Version : **2022.01**

Dernière mise-à-jour : 2022/06/20 17:08

DOF402 - Les Ressources Puppet

Contenu du Module

        - L'Attribut weekday
        - L'Attribut monthday
        - La Fonction fqdn_rand
    - LAB #6 - La Ressource exec
        - L'Attribut exec
        - L'Attribut cwd
        - L'Attribut command
        - L'Attribut creates
        - L'Attribut user
        - L'Attribut onlyif
        - L'Attribut unless
        - L'Attribut refreshonly
        - L'Attribut logoutput
        - L'Attribut timeout

LAB #1 - La Ressource File

L'Attribut Source

Créez le fichier **file_source.pp** :

```
vagrant@ubuntu-xenial:~$ sudo vi file_source.pp
vagrant@ubuntu-xenial:~$ cat file_source.pp
file { '/etc/motd':
  source => '/home/vagrant/files/motd.txt',
}
```

Creéz le répertoire **files** ainsi que le fichier **files/motd.txt** :

```
vagrant@ubuntu-xenial:~$ sudo mkdir files
vagrant@ubuntu-xenial:~$ sudo vi files/motd.txt
vagrant@ubuntu-xenial:~$ cat files/motd.txt
```

```
The best software in the world only sucks. The worst software is significantly worse than that.
-Luke Kanies
```

```
vagrant@ubuntu-xenial:~$ sudo puppet apply file_source.pp
Notice: Compiled catalog for ubuntu-xenial in environment production in 0.01 seconds
Notice: /Stage[main]/Main/File[/etc/motd]/ensure: defined content as '{md5}2400ab50861ff84ea5a3e43126d162d5'
Notice: Applied catalog in 0.04 seconds
vagrant@ubuntu-xenial:~$ cat /etc/motd
The best software in the world only sucks. The worst software is significantly worse than that.
-Luke Kanies
```

> **Important** - Notez que le contenu du fichier **/etc/motd est** le contenu du fichier **/home/vagrant/files/motd.txt** indiqué par l'attribut source.

L'attribut source peut aussi utiliser la valeur d'un URL. Créez le fichier **file_http.pp** :

```
vagrant@ubuntu-xenial:~$ sudo vi file_http.pp
vagrant@ubuntu-xenial:~$ cat file_http.pp
file { '/tmp/README.md':
  source => 'https://raw.githubusercontent.com/puppetlabs/puppet/master/README.md',
}
```

Appliquez le manifest :

```
vagrant@ubuntu-xenial:~$ sudo vi file_http.pp
vagrant@ubuntu-xenial:~$ sudo puppet apply file_http.pp
Notice: Compiled catalog for ubuntu-xenial in environment production in 0.01 seconds
Notice: /Stage[main]/Main/File[/tmp/README.md]/ensure: defined content as '{mtime}2020-02-11 10:17:59 +0000'
Notice: Applied catalog in 0.88 seconds

vagrant@ubuntu-xenial:~$ cat /tmp/README.md
```

```
Puppet
======

[![Travis
Status](https://travis-ci.com/puppetlabs/puppet.svg?branch=master)](https://travis-ci.com/puppetlabs/puppet)
[![Appveyor
Status](https://ci.appveyor.com/api/projects/status/cvhpypd4504sevqq/branch/master?svg=true)](https://ci.appveyor
.com/project/puppetlabs/puppet/branch/master)
[![Gem Version](https://badge.fury.io/rb/puppet.svg)](https://badge.fury.io/rb/puppet)
[![Inline docs](https://inch-ci.org/github/puppetlabs/puppet.svg)](https://inch-ci.org/github/puppetlabs/puppet)

Puppet, an automated administrative engine for your Linux, Unix, and Windows systems, performs
administrative tasks (such as adding users, installing packages, and updating server
configurations) based on a centralized specification.

Documentation
-------------

Documentation for Puppet and related projects can be found online at the
[Puppet Docs site](https://puppet.com/docs).

...
```

