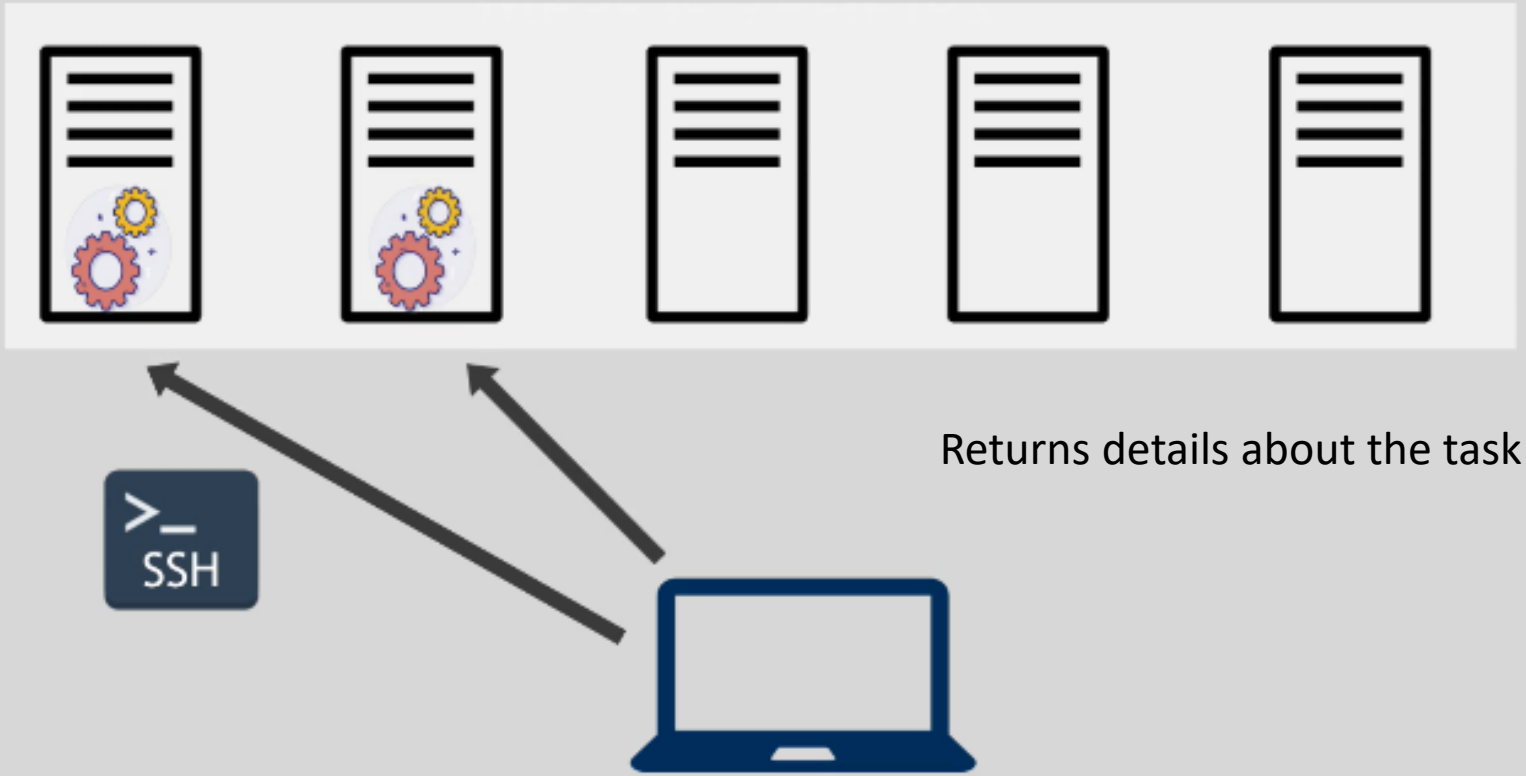
Assign Groups

Assign Permissions

How Ansible works?

