Installation Snorby
CentOS 7

Création de l'environnement

Prérequis : disposer d'un serveur ou d'une machine virtuelle connecté au réseau et en mesure d'accéder à internet. Dans un cas d'application réelle, étant donné les traitements à réaliser, il est conseillé d'avoir au moins un processeur 2Ghz et 2Go de RAM.

Démarrer la machine sur le média d'installation de CentOS 7. Lancer l'installation, penser à configurer la langue et le clavier sur Français (Attention, pas Français (variante) ni Français (oss), Français tout court). Pour le moment, on va conserver une GUI, on choisit donc un bureau GNOME et son set d'applications standards. Choisir le disque entier pour l'installation. On peut aussi configurer le réseau de manière à avoir un accès internet pendant l'installation, ce qui va nous éviter des mises à jour plus tard.

Configurer le compte admin pendant l'installation, noter le mot de passe. Créer un compte utilisateur avec les droits d'administrateur. Finaliser l'installation et redémarrer la machine. Lors du reboot il sera demandé de valider la licence d'utilisateur final.

Installer le dépôt de paquets EPEL

Le dépôt standard est relativement pauvre en paquets, nous devons accéder au dépôt EPEL qui est commun à la plupart des versions de Red Hat, CentOS et Fedora.

```
sudo yum install epel-release
sudo yum repolist
```

On a maintenant accès au dépôt EPEL.

Dépendances

Snorby nécessite un certain nombre de dépendances que nous allons installer :

```
sudo yum install mariadb mariadb-server mariadb-devel git-core readline
readline-devel httpd httpd-devel gcc-c++ gcc automake flex bison libnet
libnet-devel libxml2-devel libxslt-devel glib glib-devel patch postgresql
postgresql-devel wkhtmltopdf libcurl libcurl-devel
```

On va ensuite installer Rbenv, qui est un gestionnaire de versions pour ruby, et qui nous permettra d'installer précisément la version voulue :

```
cd
git clone https://github.com/rbenv/rbenv.git ~/.rbenv
echo 'export PATH="$HOME/.rbenv/bin:$PATH"' >> ~/.bash_profile
```

```
echo 'eval "$(rbenv init -)"' >> ~/.bash_profile
git clone https://github.com/rbenv/ruby-build.git ~/.rbenv/plugins/ruby-build
source ~/.bash_profile
```

Après ces étapes rbenv devrait être correctement installé. Entrer la commande rbenv devrait afficher une liste des actions disponibles. Si le shell indique ‘command not found’, effectuer un reboot de la machine.

A l’aide de rbenv on va installer la version voulue de ruby. Cette étape est particulièrement longue (jusqu’à 15mn), le shell peut paraître planté mais il n’en est rien. L’installation n’est par défaut pas en mode verbose, la sortie du compilateur n’ayant aucun intérêt.

```
rbenv install 2.3.0
rbenv global 2.3.0
```

Ruby étant installé, on va pouvoir passer à l’installation des ‘gems’ nécessaires, un type de package spécifique à Ruby.

```
gem install rails -v ‘3.0.0’
gem install bundler passenger
ruby -v && rails -v
```

La seconde ligne sert à interroger le système à propos du numéro de version, vérifier que c’est bien ruby 2.3.0 et rails 3.0.0 qui sont en place.

```
sudo git clone https://github.com/Snorby/snorby.git /var/www/html/snorby
# ne pas oublier le sudo sinon git ne pourra pas créer l’arborescence
```

On peut créer la base de données nécessaire au fonctionnement de Snorby :

```
sudo systemctl start mariadb
sudo systemctl enable mariadb
sudo mysql_secure_installation
```

Cette dernière commande va lancer un script qui va renforcer l’installation de mysql. Par défaut, aucun mot de passe root n’est configuré, appuyer sur entrée sans rien taper lorsque le script vous demande le mot de passe. Répondez Y (yes) lorsque le script vous demande s’il faut créer un mot de passe root. Entrer votre nouveau mot de passe, deux fois. Répondez ensuite Y (yes) à toutes les autres questions posées par le script.

