Guide d’exploitation ONYX Server sur Linux

Introduction

Cet article est un exemple type de dossier d’exploitation ONYX Server sur Linux. Il est à adapter en fonction de votre fonctionnement interne.

Il concerne exclusivement ONYX Server. L’administration d’Apache web server (arrêt / redémarrage) n’est pas décrite dans cet article.

Généralités

Sélection de l’instance Mapping

Il est possible d’installer plusieurs instances différentes de ONYX Server sur le même serveur Unix/Linux. De ce fait, avant toute action en ligne de commande, il est impératif de spécifier l’environnement Mapping désiré. Cela se fait en modifiant la variable d’environnement MAPPING_PATH.

En interactif : avec la commande “mappingenv”

```
-bash-4.2$ mappingenv
Quel environnement voulez-vous charger :
1 : Mapping_PROD   /apps/mapping/conf/mapping.conf 8002
Entrez le nom ou le numéro de l'environnement :1
bash-4.2$
```

La liste des environnements proposés est basée sur le fichier /etc/mappingtab

Par défaut, la commande mappingenv se trouve dans « /bin »

Dans un script : Il faut exporter la variable d’environnement MAPPING_PATH avec le chemin complet du fichier mapping.conf correspondant à l’environnement Mapping désiré.

```
export MAPPING_PATH=/apps/mapping/conf/mapping.conf
```

Les fichiers log

Généralités sur les fichiers log

La plupart des fichiers de log Mapping ne sont pas au format texte, et donc non éditables en l’état. Pour les visualiser, il existe 2 méthodes :

- Via l’interface web de Mapping
- En utilisant la commande /apps/mapping/bin/map_log_txt

Pour plus d’information sur cette commande :

```
/apps/mapping/bin/map_log_txt --help
```

Exemple d’affichage du contenu du fichier log du map_lpd :

```
/apps/mapping/bin/map_log_txt -log_file:/apps/mapping/spool/logs/map_lpd.log
```

Fichiers log courants

La plupart des fichiers log se trouvent dans le répertoire /apps/mapping/spool/logs.

Exemples de fichiers log :

<nom queue>.log	Logs liées aux files d’attente (imprimantes ou points d’entrée)
map_daemon.log	Log générale du spooler
map_lpd.log	Log du serveur LPD
map_lpr.log	Log du client LPR de Mapping s’il est invoqué en ligne de commande

Pour y accéder via l’interface : « Menu principal » / « Consulter la log »

Fichiers log liés aux sorties standards et erreur

```
/apps/mapping/temp/stdout.txt
/apps/mapping/temp/stderr.txt
```

Ce sont des fichiers textes éditables tels quels sans conversion.

Fichiers log des travaux présents dans le spooler

Chaque travail possède son propre fichier log. Ces fichiers se trouvent dans le répertoire /apps/mapping/spool/global. Ce sont les fichiers commençant par la lettre L suivie du numéro de travail (MAP_JOBNUM).

Les processus

Description des principaux processus Mapping

Les processus suivants sont fréquemment rencontrés lors de l’affichage de la liste des processus en cours d’exécution. Notamment avec la commande suivante :

```
ps -eaf | grep map
```

map_daemon	Processus du spooler, permettant l’orchestration des travaux dans les files d’attente.

map_lpd	Processus de réception des travaux envoyés en LPR. C'est un enfant du processus map_daemon.
map_splf	Processus permettant notamment d'ajouter un travail dans une file d'attente. C'est un enfant du processus map_lpd, et communique avec le processus map_daemon via le port défini par le paramètre PORT_SOCKET_DAEMON du fichier mapping.conf. Il peut également être déclenché directement en ligne de commande pour agir sur les files d'attentes ou sur les travaux.
map_exec	Processus prenant en charge un travail présent dans une file d'attente afin de lui appliquer un traitement (workflow, exécution d'un script, envoi vers une imprimante…). C'est un enfant du processus map_daemon. Il y en a un par file d'attente en cours de traitement.
map_lpr	Processus permettant d'envoyer un flux d'impression vers une imprimante avec le protocole LPR. C'est un enfant du processus map_exec. Mais il peut également être invoqué directement dans une commande.
map_809	Processus d'exécution d'un Workflow. C'est un enfant du processus map_exec.
mapcpysplf	Processus permettant d'invoquer la composition d'un document. Il peut être déclenché en ligne de commande, dans un script, dans le workflow (map_809)…
map_815UCS	Processus de composition d'un document (invoqué par le mapcpysplf).
map_mail	Processus permettant d'envoyer un email avec une pièce jointe.
map_scanfolder	Processus lié à un robot, permettant de scruter les fichiers dans un dossier afin de lancer un traitement de workflow sur chacun d'entre eux.

