**LABORATÓRIO 02 -** Configurando um servidor HTTP com Apache e GIT.

REQUISITOS:

* Máquina host com **Ansible**
* Máquina secundária (srvweb **CLIENT**) Debian, sem git e sem apache

Alterar o nome:  **hostnamectl set-hostname srvweb**

* Ambas máquinas disponíveis em modo Bridge

**1 - Preparando a máquina do Ansible para acessar a máquina CLIENT**

**2 - Configurar nosso primeiro playbook, na máquina do Ansible, mas antes, configure um diretório**

**3 - Erros na execução do playbook, na máquina do Ansible**

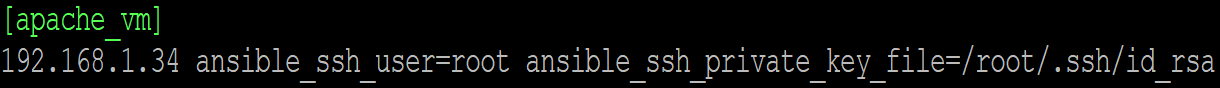
**4 - Preparar SrvWeb CLIENT e verificar serviços a partir do Ansible**

**5 - Ajustar Site no ServidorWeb CLIENT e pelo Ansible Host**

**1 - Preparando a máquina do Ansible para acessar a máquina CLIENT**

**1.1 - Editar arquivo hosts:**

Com o IP da máquina secundária em mãos, vá para a máquina host do Ansible e execute: **nano /etc/ansible/hosts**

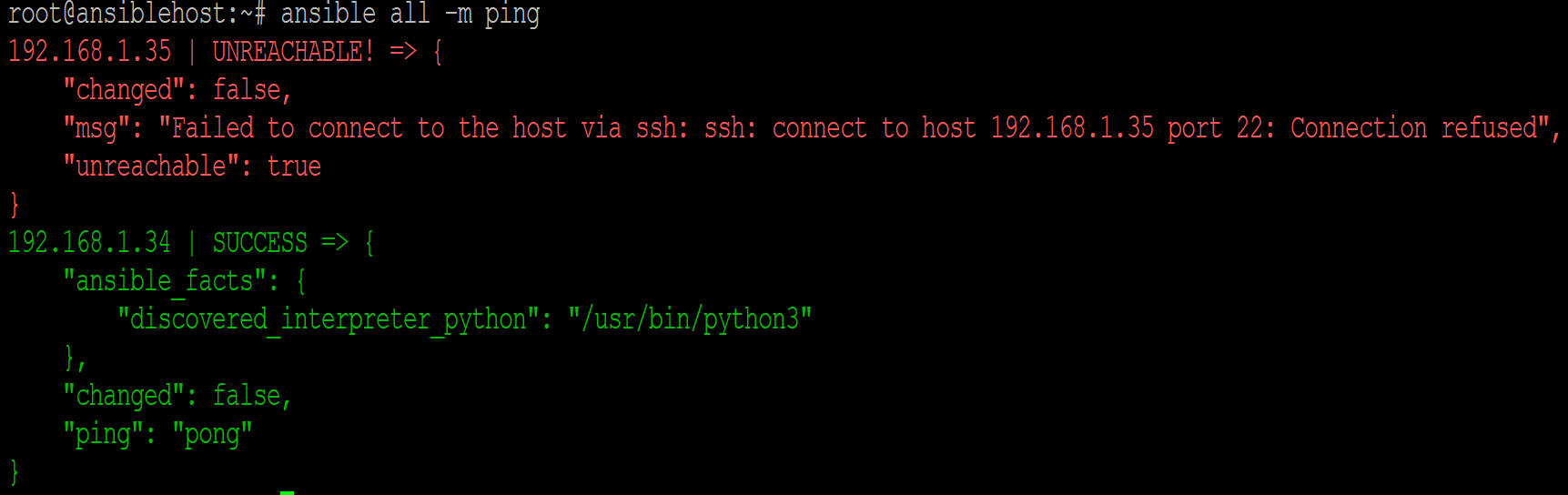


OBS: Configure igual ao da print, apenas troque o campo **<ip\_da\_maquina\_secundaria>** pelo IP da mesma, salve e feche o arquivo após configurar. Ex.:

**192.168.10.10 ansible\_ssh\_user=root ansible\_ssh\_private\_key\_file=/root/.ssh/id\_rsa**

**1.2 - Vamos testar a conexão:**

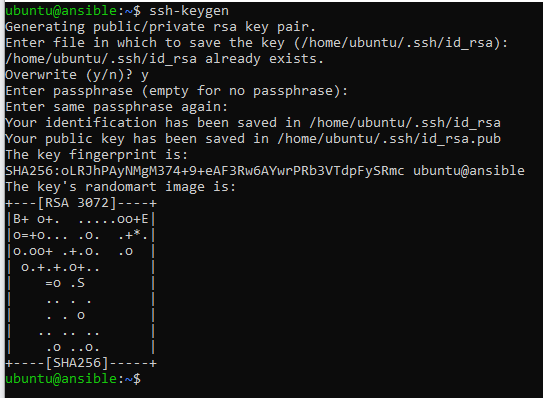
execute: **ansible all -m ping**



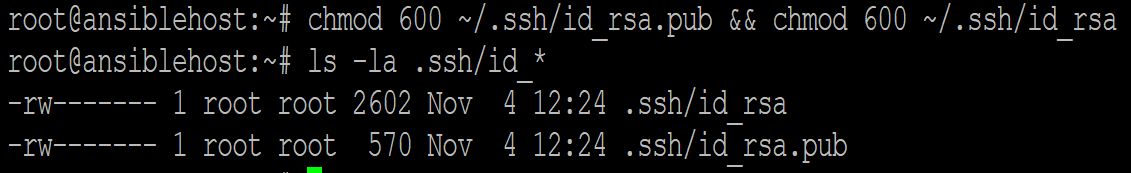
**OBS: Se você recebeu a mensagem de erro acima, siga os próximos passos**

**1.3 - Gere uma nova chave SSH na máquina do Ansible**

execute: **ssh-keygen**

****

**OBS:** Não configure nenhuma **passphrase** ou mude o local que a chave vai ser salvada

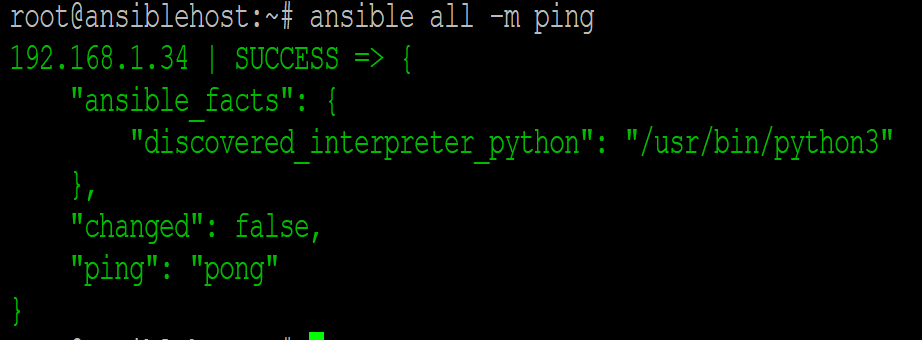
**1.4 - Com as chaves geradas, dê a permissão total para o usuário ubuntu escrever e ler a mesma:**execute: **chmod 600 ~/.ssh/id\_rsa\***

**1.5 - Copie a chave pública para o authorized\_keys da máquina secundária (onde o apache e git vão estar localizados)**

execute: **ssh-copy-id -i ~/.ssh/id\_rsa.pub root@<ip\_da\_sua\_segunda\_maquina\_CLIENT>**

**1.6 - Execute o teste do ansible novamente:**

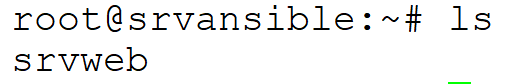
execute: **ansible all -m ping**

****

**2 - Configurar nosso primeiro playbook, na máquina do Ansible, mas antes, configure um diretório**

**2.1 - Criar Diretório para organização do Playbook**

execute: **mkdir srvweb**



No Ansible, uma das melhores maneiras de organizar seus playbooks é por diretórios, por padrão, quando você executa um playbook.

O Ansible vai procurar o primeiro **playbook.yml** que estiver disponível, ou seja, varrendo a pasta, caso não encontre, poderá retornar algum erro ou aplicar as atualizações/downloads no grupo de máquinas erradas.

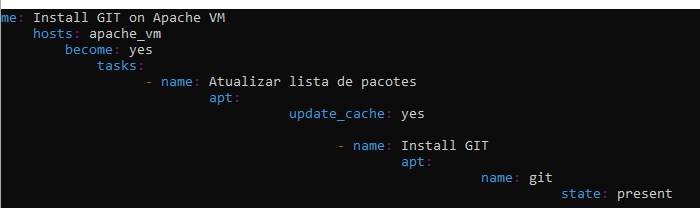
**2.2 - Crie um arquivo chamado playbook.yml**

execute: **touch srvweb/playbook.yml**

****

**2.3 - Agora, preste atenção na indentação do código, o Ansible é extremamente rígido nesse quesito, podendo dar diversos erros e muitas vezes não muito claros (Copie o conteúdo abaixo)**