L'Attribut owner

Dans l'exemple ci-dessus, l'utilisateur root a été utilisé pour exécuter les commandes puppet. Pour cette raison, les fichiers ainsi créés appartiennent à l'utilisateur **root** et sont associés avec le groupe **root** :

```
vagrant@ubuntu-xenial:~$ ls -l /etc/motd
-rw-r--r-- 1 root root 109 Feb 11 10:16 /etc/motd
```

L'appartenance d'un fichier peut être fixé grâce à l'utilisation de l'attribut **owner**. Créez le fichier **file_owner.pp** :

```
vagrant@ubuntu-xenial:~$ sudo vi file_owner.pp
vagrant@ubuntu-xenial:~$ cat file_owner.pp
file { '/etc/owned_by_ubuntu':
  ensure => present,
  owner  => 'ubuntu',
}
```

Appliquez ce manifest avec puppet :

```
vagrant@ubuntu-xenial:~$ sudo puppet apply file_owner.pp
Notice: Compiled catalog for ubuntu-xenial in environment production in 0.01 seconds
Notice: /Stage[main]/Main/File[/etc/owned_by_ubuntu]/ensure: created
Notice: Applied catalog in 0.01 seconds
```

Vérifiez maintenant le propriétaire du fichier **/etc/owned_by_ubuntu** :

```
vagrant@ubuntu-xenial:~$ ls -l /etc/owned_by_ubuntu
-rw-r--r-- 1 ubuntu root 0 Feb 11 10:20 /etc/owned_by_ubuntu
```

L'Attribut group

Le groupe d'un fichier peut être fixé grâce à l'utilisation de l'attribut **group**. Créez le fichier **file_group.pp** :

```
vagrant@ubuntu-xenial:~$ sudo vi file_group.pp
vagrant@ubuntu-xenial:~$ cat file_group.pp
file { '/etc/owned_by_ubuntu':
  ensure => present,
  owner  => 'ubuntu',
  group  => 'ubuntu',
}
```

Appliquez ce manifest avec puppet :

```
vagrant@ubuntu-xenial:~$ sudo puppet apply file_group.pp
Notice: Compiled catalog for ubuntu-xenial in environment production in 0.01 seconds
Notice: /Stage[main]/Main/File[/etc/owned_by_ubuntu]/group: group changed 'root' to 'ubuntu'
Notice: Applied catalog in 0.01 seconds
```

Vérifiez maintenant le groupe du fichier **/etc/owned_by_ubuntu** :

```
vagrant@ubuntu-xenial:~$ ls -l /etc/owned_by_ubuntu
-rw-r--r-- 1 ubuntu ubuntu 0 Feb 11 10:20 /etc/owned_by_ubuntu
```

L'Attribut mode

Les permission d'un fichier peuvent être fixées grâce à l'utilisation de l'attribut **mode**. Créez le fichier **file_mode.pp** :

```
vagrant@ubuntu-xenial:~$ sudo vi file_mode.pp
vagrant@ubuntu-xenial:~$ cat file_mode.pp
file { '/etc/owned_by_ubuntu':
  ensure => present,
  owner  => 'ubuntu',
  mode   => '0600',
}
```

Appliquez ce manifest avec puppet :

```
vagrant@ubuntu-xenial:~$ sudo puppet apply file_mode.pp
Notice: Compiled catalog for ubuntu-xenial in environment production in 0.01 seconds
Notice: /Stage[main]/Main/File[/etc/owned_by_ubuntu]/mode: mode changed '0644' to '0600'
Notice: Applied catalog in 0.01 seconds
```

Vérifiez maintenant les permissions du fichier **/etc/owned_by_ubuntu** :

```
vagrant@ubuntu-xenial:~$ ls -l /etc/owned_by_ubuntu
```

```
-rw------- 1 ubuntu ubuntu 0 Feb 11 10:20 /etc/owned_by_ubuntu
```

L'Attribut ensure

Si la valeur de l'attribut **ensure** est **directory**, la ressource **file** fait référence à un répertoire et non à un fichier :

```
vagrant@ubuntu-xenial:~$ sudo vi file_directory.pp
vagrant@ubuntu-xenial:~$ cat file_directory.pp
file { '/etc/config_dir':
  ensure => directory,
}
```

