Exploitation coté serveur

Lancer l'application en tant que service

Afin de gérer facilement les redémarrages de l'application, et notament le redémarage automatiques lors des reboots du serveur, il peut être nécessaire de configurer un service au niveau du systeme d'exploitation.

Voici un exemple de configuration de systemd pour chaque "mode" de démarrage d'esup-signature à mettre un fichier **esup-signature.service** dans /etc /systemd/system/ :

Lancement via tomcat

```
[Unit]
Description=Apache Tomcat Web Application Esup Signature
After=syslog.target network.target

[Service]
Type=forking

Environment=JAVA_HOME=/usr/lib/jvm/jdk-17.0.1
Environment=CATALINA_PID=/opt/tomcat-esup-signature/temp/tomcat.pid
Environment=CATALINA_HOME=/opt/tomcat-esup-signature
ExecStart=/opt/tomcat-esup-signature/bin/startup.sh
ExecStop=/bin/kill -15 $MAINPID

User=tomcat
Group=tomcat

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

Lancement via maven

```
[Unit]
Description=Esup Signature

[Service]
User=esup
Group=esup
Environment=JAVA_HOME=/usr/lib/jvm/jdk-17.0.1
ExecStart=mvn spring-boot:run
WorkingDirectory=/opt/esup-signature/

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

Lancement du .war

```
[Unit]
Description=Esup Signature

[Service]
User=esup
Group=esup
Environment=JAVA_OPTS="--add-exports=java.base/sun.security.pkcs=ALL-UNNAMED -Xms2048m -Xmx4096m"
#Si votre serveur se situe derrière un proxy, il est possible de positionner la configuration dans
Environnement comme suit :
#Environment=JAVA_OPTS="--add-exports=java.base/sun.security.pkcs=ALL-UNNAMED -Xms2048m -Xmx4096m -Dhttps.
proxyHost=proxy.univ-ville.fr -Dhttps.proxyPort=3128 -Dhttp.proxyHost=proxy.univ-ville.fr -Dhttp.proxyPort=3128"
ExecStart=/opt/esup-signature/target/esup-signature.war --spring.config.location=/opt/esup-signature
/application.yml -Dfile.encoding=UTF-8
WorkingDirectory=/opt/esup-signature/

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

Après l'ajout de votre fichier esup-signature.service, lancez la commande suivante pour l'activer:

```
systemctl enable esup-signature.service
```

Démarrage, redémarrage et arrêt du service

```
systemctl start esup-signature.service
systemctl restart esup-signature.service
systemctl stop esup-signature.service
```

Mise en place d'un Apache en frontal

Afin de publier esup-signature de manière securisée, il est possible de mettre un frontal Apache à l'aide du module mod_proxy.

Pour ce faire installer apache puis activer le mod_proxy :

```
a2enmod proxy proxy_http proxy_ajp
```

Voici maintenant un exemple de configuration apache qui va rediriger les requetes http vers https puis faire proxy vers l'application esup-signature

```
<VirtualHost *:80>
        ServerName esup-signature.univ-ville.fr
        ServerAlias esup-signature
        ServerAdmin  system@univ-ville.fr
        ServerSignature Off

        RewriteEngine On
        RewriteRule ^(.*) https://esup-signature.univ-ville.fr$1 [L,R]
</VirtualHost>

<VirtualHost *:443>
    ServerName esup-signature.univ-ville.fr
    ServerAlias esup-signature
    ServerAdmin  systeme@univ-ville.fr
    ServerSignature Off

    SSLEngine on
    SSLCertificateFile /etc/pki/tls/certs/crt
    SSLCertificateKeyFile /etc/pki/tls/private/key
    SSLCertificateChainFile /etc/pki/tls/certs/cacert
    SSLverifyclient none