D'autres processus préfixés par « map_ » peuvent également apparaitre s'ils sont invoqués notamment via des scripts ou dans le workflow.

Démarrage et arrêt des processus Mapping

Démarrage du spooler

La commande suivante lance le processus map_daemon ainsi que son fils map_lpd

```
export MAPPING_PATH=/apps/mapping/conf/mapping.conf
/apps/mapping/bin/map_daemon start
```

Lors du démarrage du processus map_daemon, un fichier flag contenant le pid du processus map_daemon est créé dans le répertoire spool. Sa présence bloque le démarrage du processus.

```
/apps/mapping/spool/map_daemon.ID
```

Check list pour le démarrage du spooler

- Bien s’assurer que le processus map_daemon soit arrêté
- Vérifier que le map_daemon.ID soit supprimé
- Vérifier que le port du daemon (par défaut 2000) ne soit pas utilisé
- Vérifier que le port du lpd (par défaut 515) ne soit pas utilisé

Arrêt du spooler

La commande suivante arrête proprement le processus map_daemon ainsi que son fils map_lpd

```
export MAPPING_PATH=/apps/mapping/conf/mapping.conf
/apps/mapping/bin/map_daemon stop
```

Attention : si le traitement des robots « scanfolder » consiste à envoyer un fichier dans une file d’attente du spooler, l’arrêt du map_daemon provoquera une erreur à chaque traitement de fichier. Il est donc impératif d’arrêter les robots « scanfolder » avant d’arrêter le map_daemon.

Démarrage des robots « scanfolder »

La commande suivante permet de démarrer un robot « scanfolder »

```
export MAPPING_PATH=/apps/mapping/conf/mapping.conf
/apps/mapping/bin/map_scanfolder -name:nom_du_robot
```

Exemple de script pour démarrer tous les robots :

```
export MAPPING_PATH=/apps/mapping/conf/mapping.conf
LISTROBOT=`/apps/mapping/bin/map_scanfolder -listRobot | awk ' { print $2 } '`
for ROBOT in $LISTROBOT
do
/apps/mapping/bin/map_scanfolder -name:${ROBOT}
done
```

Lors du démarrage du processus map_scanfolder, 2 fichiers flag sont créés avec le pid du processus map_scanfolder dans le répertoire temp. Leur présence bloque le démarrage du processus. Il faut donc les supprimer s’ils existent alors que le processus du scanfolder ne tourne pas :

```
/apps/mapping/temp/_apps_mapping_temp_nom_du_robot_map_scanfolder.ID
/apps/mapping/temp/xxxxx.pid
```

Attention : la suppression de ces fichiers .ID et .pid doit être effectuée en connaissance de cause, c’est à dire uniquement si le robot n’est pas en cours d’exécution. En effet, en cas de suppression de ce fichier, il est alors possible de lancer une autre instance du même robot, ce qui provoque des conflits d’accès simultanés aux mêmes fichiers.

Arrêt des robots « scanfolder »

La commande suivante arrête un robot « scanfolder »

```
export MAPPING_PATH=/apps/mapping/conf/mapping.conf
/apps/mapping/bin/map_scanfolder -stop -name:nom_du_robot
```

Exemple de script pour arrêter tous les robots :

```
export MAPPING_PATH=/apps/mapping/conf/mapping.conf
LISTROBOT=`/apps/mapping/bin/map_scanfolder -listRobot | awk ' { print $2 }
'`
for ROBOT in $LISTROBOT
do
/apps/mapping/bin/map_scanfolder -stop -name:${ROBOT}
done
```

