



**MAPSERVER**

**Installation de Mapserver sur la distribution Linux CentOS 7 - 64bits**

**le 24 Juillet 2013**

**Version 1.0**

www.memoris.fr

1 route de Gachet

44 037 Nantes CEDEX 03

Tél. +33 (0)2 40 68 50 90

Fax +33 (0)2 40 68 54 51

Fax 05 61 54 30 14

**SOMMAIRE**

Sommaire

[1. Préparation 3](#_Toc364780970)

[1.1. Configuration 3](#_Toc364780971)

[1.2. Installer le client SVN 3](#_Toc364780972)

[1.3. Chargement de paquets nécessaire aux compilations 3](#_Toc364780973)

[1.4. Installation des librairies requises 3](#_Toc364780974)

[1.4.a. Bibliothèques géographiques 3](#_Toc364780975)

[1.4.b. Installation de Libecwj 4](#_Toc364780976)

[1.4.c. Installation de curl et gd 4](#_Toc364780977)

[1.4.d. Installation expression régulière PERL 4](#_Toc364780978)

[1.4.e. Installation des bibliothèques graphiques et typographiques 4](#_Toc364780979)

[2. Base de données 4](#_Toc364780980)

[2.1. Installation de postgresql 4](#_Toc364780981)

[2.2. Nouvelle librairie 4](#_Toc364780982)

[2.3. Installation de postgis 5](#_Toc364780983)

[2.4. Initialisation des services 5](#_Toc364780984)

[2.4.a. Initialiser le service 5](#_Toc364780985)

[2.4.b. Initialiser la base de données 5](#_Toc364780986)

[2.4.c. Paramétrage de PostgreSQL 5](#_Toc364780987)

[3. Apache et MapServer 6](#_Toc364780988)

[3.1. Apache 2 6](#_Toc364780989)

[3.1.a. Installation 6](#_Toc364780990)

[3.1.b. Initialiser le service 6](#_Toc364780991)

[3.2. Installation et test de php 6](#_Toc364780992)

[3.3. Installation de mapserver et mapscript 7](#_Toc364780993)

[3.4. MapCache 7](#_Toc364780994)

[3.4.a. Installation 7](#_Toc364780995)

[3.4.b. Configuration 8](#_Toc364780996)

[3.4.a. Utilisation 8](#_Toc364780997)

Pour procéder à l’installation, insérer les lignes de commande (commençant

par #) dans un terminal, en tant qu’utilisateur root.

Cette installation est validée avec les versions suivantes :

Applications installées :

* PostgreSQL 9.3.5
* Apache 2.4.6
* PHP 5.4.35
* Mapserver 6.4.1
* MapCache Master b234a51

Librairies :

Geos 3.4.2 / Proj 4.8.0 / Libecw 3.3 / GDAL 1.11.1 / Curl 7.39.0 / GD 2.0.35 / APR 1.5.1 / APR-Util 1.5.4 / FCGID 2.3.9

Les parties en rouge sont utiles uniquement pour la prise en charge d’Oracle (non testé)

Les parties en bleu sont requises pour une utilisation avec ArcGis

# Préparation

## Installer CentOS

Installer CentOS à partir du fichier [CentOS-7.0-1406-x86\_64-Minimal.iso](http://mirrors.atosworldline.com/public/centos/7.0.1406/isos/x86_64/CentOS-7.0-1406-x86_64-Minimal.iso)

Activer l’interface réseau :

#nmtui

Edit a connection > enp0s3 > Cocher la case Automatically connect

#service network restart

#rpm -ivh http://epel.mirrors.arminco.com/7/x86\_64/e/epel-release-7-2.noarch.rpm

#yum –y update

Pour connaître l’adresse IP de la VM et ainsi travailler en SSH, utiliser la commande :

#ip addr

## **Ajouter une partition de données**

Eteignez la machine virtuelle puis ouvrir les paramètres.

Dans l’onglet « Stockage » cliquez sur « Ajouter un disque dur » de la ligne « Contrôleur : SATA ». Cliquez sur « Créer un nouveau disque ».