Appliquez ce manifest avec puppet :

```
vagrant@ubuntu-xenial:~$ sudo puppet apply file_directory.pp
Notice: Compiled catalog for ubuntu-xenial in environment production in 0.01 seconds
Notice: /Stage[main]/Main/File[/etc/config_dir]/ensure: created
Notice: Applied catalog in 0.01 seconds
```

Vérifiez maintenant la création du répertoire **/etc/config_dir** :

```
vagrant@ubuntu-xenial:~$ ls -l /etc | grep config_
drwxr-xr-x 2 root   root    4096 Feb 11 10:36 config_dir
```

Si la valeur de l'attribut **ensure** est **link**, la ressource **file** fait référence à **lien symbolique** :

```
vagrant@ubuntu-xenial:~$ sudo vi file_symlink.pp
vagrant@ubuntu-xenial:~$ cat file_symlink.pp
file { '/etc/this_is_a_link':
  ensure => link,
  target => '/etc/motd',
}
```

Appliquez ce manifest avec puppet :

```
vagrant@ubuntu-xenial:~$ sudo puppet apply file_symlink.pp
Notice: Compiled catalog for ubuntu-xenial in environment production in 0.01 seconds
Notice: /Stage[main]/Main/File[/etc/this_is_a_link]/ensure: created
Notice: Applied catalog in 0.01 seconds
```

Vérifiez maintenant la création du lien :

```
vagrant@ubuntu-xenial:~$ ls -l /etc/this_is_a_link
lrwxrwxrwx 1 root root 9 Feb 11 10:37 /etc/this_is_a_link -> /etc/motd
```

L'Attribut recurse

Cet attribut permet de copier une arborescence complète. Créez le répertoire **files/config** contenant trois fichiers **a**, **b** et **c** :

```
vagrant@ubuntu-xenial:~$ sudo mkdir files/config
vagrant@ubuntu-xenial:~$ sudo touch files/config/a files/config/b files/config/c
```

Creéz le manifest **file_tree.pp** :

```
vagrant@ubuntu-xenial:~$ sudo vi file_tree.pp
vagrant@ubuntu-xenial:~$ cat file_tree.pp
file { '/etc/config_dir':
  source  => '/home/vagrant/files/config',
  recurse => true,
}
```

Appliquez le manifest :

```
vagrant@ubuntu-xenial:~$ sudo puppet apply file_tree.pp
Notice: Compiled catalog for ubuntu-xenial in environment production in 0.01 seconds
```

```
Notice: /Stage[main]/Main/File[/etc/config_dir/a]/ensure: defined content as
'{md5}d41d8cd98f00b204e9800998ecf8427e'
Notice: /Stage[main]/Main/File[/etc/config_dir/b]/ensure: defined content as
'{md5}d41d8cd98f00b204e9800998ecf8427e'
Notice: /Stage[main]/Main/File[/etc/config_dir/c]/ensure: defined content as
'{md5}d41d8cd98f00b204e9800998ecf8427e'
Notice: Applied catalog in 0.04 seconds
```

Vérifiez maintenant le contenu du répertoire **/etc/config_dir** :

```
vagrant@ubuntu-xenial:~$ ls -lR /etc/config_dir
/etc/config_dir:
total 0
-rw-r--r-- 1 root root 0 Feb 11 10:38 a
-rw-r--r-- 1 root root 0 Feb 11 10:38 b
-rw-r--r-- 1 root root 0 Feb 11 10:38 c
```

> **Important** - Si le répertoire cible existe et il contient déjà des fichiers, Puppet ne fera rien. Ceci peut être modifié en utilisant l'attribut **purge** au quel cas Puppet supprimera les fichiers et sous-répertoires dans le répertoire cible qui ne sont pas présents dans le répertoire source.

LAB #2 - La Ressource package

L'Attribut ensure

Installation de Paquets

Normalement, la valeur de cet attribut est **installed**, comme vu dans l'exemple précédent :

```
vagrant@ubuntu-xenial:~$ cat package.pp
package { 'cowsay':
  ensure => installed,
}
```

Il est cependant possible de spécifier une version spécifique d'un paquet, comme démontre l'exemple ci-dessous :

```
package { 'openssl':
  ensure => '1.0.2g-1ubuntu4.8',
}
```

> **Important** - Si de multiple versions d'un paquet existe, en spécifiant **ensure ⇒ to install** Puppet installera la version par défaut. Par contre, en spécifiant **ensure ⇒ latest**, Puppet installera la version la plus récente.

