Primeiros passos com o Ansible

Caio Volpato (**REDACTED**)

3 de agosto de 2022

Texto original

Essa palestra foi originada de um texto do mesmo autor:
ansible localhost -a "/bin/echo Primeiros passos com o Ansible"

Somos um grupo de divulgação cientifica no medium, escrevemos sobre Computação, Matemática, Estatística e Ciência de Dados.

- Link: medium.com/computando-arte/
- Fundado em Nov/2020 🎂
- A informação quer ser livre: Licenciado sob CC BY-SA 4.0
- /join: Por que e como fazer um blog técnico

O que é o ansible?

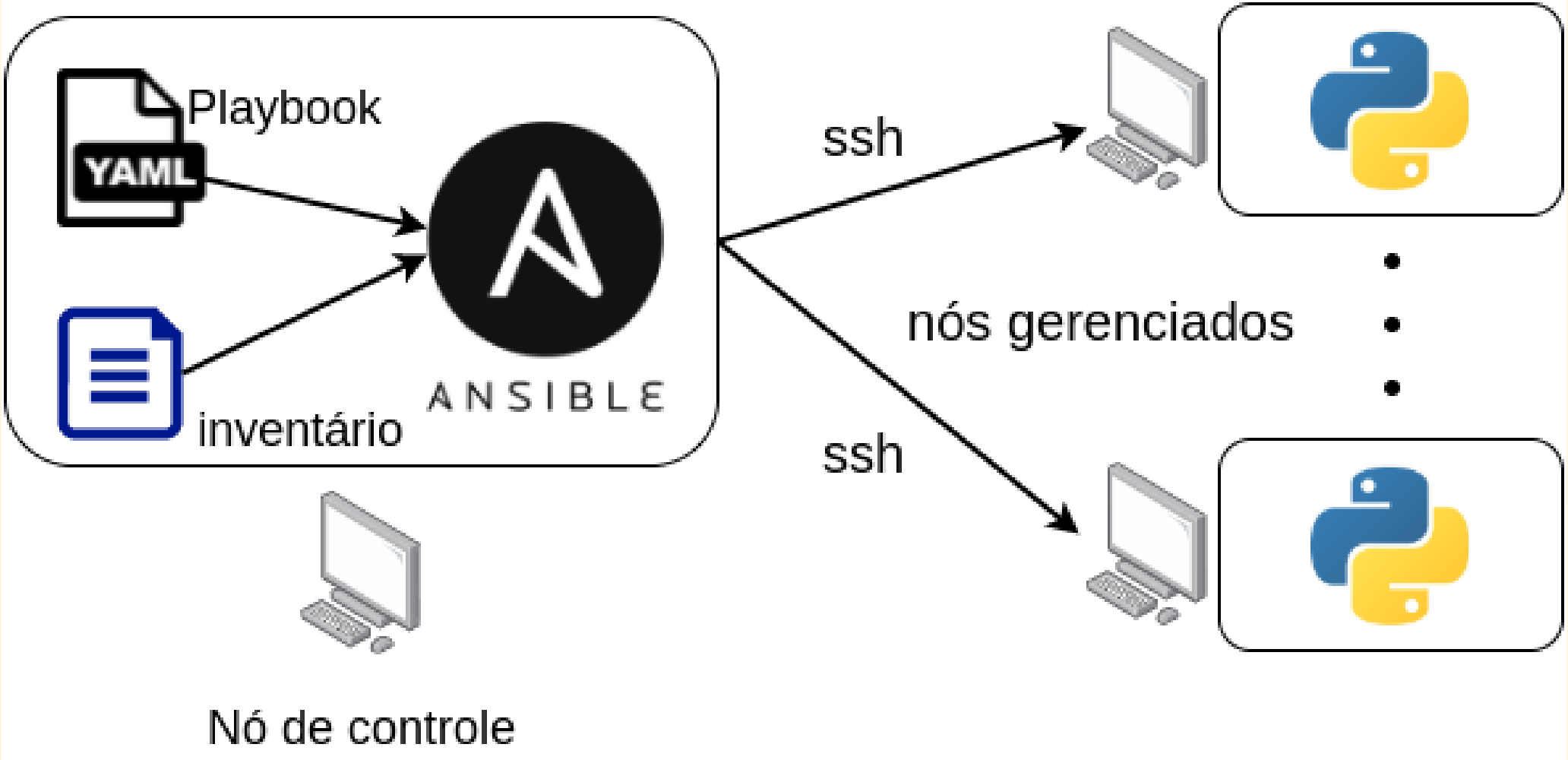
Ansible é uma ferramenta de gestão de configuração, que permite:

- Gerenciar diversas maquinas remotamente, via ssh
- Automação e manutenção de configuração de ambientes

Historia do ansible

- Criado em 2012, comprado pela RedHat em 2015
- Escrito em python
- Licença: GPL-3.0
- Produto comercial é o Ansible Automation Platform, quem quiser ter um gostinho tem a versão open source: awx

como o ansible funciona?



Vantagens e desvantagens do ansible

Quando comparado com ferramentas similares (puppet e chef)

o ansible tem as seguintes vantagens:

- Fácil aprendizado
- Não precisa de agents

Desvantagens:

- por não ter um agent, não escala muito bem (nó de controle precisa escalar verticalmente).

Instalando o ansible

Pra instalar o ansible, basta fazer

```
pip install ansible
```

No nó de controle.

Para funcionar de forma completa, é necessário ter o python nas maquinas gerenciadas.

Entretanto, é possível usar o ansible de forma limitada e instalar o python, contornando esse pré requisito ;)

montando o inventario

O arquivo inventario contem informações das maquinas gerenciadas, quais maquinas, como conectar nelas, grupos e variáveis.

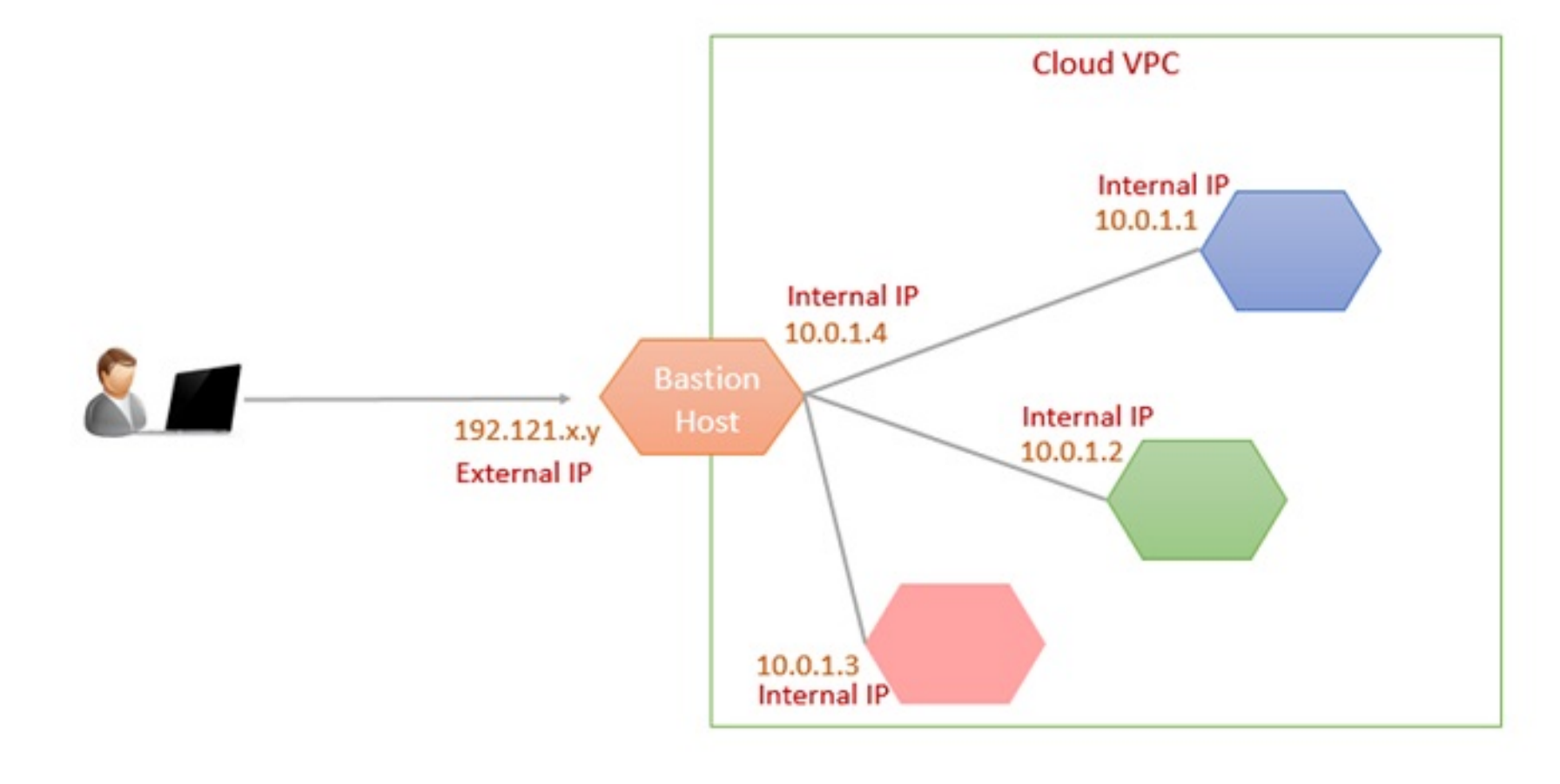
Pode ser escrito no formato yaml ou ini.

```
[pies]
rpi4 ansible_host=192.168.1.205 ansible_user=piuser ansible_password="piepass"

[vms]
rockylinux ansible_host=192.168.121.182 ansible_user=rockyuser