Utilisez le format VMDK si vous voulez que le disque dur soit utilisable par d’autre logiciel de virtualisation comme VMWare.

Choisissez « Dynamiquement alloué » de préférence puis choisissez la taille.

Démarrez la VM. Pour info voici les disques et partitions présentent sur votre VM :

* sda : disque dur principale
* sda1 : partition de boot monté sur /boot (grub)
* sda2 : partition principale montée sur / ainsi que le SWAP
* sdb : disque dur de données précédemment créé

Pour ajouter une partition sdb1 à CentOS lancez l’utilitaire fdisk : #fdisk /dev/sdb

Taper ces touches :

n (créer une nouvelle partition)

p (primaire)

1

[Entrée] (premier secteur)

[Entrée] (dernier secteur)

w (écrire la table de patition et quitter l’utilitaire)

Formater la partition avec la commande : #mke2fs /dev/sdb1

Créer le dossier sur lequel sera monté la partition : #mkdir /data

Ajouter cette ligne au fichier /etc/fstab :

/dev/sdb1 /data auto defaults 0 0

Monter la partition : #mount -a

## **Ajouter une partition de données**

Eteignez la machine virtuelle puis ouvrir les paramètres.

Dans l’onglet « Stockage » cliquez sur « Ajouter un disque dur » de la ligne « Contrôleur : SATA ». Cliquez sur « Créer un nouveau disque ».

Utilisez le format VMDK si vous voulez que le disque dur soit utilisable par d’autre logiciel de virtualisation comme VMWare.

Choisissez « Dynamiquement alloué » de préférence puis choisissez la taille.

Démarrez la VM. Pour info voici les disques et partitions présentent sur votre VM :

* sda : disque dur principale
* sda1 : partition de boot monté sur /boot (grub)
* sda2 : partition principale montée sur / ainsi que le SWAP
* sdb : disque dur de données précédemment créé

Pour ajouter une partition sdb1 à CentOS lancez l’utilitaire fdisk : #fdisk /dev/sdb

Taper ces touches :

n (créer une nouvelle partition)

p (primaire)

1

[Entrée] (premier secteur)

[Entrée] (dernier secteur)

w (écrire la table de patition et quitter l’utilitaire)

Formater la partition avec la commande : #mke2fs /dev/sdb1

Créer le dossier sur lequel sera monté la partition : #mkdir /data

Ajouter cette ligne au fichier /etc/fstab :

/dev/sdb1 /data auto defaults 0 0

Monter la partition : #mount -a

## Installer les utilitaires

#yum install –y subversion mlocate wget telnet telnet-server net-tools

#echo 'export SVN\_EDITOR=vim'>>~/.bashrc

#systemctl enable telnet.socket; systemctl start telnet.socket

#firewall-cmd --permanent --add-service=http --add-service=https --add-service=telnet

## Chargement de paquets nécessaire aux compilations

#yum install -y autoconf gcc make zlib-devel libxml2-devel libmcrypt-devel libpng-devel libjpeg-devel bzip2 unzip gcc-c++ expat-devel pixman-devel giflib-devel pcre-devel freetype-devel readline-devel libtiff-devel python-devel

python-devel xerces-j2 libpng libjpeg

## Installation des librairies requises

#mkdir –p /\_install/\_archive

### Bibliothèques géographiques

#tar xvfj geos-3\*.tar.bz2 ; mv geos-3\*.tar.bz2 \_archive/ ; cd geos-3\* / ; ./configure ; make ; make install ; make clean; cd ..

Tester avec: #geos-config --version

#tar xf proj-4.\*tar.gz ; mv proj-4.\*tar.gz \_archive/ ; cd proj-4.\*/ ; ./configure ; make ; make install ; cd .. ; ldconfig

Tester avec: #proj

### Installation de Libecwj

#wget <https://github.com/makinacorpus/libecw/archive/master.zip> ; unzip master.zip ; mv master.zip \_archive/ecw-master.zip ; cd libecw-master ; ./configure CFLAGS="-O0" CXXFLAGS="-O0" --enable-shared --enable-static ; make ; make install ; cd ..

