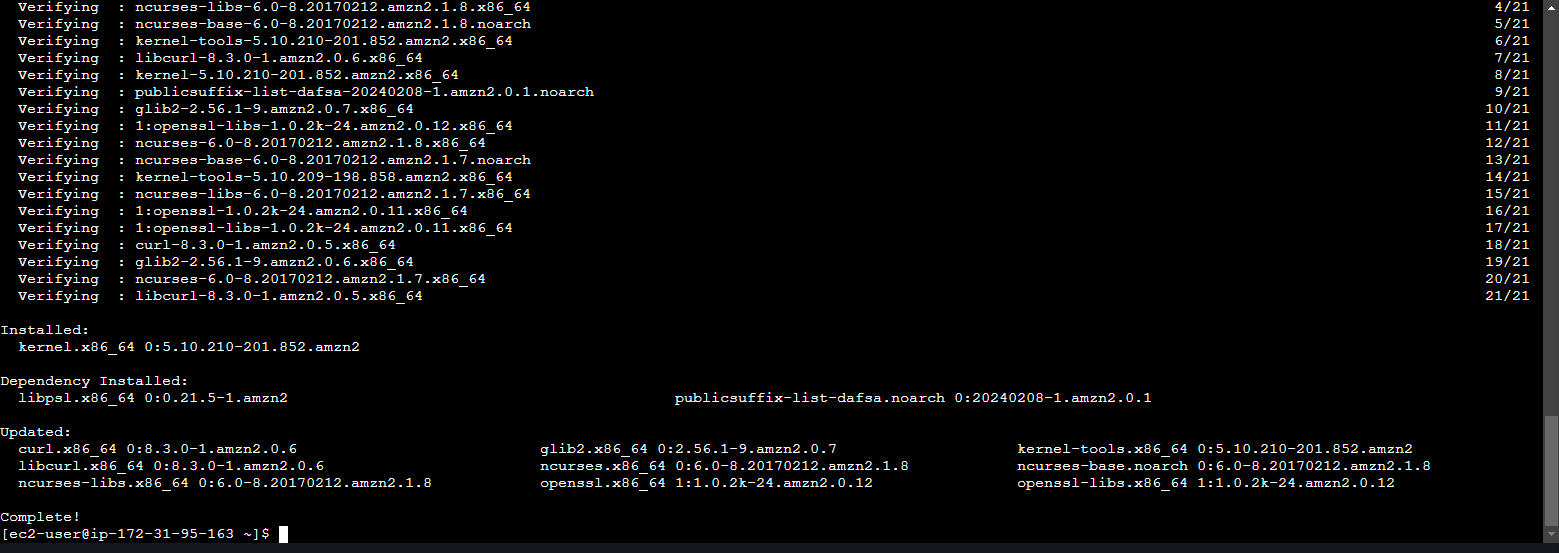
1. **Etape :**

**A. Dans le serveur maître**

* + Update the server

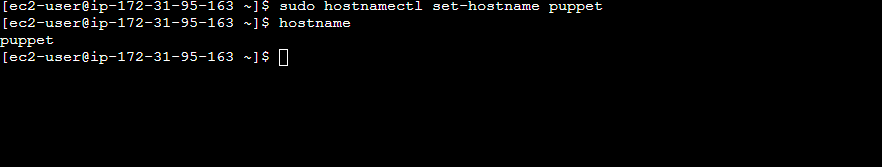
sudo yum update

[](https://github.com/AWS-Re-Start-RDC-KINSHASA-1/pupputJeremy/blob/main/update%20result.png)

 Renommez le nom d’hôte maître en puppet et vérifiez le nouveau nom d’hôte

sudo hostnamectl set-hostname puppet

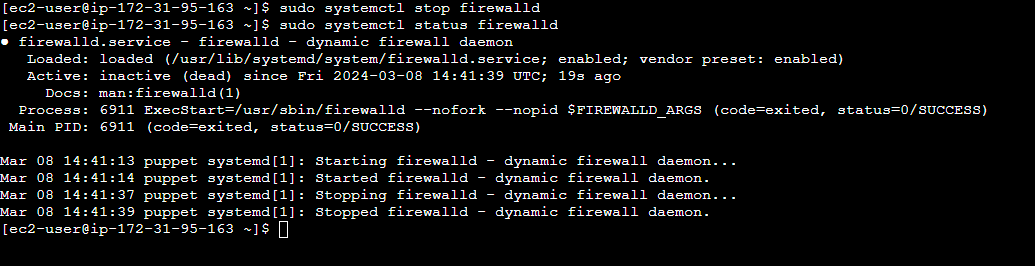
hostname

[](https://github.com/AWS-Re-Start-RDC-KINSHASA-1/pupputJeremy/blob/main/hostname%20result.png)

 Arrêtez tout pare-feu en cours d’exécution si nécessaire et vérifiez l’état du pare-feu