Le Cas Spécifique des Gems de Ruby

Installez Ruby :

```
vagrant@ubuntu-xenial:~$ sudo vi package_gem.pp
vagrant@ubuntu-xenial:~$ cat package_gem.pp
package { 'ruby':
  ensure => installed,
}

package { 'puppet-lint':
  ensure   => installed,
  provider => gem,
}
```

Important - Un Ruby Gem est une bibliothèque Ruby qui étend les fonctionalités de ce premier. Dans le fichier précédent, **puppet-lint** est un Gem et ne peut pas être installé par le gestionnaire de paquet. Puppet est informé de ceci par la ligne **provider ⇒ gem**.

Appliquez le manifest :

```
vagrant@ubuntu-xenial:~$ sudo puppet apply package_gem.pp
Notice: Compiled catalog for ubuntu-xenial in environment production in 0.46 seconds
Notice: /Stage[main]/Main/Package[ruby]/ensure: created
Notice: /Stage[main]/Main/Package[puppet-lint]/ensure: created
Notice: Applied catalog in 5.58 seconds
```

L'outil **puppet-lint** permet de vérifier les manifests pour des erreurs, par exemple :

```
vagrant@ubuntu-xenial:~$ sudo vi lint_test.pp
vagrant@ubuntu-xenial:~$ cat lint_test.pp
file { '/tmp/lint.txt':
  ensure => file,
  content => "puppet-lint is your friend\n",
}

vagrant@ubuntu-xenial:~$ sudo puppet-lint lint_test.pp
WARNING: indentation of => is not properly aligned (expected in column 11, but found it in column 10) on line 2
```

Bien que Puppet soit partielleemnt écrit en Ruby, il ne peut pas utiliser des Gems du **provider ⇒ gem**. En effet, il convient d'utiliser le **provider ⇒ puppet_gem** afin d'installer un Gem dans le contexte de Puppet et non dans le contxet du système :

```
vagrant@ubuntu-xenial:~$ sudo vi package_puppet_gem.pp
vagrant@ubuntu-xenial:~$ cat package_puppet_gem.pp
package { 'r10k':
  ensure   => installed,
  provider => puppet_gem,
```

```
}
vagrant@ubuntu-xenial:~$ sudo puppet apply package_puppet_gem.pp
Notice: Compiled catalog for ubuntu-xenial in environment production in 0.47 seconds
Notice: Applied catalog in 0.17 seconds
```

Pour vérifier si le Gem a été installé dans le contexte de Puppet, utilisez la commande suivante :

```
vagrant@ubuntu-xenial:~$ sudo /opt/puppetlabs/puppet/bin/gem list | grep r10k
r10k (3.4.0)
```

Suppression de Paquets

Il est aussi possible de désinstaller un paquet :

```
vagrant@ubuntu-xenial:~$ sudo vi package_remove.pp
vagrant@ubuntu-xenial:~$ cat package_remove.pp
package { 'apparmor':
  ensure => absent,
}

vagrant@ubuntu-xenial:~$ dpkg -l | grep apparmor
ii  apparmor                         2.10.95-0ubuntu2.11                     amd64        user-space parser
utility for AppArmor
ii  libapparmor-perl                 2.10.95-0ubuntu2.11                     amd64        AppArmor library
Perl bindings
ii  libapparmor1:amd64               2.10.95-0ubuntu2.11                     amd64        changehat AppArmor
library
```

Appliquez les manifest avec Puppet :

```
vagrant@ubuntu-xenial:~$ sudo puppet apply package_remove.pp
Notice: Compiled catalog for ubuntu-xenial in environment production in 0.47 seconds
```

```
Notice: /Stage[main]/Main/Package[apparmor]/ensure: removed
Notice: Applied catalog in 2.48 seconds

vagrant@ubuntu-xenial:~$ dpkg -l | grep apparmor
rc  apparmor                        2.10.95-0ubuntu2.11                    amd64        user-space parser
utility for AppArmor
ii  libapparmor-perl                2.10.95-0ubuntu2.11                    amd64        AppArmor library
Perl bindings
ii  libapparmor1:amd64              2.10.95-0ubuntu2.11                    amd64        changehat AppArmor
library
```

> **Important** - Par défaut, quand Puppet supprime un paquet, il laisse les fichiers associés avec le paquet en place. Pour purger ces fichiers, il convient d'utiliser **purge**.