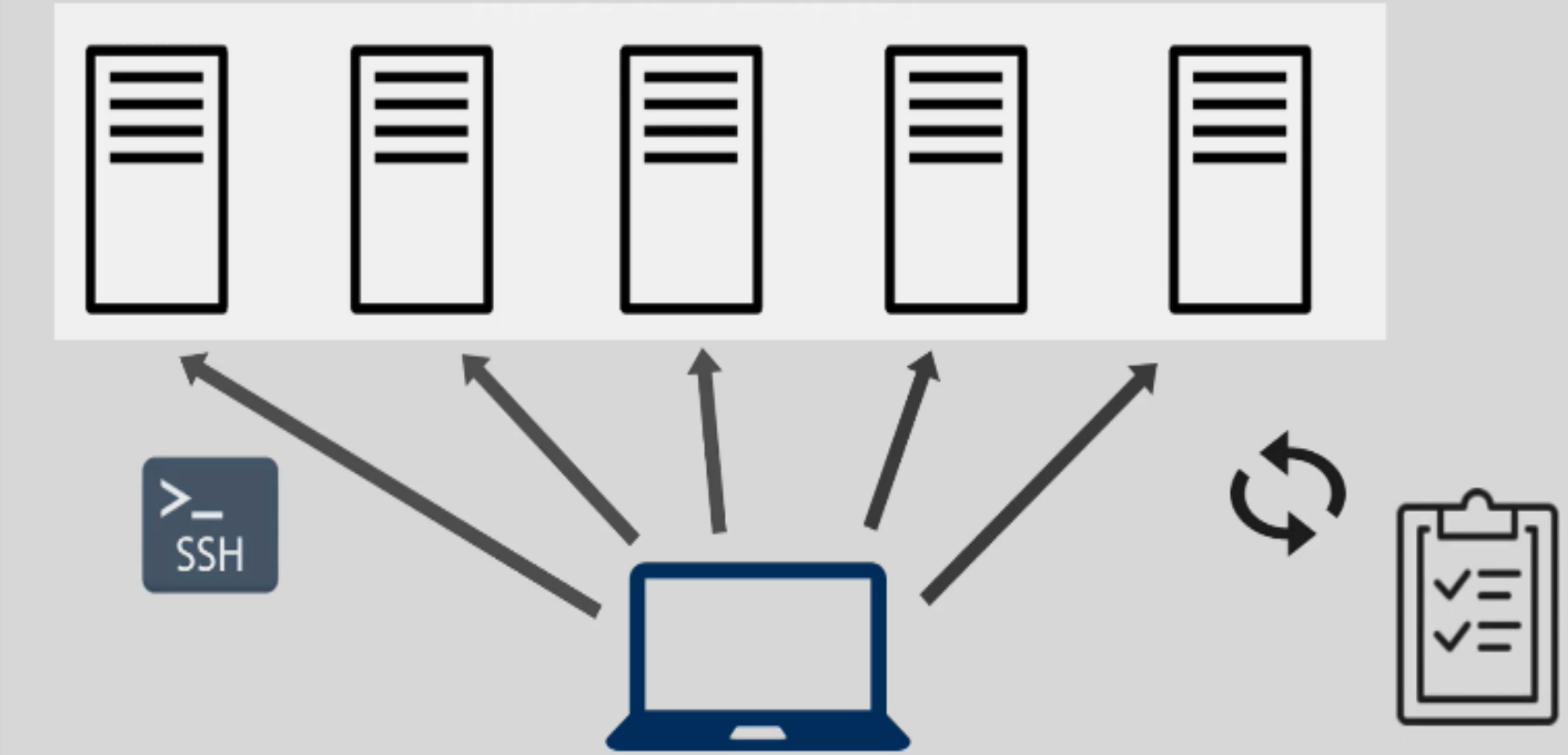


How Ansible works?



Returns details about the task and the target hosts

Why to use Ansible?



Why to use Ansible?

What did I do the last time? 🤔

Why to use Ansible?

Taking notes 💡

Why to use Ansible?