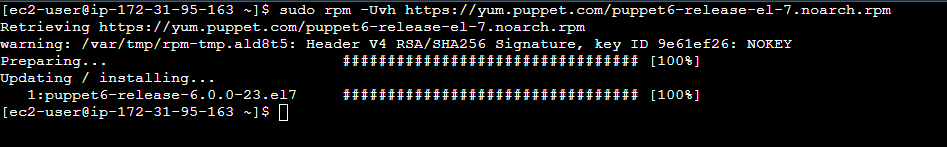
sudo systemctl stop firewalld

sudo systemctl status firewalld

[](https://github.com/AWS-Re-Start-RDC-KINSHASA-1/pupputJeremy/blob/main/firewalld%20status.png)

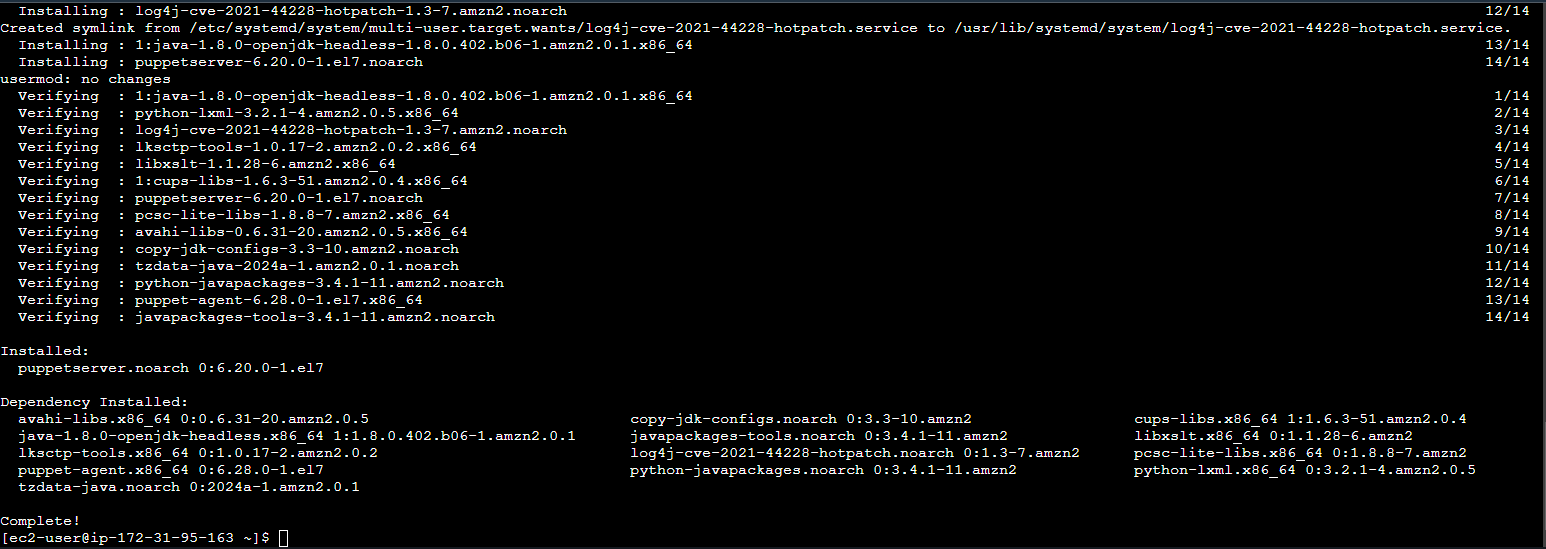
 Télécharger Puppet labs

sudo rpm -Uvh https://yum.puppet.com/puppet6-release-el-7.noarch.rpm

[](https://github.com/AWS-Re-Start-RDC-KINSHASA-1/pupputJeremy/blob/main/dowload%20puppetlabs%20result.png)