### Installation de curl et gd

#tar xf curl-7.\*.tar.gz ; mv curl-7.\*.tar.gz \_archive/ ; cd curl-7.\*/ ; ./configure ; make ; make install ; make clean ; cd ..

#tar xf gd-2.\*.tar.gz ; mv gd-2.\*.tar.gz \_archive/ ; cd gd/2.\*/ ; ./configure ; make ; make install ; make clean ; cd ../..

## Ouvrir un port

Si besoin, vous pouvez ouvrir un port grâce à ces commandes :

firewall-cmd --zone=public --add-port=8080/tcp --permanent

firewall-cmd --reload

# Base de données

## Installation de postgresql

#tar xf postgresql-9.\*.tar.gz ; mv postgresql-9.\*.tar.gz \_archive/ ; cd postgresql-9.\*/ ; ./configure --with-libxml ; make ; make install

#adduser postgres ; mkdir -p /data/pgsql/data ; chown postgres /data/pgsql/data

#echo /usr/local/lib >> /etc/ld.so.conf !! FAIT PLANTER YUM (libcurl.so.4) !!

Supprimer /usr/local/lib du fichier ld.so.conf et refaire un ldconfig pour utiliser yum

#echo /usr/local/pgsql/lib >> /etc/ld.so.conf

#/sbin/ldconfig ; cd ..

## Nouvelle librairie

Pour support oracle, choisir instantclient 32 ou 64bits selon votre OS

#unzip 'instantclient\*.zip' -d /usr/local ; mv instantclient-\* \_archive ; ln -s /usr/local/instantclient\_11\_2/ /usr/local/instantclient\_11\_2/lib ; echo "/usr/local/instantclient\_11\_2/" > /etc/ld.so.conf.d/oracle\_lib.conf ; ldconfig ; export ORACLE\_HOME=/usr/local/instantclient\_11\_2 ; export LD\_LIBRARY\_PATH=/usr/local/instantclient\_11\_2 ; export PATH=$PATH:$ORACLE\_HOME ; #echo -e "\nexport PATH=\$PATH:/usr/local/instantclient\_11\_2" >> /etc/profile

Vérifier l’absence de Not Found à la sortie de la commande

#ldd /usr/local/instantclient\_11\_2/\*.so

Faire éventuellement :

#find /usr/lib/vmware-tools/lib32/ -maxdepth 1 -name '\*' -exec echo {} >> /etc/ld.so.conf.d/vmware-tools-libraries.conf \ ; ldconfig

Ajouter [IP\_de\_la\_machine] [nom\_de\_la\_machine] dans /etc/hosts

Gdal :

#tar xf gdal-1.\*.tar.gz ; mv gdal-1.\*.tar.gz \_archive/ ; cd gdal-1.\* ; ./configure --with-python --with-ecw=yes --with-expat=/usr/lib --with-pg=/usr/local/pgsql/bin/pg\_config --with-oci=yes ; make ; make install ; make clean ; cd .. ; ldconfig

Tester avec : #gdal-config --version

## Installation de postgis

#mv postgis-2.\*.tar.gz postgresql-9.\*/contrib/ ; cd postgresql-9.\*/contrib ; tar xf postgis-2.\*.tar.gz ; mv postgis-2.\*.tar.gz ../../\_archive/ ; cd postgis-2.\*/ ; ./configure --with-pgconfig=/usr/local/pgsql/bin/pg\_config ; make ; make install ; make clean

## Initialisation des services

Exécuter :

#echo -e "\nexport PATH=\$PATH:/usr/local/pgsql/bin" >> /home/postgres/.bashrc

#echo -e "\nexport PATH=\$PATH:/usr/local/pgsql/bin" >> /etc/profile

### Initialiser le service

Copier le fichier **postgres** fournis dans le répertoire **/etc/init.d/**

Exécuter :

#chmod 755 /etc/init.d/postgres

Cd

Ajouter le service ‘postgres’ à la sequence de demurrage du serveur

#chkconfig --add postgres ; chkconfig postgres on

### Initialiser la base de données

#su postgres

#/usr/local/pgsql/bin/initdb -D /data/pgsql/data --no-locale

#sudo service postgres start

Initialiser un template de Bdd pour le LATIN1 :