192.168.52.[02:10:2]

[campinas:children]
rpi4
rockylinux

[all:vars]
ansible_become=yes
ansible_become_method=sudo
ansible_become_password="rootpass"
```

Inventario: vars

- ansible_host: ip ou dominio da maquina
- ansible_{become,method}: se o ansible deve ou não elevar privilegios, e como o metodo, como sudo, doas e su
- ansible_password: senha para conectar via ssh.
- ansible_become_password: senha para elevar privilegios, com sudo/doas é a mesma senha (ansible_password).

Inventario: Como lidar com hosts bastião?



Fonte: [What is Bastion Host]

Inventario: Como lidar com hosts bastião?

O ssh suporta nativamente fazer um "proxy" através do bastião, é a opção ProxyJump, através do argumento -J

```
ssh -J bastion_host internal_machine
```

Dessa forma, basta instruir o ansible usar a opção ProxyJump, no inventario use a opção ansible_ssh_common_args.

Outra abordagem semelhante é criar um arquivo ssh_config e passar o mesmo no inventario.

```
# ssh_config:

Host bastion
   HostName bastion.foobar.tld
   User connect_user
   Port 22
   Compression yes
   ServerAliveInterval 10
   ServerAliveCountMax 30

Host internal_machine
   ProxyJump bastion
```

ansible_ssh_common_args='-F ssh_config'

ansible-vault

Como proteger segredos como senhas?

No ansible, tem uma ferramenta inclusa chamada ansible-vault

como usar?