more efficient

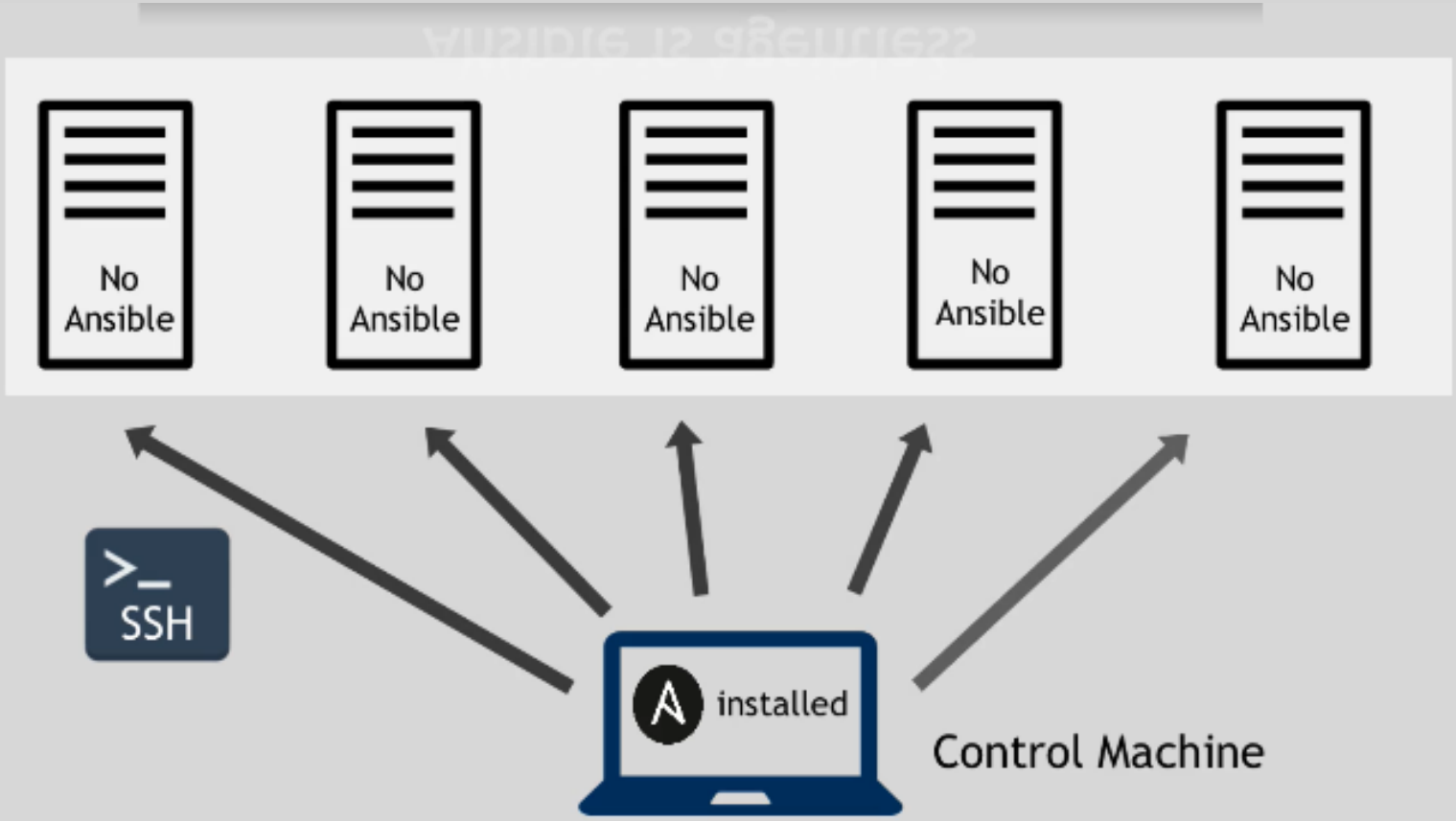
less time consuming

In 4 different ways

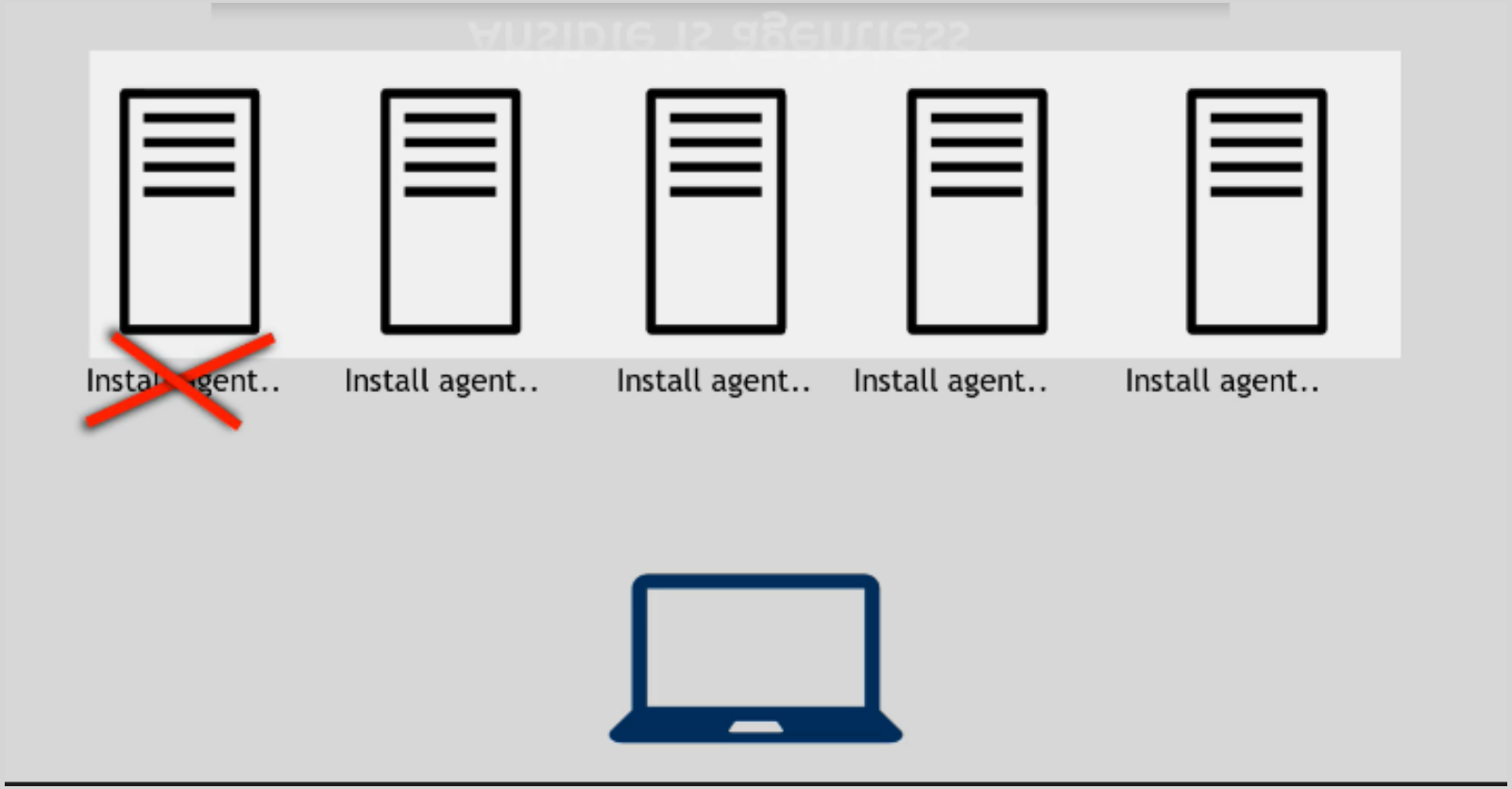
- 1. Execute tasks from your own machine
- 2. Configuration/Installation/Deployment steps in a single YAML File
- 3. Re-use same file multiple times and for different environments
- 4. More reliable and less likely for errors