**OBS:** Não pode ficar assim:

****

**O Ansible não vai aceitar, mas aqui vai um truque de mestre dos magos, saia do arquivo (sem salvar mesmo)**

**Execute o comando abaixo direto no shell:**

*cat <<EOF> srvweb/playbook.yml*

*---*

*- name: Instalar GIT e Apache na VM*

*hosts: srvwebs*

*become: yes*

*tasks:*

*- name: Atualizar lista de pacotes*

*apt:*

*update\_cache: yes*

*- name: Instalar GIT*

*apt:*

*name: git*

*state: present*

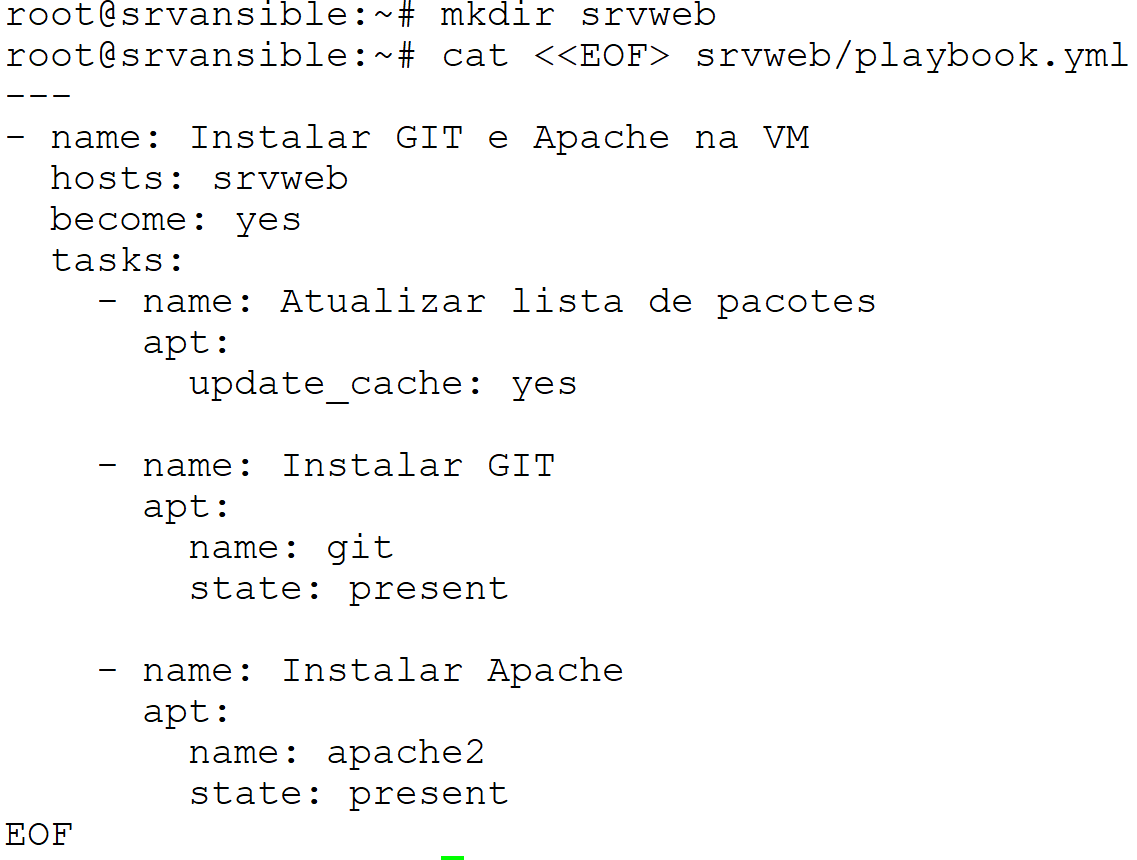
*- name: Instalar Apache*

*apt:*

*name: apache2*

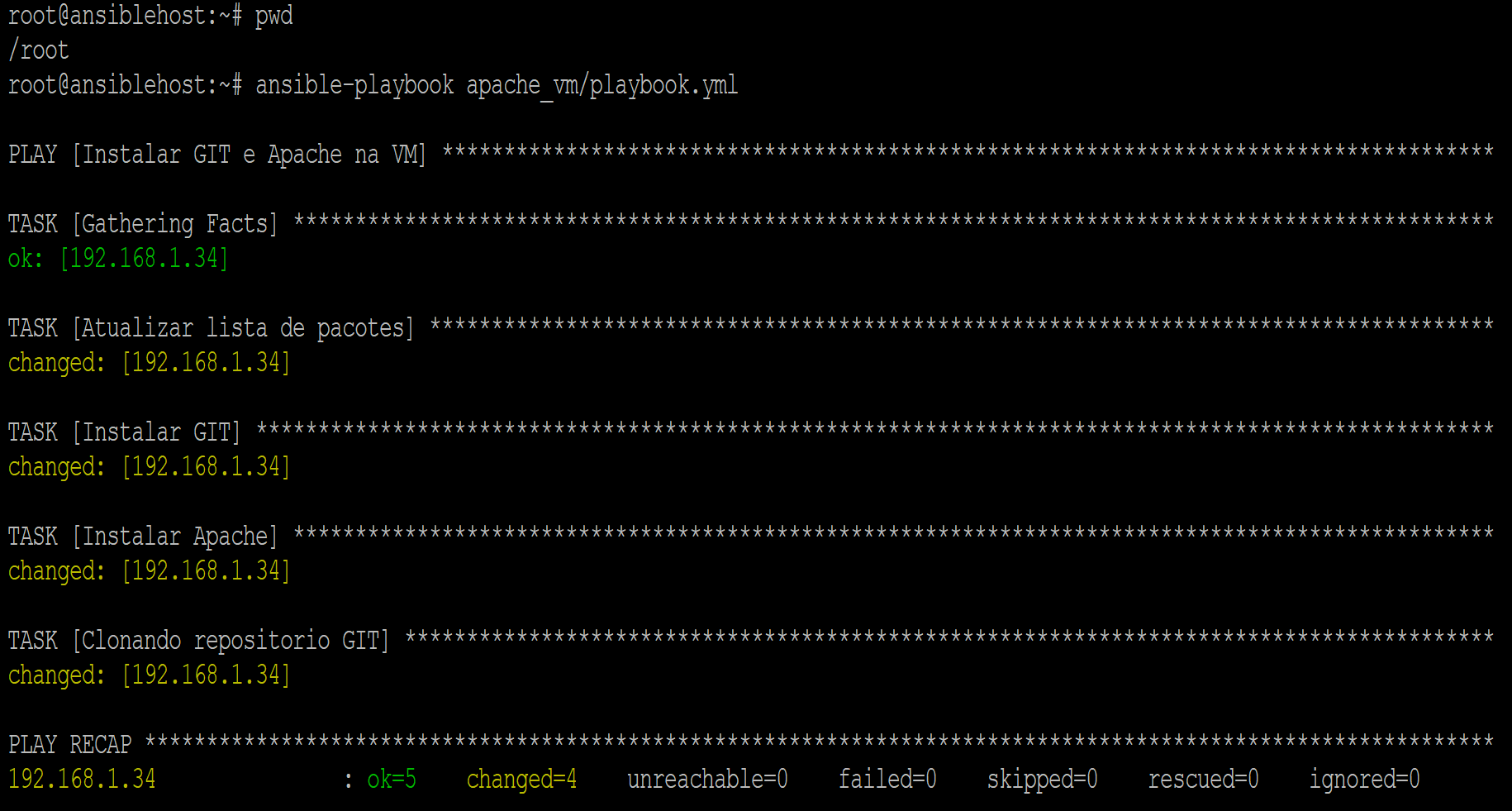
*state: present*

*EOF*

****

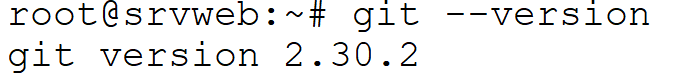
**2.4 - Agora vamos executar nosso playbook, para isso, se a instalação do ansible foi bem sucedida, teremos a linha de comando *ansible-playbook*:**

execute: **ansible-playbook srvweb/playbook.yml**