LAB #3 - La Ressource service

Puppet a besoin de vérifier si un service a été démarré :

```
vagrant@ubuntu-xenial:~$ sudo vi service_hasstatus.pp
vagrant@ubuntu-xenial:~$ cat service_hasstatus.pp
package { 'ntp':
  ensure => installed,
}

service { 'ntp':
  ensure    => running,
  enable    => true,
  hasstatus => false,
}
```

L'Attribut hasstatus

La façon que cette vérification a lieu dépend du système d'exploitation du nœud. Par exemple dans le cas d'Ubuntu 16.04, Puppet utilisera la commande **systemctl**.

Dans certains cas Puppet essayera de démarrer un service chaque fois qu'il est exécuté. Ceci implique en règle générale que Puppet ne dispose pas de façon à vérifier si le service a déjà démarré. Dans ce cas, il convient d'utiliser l'attribut **hasstatus ⇒ false**. Dans ce cas, Puppet n'utilise pas les outils du système d'exploitation mais regarde dans le tableau des processus pour un processus dont le nom est le même que le service :

```
vagrant@ubuntu-xenial:~$ sudo puppet apply service_hasstatus.pp
Notice: Compiled catalog for ubuntu-xenial in environment production in 0.68 seconds
Notice: /Stage[main]/Main/Package[ntp]/ensure: created
Notice: Applied catalog in 3.07 seconds

vagrant@ubuntu-xenial:~$ sudo systemctl status ntp
● ntp.service - LSB: Start NTP daemon
   Loaded: loaded (/etc/init.d/ntp; bad; vendor preset: enabled)
   Active: active (running) since Tue 2020-02-11 11:22:07 UTC; 32s ago
     Docs: man:systemd-sysv-generator(8)
   CGroup: /system.slice/ntp.service
           └─2732 /usr/sbin/ntpd -p /var/run/ntpd.pid -g -u 112:116

Feb 11 11:22:12 ubuntu-xenial ntpd[2732]: Soliciting pool server 37.187.122.11
Feb 11 11:22:12 ubuntu-xenial ntpd[2732]: Soliciting pool server 91.189.91.157
Feb 11 11:22:13 ubuntu-xenial ntpd[2732]: Soliciting pool server 91.189.89.199
Feb 11 11:22:13 ubuntu-xenial ntpd[2732]: Soliciting pool server 79.143.250.119
Feb 11 11:22:13 ubuntu-xenial ntpd[2732]: Soliciting pool server 95.81.173.8
Feb 11 11:22:14 ubuntu-xenial ntpd[2732]: Soliciting pool server 37.187.174.185
Feb 11 11:22:14 ubuntu-xenial ntpd[2732]: Soliciting pool server 2001:bc8:30d3:ff00::2
Feb 11 11:22:15 ubuntu-xenial ntpd[2732]: Soliciting pool server 212.83.158.83
Feb 11 11:22:16 ubuntu-xenial ntpd[2732]: Soliciting pool server 162.159.200.123
Feb 11 11:22:17 ubuntu-xenial ntpd[2732]: Soliciting pool server 51.15.203.2
```

L'Attribut pattern

Dans le cas où le processus n'a pas le même nom que le processus, il convient d'utilise l'attribut **pattern** pour aider Puppet a trouvé la correspondance :

```
vagrant@ubuntu-xenial:~$ ps aux | grep ntp
ntp       2732  0.0  0.4 110032  5056 ?        Ssl  11:22   0:00 /usr/sbin/ntpd -p /var/run/ntpd.pid -g -u
112:116
vagrant   3185  0.0  0.0  12940   980 pts/0    S+   11:23   0:00 grep --color=auto ntp
```

Par exemple :

```
service { 'ntp':
  ensure    => running,
  enable    => true,
  hasstatus => false,
  pattern   => 'ntpd',
}
```

Les Attributs hasrestart et restart

Par défaut, quand une ressource **file** utilise l'attribut **notify** pour informer Puppet que son fichier de configuration a été modifié, Puppet arrête puis démarre le service.