Supporting all infrastructure

Ansible as Agentless



Ansible as Agentless



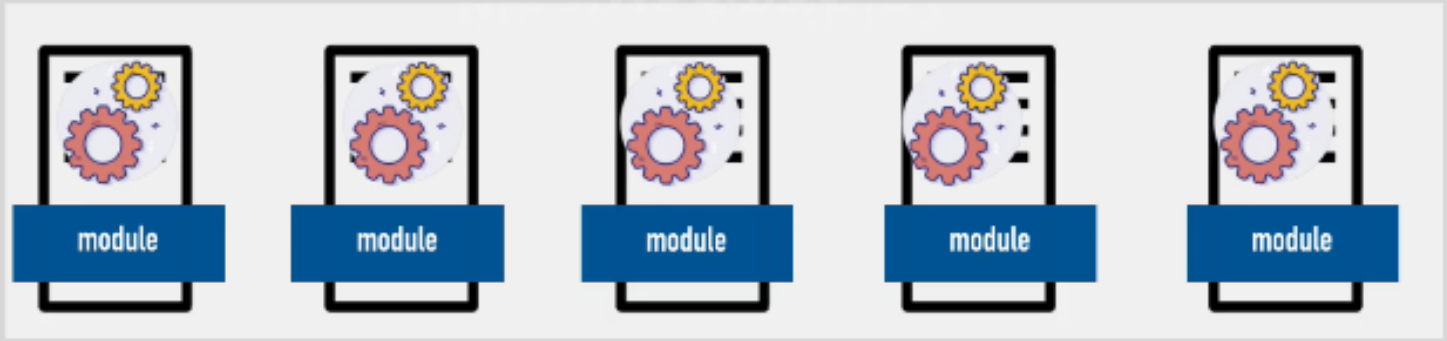
Ansible as Agentless

- No deployment effort in beginning
- No upgrade effort
- No worries

Ansible Architecture

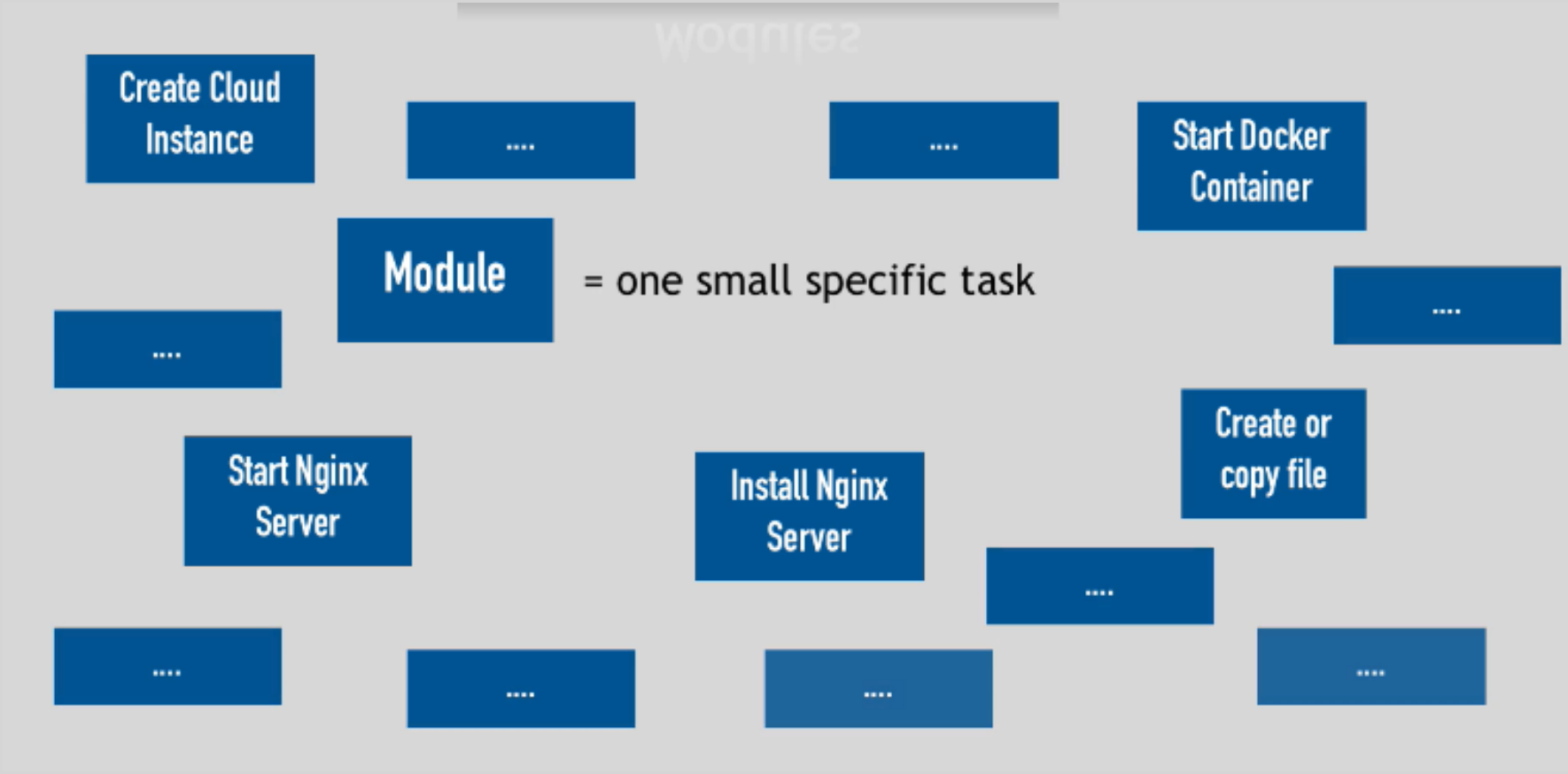
Modules

Modules



Get pushed to the target server.