****

**2.5 -Testar o comando Git no SrvWeb CLIENT**

execute: **git --version**

****

**3 - Erros na execução do playbook, na máquina do Ansible**

Se houver um problema na hora de iniciarmos o nosso playbook, isso ocorreu por conta que no nosso playbook nós definimos um campo chamado

**become = true**, esse campo indica que as ações a seguir (tasks) deverão ser executadas com o usuário root ou usuário com sudo.

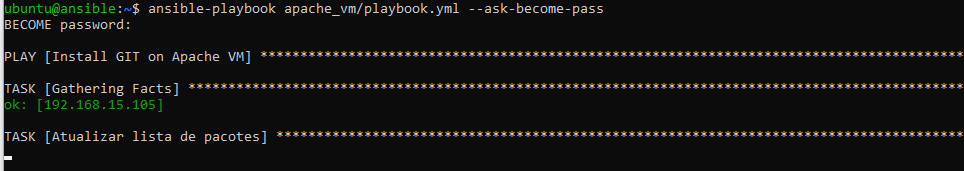
Há diversas maneiras de resolver isso:  
  
1 - Acessando a máquina secundária e tirando a senha do usuário root (NÃO RECOMENDADO)

2 - Permitindo que o usuário default da imagem (aluno) consiga baixar pacotes (NÃO RECOMENDADO)