Toujours sous l’utilisateur postgres saisir la commande *psql* afin d’obtenir l’invite de commande SQL sous la forme : *postgres=#*

CREATE DATABASE template2 TEMPLATE template0 LC\_COLLATE 'C' LC\_CTYPE 'C' ENCODING 'LATIN1';

UPDATE pg\_database SET datistemplate = TRUE WHERE datname = 'template2';

\q

Copier le fichier makedb2.0.sh dans le répertoire /home/postgres/makedb.sh et exécuter :

#su root

#chmod 755 /home/postgres/makedb2.0.sh ; su postgres

#/home/postgres/./makedb2.0.sh postgis

### Paramétrage de PostgreSQL

Modifier le fichier **/data/pgsql/data/postgresql.conf** comme cela :

listen\_addresses = '\*'

Exécuter :

#echo -e "host all all 80.14.140.201/32 trust\nhost all all 217.108.75.1/32 trust" >> /data/pgsql/data/pg\_hba.conf

#su root

#service postgres restart

# Apache et MapServer

## Apache 2

### Installation

#cd /\_install ; tar xf httpd-2.\*.tar.gz ; tar xf apr-1.\*.tar.gz ; tar xf apr-util-1.\*.tar.gz ; mv apr-1.\*.tar.gz apr-util-1.\*.tar.gz httpd-2.\*.tar.gz \_archive/ ; mv apr-1.\* httpd-2.4.12/srclib/apr ; mv apr-util-1.\* httpd-2.4.12/srclib/apr-util ; cd httpd-2.\*/

#./configure --enable-so --enable-cgi --enable-deflate --enable-headers --enable-rewrite --with-included-apr --with-pcre=/usr/local/pcre --enable-fcgid ; make ; make install ; cd ..

#echo -e "\nexport PATH=\$PATH:/usr/local/apache2/bin" >> /etc/profile

### Ajout du mod FCGI

#export APXS=/usr/local/apache2/bin/apxs ; tar xf mod\_fcgid-2\*.tar.gz ; mv mod\_fcgid-2\*.tar.gz \_archive/ ; cd mod\_fcgid-2\* ; ./configure.apxs ; make ; make install

Ajouter au fichier httpd.conf : LoadModule fcgid\_module modules/mod\_fcgid.so

### Initialiser le service

Copier le fichier **apache2** fournis dans le répertoire **/etc/init.d/**

Exécuter :

#chmod 755 /etc/init.d/apache2 /etc/init.d/functions

Ajouter le service ‘apache’ à la sequence de demurrage du serveur

#chkconfig --add apache2 ; chkconfig apache2 on

#echo -e "\nServerName 127.0.0.1" >> /usr/local/apache2/conf/httpd.conf

#systemctl daemon-reload ; service apache2 start (ou #/etc/init.d/apache2 start)

Tester avec un navigateur l’adresse <http://127.0.0.1/> : « It works »

Ajout automatique dans la config apache :

#mkdir -p /var/log/httpd /var/www/html ; ln -s /var/www/html /projets ; echo -e "User-agent : \*\nDisallow : /" > /projets/robots.txt

## Installation et test de php

#mkdir -p /usr/local/apache2/php/php\_extensions/ ; tar xf php-5.4\*.tar.gz ; mv php-5.4\*.tar.gz \_archive/ ; cd php-5.4\*/

#./configure --prefix=/usr/local/apache2/ --with-apxs2=/usr/local/apache2/bin/apxs --with-config-file-path=/usr/local/apache2/php/ --with-regex=system --with-jpeg-dir=/usr --with-png-dir=/usr --with-zlib --with-gd --with-freetype-dir=/usr/local/ --enable-gd-native-ttf --disable-cgi --enable-mbstring --with-pgsql=/usr/local/pgsql/bin/pg\_config --with-pdo-pgsql=/usr/local/pgsql/bin/pg\_config --with-pear --with-mcrypt --with-curl --with-pdo-mysql --with-mysql --with-mysqli --with-oci8=shared,instantclient,/usr/local/instantclient\_11\_2