Cependant, certains services acceptent la commande **restart**. Dans ce cas, il convient d'informer Puppet en incluant dans le manifest l'attribut **hasrestart** :

```
vagrant@ubuntu-xenial:~$ sudo vi service_hasrestart.pp
vagrant@ubuntu-xenial:~$ cat service_hasrestart.pp
service { 'ntp':
  ensure     => running,
  enable     => true,
```

```
  hasrestart => true,
}

vagrant@ubuntu-xenial:~$ sudo puppet apply service_hasrestart.pp
Notice: Compiled catalog for ubuntu-xenial in environment production in 0.21 seconds
Notice: Applied catalog in 0.03 seconds
```

Par contre, que se passe-t-il si la commande restart par défaut du système d'exploitation ne fonctionne pas ? Dans ce cas, il est possible de stipulé une commande avec l'attribut **restart** :

```
service { 'ntp':
  ensure  => running,
  enable  => true,
  restart => '/bin/echo Restarting >>/tmp/debug.log && systemctl restart ntp',
}
```

LAB #4 - La Ressource user

Un utilisateur est un objet qui peut :

- posséder des fichiers,
- exécuter des commandes,
- éventuellement peut lire ou modifier les fichiers d'autres utilisateurs.

Créer un Utilisateur

Créez le fichier **user.pp** :

```
vagrant@ubuntu-xenial:~$ sudo vi user.pp
vagrant@ubuntu-xenial:~$ cat user.pp
group { 'devs':
```

```
  ensure => present,
  gid    => 3000,
}

user { 'trainee':
  ensure => present,
  uid    => '3002',
  home   => '/home/trainee',
  shell  => '/bin/bash',
  groups => ['devs'],
}
```

Appliquez ce manifest avec puppet :

```
vagrant@ubuntu-xenial:~$ sudo puppet apply user.pp
Notice: Compiled catalog for ubuntu-xenial in environment production in 0.01 seconds
Notice: /Stage[main]/Main/Group[devs]/ensure: created
Notice: /Stage[main]/Main/User[trainee]/ensure: created
Notice: Applied catalog in 0.04 seconds
```

Important - Le type de la ressource est le login de l'utilisateur spécifié par l'attribut **user**. Notez que l'UID est spécifié de façon à ce que l'utilisateur possède le même UID sur tous les nœuds.

Contrôlez la présence de **trainee** dans le fichier **/etc/passwd** :

```
vagrant@ubuntu-xenial:~$ tail /etc/passwd
messagebus:x:107:111::/var/run/dbus:/bin/false
uuidd:x:108:112::/run/uuidd:/bin/false
dnsmasq:x:109:65534:dnsmasq,,,:/var/lib/misc:/bin/false
sshd:x:110:65534::/var/run/sshd:/usr/sbin/nologin
pollinate:x:111:1::/var/cache/pollinate:/bin/false
```

```
vagrant:x:1000:1000:,,,:/home/vagrant:/bin/bash
ubuntu:x:1001:1001:Ubuntu:/home/ubuntu:/bin/bash
vboxadd:x:999:1::/var/run/vboxadd:/bin/false
ntp:x:112:116::/home/ntp:/bin/false
trainee:x:3002:3002::/home/trainee:/bin/bash
```

Notez cependant que le répertoire personnel n'a pas été créé :

```
vagrant@ubuntu-xenial:~$ ls -l /home
total 8
drwxr-xr-x 3 ubuntu  ubuntu  4096 Feb 11 08:30 ubuntu
drwxr-xr-x 7 vagrant vagrant 4096 Feb 11 12:42 vagrant
```