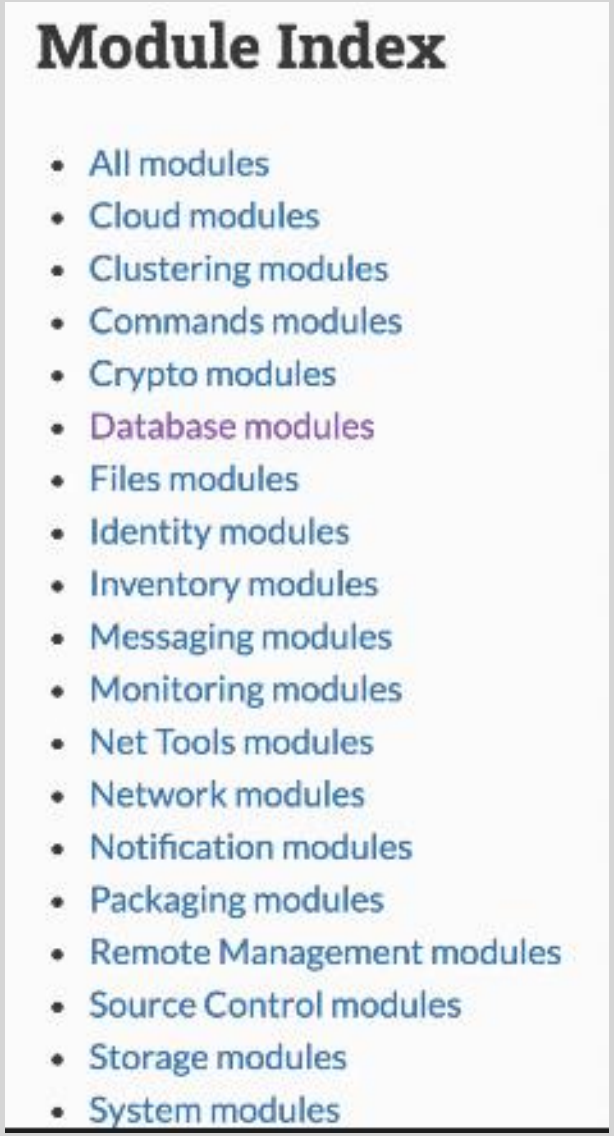
Modules


Create Cloud Instance
....
....
Start Docker Container
Module
= one small specific task
....
....
Create or copy file
Start Nginx Server
Install Nginx Server
....
....
....
....

Modules

Module = one small specific task

List of Modules in Ansible Official Documentation:



Comparable Tools

Ansible

- Simple YAML
- agentless

Puppet and Chef

- Ruby
- More difficult to learn
- Installation needed
- So need for managing updates on target servers

ANSIBLE VE TERRAFORM

FARKLAR

Terraform

- - Genellikle Infrastructure provisioning tool(Altyapı sağlama aracı) olarak kullanılır.
- - Görece daha yenidir. (2014)
- - Orchestration yeteneği daha gelişmiştir.
- Orchestration, bilgisayar sistemlerinin, uygulamaların ve hizmetlerin otomatik yapılandırması, yönetimi ve koordinasyonudur. IT departmanının karmaşık görevleri ve iş akışlarını daha kolay yönetmesine yardımcı olur.

Ansible

- - Genellikle configuration tool olarak kullanılır. Yani önce infrastructure oluşturursun. Sonra onu configure etmek için Ansible kullanırsın.
- - Terraforma göre daha eskidir. (2012)

ANSIBLE VE TERRAFORM

BENZERLİKLER

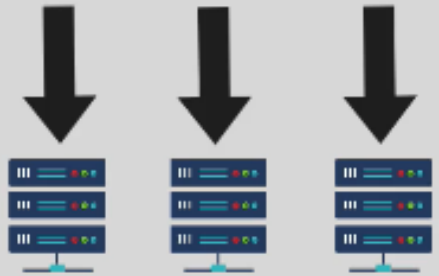
- İkisi de IAC tool'u olarak kullanılır. Yani ikisiyle de altyapıyı sağlarız, yapılandırırırz ve yönetiriz.

Install Ansible

Control Node

- the machine that runs Ansible
- Windows is not supported for the control node
- and manages target servers

on a remote server



Ansible Playbooks

Ansible Playbooks

Execute multiple modules in a sequence:

```yaml
tasks:
  - name: create directory for nginx
    file:
      path: /path/to/nginx/dir
      state: directory

  - name: install nginx latest version
    yum:
      name: nginx
      state: latest

  - name: start nginx
    service:
      name: nginx
      state: started
```

Module name

Arguments

1 configuration

Ansible Inventories

Default inventory file
hosts

```
[amazon]
amazon ansible_host=3.227.231.20 ansible_user=ec2-user

[aws:vars]
ansible_ssh_private_key_file=/home/ec2-user/deneme.pem
```

Why Ansible?