#make ; make install

#echo -e "AddType application/x-httpd-php .php .php5 .phtml\nAddType application/x-httpd-php-source .phps">>/usr/local/apache2/conf/httpd.conf

#echo "<?php phpinfo();?>">/var/www/html/index.php

#cp php.ini-development /usr/local/apache2/php/php.ini

#cp ext/oci8/.libs/oci8.so /usr/local/apache2/php/php\_extensions/oci8.so

#chmod 777 /usr/local/apache2/php/php\_extensions/oci8.so

#cd .. ; mv php-5.4.\*/ /usr/local/src/

Mise à jour de la config PHP **/usr/local/apache2/php/php.ini**

default\_charset = "iso-8859-1"

include\_path = ".:/php/includes"

extension\_dir = "/usr/local/apache2/php/php\_extensions/"

date.timezone = "Europe/Paris"

Mise à jour manuelle de la config apache **/usr/local/apache2/conf/httpd.conf**

DocumentRoot "/var/www/html"

DirectoryIndex index.html index.php

<Directory "/var/www/html">

Options Indexes FollowSymLinks

AllowOverride All

Require all granted

</Directory>

Décommenter la ligne LoadModule cgi\_module modules/mod\_cgi.so

Et ajouter LoadModule php5\_module modules/libphp5.so

Tester avec un navigateur l’adresse [http://127.0.0.1/](http://127.0.0.1/test.php) et vérifier la variable extension\_dir

## Installation de mapserver et mapscript

#mkdir /usr/lib/oracle/10.2.0.5/lib/sdk

#ln -s /usr/include/oracle/10.2.0.5/client /usr/lib/oracle/10.2.0.5/lib/sdk/include

#export LD\_LIBRARY\_PATH=/usr/lib/oracle/10.2.0.5/client/lib/

#echo "/usr/lib/oracle/10.2.0.5/client/lib">>/etc/ld.so.conf

#/sbin/ldconfig

#yum install –y cmake fcgi-devel cairo-devel ; tar xf mapserver-6.\*.tar.gz ; mv mapserver-6.\*.tar.gz \_archive/ ; cd mapserver-6.\*/ ; mkdir build ; cd build

#rm -rf ../CMakeCache.txt ../CMakeFiles/ ./\*

#cmake .. -DCMAKE\_INSTALL\_PREFIX=/usr/local/mapserver \

-DWITH\_FCGI=1 -DWITH\_POSTGIS=1 -DWITH\_PHP=1 -DWITH\_CURL=1 -DWITH\_CLIENT\_WMS=1 -DWITH\_CLIENT\_WFS=1 -DWITH\_GD=1 -DWITH\_FRIBIDI=0 \

-DPHP5\_EXTENSION\_DIR=/usr/local/apache2/php/php\_extensions

#make ; make install

#cp /\_install/mapserver-6.4.1/build/mapscript/php/php\_mapscript.so /usr/local/apache2/php/php\_extensions/

#echo extension=php\_mapscript.so >> /usr/local/apache2/php/php.ini

# echo -e "

<IfModule fcgid\_module>

\tFcgidMaxProcessesPerClass 30

\tFcgidInitialEnv PROJ\_LIB /usr/local/share/proj

\tFcgidInitialEnv LD\_LIBRARY\_PATH "/usr/local/lib:/usr/local/pgsql/lib"

\t<Location /mapserv>

\t\tRequire all granted

\t\tSetHandler fcgid-script

\t</Location>

\tScriptAlias /mapserv /usr/local/mapserver/bin/mapserv

</IfModule>" >> /usr/local/apache2/conf/httpd.conf

#cd ..

Redémarrer de apache :

#service apache2 restart ; mkdir -p /data/raster/\_mapfiles

Tester avec un navigateur l’adresse <http://127.0.0.1/> et vérifier la présence de

MapServer version 6.4.1

et de

input=oraclespatial

Tester le fcgi mapserv avec l'URL <http://127.0.0.1/mapserv>, vous devez avoir :

No query information to decode. QUERY\_STRING is set, but empty.