* + Installer puppetserver

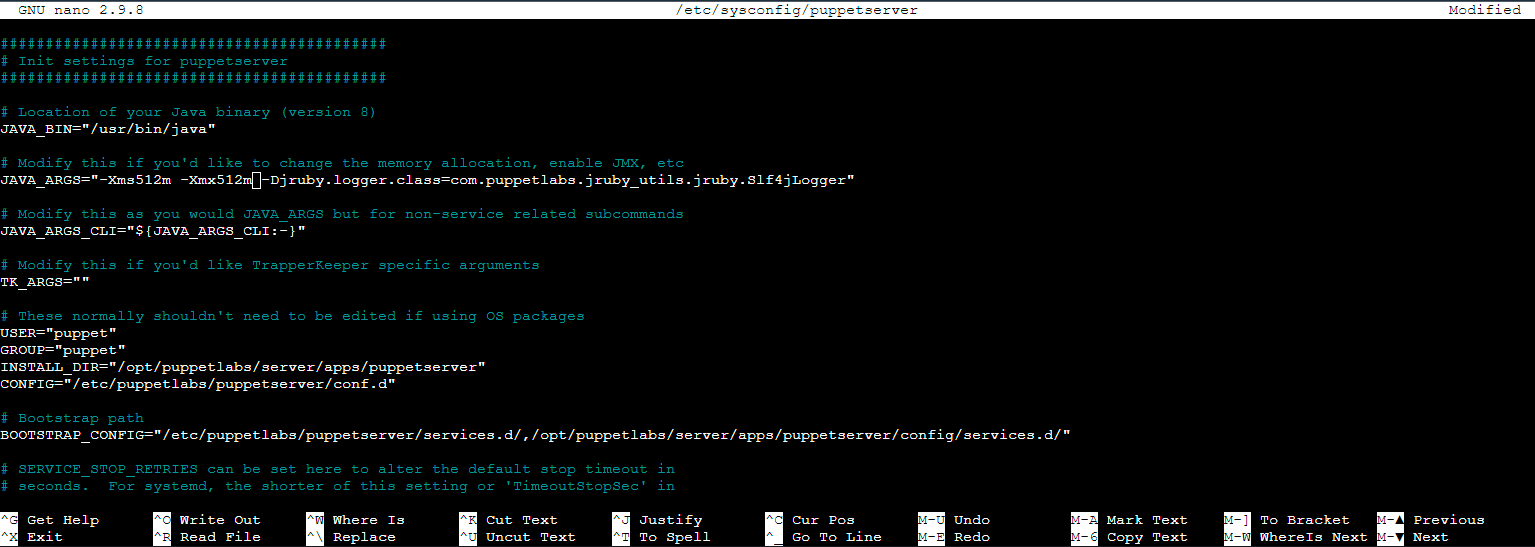
sudo yum install puppetserver -y

[](https://github.com/AWS-Re-Start-RDC-KINSHASA-1/pupputJeremy/blob/main/puppetserver%20install%20result.png)

* + Configurer puppetserver

Ouvrez le fichier de configuration avec nano et changez la taille de la mémoire de 2g à 512m sur « JAVA-AGS »

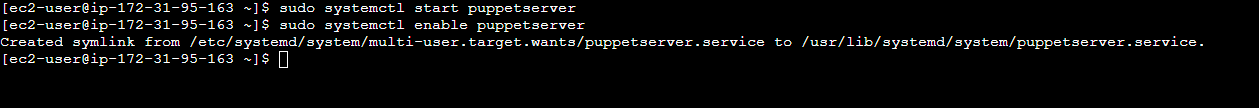
sudo nano /etc/sysconfig/puppetserver

[](https://github.com/AWS-Re-Start-RDC-KINSHASA-1/pupputJeremy/blob/main/configuration%20file.png)  
*Press Ctr + O and Enter and Ctr + x to save and quit*

* + Start et enable puppetserver

sudo systemctl start puppetserver

sudo systemctl enable puppetserver

[](https://github.com/AWS-Re-Start-RDC-KINSHASA-1/pupputJeremy/blob/main/start%20puppetserver.png)

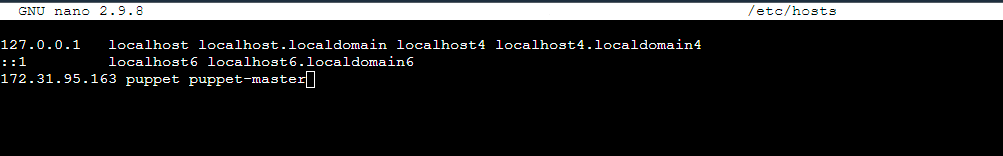
**B. Dans le serveur esclave (client node)**

Note : effectuez les étapes suivantes dans chacun de vos nœuds clients (serveurs esclaves)

 Ajouter l’adresse IP privée du maître dans le fichier de configuration de l’hôte esclave

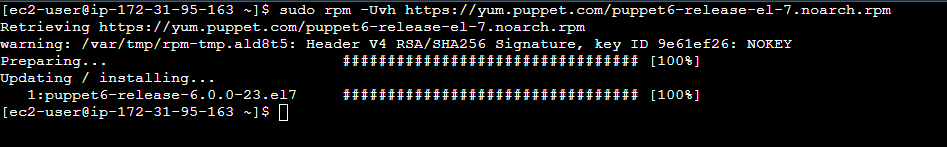
Ouvrez le fichier de configuration avec nano et ajoutez la ligne suivante à la fin du fichier

your\_master\_private\_ip\_address marionnettiste marionnettistesudo nano /etc/hosts

[](https://github.com/AWS-Re-Start-RDC-KINSHASA-1/pupputJeremy/blob/main/pngchange%20host.png)  
*Tape Ctr + O, Enter, Ctr + x to save and quit*