Les processus à monitorer

map_daemon et map_lpd

Ces 2 processus sont des services et doivent être constamment en cours d'exécution. Comme ils sont forkés régulièrement, il peut éventuellement y en avoir plusieurs à instant donné.

```
-bash-4.1$ ps -eaf | grep map_
mapadmin 29551     1  1 Oct25 ?   04:00:29 /apps/mapping/bin/map_daemon
start
mapadmin 27424 29551  0 Oct30 ?   00:00:00 /apps/mapping/bin/map_lpd -
port:515 -nbfork:100
```

map_scanfolder

Il y a autant de processus map_scanfolder que de robot en cours d'exécution. Ils sont identifiés grâce à leur option « -name ». Ils sont indépendants les uns des autres, mais également indépendants du map_daemon.

```
mapadmin   8271  1  0 17:02 ? 00:00:00 /apps/mapping/bin/map_scanfolder
-name:test
```

httpd

Le bon fonctionnement d'Apache peut être contrôlé par la présence du processus httpd. Ce processus est souvent forké, d'où la présence de plusieurs lignes dans la liste des processus.

```
-bash-4.2$ ps -eaf | grep httpd
apache    4590 11849  0 10:56 ?        00:00:00 /usr/sbin/httpd -
```

```
DFOREGROUND
apache    4874 11849  0 08:23 ?        00:00:00 /usr/sbin/httpd -
DFOREGROUND
apache    4876 11849  0 08:23 ?        00:00:00 /usr/sbin/httpd -
DFOREGROUND
apache    4890 11849  0 08:23 ?        00:00:00 /usr/sbin/httpd -
DFOREGROUND
root     11849     1  0 oct.15 ?       00:01:42 /usr/sbin/httpd -
DFOREGROUND
apache   15641 11849  0 08:54 ?        00:00:00 /usr/sbin/httpd -
DFOREGROUND
apache   16301 11849  0 08:56 ?        00:00:00 /usr/sbin/httpd -
DFOREGROUND
```

Actions en cas de crash d’un processus Mapping

Crash du map_daemon

1- Tenter un arrêt propre du spooler Mapping :

```
export MAPPING_PATH=/apps/mapping/conf/mapping.conf
/apps/mapping/bin/map_daemon stop
```

2- Arrêter proprement les processus scanfolder actifs (cf. démarrage et arrêt des processus Mapping)

3- Killer (kill -9) les autres processus mapping encore actifs :

- map_daemon
- map_lpd
- map_exec
- map_lpr
- map_809

A noter que tous ces processus peuvent devenir fils du processus système (PID=1) en cas de crash de leur processus « père », point commun : leur propriétaire est “mapadmin”.

Exemple (à adapter selon le système et la configuration) :

```
ps -f -U mapadmin | grep -v bash | grep -v "ps \-f" | grep -v "UID" | while
read LINE
do
 PID=`echo $LINE | awk ' { print $2 } '`
 kill -9 $PID
done
```

4- Supprimer le fichier flag /apps/mapping/spool/map_daemon.ID

5- Démarrer le spooler Mapping

```
/apps/mapping/bin/map_daemon start
```

Crash du map_lpd

Si le processus map_daemon est toujours actif, il est possible de redémarrer le map_lpd à l'aide de la commande suivante :

```
export MAPPING_PATH=/apps/mapping/conf/mapping.conf
/apps/mapping/bin/map_control -start_lpd
```

Crash d'un robot « scanfolder »

Lors du démarrage d'un robot, un fichier avec le nom du robot et l'extension ".ID" est créé, ainsi qu'un fichier avec le numéro de process et l'extension .pid Exemple :

```
_apps_mapping_temp_nom_du_robot_map_scanfolder.ID
4656.pid
```

En cas de crash du robot, ces fichiers ne sont pas supprimés et empêche le redémarrage du robot. Par conséquent, pour permettre le redémarrage du robot, le fichier « .ID » correspondant au robot arrêté doit être supprimé du répertoire « /apps/mapping/temp »

Opérations de maintenance

Nettoyage courant

Les commandes suivantes peuvent être planifiées très régulièrement. Dans l'idéal, au moins une fois par jour.