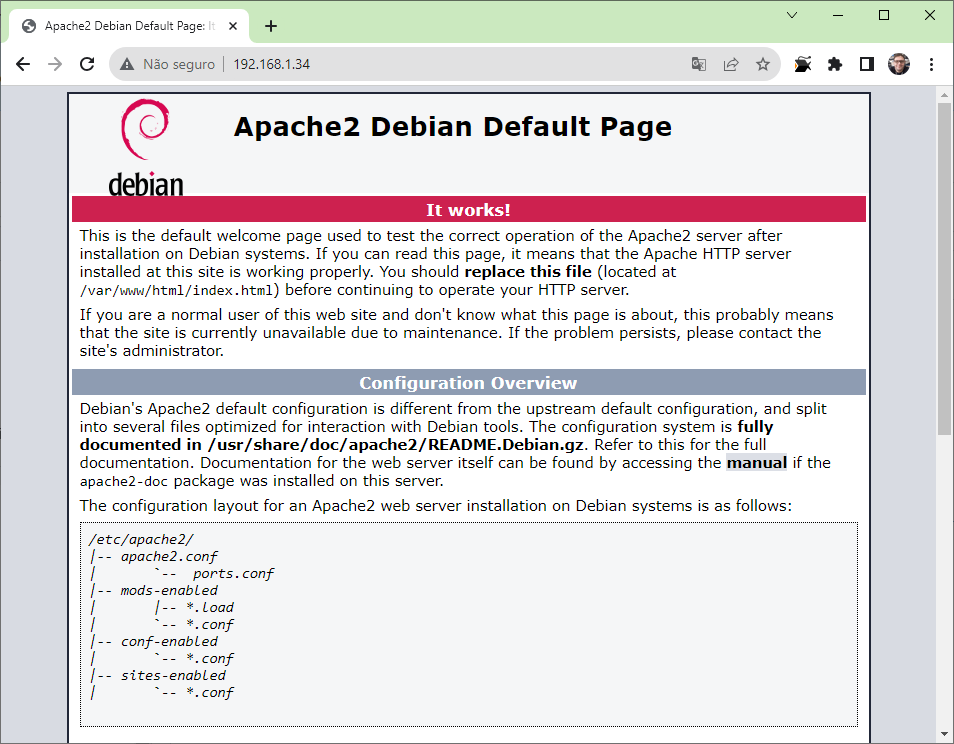
3 - Criando um usuário do ansible com a permissão para baixar pacotes e fazer determinadas ações (RECOMENDADO)  
  
4 - Adicionar o parâmetro ***--ask-become-pass*** para que, quando rodarmos o comando novamente, o ansible nos pergunte sobre a senha do usuário root (NÃO RECOMENDADO EM AMBIENTES DE PRODUÇÃO)

Por questões de agilidade e tutorial, vamos utilizar a opção **4** portanto, execute o novo comando: **ansible-playbook srvweb/playbook.yml --ask-become-pass**

****

**Digite a senha do seu usuário ROOT**

**Aguarde as execuções das tarefas e Resultado esperado acima**

**3.1 - Verifique se consegue acessar o site do Apache:** **acesse o IP da máquina virtual secundária (srvweb) no Google**

Nosso apache foi instalado com sucesso e a página de testes parece funcionar. Vamos voltar para a máquina Ansible Host e ajustar o site, siga os próximos passos.

**4 - Criar Site no SrvWeb CLIENT pelo Ansible Host**

**4.1 - Na instância Ansible Host, vamos criar um outro playbook, montando um deploy de um site, ajustando nosso site para o site principal do apache**

**become: yes = ativar o sudo**

execute:

*cat <<EOF> srvweb/sitebike\_playbook.yml*

*---*

*- name: Configurando o site*

*hosts: srvwebs*

*become: yes*

*tasks:*

*- name: Clonando repositorio GIT*

*shell: git clone https://github.com/FofuxoSibov/sitebike.git*

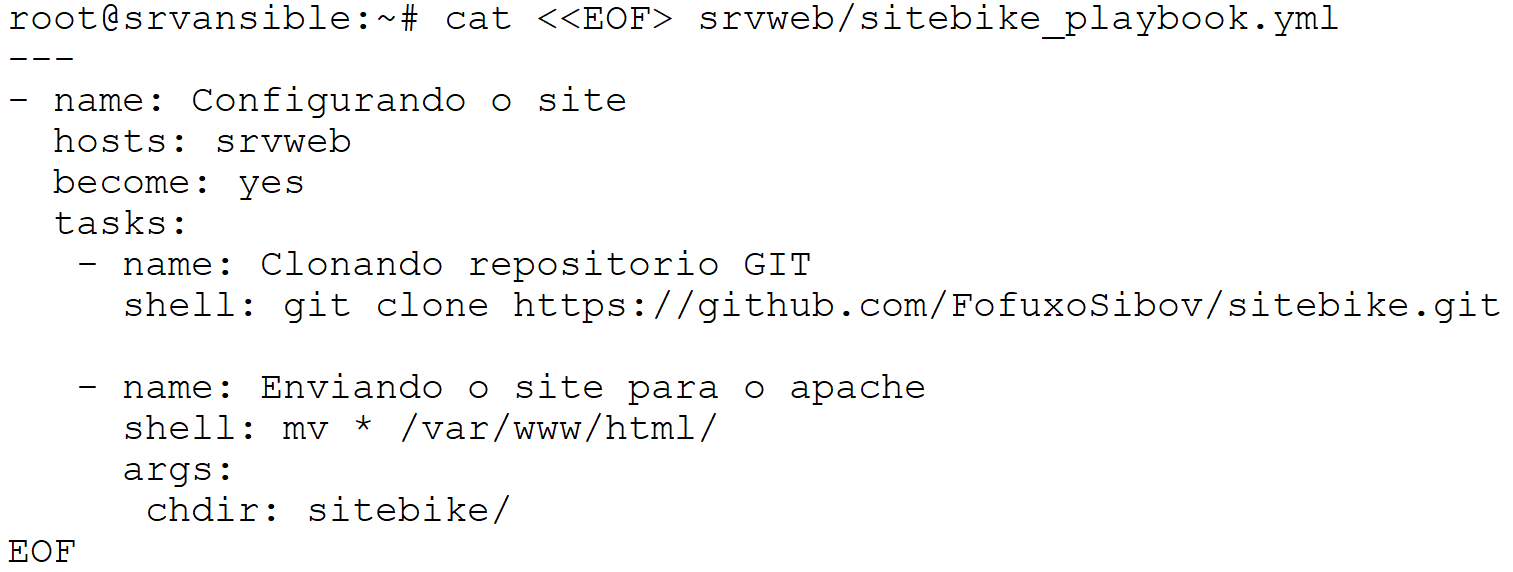