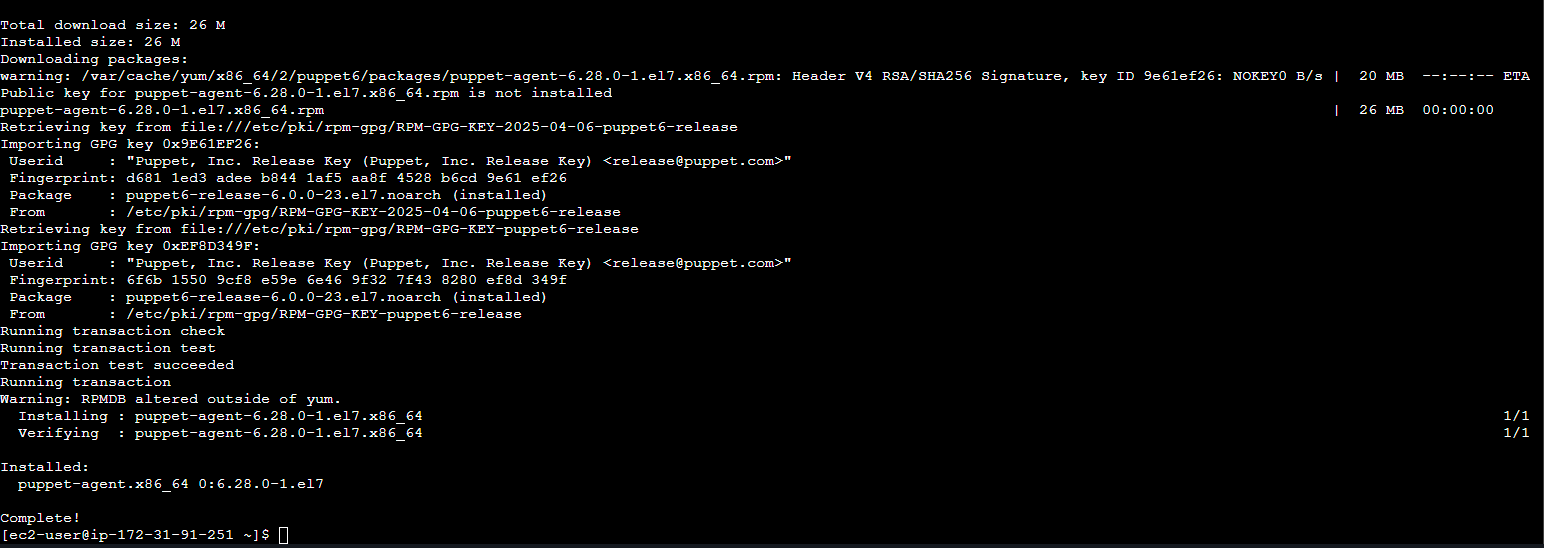
* + Telecharger Puppet labs

sudo rpm -Uvh https://yum.puppet.com/puppet6-release-el-7.noarch.rpm

[](https://github.com/AWS-Re-Start-RDC-KINSHASA-1/pupputJeremy/blob/main/dowload%20puppetlabs%20result.png)

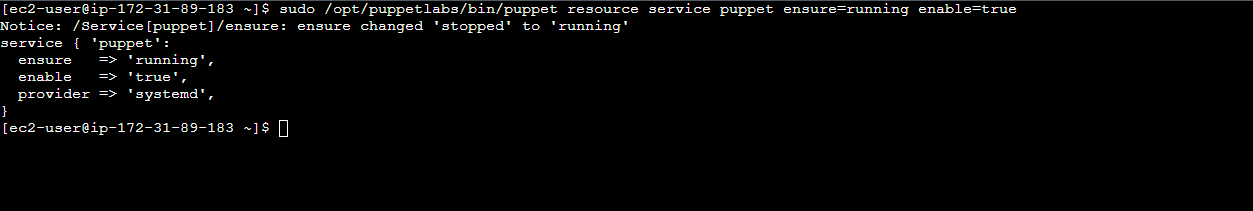
* + Installer puppet agent

sudo yum install -y puppet-agent

[](https://github.com/AWS-Re-Start-RDC-KINSHASA-1/pupputJeremy/blob/main/install%20agent.png)