- Provisioning
- Configuration Management
- Continuous Delivery
- Application Deployment
- Security Compliance

Scripts
- Coding Skills
- Maintenance

- Simple
- Powerful
- Agentless

Scipts vs Ansible Playbook

```
#!/bin/bash
# Script to add a user to Linux
system  if [ $(id -u) -eq 0 ]; then
   $username=johndoe
   read -s -p "Enter password : " password
   egrep "^$username" /etc/passwd
   >/dev/null  if [ $? -eq 0 ]; then
      echo "$username
      exists!"  exit 1
   else
      useradd -m -p $password $username
      [ $? -eq 0 ] && echo "User has been
added  to system!" || echo "Failed to add a
user!"
   fi
```

```
- hosts:
  all_my_wbe_bs_esrevrevresrs_in
  _DclRoud          tasks:
    - user:
        name: johndoe
        Password : Denemel
        Status : present
        Shell : /bin/sh
        Home : /home/johndoe
```

Installation

https://docs.ansible.com/ansible/latest/installation_guide/intro_installation.html

- For AWS EC2:

```
sudo yum update -y
sudo amazon-linux-extras install ansible2
```

- For PIP:

```
sudo pip install ansible
```

- For Ubuntu:

```
sudo apt-get update
sudo apt-get install ansible
```

Inventory File

Machine Names

Variables to be Used

```
server1.company.com                ansible_connection=ssh    ansible_user=root
server2.company.com                ansible_connection=winrm  ansible_user=admin
server3.company.com                ansible_connection=ssh    ansible_ssh_pass=P@#
localhost                          ansible_connection=localhost
server4.company.com                ansible_connection=winrm
```

Playbook

- Playbook is a single YAML file
- Playbook defines set of activities(tasks) to be run on hosts
  - Task is an action to be performed on the host.

```yaml
tasks:
  - name: create directory for nginx
    file:
      path: /path/to/nginx/dir
      state: directory

  - name: install nginx latest version
    yum:
      name: nginx
      state: latest

  - name: start nginx
    service:
      name: nginx
      state: started
```

Hosts

```
name  Play 1
hosts
localhost
tasks

    name  Execute command
    ‘date’

    command  date

    name  Execute script on server script
    test_script.sh

    name  Install httpd
    service yum

    name  httpd
    state
    present

    name  Start web
    server service

    name  httpd
    state
    started
```

```
name  Play 1
hosts
localhost
tasks
name  Execute command
‘date’
command  date
name  Execute script on
server
script
test_script.sh
name  Install httpd
service
```

Run

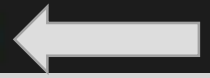
- Execute Ansible Playbook
- Syntax: ansible-playbook <playbook file name>

```
ansible-playbook playbook.yml

ansible-playbook --help
```

Idempotency

- Why **started** and not **start**?

```
tasks:
  - name: create directory for nginx
    file:
      path: /path/to/nginx/dir
      state: directory

  - name: install nginx latest version
    yum:
      name: nginx
      state: latest

  - name: start nginx
    service:
      name: nginx
      state: started
```



- Ensure service nginx is started.
- If service nginx has not already started;
  start
- Else;
  Do nothing