Suppression des travaux terminés et arrivé à échéance

```
export MAPPING_PATH=/apps/mapping/conf/mapping.conf
/apps/mapping/bin/map_cron -date
```

Suppression des travaux non effectués (quel que soit leur statut), après 30 jours

```
export MAPPING_PATH=/apps/mapping/conf/mapping.conf
/apps/mapping/bin/map_cron -cleanspool -relative_date:30/0/0.  #
jour/mois/années
```

Suppression des fichiers de session web obsolètes

Fichiers context_menu.*, historique.*, et *.id du dossier /apps/mapping/temp

```
export MAPPING_PATH=/apps/mapping/conf/mapping.conf
```

```
/apps/mapping/bin/map_cron -cleanid
```

Suppression des fichiers temporaires de conversion XPS et de Workflow

Ces fichiers se trouvent dans le dossier temporaire /apps/mapping/temp. Normalement, ils sont supprimés automatiquement. Mais en cas de problème (crash, interruption forcée…), ils peuvent subsister dans le dossier temp. Ce n'est donc pas une situation normale.

```
find /apps/mapping/temp -name "*cri.tmp"     -mtime +2 -exec rm -f {} \;
find /apps/mapping/temp -name "*ttf.tmp"     -mtime +2 -exec rm -f {} \;
find /apps/mapping/temp -name "compress_*"   -mtime +2 -exec rm -f {} \;
find /apps/mapping/temp -name "*.[0-9]"      -mtime +2 -exec rm -f {} \;
find /apps/mapping/temp -name "*.[0-9][0-9]" -mtime +2 -exec rm -f {} \;
find /apps/mapping/temp -name "xps_*"        -mtime +2 -exec rm -f {} \;
find /apps/mapping/temp -name "tmp_report*"  -mtime +2 -exec rm -f {} \;
```

Opérations de nettoyage et de rotations des logs

Pour des raisons pratiques, les commandes suivantes ne doivent pas être planifiées plus d'une fois par jour. Cela afin de ne pas multiplier les archives créées, et éviter de perdre des derniers jours de logs. La fréquence idéale est de 2 fois par semaine, par exemple le lundi soir et le jeudi soir, avant le déclenchement des batchs de nuit.

Archivage des logs

Cette commande permet de faire une conversion texte de tous les fichiers de logs, puis de les archiver dans un fichier zip horodaté.

```
export MAPPING_PATH=/apps/mapping/conf/mapping.conf
/apps/mapping/bin/map_cron -cleanlog -folder:/apps/mapping/spool/logs -
format:TXT
```

Résultat :

```
bash-4.2$ ls -lrt /apps/mapping/spool/logs/*.zip
-rw-r--r-- 1 mapadmin staff 254186924 30 oct.
/apps/mapping/spool/logs/2018_10_30_19_51.zip
-rw-r--r-- 1 mapadmin staff  58479965  7 nov.
/apps/mapping/spool/logs/2018_11_07_12_32.zip
```

Rotation des logs des sorties standard et erreur

```
mv  /apps/mapping/temp/stderr.txt  /apps/mapping/temp/stderr.txt.bak
mv  /apps/mapping/temp/stdout.txt  /apps/mapping/temp/stdout.txt.bak
```

Suppression des archives de logs de plus de 30 jours

```
find /apps/mapping/spool/logs -name "*.zip" -mtime +30 -exec rm -f {} \;
```

Archivage des logs

Cette commande permet de faire une conversion texte de tous les fichiers de logs, puis de les archiver dans un fichier zip horodaté. export MAPPING_PATH=/apps/mapping/conf/mapping.conf

```
/apps/mapping/bin/map_cron -cleanlog -folder:/apps/mapping/spool/logs -format:TXT
```

Résultat :

```
bash-4.2$ ls -lrt /apps/mapping/spool/logs/*.zip
-rw-r--r-- 1 mapadmin staff 254186924 30 oct.
/apps/mapping/spool/logs/2018_10_30_19_51.zip
-rw-r--r-- 1 mapadmin staff  58479965  7 nov.
/apps/mapping/spool/logs/2018_11_07_12_32.zip
```