* + Start puppet-agent

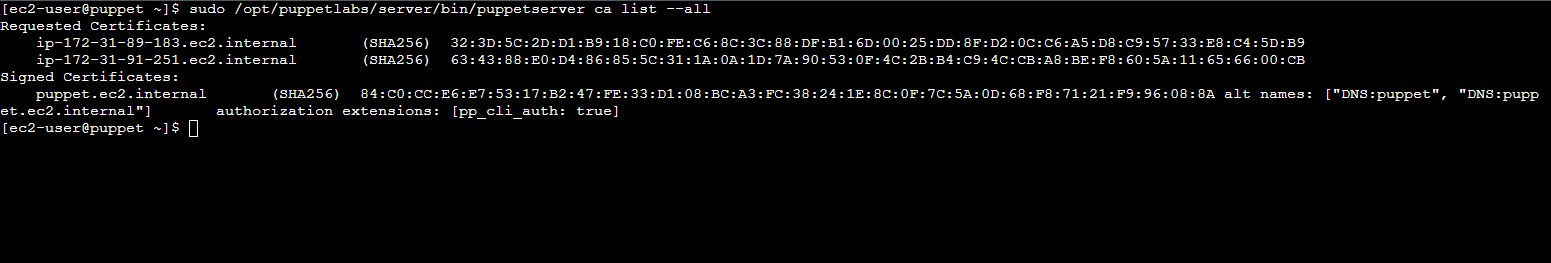
sudo /opt/puppetlabs/bin/puppet resource service puppet ensure=running enable=true

[](https://github.com/AWS-Re-Start-RDC-KINSHASA-1/pupputJeremy/blob/main/start%20puppetagent.png)

**C. Dans le serveur maître**

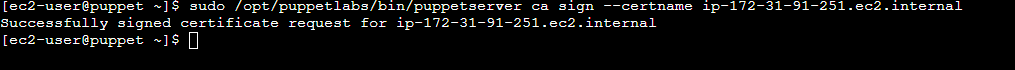
 Affichage de la liste des certificats reçus

sudo /opt/puppetlabs/server/bin/puppetserver ca list --all

[](https://github.com/AWS-Re-Start-RDC-KINSHASA-1/pupputJeremy/blob/main/list%20of%20certificates.png)  
Note : Notre maître a reçu deux certificats car nous avons travaillé avec deux nœuds clients (instances esclaves)

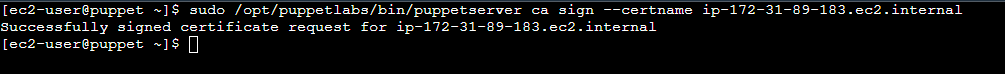
* + Signe certificates
    - Signe le premier client node certificate

sudo /opt/puppetlabs/bin/puppetserver ca sign --certname your\_first\_client\_node\_hostname

[](https://github.com/AWS-Re-Start-RDC-KINSHASA-1/pupputJeremy/blob/main/sign1.png)

* + - Signe le deuxieme client node certificate

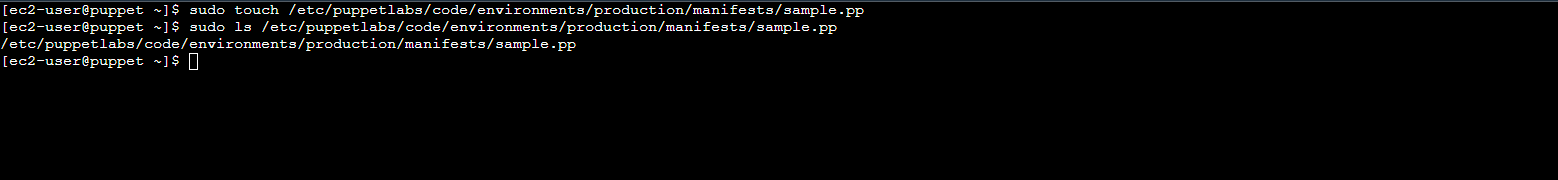
sudo /opt/puppetlabs/bin/puppetserver ca sign --certname your\_second\_client\_node\_hostname

[](https://github.com/AWS-Re-Start-RDC-KINSHASA-1/pupputJeremy/blob/main/sign2.png)

 Créez un fichier manifeste et vérifiez si le fichier a été créé

sudo touch /etc/puppetlabs/code/environments/production/manifests/sample.pp

sudo ls /etc/puppetlabs/code/environments/production/manifests/sample.pp

[](https://github.com/AWS-Re-Start-RDC-KINSHASA-1/pupputJeremy/blob/main/manifest%20created.png)

D. Dans le serveur maître, écrivez la configuration que vous prévoyez d’implémenter et de gérer sur vos serveurs esclaves (clients node)

 Ouvrez le fichier d’exemple de manifeste avec nano

sudo nano /etc/puppetlabs/code/environments/production/manifests/sample.pp

* + Copie les instructions suivantes :

node 'your\_slave1\_hostname', 'your\_slave2\_hostname'