Troque no inventario a senha por uma referencia a uma variável

```
# inventario.ini
[...]
[all:vars]
foobar_password="{{ var_name1 }}"
```

ansible-vault

E crie um yaml com as senhas:

vault.yml:

```
---
var_name1: 'psicohistoria'
```

Para criptografar: `ansible-vault encrypt vault.yml`

Para importar o vault:

```
ansible -i inventario.ini -m ping all -e @vault.yml --ask-vault-pass
```

ansible.cfg

O arquivo ansible.cfg permite configurar diversas opções do ansible, facilitando assim sua execução.

```
# ansible.cfg
[defaults]
host_key_checking = True
inventory = inventario.ini
vault_password_file = ../.vault_pass_file.txt
```

```
ansible -m ping -e @vault.yml all
```

comandos adhoc

Com inventario montado, podemos executar alguns comandos:

No ansible as tarefas são executadas através de modulos, por ex:

```
ansible -m ping all
```

```
ansible -m command -a id all
```

```
ansible -a hostnamectl all
```

```
ansible -m package -a "name=neofetch state=present" all
```

Escrevendo nosso primeiro playbook

Até agora sempre executamos comandos isoladamente, podemos também escrever uma "receita" com a sequencia de tarefas com um objetivo especifico, os playbooks.

Exemplo: Colocar um site estático no ar, usando o apache webserver.

exemplo apache

Passos da tarefa:

1. Instala o apache
2. Copia o site
3. Reinicia o apache

Como instala o apache? Só mandar um `sudo apt install apache2` ?

No debian sim! Porém no rockylinux o pacote chama httpd

Tem como o ansible detectar qual distro?

variavel ansible_facts

O ansible quando conecta ele identifica algumas infos da maquina e as disponibiliza na variabel ansible_facts.

Para acessar essas infos use o modulo setup:

```
ansible -m setup all
```

Podemos usar a ansible_os_family ou ansible_distribution

- ansible_distribution = Rocky, Debian
- ansible_os_family = RedHat, Debian

Playbook:

```
---
- name: Configura um site estatico.
  hosts: all
  become: true

  vars:
    apache_name_dict:
      "Debian": apache2
      "RedHat": httpd

  handlers:
    - name: restart apache
      service:
        name: "{{ apache_name_dict[ansible_os_family] }}"
        state: restarted
```

```yaml
tasks:
  - name: Instala o apache.
    package:
      name: "{{ apache_name_dict[ansible_os_family] }}"
      state: present

  - name: Copia o html da pag.
    copy:
      src: page.html
      remote_src: false
      dest: "/var/www/html/index.html"
      mode: 0664
    notify: restart apache

  - name: Habilita o servico do apache.
    service:
      name: "{{ apache_name_dict[ansible_os_family] }}"
```

rodando o playbook

Rodando o playbook:

```
ansible-playbook playbook.yml
```

Idempotência

Uma coisa que você precisa se atentar é se seus playbooks são idempotentes.

Idempotencia significa que o efeito do seu playbook é o mesmo independente de quantas vezes o mesmo foi executado.

Ou seja, seu playbook não pode ter efeitos colaterais

Na pratica, execute o playbook 2x e veja que na 2a vez não teve nenhuma task como changed.

Idempotência: desafios

O principal desafio de garantir a idempotencia é quando usamos um command ou shell.

Como o ansible vai saber se o mesmo foi executado anteriormente?

Nesses casos o ansible tem a opção creates, onde é criado um arquivo "marcador"

```yaml
---
- hosts: all
  become: true

  tasks:
    - name: inicializa um arquivo
      shell: echo "passei por aqui" > /etc/marcador_xpto.txt
      args:
        creates: /etc/marcador_xpto.txt
```

uso excessivo de shell

Dicas uteis

Use o **vagrant** para facilitar a criação de VMs, ajudando a testar seus playbooks.

ansible-lint te ajuda a seguir as boas praticas do ansible.

referências

Livro Ansible for DevOps do Jeff Geerling.

- lives: Ansible 101 playlist.
- repo: github.com/geerlingguy/ansible-for-devops