Instalaciones y compilaciones

Instalación

Instalación básica CentOS

Desactivación SeLinux

Para desactivar el selinux editar el fichero /etc/sysconfig/selinux, modificar el parametro SELINUX a disabled. Verificar que en /etc/selinux/config también aparece como disabled y reiniciar (en un principio el fichero editado es un enlace al config). Una vez reiniciado, si lanzamos el comando sestatus debería aparecer ya como disabled.

Activación rc.local

```
chmod u+x /etc/rc.local.
```

Paquetes necesarios

```
yum -y install epel-release
yum  install  wget Make gcc ImageMagick mcrypt libmcrypt-devel ImageMagick-devel  ImageMagick-perl ImageMa
aspell-devel beecrypt-devel boost distcache distcache-devel docbook-style-xsl elfutils-devel fonts-xorg-truetype fle
libtool-libs nmap openjade postgresql-libs pygtk2 pygtk2-libglade screen  syslinux tk libart_lgpl libc-client libc-clien
net-snmp-devel net-snmp-libs pam-devel passivetex  pcre-devel postgresql postgresql-devel sqlite-devel system-c
perl-Crypt-SSLeay perl-XML-Dumper perl-XML-NamespaceSupport perl-XML-SAX perl-XML-Twig perl-XML-LibXML-C
libXpm-devel libXpm freetype freetype-devel libX11 libX11-devel libmcrypt-devel libtool-ltdl-devel libxml2-devel op
freetype-devel openldap-devel libmcrypt-devel libxslt-devel perl-ExtUtils-Embed pcre-devel bison yaml psmisc
```

Instalación Rabbit / Erlang

Actualmente la versión de apache instalada en el servidor es 2.4.23, esta se puede encontrar en:

```
https://archive.apache.org/dist/httpd/httpd-2.4.23.tar.bz2
```

```
cd /usr/local/src
wget https://archive.apache.org/dist/httpd/httpd-2.4.23.tar.bz2
tar -jxvf httpd-2.4.23.tar.bz2
cd httpd-2.4.23
```

Para compilar el apache con el APR bajar el código del apr y el apr-util (los dos se pueden entocontar en:

```
https://apr.apache.org/download.cgi) . Bajar por ejemplo los paquetes
http://apache.rediris.es//apr/apr-1.6.5.tar.gz y http://apache.rediris.es//apr/apr-util-1.6.1.tar.gz.
```

Descomprimirlos en srclib con el nombre de apr y apt-util respectivamente.

```
cd srclib
wget  http://apache.rediris.es//apr/apr-1.6.5.tar.gz
wget http://apache.rediris.es//apr/apr-util-1.6.1.tar.gz
tar -zxvf apr-1.6.5.tar.gz
tar -zxvf apr-util-1.6.1.tar.gz
mv apr-1.6.5. apr
mv apr-util-1.6.1 apr-util
```

Una vez hecho esto lanzar el configure, make y make install

```
# ./configure --prefix=/www --enable-so --enable-cgi --enable-info --enable-rewrite --enable-speling --enable-usertr
# make
# make install
```

Instalar el GeoIP, primero des librerías de desarrollo y luego el módulo (lo podemos encontrar en https://github.com/maxmind/geoip-api-mod_geoip2)

```
yum install GeoIP-devel
cd /usr/local/src
curl https://codeload.github.com/maxmind/geoip-api-mod_geoip2/zip/1.2.10 -o mod_geoip2.zip
unzip mod_geoip2.zip
cd geoip-api-mod_geoip2-1.2.10
```

```
/www/bin/apxs -i -a -L/usr/local/lib -I/usr/local/include -lGeoIP -c mod_geoip.c
```

Instalación MongoDB

Instalar el repo del mongodb den /etc/yum.repos.d

mongodb.repo

```
[mongodb-org-3.6]
name=MongoDB Repository
baseurl=https://repo.mongodb.org/yum/redhat/$releasever/mongodb-org/3.6/x86_64/
gpgcheck=1
enabled=1
gpgkey=https://www.mongodb.org/static/pgp/server-3.6.asc
```

Una vez instalado se puede hacer el yum del mongodb-org y este ya instalará todas las dependencias:

```
yum install mongodb-org (instalar las 4 dependencias con el server, el tools y el cli)
```

Por último copiar el mongodb.conf en el /etc: **Mongodb.conf**

```
# mongod.conf
# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/
# where to write logging data.
systemLog:
 destination: file
 logAppend: true
 path: /var/log/mongodb/mongod.log
# Where and how to store data.
storage:
 dbPath: /var/lib/mongo
 journal:
   enabled: true
#  engine:
#  mmapv1:
#  wiredTiger:
# how the process runs
processManagement:
 fork: true  # fork and run in background
 pidFilePath: /var/run/mongodb/mongod.pid  # location of pidfile
 timeZoneInfo: /usr/share/zoneinfo
# network interfaces
net:
 port: 27017
```

```
 bindIp: 127.0.0.1  # Listen to local interface only, comment to listen on all interfaces.
security:
 authorization: enabled
#operationProfiling:
#replication:
#sharding:
## Enterprise-Only Options
#auditLog:
#snmp:
```