{

#Insure that httpd is installed

package{ 'httpd' :

ensure=> installed

}

file { '/etc/config':

ensure => 'directory',

}

#insure that there is an existing config.txt file

file{

'/etc/config/config.txt':

ensure=>file

}

#Add an exec command to check for the presence of the file

exec { 'check\_config':

command => '/bin/test -f /etc/config/config.txt',

# We use 'onlyif' option to run this command only if config.txt file does not exist

onlyif => '/bin/test ! -f /etc/config/config.txt',

}

#Install Python 3 if not already installed

package { 'python3':

ensure => installed,

}

#Add an exec command to check the installed version of Python

exec { 'check\_python\_version':

command => '/usr/bin/python3 --version',

#We use the 'unless' option to run this command only if Python 3 is not installed

unless => '/usr/bin/python3 --version',

logoutput => true, # Afficher la sortie de la commande dans les logs Puppet

}

#Install Flask with pip if it is not already installed

package { 'flask':

ensure => 'installed',

provider => 'pip3',

require => Package['python3'], # Assurez-vous que Python 3 est installé avant d'installer Flask

}

#Add an exec command to check if Flask is installed

exec { 'check\_flask':

command => '/usr/bin/pip3 show flask',

logoutput => true, # Afficher la sortie de la commande dans les logs Puppet

unless => '/usr/bin/pip3 show flask', # Exécuter cette commande uniquement si Flask n'est pas déjà installé

}

#Install Git if it is not already installed

package { 'git':

ensure => 'installed',

}

#Add an exec command to check if Git is installed

exec { 'check\_git':

command => '/usr/bin/git --version',

logoutput => true, # Afficher la sortie de la commande dans les logs Puppet

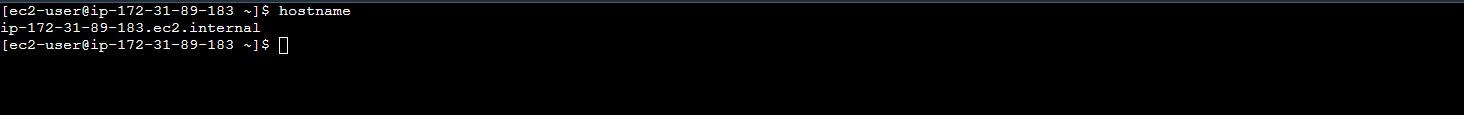
unless => '/usr/bin/git --version', # Exécuter cette commande uniquement si Git n'est pas déjà installé

}

}

Notez que remplacez your\_slave1\_hostname et your\_slave2\_hostname dans les instructions ci-dessus par les noms d’hôtes de vos serveurs esclaves, enregistrez la commande suivante dans vos serveurs esclaves pour obtenir leurs noms d’hôte

hostname

[](https://github.com/AWS-Re-Start-RDC-KINSHASA-1/pupputJeremy/blob/main/hostname%20result2.png)

1. **Running the slaves catalogs**
   * + 1. **Create a scripth.sh file in the Slave Servers home directory**

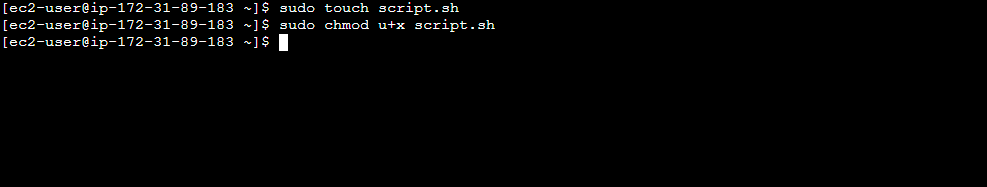
**Remarque : Effectuez les étapes suivantes sur chacun de vos serveurs esclaves**

* + creer un script.sh file

sudo touch script.sh

* + Donner les permissions a ce file

sudo chmod 777 script.sh

[](https://github.com/AWS-Re-Start-RDC-KINSHASA-1/pupputJeremy/blob/main/permission.png)

 Ouvrez le fichier script.sh et collez les instructions suivantes

sudo nano script.sh

*paste the following instructions*

#!/bin/bash

sudo /opt/puppetlabs/bin/puppet agent --test

#Chemin local de votre application

APPLICATION\_PATH="/home/Tp1LoginJeremy"

#URL du dépôt GitHub

GITHUB\_REPO="<https://github.com/AWS-Re-Start-RDC-KINSHASA-1/Tp1LoginJeremy.git>"

#Vérifier si le répertoire de l'application existe

if [ -d "$APPLICATION\_PATH" ]; then

echo "Le répertoire de l'application existe. Effectuer un pull depuis GitHub..."

cd "$APPLICATION\_PATH" || exit

sudo git pull origin main

else

echo "Le répertoire de l'application n'existe pas. Cloner depuis GitHub..."

sudo git clone "$GITHUB\_REPO" "$APPLICATION\_PATH"

fi

#Chemin vers votre application Flask

APPLICATION\_PATH="/home/Tp1LoginJeremy/app.py"

#Vérifier si l'application Flask est déjà en cours d'exécution

if pgrep -f "$APPLICATION\_PATH" > /dev/null; then

echo "L'application Flask est déjà en cours d'exécution."