Dans les actions effectuées on note entre autres la protection contre la connection distante du compte root à la base de données. En clair, root sera utilisable sur notre base uniquement depuis localhost.

```
mysql -u root -p
create database snorby ;
create user ‘snorby’@’localhost’ identified by ‘password’ ;
# remplacer password ci-dessus par le mot de passe voulu, penser à le noter
grant all on snorby.* to ‘snort’@'localhost' ;
flush privileges ;
exit
```

On a ainsi créé le compte dont Snorby se sevira pour accéder à la base et la lire. Lors du déploiement d'un senseur que l'on veut agréger dans Snorby, il faudra installer Barnyard2 sur le senseur et le configurer de manière à ce qu'il se connecte à cette base de donnée à travers le réseau. Il faudra alors, dans la base, créer un compte utilisateur par senseur, de manière similaire à celui-ci mais en remplaçant `localhost` par l'adresse IP du senseur concerné.

On va ensuite installer la myriade de dépendances 'gem' restantes. C'est une étape complexe due au fait que Snorby n'est plus maintenu. Voici donc la commande à exécuter en premier, accompagné d'une liste des potentielles erreurs et leur solution. A chaque erreur rencontrée, appliquer le correctif comme indiqué, puis relancer la commande d'installation.

```
cd /var/www/html/snorby
# la ligne ci-dessous est la commande d'installation
bundle install
```

Voici les cas d'erreur possibles :

- ✗ problème concernant les dépendances de do_mysql
  - ✔ bundle update do_mysql
- ✗ problème concernant les dépendances de dm-mysql-dapater
  - ✔ bundle update dm-mysql-adapter
- ✗ problème de permissions concernant Gemfile.lock
  - ✔ changer le propriétaire du fichier de root à l'utilisateur courant (recommandé)
  - ✔ OU donner les droits en écriture sur le fichier à tous les utilisateurs
  - ✔ OU relancer la commande en mode root

Toute autre erreur de type 'build failed' ou approchant serait plus probablement lié à une installation de dépendance oubliée, dont la liste complète se trouve au début de ce document.

Configuration de Snorby

Mise en place des fichiers de configuration de Snorby :

```
cd config
cp snorby-config.yml.example snorby_config.yml
cp database.yml.example database.yml
```

Ouvrir le fichier snorby_config.yml avec un éditeur de texte (vim, nano, gedit…) et appliquer les modifications suivantes :

```
time_zone: 'Europe/Paris'
```

(attention à bien dé-commenter la ligne)

Ouvrir le fichier database.yml avec un éditeur de texte (vim, nano, gedit…) et appliquer les modifications suivantes :

```
username: snorby
password: "notre_PW_de_la_BD"
```

(remplacer l'expression entre guillemets par le mot de passe du compte snorby de la BD)

Configuration de Passenger

Passenger est une gem permettant aux applications ruby-on-rails de s'exécuter sur un serveur apache (httpd sur CentOS, nous l'avons installé en même temps que les autres dépendances). Entrez la commande suivante, un installeur va être lancé :

```
passenger-install-apache2-module
```

Nouveau passage un peu complexe, de par la nature de Passenger. Voici les instructions à suivre.

- L'installeur affiche un message de bienvenue et quelques recommandations, appuyer sur Entrée
- L'installeur signale que Passenger est installé en version gem et recommande de l'installer via RPM, ne pas tenir compte de cet avertissement et appuyer sur Entrée
- Dans la liste des langages proposées, Ruby et Python sont normalement cochés par défaut. Décocher Python (avec espace) et s'assurer que seul Ruby est coché, le reste ne nous intéresse pas ici. Appuyer ensuite sur Entrée
- Si un avertissement concernant des permissions à modifier apparaît, appliquer toutes les suggestions faites par l'installeur : quitter le programme avec Ctrl+C, modifier toutes les permissions comme demandé, et relancer l'installeur par la même commande que

précédemment. Il est normal de repasser par les étapes précédentes, appliquer les mêmes instructions.