        ProxyPreserveHost On

    #ProxyPass / ajp://localhost:6009/ ttl=10 timeout=3600 loadfactor=100 retry=1
        # ou
    ProxyPass / http://localhost:8080/ ttl=10 timeout=3600 loadfactor=100 retry=1
        # avec proxy http (et non ajp), il faut ajouter le X Forward
    RequestHeader set "X-Forwarded-Proto" expr=%{REQUEST_SCHEME}
    RequestHeader set "X-Forwarded-SSL" expr=%{HTTPS}
    ProxyPreserveHost On
  </VirtualHost>
```

Enfin recharger la configuration Apache :

```
service apache2 reload
```

⚠ Dans le cas de l'utilisation de Proxypass il faut utiliser le module remoteip qui permet de transmettre l'adresse IP réel du client au serveur tomcat (cela n'est pas necessaire en AJP). Pour ce faire il faut activer le module coté Apache:

ajouter le fichier mod_remoteip.conf dans le dossier conf.d/ et y inscrire :

```
LoadModule remoteip_module modules/mod_remoteip.so
RemoteIPHeader X-Forwarded-For
RemoteIPTrustedProxy 127.0.0.1 ::1
```

Si vous utilisez votre propre tomcat, il faut ajouter une valve dans la configuration du serveur tomcat (server.xml) :

```
<Valve className="org.apache.catalina.valves.RemoteIpValve"
remoteIpHeader="x-forwarded-for"
 proxiesHeader="x-forwarded-by"
 protocolHeader="x-forwarded-proto" />
```

Il est aussi possible d'utiliser le protocole AJP. On peut configurer la port ajp dans application.yml

Gestion des sources avec git

Pour gérer plus facilement les sources sur le serveur d'exploitation il est conseillé d'utiliser git.

Le dépôt d'origine d'esup-signature se trouve sur github https://github.com/EsupPortail/esup-signature. Le dépôt fonctionne ainsi :

- la branche "master" contient le code de la dernière release
- la branche "test" contient les commits compris entre la dernière et la prochaine release

Des tags sont créés pour chaque version importante du code voir : Change log , https://github.com/EsupPortail/esup-signature/tags

Clonage

Comme expliqué dans la documentation d'installation, il est préférable de cloner le dépôt git sur le serveur qui héberge esup-signature. Pour cela on commence par faire :

```
git clone https://github.com/EsupPortail/esup-signature.git
```

Le dépôt sera copié dans ./esup-signature, vous pouvez rester sur la branche **master** (branche git par défaut)

Commit de la configuration de production

Dans le cas ou vous voulez concerver le fichier de configuration dans votre dépôt local, vous devez créer un commit de production pour "sauvegarder" vos modifications. Grace à cela lorsque que vous récupérerez les prochaines versions vos modifications seront reprises (et pourront potentiellement donner lieu à des conflits)

Pour consulter vos modifications :

```
git status
# ou
git diff
```

Pour sélectionner les fichiers à "commiter":

```
# pour tout prendre
git add .
# ou pour de l'unitaire
git add src/resources/application.yml
```

Pour créer un nouveau commit :

```
git commit -m "ma conf de prod"
```

Il faudra répéter cette opération pour chaque modification effectuée sur le code

Si votre configuration est externalisée (utilisation de --spring.config.location), on conseil de créer un dépôt local contenant votre fichier. Cela pemettra de revenir sur une ancièn version si besoin.

Mise à jour d'esup-signature

Il faut tout d'abord mettre à jour votre dépôt local depuis le dépôt github

```
git fetch
```

Vous pouvez consulter la liste des tags et ainsi contrôler que les dernières modifications ont bien été téléchargées:

```
git tag --sort=-v:refname
```

(--sort=-v:refname permet de lister dans l'ordre de version de tag la plus récente en premier)

Il faut de plus s'assurer que toutes les modifications de code ont bien été sauvegardées ("commitées") :