exit 1

fi

#Démarrer l'application Flask en arrière-plan avec nohup

nohup python3 "$APPLICATION\_PATH" &

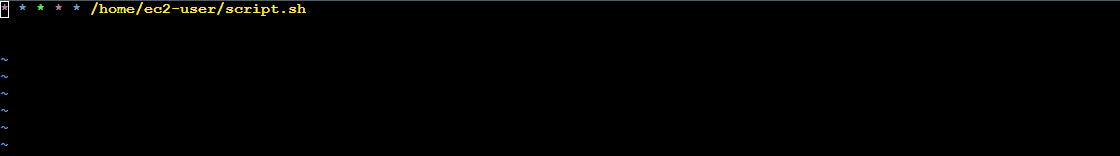
**B. execute le script.sh file avec le crontab**

* + Modifier le crontab

crontab –e

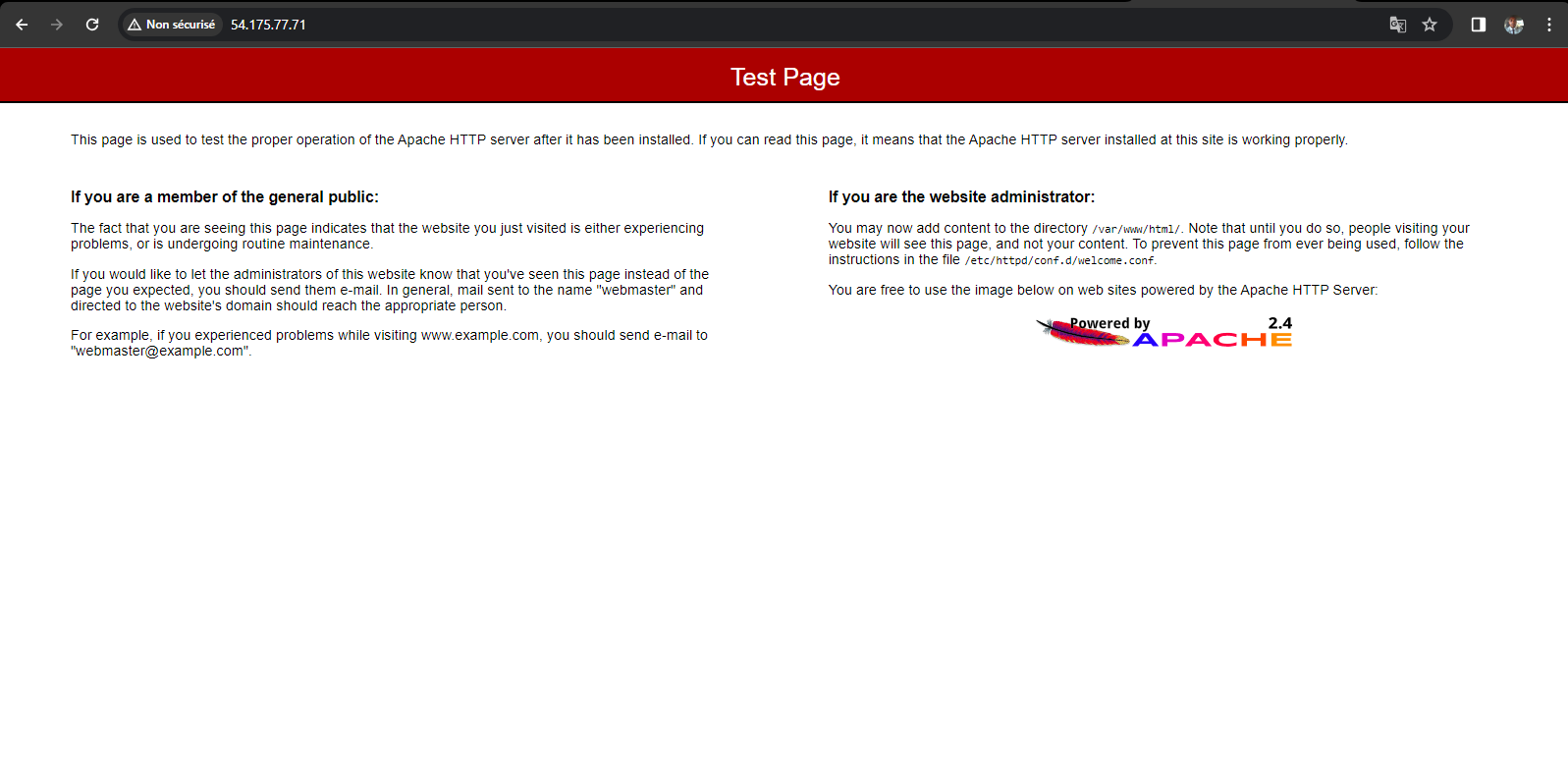
Collez l’instruction suivante pour exécuter votre fichier script.sh toutes les minutes avec un crontab