*- name: Enviando o site para o apache*

*shell: mv \* /var/www/html/*

*args:*

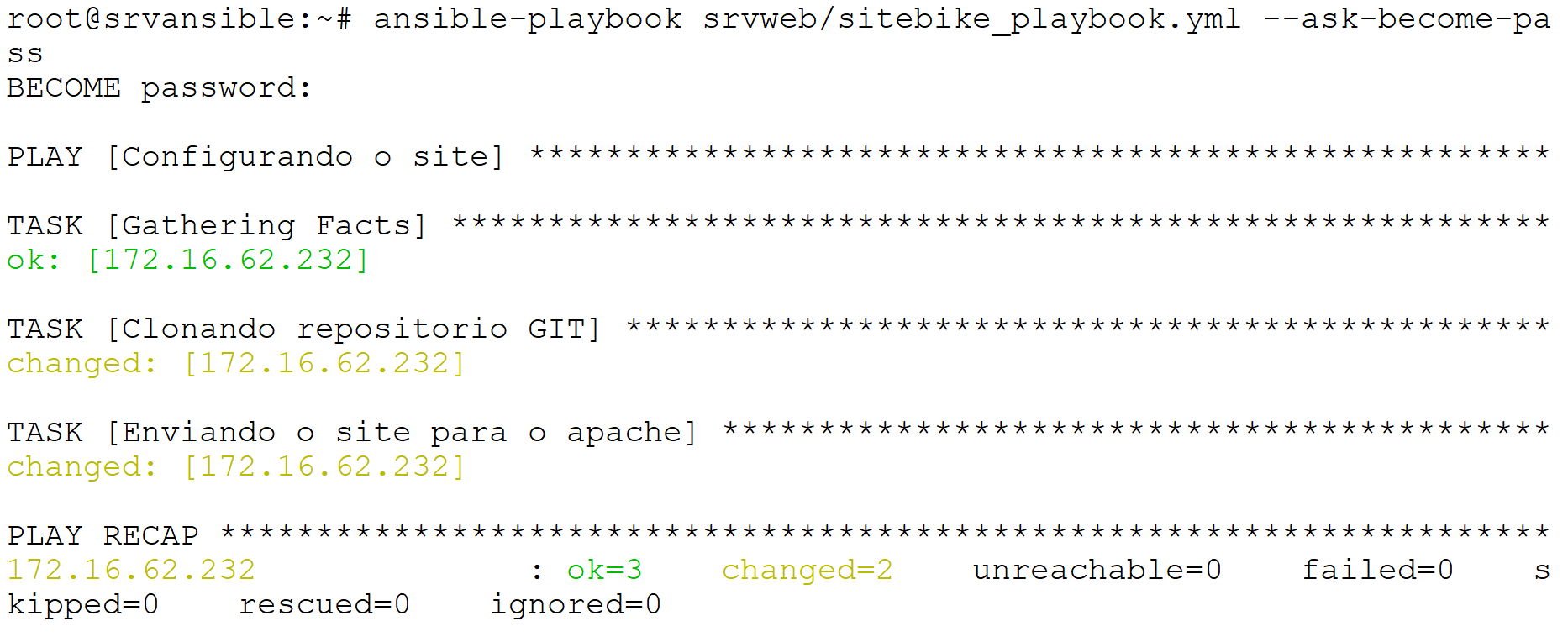
*chdir: sitebike/*

*EOF*



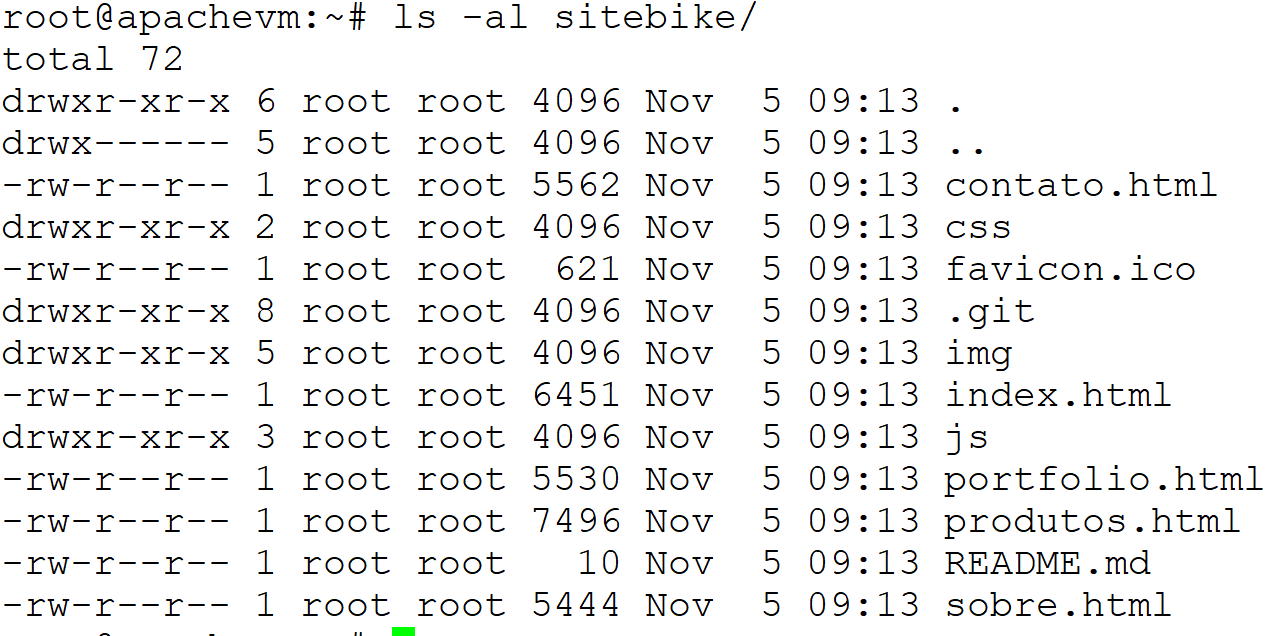
**4.2 - Agora vamos subir nosso novo site?**

