

Déploiement Springboot

Via Gitlab CI/CD

Mise en place du déploiement automatique d'une application Springboot vers un serveur externe (VM mise à disposition) via CI/CD Gitlab.

Configuration VM

Se connecter en root avec su :

```
su -l
```

Installer Java

Choisir une version compatible (supérieure ou égale à la version java déclarée dans le **pom.xml**):

```
cd /tmp
wget https://download.oracle.com/java/17/latest/jdk-17_linux-x64_bin.deb
dpkg -i jdk-17_linux-x64_bin.deb
java -version
```

Installer Tomcat 10

Ajouter un utilisateur **tomcat** :

```
useradd -m -d /opt/tomcat -U -s /bin/false tomcat
```

Télécharger et dézipper Tomcat :

```
cd /tmp
wget https://dlcdn.apache.org/tomcat/tomcat-10/v10.1.16/bin/apache-tomcat-10.1.16.tar.gz
sudo tar xzvf apache-tomcat-10*.tar.gz -C /opt/tomcat --strip-components=1
```

Définir les permissions sur les fichiers :

```
chown -R tomcat:tomcat /opt/tomcat/
chmod -R u+x /opt/tomcat/bin
```

Créer le tomcat-manager pour manager les sites à distance via l'interface web :

```
nano /opt/tomcat/conf/tomcat-users.xml
```

```
<!-- user manager can access only manager section -->
<role rolename="manager-gui" />
<user username="manager" password="_SECRET_PASSWORD_" roles="manager-gui" />

<!-- user admin can access manager and admin section both -->
<role rolename="admin-gui" />
<user username="admin" password="_SECRET_PASSWORD_" roles="manager-gui,admin-gui" />
```

Créer un service Tomcat

Pour gérer tomcat plus facilement, créer un service Tomcat :

```
nano /etc/systemd/system/tomcat.service
```

```
[Unit]
Description=Tomcat
After=network.target

[Service]
Type=forking

User=tomcat
Group=tomcat

Environment="JAVA_HOME=/usr/lib/jvm/jdk-17-oracle-x64"
Environment="JAVA_OPTS=-Djava.security.egd=file:///dev/urandom"
Environment="CATALINA_BASE=/opt/tomcat"
Environment="CATALINA_HOME=/opt/tomcat"
Environment="CATALINA_PID=/opt/tomcat/temp/tomcat.pid"
Environment="CATALINA_OPTS=-Xms512M -Xmx1024M -server -XX:+UseParallelGC"

ExecStart=/opt/tomcat/bin/startup.sh
ExecStop=/opt/tomcat/bin/shutdown.sh

[Install]
WantedBy=multi-user.target
```

Recharger le daemon service pour prendre en compte le nouveau service :

```
systemctl daemon-reload
```

Démarrer Tomcat :

```
systemctl start tomcat.service
```

Activer Tomcat pour qu'il redémarre automatiquement à chaque reboot :

```
systemctl enable tomcat.service
```

Afficher son statut :

```
systemctl status tomcat.service
```

```
● tomcat.service - Tomcat
   Loaded: loaded (/etc/systemd/system/tomcat.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2023-11-21 10:53:06 CET; 46min ago
     Process: 1235 ExecStart=/opt/tomcat/bin/startup.sh (code=exited, status=0/SUCCESS)
    Main PID: 1242 (java)
       Tasks: 31 (limit: 2323)
      Memory: 336.6M
         CPU: 12.349s
    CGroup: /system.slice/tomcat.service
            └─1242 /usr/lib/jvm/jdk-17-oracle-x64/bin/java -Djava.util.logging.config.file=/opt/tomcat>
```

Accéder à l'interface Web via l'adresse DNS fournie :

- [http\[s\]://srv2-vm-\[number\].sts-sio-caen.info/](http[s]://srv2-vm-[number].sts-sio-caen.info/)
- [http\[s\]://srv2-vm-\[number\].sts-sio-caen.info/manager/](http[s]://srv2-vm-[number].sts-sio-caen.info/manager/)

Mise en place CI/CD

gitlab-ci user sur la VM

Sur la VM :

- Créer un utilisateur **gitlab-ci**
- Faire en sorte qu'il puisse accéder à **sudo**
- Changer son password : **changeMyPassword**
- Permettre qu'il puisse s'authentifier en SSH avec login/password
- Redémarrer SSH
- Faire en sorte qu'il n'ait pas besoin de saisir un password en utilisant **sudo** avec les commandes **mv**, **cp**, **systemctl**

```
adduser --quiet --shell $SHELL --disabled-password --gecos 'GitlabCI User' gitlab-ci
usermod -a -G sudo gitlab-ci
echo 'gitlab-ci:changeMyPassword' | chpasswd
printf 'Match User gitlab-ci\n\tPasswordAuthentication yes\n' >>
/etc/ssh/sshd_config
systemctl restart sshd
echo 'gitlab-ci ALL=(ALL) NOPASSWD: /bin/mv, NOPASSWD: /usr/bin/systemctl,
NOPASSWD: /bin/cp' | sudo EDITOR='tee -a' visudo
```

Variable CI/CD sur Gitlab

Créer une variable dans Gitlab pour stocker le mot de passe de l'utilisateur **gitlab-ci** qui se connectera en SSH :

Dans **Settings-CI/CD**, créer la variable **CI_USER_PASS** :

CI/CD Variables </> 1				Reveal values	Add variable
↑ Key	Value	Environments	Actions		
CI_USER_PASS  Protected Expanded	***** 	All (default) 	 		

Configuration du projet

Modifier **pom.xml** pour définir le nom du fichier WAR déployé :

```
<build>
  <finalName>ssh-deploy</finalName>
</build>
```

Vérifiez que le déploiement sera bien fait en WAR :

```
<packaging>war</packaging>
```

Configuration Gitlab CI/CD

Créer ou modifier le fichier **.gitlab-ci.yml** :

```
stages:
  - build
  - deploy

maven-build:
  image: maven:3.9.5-amazoncorretto-17-debian
  stage: build
  script: "mvn package -B"
  artifacts:
    paths:
      - target/ssh-deploy.war

deploy-master:
  variables:
    HOST: "149.202.94.223"
    PORT: "7839"
    USER: "gitlab-ci"
    WAR: "ssh-deploy.war"
  rules:
```

```
- if: '$CI_COMMIT_BRANCH =~ /^main$/'
before_script:
- apt-get update -qq && apt-get install -y -qq sshpass sudo
- echo "Host= $HOST"
stage: deploy
script:
- sudo whoami # Vérifiez si sudo est disponible
- which mv
- sshpass -V
- export SHHPASS=$CI_USER_PASS
- sshpass -e scp -o StrictHostKeyChecking=no -P $PORT target/$WAR
$USER@$HOST:/home/$USER
- sshpass -e ssh -tt -o StrictHostKeyChecking=no -p $PORT $USER@$HOST echo
$PATH
- sshpass -e ssh -tt -o StrictHostKeyChecking=no -p $PORT $USER@$HOST sudo mv
/home/$USER/$WAR /opt/tomcat/webapps
- sshpass -e ssh -tt -o StrictHostKeyChecking=no -p $PORT $USER@$HOST sudo
systemctl restart tomcat.service
```

From:

<http://slamwiki2.kobject.net/> - **SlamWiki 2.1**

Permanent link:

<http://slamwiki2.kobject.net/web/framework/spring/deployment?rev=1700590229>

Last update: **2023/11/21 19:10**