Supprimer un Utilisateur

Créez le fichier **user_remove.pp** :

```
vagrant@ubuntu-xenial:~$ sudo vi user_remove.pp
vagrant@ubuntu-xenial:~$ cat user_remove.pp
user { 'trainee':
  ensure => absent,
}
```

Appliquez ce manifest avec puppet :

```
vagrant@ubuntu-xenial:~$ sudo puppet apply user_remove.pp
Notice: Compiled catalog for ubuntu-xenial in environment production in 0.01 seconds
Notice: /Stage[main]/Main/User[trainee]/ensure: removed
Notice: Applied catalog in 0.05 seconds
```

Contrôlez la présence de **trainee** dans le fichier **/etc/passwd** :

```
vagrant@ubuntu-xenial:~$ tail /etc/passwd
lxd:x:106:65534::/var/lib/lxd/:/bin/false
messagebus:x:107:111::/var/run/dbus:/bin/false
uuidd:x:108:112::/run/uuidd:/bin/false
dnsmasq:x:109:65534:dnsmasq,,,:/var/lib/misc:/bin/false
sshd:x:110:65534::/var/run/sshd:/usr/sbin/nologin
pollinate:x:111:1::/var/cache/pollinate:/bin/false
vagrant:x:1000:1000:,,,:/home/vagrant:/bin/bash
ubuntu:x:1001:1001:Ubuntu:/home/ubuntu:/bin/bash
vboxadd:x:999:1::/var/run/vboxadd:/bin/false
ntp:x:112:116::/home/ntp:/bin/false
```

LAB #5 - La Ressource cron

Consultez le fichier **~/puppet/manifests/run-puppet.pp** :

```
vagrant@ubuntu-xenial:~$ cat puppet/manifests/run-puppet.pp
# Set up regular Puppet runs
file { '/usr/local/bin/run-puppet':
  source => '/etc/puppetlabs/code/environments/production/files/run-puppet.sh',
  mode   => '0755',
}

cron { 'run-puppet':
  command => '/usr/local/bin/run-puppet',
  hour    => '*',
  minute  => '*/15',
}
```

Rappelez-vous que ce manifest avait créé le fichier crontab de **root** :

```
vagrant@ubuntu-xenial:~$ sudo crontab -l
```

```
# HEADER: This file was autogenerated at 2020-02-11 10:08:47 +0000 by puppet.
# HEADER: While it can still be managed manually, it is definitely not recommended.
# HEADER: Note particularly that the comments starting with 'Puppet Name' should
# HEADER: not be deleted, as doing so could cause duplicate cron jobs.
# Puppet Name: run-puppet
*/15 * * * * /usr/local/bin/run-puppet
```

Consultez l'exemple suivant :

```
cron { 'cron example':
  command     => '/bin/date +%F',
  user        => 'ubuntu',
  environment => ['MAILTO=admin@example.com', 'PATH=/bin'],
  hour        => '0',
  minute      => '0',
  weekday     => ['Saturday', 'Sunday'],
}
```

L'Attribut user

Cet attribut spécifie qui exécute le cron job. Si ce n'est pas spécifié, le job est exécuté par root.

L'Attribut environment

Cet attribut permet de définir des variables système pour cron.

L'Attribut weekday

Cet attribut permet de spécifier les jours de la semaine.

L'Attribut monthday

Cet attribut permet de spécifier le jour du mois entre 1 et 31.

La Fonction fqdn_rand

Afin d'éviter à ce que les mêmes cron jobs ne s'exécutent sur tous les noeuds en même temps, Puppet utilise la fonction **fqdn_rand**.

Cette fonction fournit un nombre aléatoire de 0 à 23 dans le cas de l'exemple ci-dessous :

```
cron { 'run daily backup':
  command => '/usr/local/bin/backup',
  minute  => '0',
  hour    => fqdn_rand(24, 'run daily backup'),
}

cron { 'run daily backup sync':
  command => '/usr/local/bin/backup_sync',
  minute  => '0',
  hour    => fqdn_rand(24, 'run daily backup sync'),
}
```

LAB #6 - La Ressource exec

La ressource **exec** permet d'exécuter toute commande sur un noeud. Ce qui peut être exécuté en ligne de commande peut être exécuté en utilisant **exec**.