Rotation des logs des sorties standard et erreur

```
mv  /apps/mapping/temp/stderr.txt  /apps/mapping/temp/stderr.txt.bak
mv  /apps/mapping/temp/stdout.txt  /apps/mapping/temp/stdout.txt.bak
```

Suppression des archives de logs de plus de 30 jours

```
find /apps/mapping/spool/logs -name "*.zip" -mtime +30 -exec rm -f {} \;
```

Exemple de script de purge

```
#Purge script example
#!/bin/bash
echo "------------- Initialization of the parameters necessary for the execution of the script ---------------------"
echo "Set environment"
export MAPPING_PATH=/apps/mapping/conf/mapping.conf
echo "Variable creation containing the path of the Mapping binaries"
PATH_BIN="/apps/mapping/bin"
echo "Creation of the other variables used in the script"
TIMEDATE=`date "+%Y_%m_%d_%H_%M_%S"`
echo "My variable PATH_BIN :  $PATH_BIN"
echo "MAPPING_PATH : $MAPPING_PATH"
echo "Variable creation and population set with the content of the mapping.conf file used by the script"
```

```
PATH_LOG_BACKUP=`$PATH_BIN/map_004 PATH_LOG_BACKUP`
PATH_BASE_MAPPING=`$PATH_BIN/map_004 PATH_BASE_MAPPING`
echo "My variable PATH_LOG_BACKUP : $PATH_LOG_BACKUP"
echo "My variable PATH_BASE_MAPPING : $PATH_BASE_MAPPING"
echo "My timedate : $TIMEDATE"
echo "------------------------- Compression and archiving of the log
files/folders ---------------------------------------------"
echo "Archiivng of the log folder /apps/mapping/spool/logs in to the log
backup folder"
$PATH_BIN/map_cron -cleanlog -folder:$PATH_BASE_MAPPING/mapping_temp/log_temp
echo "Packaging of the content of the mapping log folder and mapping Log
Custom folder"
tar -cvf $PATH_BASE_MAPPING/mapping_temp/log_temp/Archive_M-
Connect_$TIMEDATE.tar $PATH_BASE_MAPPING/spool/mapping_log_custom
tar -cvf $PATH_BASE_MAPPING/log_archive/log_Archive_$TIMEDATE.tar
$PATH_BASE_MAPPING/mapping_temp/log_temp
echo "Compression of the archive log folder"
gzip $PATH_BASE_MAPPING/log_archive/log_Archive_$TIMEDATE.tar
echo "--------------------- Clean of the log folder and mapping log custom
folder -------------------------"
echo "Suppression du contenu du répertoire de log de M-Connect"
rm -f $PATH_BASE_MAPPING/spool/mapping_log_custom/*
echo "Suppression du contenu du répertoire de log temporaire"
rm -f $PATH_BASE_MAPPING/mapping_temp/log_temp/*
echo "clean of the saved spool files"
$PATH_BIN/map_cron -cleanspool -state:saved
echo "Clean of .id , .pid , historique , context_menu files of the mapping
/apps/mapping/temp folder"
$PATH_BIN/map_cron -cleanid
echo "Clean of the temporally files older than 2 days of the
/apps/mapping/temp folder"
find $PATH_BASE_MAPPING/temp -name "*cri.tmp" -mtime +2 -exec rm -f {} \;
find $PATH_BASE_MAPPING/temp -name "*ttf.tmp" -mtime +2 -exec rm -f {} \;
find $PATH_BASE_MAPPING/temp -name "compress_*" -mtime +2 -exec rm -f {} \;
find $PATH_BASE_MAPPING/temp -name "*.[0-9]" -mtime +2 -exec rm -f {} \;
find $PATH_BASE_MAPPING/temp -name "*.[0-9][0-9]" -mtime +2 -exec rm -f {} \;
find $PATH_BASE_MAPPING/temp -name "xps_*" -mtime +2 -exec rm -f {} \;
find $PATH_BASE_MAPPING/temp -name "tmp_report*" -mtime +2 -exec rm -f {} \;
echo "Suppression du contenu de plus de 30 jours du répertoire d'archivage
/apps/mapping/log_archive"
find $PATH_BASE_MAPPING/log_archive -name "*" -mtime +30 -exec rm -f {} \;
```