## MapCache

### Installation

#wget <https://github.com/mapserver/mapcache/zipball/master> ; unzip master ; mv master \_archive/mapcache.zip ; cd mapserver-mapcache\* ; mkdir build ; cd build

#rm -rf ../CMakeCache.txt ../CMakeFiles/ ./\* ; cmake .. \

-DWITH\_SQLITE=0 \

-DCMAKE\_PREFIX\_PATH=/usr/local/apache2

#make ; make install

#mkdir -p /data/mapcache /var/www/mapcache ; cd .. ; cp mapcache.xml /var/www/mapcache/mapcache.xml

#echo -e "

<IfModule fcgid\_module>

\tFcgidInitialEnv "MAPCACHE\_CONFIG\_FILE" "/var/www/mapcache/mapcache.xml"

\tIPCCommTimeout 120

\tMaxProcessCount 10

\t<Location /mapcache>

\t\tRequire all granted

\t\tSetHandler fcgid-script

\t</Location>

\tScriptAlias /mapcache "/usr/local/bin/mapcache.fcgi"

</IfModule>" >> /usr/local/apache2/conf/httpd.conf

#service apache2 restart

Tester avec l’URL : http://localhost/mapcache/demo/wms

### Configuration

Modifier le fichier **/var/www/mapcache/mapcache.xml**

L’élément <grid> est commun à toutes les couches

Pour chaque couche un élément <source> et un élément <tileset> doivent être configurés  
Modifier l’élément <cache> comme ceci :

<cache name="disk" type="disk">

<base>/data/mapcache</base>

<symlink\_blank/>

</cache>

Et l’élément <lock\_dir> comme cela :

<lock\_dir>/data/mapcache</lock\_dir>

### **Utilisation**

Construire le catalogue pour les rasters (génère un fichier shape)

# gdaltindex [absolute\_path\_to\_shp\_directory] [absolute\_path\_to\_raster]

Ex : #gdaltindex /data/raster/ortho/ortho /data/raster/ortho/\*.ecw

Créer un mapfile dans le dossier **/data/raster/\_mapfile/** par exemple **ortho.map** :

MAP

EXTENT 100000 6120000 1100000 7120000

IMAGECOLOR 255 255 254

IMAGETYPE PNG

RESOLUTION 96

SIZE 600 400

STATUS ON

TRANSPARENT TRUE

UNITS METERS

NAME "ortho"

#FONTSET "fonts/fonts.txt"

MAXSIZE 4096

#DEBUG 5

#CONFIG "MS\_ERRORFILE" "/data/mapcache/MS\_Error.log"

PROJECTION

"init=epsg:2154"

END

LAYER

NAME "ortho"

PROJECTION

"init=epsg:2154"

END

TYPE RASTER

TILEINDEX "/data/raster/ortho/ortho/ortho.shp" #Référence au shapefile

TILEITEM "LOCATION"

STATUS ON

#MINSCALE 175000

#PROCESSING "RESAMPLE=BILINEAR"

END

END

Ajouter une configuration dans mapcache.xml qui fait référence au fichier mapfile (source qui sera utilisée pour générer les tuiles) :

<source name="ortho" type="wms">

<getmap>

<params>

<FORMAT>image/jpeg</FORMAT>

<LAYERS>ortho</LAYERS>

</params>

</getmap>

<http>

<url>http://localhost/cgi-bin/mapserv?map=/data/raster/\_mapfiles/ortho.map</url> <!--Référence au mapfile -->

<connection\_timeout>500</connection\_timeout>

</http>

</source>

<tileset name="ortho">

<source>ortho</source>

<cache>disk</cache>

<grid>chinook</grid>

<format>JPEG</format>

<metabuffer>10</metabuffer-->

<expires>3600</expires>

<metadata>

<title>Ortho</title>

<abstract>Abstract de la couche Ortho</abstract>

</metadata>

</tileset>

Lancer la génération du cache :

#mapcache\_seed –c /var/www/mapcache/mapcache.xml –g chinook –t ortho

Tous les paramètres sont accessibles via la commande #mapcache\_seed