Considérez l'exemple suivant concernant la compilation d'un logiciel imaginaire :

```
exec { 'install-cat-picture-generator':
```

```
  cwd     => '/tmp/cat-picture-generator',
  command => '/tmp/cat-picture/generator/configure && /usr/bin/make install',
  creates => '/usr/local/bin/cat-picture-generator',
}
```

L'Attribut exec

Cet attribut peut être ce que vous voulez mais doit être unique.

L'Attribut cwd

Cet attribut fixe le répertoire de travail.

L'Attribut command

Cet attribut indique la ou les commandes à exécuter. Il faut indiquer le chemin complet.

L'Attribut creates

Cet attribut indique un fichier qui sera présent après l'exécution de la commande. Si ce fichier existe, Puppet n'exécutera pas la commande une deuxième fois.

L'Attribut user

Créez maintenant le fichier **exec_user.pp** :

```
vagrant@ubuntu-xenial:~$ sudo vi exec_user.pp
```

```
vagrant@ubuntu-xenial:~$ cat exec_user.pp
exec { 'say-hello':
  command => '/bin/echo Hello, this is `whoami` >/tmp/hello-ubuntu.txt',
  user    => 'ubuntu',
  creates => '/tmp/hello-ubuntu.txt',
}
```

Dans le cas de ce manifest, la commande sera exécutée par l'utilisateur **ubuntu** :

```
vagrant@ubuntu-xenial:~$ sudo puppet apply exec_user.pp
Notice: Compiled catalog for ubuntu-xenial in environment production in 0.06 seconds
Notice: /Stage[main]/Main/Exec[say-hello]/returns: executed successfully
Notice: Applied catalog in 0.03 seconds

vagrant@ubuntu-xenial:~$ cat /tmp/hello-ubuntu.txt
Hello, this is ubuntu
```

Important - Si l'utilisateur n'est pas spécifié, la commande sera exécutée par **root**.

L'Attribut onlyif

Certaines commandes ne doivent être exécutées que dans certaines conditions. Pour cette situation, Puppet possède l'attribut **onlyif** :

```
exec { 'process-incoming-cat-pictures':
  command => '/usr/local/bin/cat-picture-generator --import /tmp/incoming/*',
  onlyif  => '/bin/ls /tmp/incoming/*',
}
```

Important - L'attribut **onlyif** spécifie la commande qui est exécuté en premier. Dans le manifest ci-dessus, si le code retour de la commande **/bin/ls /tmp/incoming/*** est autre que 0, la commande spécifiée par l'attribut **command** ne sera **pas** exécutée.

L'Attribut unless

L'attribut **unless** fait l'opposé de l'attribut **onlyif** à savoir, la commande sera toujours exécutée sauf si le code retour de la commande spécifiée par l'attribut **unless** est un 0.

L'Attribut refreshonly

Considérez l'exemple suivant :

```
package { 'postfix':
  ensure => installed,
}

file { '/etc/aliases':
  content => 'root: admin@example.com',
  notify  => Exec['newaliases'],
}

exec { 'newaliases':
  command     => '/usr/bin/newaliases',
  refreshonly => true,
}
```

Dans les cas de ce manifest, la commande **/usr/bin/newaliases** n'est exécutée que dans le cas où le fichier **/etc/aliases** a été modifié.

L'Attribut logoutput

Cet attribut permet de modifier la journalisation de la commande.

L'attribut peut prendre trois valeurs :

- **true**,
  - la sortie de la commande est toujours journalisée,
- **false**,
  - la sortie de la commande n'est jamais journalisée,
- **on failure**,
  - la sortie de la commande n'est journalisée que dans le cas d'une erreur.

Par exemple :

```
exec { 'newaliases':
 command => '/usr/bin/newaliases',
 logoutput => true,
}
```

L'Attribut timeout

Par défaut, Puppet attend 300 secondes pour qu'une commande se termine. Une valeur de 0 permet la commande de s'exécuter à l'infini.

Pour toute autre valeur, il convient d'utiliser l'attribut **timeout**.