Variable

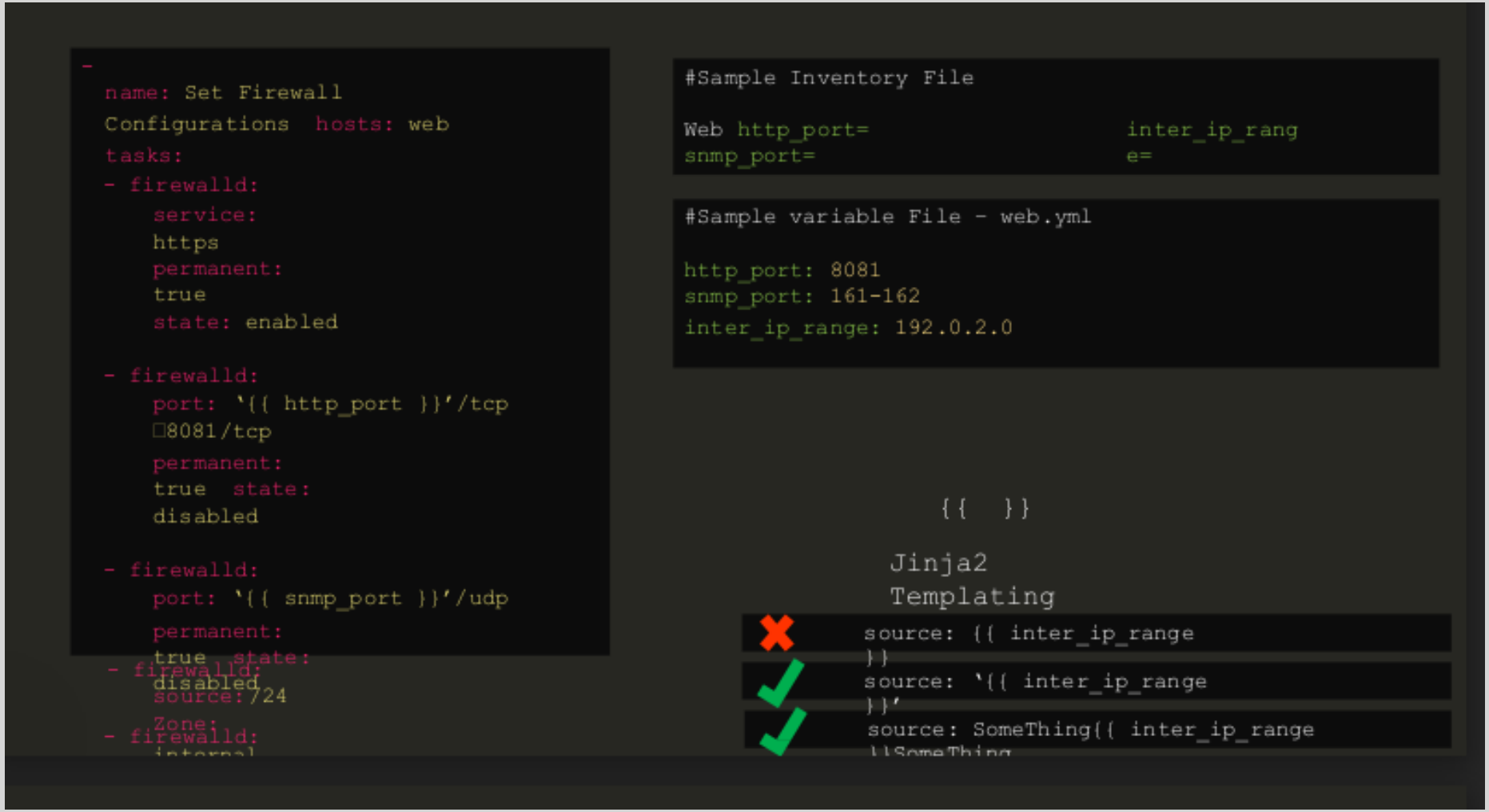
- Stores information that varies with each host

```
# inventory

Web1 ansible_host=server1.company.com ansible_connection=ssh
ansible_shh_pass=P@ssW db
ansible_host=server2.company.com
ansible_connection=winrm ansible_shh_pass=P@s Web2
ansible_host=server3.company.com ansible_connection=ssh
ansible_shh_pass=P@ssW
```

```
# Playbook.yml

name: Add DNS server to
resolv.conf  hosts: localhost
vars_file : /var.yml
tasks:
   - lineinfile:

        path: /etc/resolv.conf
        line: 'nameserver
        10.1.250.10'
```

```
# variables.yml

variable1:
value1
variable2:
value2
```

Using Variables



Loops

```
-
  name: Create users
  hosts: localhost
  tasks:
   - user: name='{{ item }}'   state=present
     loop:
       - joe
       - george
       - ravi
       - mani
       - kiran
       - jazlan
       - emaan
       - mazin
       - izaan
       - mike
       - menaal
       - shoeb
       - rani
```

With_*

```
-
  name: Create
  users  hosts:
  localhost
  tasks:                          state=prese
   - user: name=  '{{ item        nt
     }}'

     loop:
       - joe
       - george
       - ravi
       - mani
```

```
-
  name : Create
  users  hosts:
  localhost
  tasks:                          state=prese
   - user: name=  '{{ item        nt
     }}'

     with_items:
       - joe
       - george
       - ravi
       - mani
```

With_*

```
-
  name: Create
  users  hosts:
  localhost
  tasks:
   - user: name=  '{{ item    state=prese
     }}'                     nt

     loop:
       - joe
       - george
       - ravi
       - mani
```

```
-
  name: Create
  users  hosts:
  localhost
  tasks:
   - user: name=  '{{ item    state=prese
     }}'                     nt

     with_items:
       - joe
       - george
       - ravi
       - mani
```

Conditionals & Register

```
name  Check status of a service and email if its down
hosts  localhost
tasks
    command  service httpd status
    register  result
    mail
      to  admin@company.com
      subject  Service Alert
      body  Httpd Service is down
      when  result.stdout.find('down') != -1
```