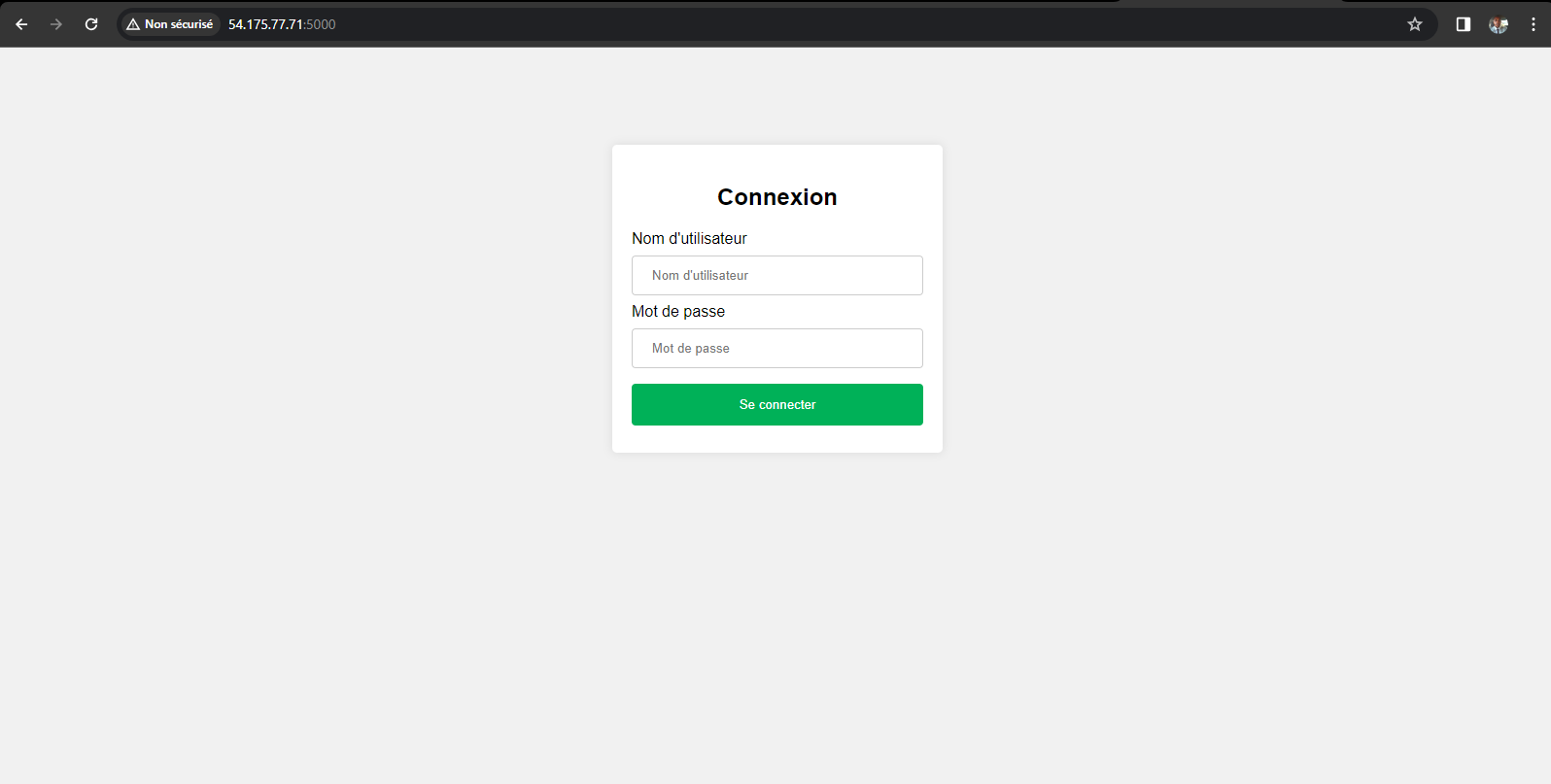
\* \* \* \* \* /home/ec2-user/script.sh

*Note: press i to switch in inserting mode*  
[](https://github.com/AWS-Re-Start-RDC-KINSHASA-1/pupputJeremy/blob/main/crontab.png)

* + To quit crontab editing press :wq and enter
  + Restart your server to make your crontab start

1. Tester le resultat
   * Ouvre le browser
   * Copie ton instance esclave public address

[](https://github.com/AWS-Re-Start-RDC-KINSHASA-1/pupputJeremy/blob/main/apache.png) Remarque : la page par défaut d’Apache s’affiche

*Ajoutez maintenant un port 5000 après l’adresse publique de votre serveuse esclave comme your\_slave\_server\_public\_address :5000Review the flask login page application displayed* [](https://github.com/AWS-Re-Start-RDC-KINSHASA-1/pupputJeremy/blob/main/flaskapp.png)