- Au moment de la vérification des dépendances, l'installeur devrait normalement tout indiquer comme OK, car tout a été installé au début. Néanmoins, si une des dépendances a été oubliée, suivre les instructions de l'installeur et installer toutes les dépendances manquantes. Relancer ensuite l'installeur par la même commande que précédemment. Il est normal de repasser par les étapes précédentes, appliquer les mêmes instructions.
- Une longue phase de compilation débute, attendre la fin de cette phase sans tenir compte de la sortie du compilateur qui recouvre l'écran. Un grand nombre d'avertissements est à attendre, sans importance pour la suite. Si la compilation s'arrête à cause d'un problème de dépendance, installer la dépendance manquante puis relancer l'installeur et repasser par les étapes précédentes en appliquant les mêmes instructions.
- Une fois la compilation terminée, l'installeur fournie plusieurs lignes de code. Les copier dans un fichier texte afin de ne pas les perdre. Appuyer ensuite sur Entrée pour continuer.
- L'installeur effectue quelques dernières vérifications. Tout devrait être correct à l'exception d'un avertissement concernant la configuration d'apache. Cet avertissement est normal, nous allons remédier à cette situation juste après. Appuyer sur Entrée.
- L'installeur affiche encore quelques lignes avec divers liens de documentation puis se ferme.

Configuration d'Apache (HTTPD)

On va utiliser les lignes fournies par Passenger pour configurer le serveur web comme désiré.

```
cd /etc/httpd/conf.d
sudo touch passenger.conf
sudo touch snorby.conf
```

Ouvrir ensuite le fichier passenger.conf avec un éditeur de texte (vim, nano, gedit) et y coller les lignes précédemment copiées. Enregistrer et fermer le fichier. Ouvrir le fichier snorby.conf avec un éditeur de texte (vim, nano, gedit) et y entrer les lignes suivantes :

```
Servername 192.168.0.7
 DocumentRoot /var/www/html/snorby/public
 <Directory /var/www/html/snorby/public>
  AllowOverride all
  Order allow,deny
  Allow from all
  Options -MultiViews
 </Directory>
```

Remplacer l'adresse IP présente ci-dessus par l'IP de la machine concernée.

Exécuter la commande suivante, afin de donner les droits d'exploitation de l'application au serveur web :

```
sudo chown -R apache.apache /var/www/html/snorby
```

Dans le fichier `/etc/sysconfig/selinux`, passer la valeur `SELINUX=enforcing` à `SELINUX=disabled`. Ensuite, ouvrez le port 3306 du serveur dans le pare-feu pour permettre aux senseurs d'écrire dans la base MySQL de ce serveur. Le bon remplissage de la base de données est vérifiable avec la commande `select count(*) from event;` depuis MySQL. Si après avoir terminé l'installation sans erreur, les senseurs ne sont pas en mesure d'écrire dans la base de données (message d'erreur au démarrage de Barnyard2 ou bien base de données qui reste vide) le problème vient probablement de l'ouverture du port 3306 qui n'est pas correctement effectué. CentOS 7 semble présenter un système de double pare-feu (iptables ET firewall-cmd) dont les règles ne sont parfois pas couplées correctement. Il est donc essentiel de vérifier prioritairement ce point avant de chercher le problème ailleurs.

Redémarrer la machine. Après le redémarrage, exécuter les quatre commandes suivantes pour démarrer l'application :

```
cd /var/www/html/snorby
su root
RAILS_ENV=production bundle exec rake snorby:update
exit
```

Snorby ne peut plus être lancé autrement qu'en mode root (pas avec sudo, avec le compte root uniquement). Suite à l'arrêt du développement de Snorby, ce problème de permissions n'a jamais été réglé. Si après un reboot le démarrage de Snorby n'est pas fait en root, l'interface apparaîtra et l'utilisateur pourra se connecter et visualiser toutes les alertes qui ont déjà été traité, mais ne pourra en traiter ni en visualiser aucune nouvelle.

Cela est dû au fait que ce programme respecte l'unicité des tâches : l'affichage se contente d'afficher, mais tous les calculs et traitements sont effectués de manière asynchrone par un composant dénommé Snorby::Worker, et c'est ce composant qui ne peut démarrer correctement que si lancé par root. Si le composant est arrêté, l'interface web affichera un message d'avertissement sous forme de bandeau au dessus du tableau de bord.

Dès que le Snorby::Worker aura démarré correctement, toutes les alertes injectées par les senseurs dans la base de données entre temps seront rattrapées, traitées, et affichées par l'interface. Aucune donnée n'est perdue.