execute: **ansible-playbook srvweb/sitebike\_playbook.yml --ask-become-pass**

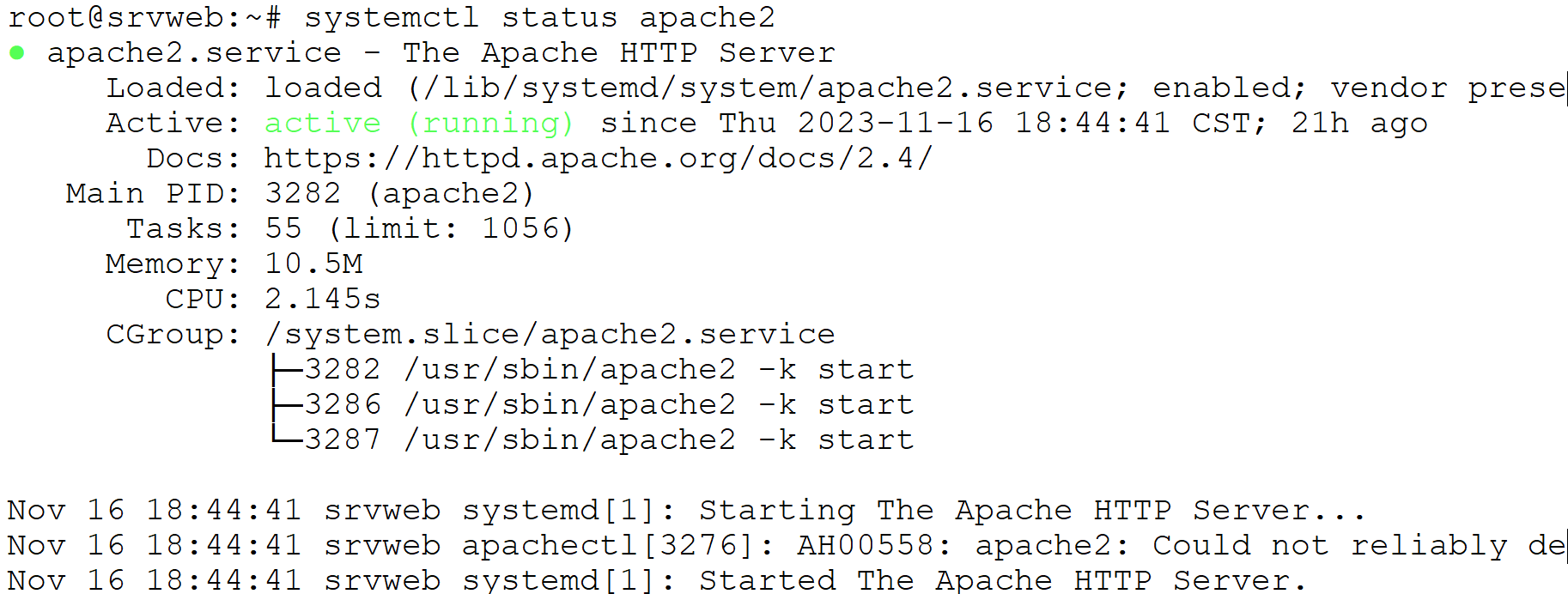


**5 - Preparar SrvWeb CLIENT e verificar serviços a partir do Ansible**

**5.1 - Verifique se o repositório foi clonado corretamente:** **ls -al sitebike/**

****

**5.2 - Verifique se o apache foi instalado corretamente:** **systemctl status apache2**

****

**5.3 - Se tudo deu certo, deu certo tudo, vamos validar nosso site, acesse novamente o IP da máquina secundária na WEB:**

**6 - Preparar Firewall do SrvWeb CLIENT a partir do Ansible**

**Lucas Gregório 2RT-2023**

*cat <<EOF> srvweb/firewallv2.yml*

*---*

*- name: Configuração do Firewall com iptables*

*hosts: srvwebs*

*become: yes*

*tasks:*

*- name: Permitir conexões SSH do Servidor Ansible*

*iptables:*

*chain: INPUT*

*protocol: tcp*

*source: 172.16.63.100*

*destination\_port: "22"*

*jump: ACCEPT*

*- name: Permitir conexões SSH do Admin Fofs Notebook*

*iptables:*

*chain: INPUT*

*protocol: tcp*

*source: 172.16.63.103*

*destination\_port: "22"*

*jump: ACCEPT*

*- name: Permitir conexões HTTP para o Mundo*

*iptables:*

*chain: INPUT*

*protocol: tcp*

*destination\_port: "80"*

*jump: ACCEPT*

*- name: Permitir conexões HTTPS para o Mundo*

*iptables:*

*chain: INPUT*

*protocol: tcp*

*destination\_port: "443"*

*jump: ACCEPT*

*- name: Permitir pacotes Related e Established*

*command: iptables -I INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT*

*- name: Definir políticas padrão como DROP*

*iptables:*

*chain: "{{ item }}"*

*policy: DROP*

*with\_items:*

*- INPUT*

*- FORWARD*

*EOF*

*cat <<EOF> srvweb/firewallv2.yml*

*- name: Configuração do Firewall com iptables*

*hosts: srvwebs*

*become: yes*

*tasks:*

*- name: Permitir conexões SSH do Servidor Ansible*

*iptables:*

*chain: INPUT*

*protocol: tcp*

*source: 172.16.63.100*

*destination\_port: "22"*

*jump: ACCEPT*

*- name: Permitir conexões SSH do Admin Fofs Notebook*

*iptables:*

*chain: INPUT*

*protocol: tcp*

*source: 172.16.63.103*

*destination\_port: "22"*

*jump: ACCEPT*

*- name: Permitir conexões HTTP para o Mundo*

*iptables:*

*chain: INPUT*

*protocol: tcp*

*destination\_port: "80"*

*jump: ACCEPT*

*- name: Permitir conexões HTTPS para o Mundo*

*iptables:*

*chain: INPUT*

*protocol: tcp*

*destination\_port: "443"*

*jump: ACCEPT*

*- name: Definir políticas padrão como DROP*

*iptables:*

*chain: "{{ item }}"*

*policy: DROP*

*with\_items:*

*- INPUT*

*- FORWARD*

*- name: Permitir pacotes related e established*

*iptables:*

*chain: INPUT*

*ctstate: "{{ item }}"*

*jump: ACCEPT*

*with\_items:*

*- RELATED*

*- ESTABLISHED*

*EOF*