```
git status
```

S'il y a des modifications à valider, faire un commit comme expliqué plus haut.

On peut alors fusionner la branche master locale avec la version (le tag) correspondant à la version suivante d'esup-signature.

Dans la page Change log on conseille de passer par chaque version intermédiaire du code. Cela n'est nécessaire que dans le cas où il y a des mises à jours de base de données. Il faut donc bien vérifier la présence ou non de script à passer en consultant le Change log et/ou en regardant la liste des scripts dans src/resources/ (les scripts update_X.X.sql).

Pour fusionner la version (tag) voulue :

```
git merge X.X.X
```

git va tenter de fusionner le code provenant du tag sur votre code de la branche locale **master**. A ce moment il se peut que des conflits se produisent. Si une même partie de code à été modifiée à la fois dans la branche master locale (configuration ou personnalisation) et sur le tag provenant de github, git va les marquer en conflit. Un retour à l'écran précise les fichiers à corriger, dans ce cas le projet ne compilera pas. Il faut donc éditer chaque fichier en conflit pour corriger les incohérences.

Lorsque les conflits sont résolus et que le projet compile (**mvn clean package**) correctement, Il faut de nouveau faire un commit sur la votre branche master locale (le commentaire n'est pas obligatoire c'est git qui le gère pour le merge):

```
git commit
```


Si précisé dans la page Change log , ne pas oublier de passer le script de mise à jour de base de données après avoir fait "**mvn clean package**"

Mise à jour de la base de données

Pour monter de version il se peut que des scripts doivent être lancer. Pour ce faire il faut d'abord compiler une première fois les sources pour que le schéma de la base soit à jour. Ensuite, il faut passer le script correspondant présent dans "src/main/resources/update_X.X.sql".

Pour lancer un script SQL sous PostgreSQL :

```
su postgres
psql -d 'nom_base_esupsignature' < 'chemin/nom_script.sql'
```

Sauvegarde / Restauration

Voici un exemple de tache planifiée pour une sauvegarde quotidienne :

```
08 23 * * * postgres rm -rf /opt/pg-backup/esupsignature-dump && pg_dump -b -F d -f /opt/pg-backup
/esupsignature-dump esupsignature
```

Pour la restauration, il faut arreter le service esup-signature puis supprimer la base de données, puis recréer la base :

```
su postgres
psql
drop database esupsignature;
create database esupsignature;
create USER esupsignature with password 'esup';
grant ALL ON DATABASE esupsignature to esupsignature;
exit
```

Enfin lancer la commande de restoration de la base :

```
pg_restore -d esupsignature --no-owner --role <nom_de_l_utilisateur> /backup/esup-signature/esupsignature-dump/
```

Contrôle / Archivage / Purge des données

Lorsqu'une adresse est configurée au niveau la propriété archive-uri du fichier de configuration application.yml le système va tenter d'y archiver les documents signés (Demandes terminées).

Le paramètre delay-before-cleaning permet de régler un délais en jours avant la suppression des fichiers présents en base pour les demandes qui sont au statut "Exporté". Si la valeur est 0, les demandes seront archivées dés quelles auront été exportées.

> ⚠ Pour que le système d'archivage soit activé, il faut mettre le paramètre enable-scheduled-cleanup à true
> Assurez-vous également d'avoir bien installé l'extension postgresql-contrib dans la partie installation de la BDD : PostgreSQL

Les demandes restent consultables depuis esup-signature qui fait, alors, la passerelle avec l'espace d'archivage.

Pour consulter la taille de la base depuis la console psql :

```
SELECT pg_size_pretty( pg_database_size('esupsignature'));
```

Pour purger la base des larges objects :

```
vacuumlo -n esupsignature  # vérifer le nombre de large objects à supprimer
```

Puis à nouveau depuis psql :

```
psql esupsignature;
VACUUM FULL;
```

Pour voir ce qui se déroule vous pouvez rendre VACUUM plus verbeux :

```
VACUUM FULL VERBOSE;
```