Lab 2 - Préparez l'installation de Puppet sur la VM puppet

Aperçu

Dans cet atelier, nous effectuerons certaines étapes de pré-installation pour préparer notre VM à prendre Puppet

Démarrez vos VM de formation

Si ce n'est déjà fait ...

```
vagrant up puppet
```

Si vous ne connaissez pas l'état de vos VM contrôlées par vagrant, vous pouvez toujours vérifier avec ...

```
$ vagrant global-status
id       name          provider   state    directory
-----------------------------------------------------------------------
4dd5ed7  puppet        virtualbox running  ~/puppet-training
070258c  gitlab        virtualbox poweroff ~/puppet-training
682eafe  agent         virtualbox poweroff ~/puppet-training
```

Notez que la VM puppet est en cours d'exécution, mais les VM gitlab et agent sont éteintes.

Connectez-vous à votre machine virtuelle de formation

```
$ vagrant ssh puppet
```

Une fois que vous êtes connecté à la VM **puppet**, notez que le prompt du shell bash ressemble à ceci:

```
[vagrant@puppet ~]$
```

Devenez root ...

```
[vagrant@puppet ~]$ sudo su -
```

... et notez que l'invite du shell change en:

```
Last login: Fri Oct 21 15:37:10 UTC 2020 on pts/0
[root@puppet ~]#
```

Maintenant, quittez votre shell root, et revenez au shell de l'utilisateur vagrant ...

```
[root@puppet ~]# exit
logout
[vagrant@puppet ~]$
```

Tout au long des travaux pratiques, vous devrez exécuter de nombreuses commandes (la plupart) en tant que root.

Étapes de pré-installation

Il y a quelques choses que nous devons faire pour que notre VM soit prête à recevoir Puppet:

- Ajouter quelques entrées au fichier /etc/hosts (si ce n'est pas déjà fait)
- Ouvrir certains ports via le pare-feu hôte

Modifier /etc/hosts

Le fichier `/etc/hosts` devrait déjà être configuré correctement, mais si pour une raison quelconque vous trouvez que ce n'est pas le cas, allez-y et modifiez-le comme suit.

Modifiez `/etc/hosts` et ajoutez des entrées pour localhost, ainsi que nos 3 machines virtuelles du lab, les deux noms long (FQDN) et court.

```
sudo vi /etc/hosts
```

Nous voulons que notre fichier `/etc/hosts` ressemble à ceci:

```
127.0.0.1      localhost
192.168.198.10 puppet.example.com puppet
192.168.198.11 agent.example.com  agent
192.168.198.12 gitlab.example.com gitlab
```

Dans un lab ultérieur, nous configurerons Puppet pour gérer les entrées /etc/hosts pour nous.

Configurer le pare-feu

Configurer le pare-feu hôte pour permettre à Puppet de fonctionner conformément au PE Install Guide - Firewall Config

```
sudo su -
firewall-cmd --permanent --add-service=https   # PE Console (default port 443)
firewall-cmd --permanent --add-port=3000/tcp   # PE web-based installer
firewall-cmd --permanent --add-port=8080/tcp   # PuppetDB
firewall-cmd --permanent --add-port=8081/tcp   # PuppetDB
firewall-cmd --permanent --add-port=8140/tcp   #Puppet Master,Certificate Authority
firewall-cmd --permanent --add-port=8142/tcp   # Orchestration services
firewall-cmd --permanent --add-port=8143/tcp   # The Orchestrator client uses this
port to communicate with the orchestration services
firewall-cmd --permanent --add-port=61613/tcp  # MCollective / ActiveMQ
firewall-cmd --permanent --add-port=4432/tcp   # Local connections for node
classifier, activity service, and RBAC status checks
firewall-cmd --permanent --add-port=4433/tcp   #Classifier/Console Services API endpoint
firewall-cmd --permanent --add-port=8170/tcp   # Code Manager
firewall-cmd --reload
firewall-cmd --list-all
exit # drop out of root shell
```

La sortie de votre `firewall-cmd --list-all` devrait ressembler à ceci:

```
[vagrant@puppet ~]$ firewall-cmd --list-all
public (default, active)
  interfaces: enp0s3 enp0s8
  sources:
  services: dhcpv6-client https ssh
  ports: 3000/tcp 8140/tcp 8170/tcp 8080/tcp 4433/tcp 8081/tcp
4432/tcp 8143/tcp 8142/tcp 61613/tcp
  masquerade: no
  forward-ports:
  icmp-blocks:
  rich rules:
```

D'accord, nous sommes maintenant prêts à exécuter le programme d'installation de Puppet ...