Instalación

Wordpress con wp-cli

Prodecimiento de instalación

```
$ curl -O https://raw.githubusercontent.com/wp-cli/builds/gh-pages/phar/wp-cli.phar
```

Meter un enlace en el /usr/bin: ln -s /www/php/bin/wp-cli /usr/bin/wp-cli or wp

Después de haber descargado el core, procedemos al cambio de permisos que es necesario.

```
sudo -u www-data wp core donwload
```

Descarga del core

```
wp core download --locale=es_ES
```

Creación base de datos

```
CREATE DATABASE nombredominio_db;
CREATE USER 'nombredominio_user'@'localhost' IDENTIFIED BY 'password_complicado';
GRANT ALL ON nombredominio_db.* TO 'nombredominio_user'@'localhost';
```

==Creación del fichero wp-config.php

```
sudo -u www-data wp core config --dbname=nombredominio_db --dbuser=nombredominio _user --dbpass=passwo
```

Instalar wordpress

```
sudo -u www-data  wp core install --url=nombredominio --title=flozep --admin_user=admin --admin_password=pas
```

Script actualización WP

```
#!/bin/bash
WPPATH="/var/www/public"
WP="/usr/local/bin/wp"
BACKUPDIR="/var/backup"
cd $WPPATH/$i
$WP db export --url=wp1.com
mv *.sql $BACKUPDIR
tar -czf $i.tgz .
mv $i.tgz $BACKUPDIR
$WP core update --url=wp1.com
$WP core update-db --network
$WP plugin update --all --url=wp1.com
$WP theme update --all --url=wp1.com
```

Restricciones WP

- Change wp-login.php to something unique; e.g. my_new_login
- Change /wp-admin/ to something unique; e.g. my_new_admin
- Change /wp-login.php?action=register to something unique; e.g. my_new_registeration
- Add the following to the wp-config.php file (at the very end):
- define('DISALLOW_FILE_EDIT', true);
- Quitar el listado en el Directorio principal:
- Options All -Indexes

Con Apache (Restricción del php / bloqueo del xmlrpc.php / Bloque de acceso al wp-admin)

```
<Directory "/web/nombre_dominio/wp-content/uploads">
  php_admin_flag engine off
  AllowOverride None
  DirectoryIndex Off
  RewriteEngine On
  RewriteRule \.php$ - [F,L]
</Directory>
```

Bloqueo del xmlrpc.php

```
<Files xmlrpc.php>
  order deny,allow
  deny from all
</Files>
```

Bloqueo por IP al admin

```
<Directory /web/nombre_dominio/wp-admin >
  AllowOverride All
  Order Deny,Allow
  Deny from all
  Allow from 2.139.214.8
  Allow from 217.13.118.104
  php_flag display_errors on
  php_flag log_errors on
</directory>
```

Misma conf con nginx:

```
location ~* /wp-upload/.*.php$ {
   deny all;
   access_log off;
   log_not_found off;
}
location /xmlrpc.php  {
              allow   62.43.212.102;
allow   213.97.150.34;
deny    all;
}
location /wp-admin {
 allow   62.43.212.102;
 allow   213.97.150.34;
deny    all;
}
```

Instalación Composer

El composer se puede bajar en la web: https://getcomposer.org/. Bajamos el ejecutable, lo lanzamos y creamos los enlaces:

```
# cd /www/php/bin
# curl "https://getcomposer.org/installer" -o php_composer
#  chmod 755 php_composer
#  ./php php_composer
# rm php_composer
# cd /usr/bin
# ln -s /www/php/bin/php php
#  ln -s /www/php/bin/phar phar
#  ln -s /www/php/bin/composer.phar composer.phar
#  ln -s /www/php/bin/composer.phar composer
```

Instalación

Gitlab-runner

```
# yum install gitlab-runner
# gitlab-runner
# gitlab-runner register
# /etc/gitlab-runner/config.toml
# chown devel:root /etc/gitlab-runner//etc/systemd/system/gitlab-runner.service
# chmod 755 /etc/gitlab-runner/config.toml
# chmod a+x  /etc/systemd/system/gitlab-runner.service
```

Config.toml

```
concurrent = 1
check_interval = 0
```

```
[runners]
 name = "cronmediation"
 url = "http://217.13.124.137:9999/"
 token = "be410c213905833833d28670bfea6e"
 executor = "shell"
 clone_url = "http://217.13.124.137:9999"
 [runners.cache]
```

gitlab-runner.service

```
[Unit]
Description=GitLab Runner
After=syslog.target network.target
ConditionFileIsExecutable=/usr/lib/gitlab-runner/gitlab-runner
[Service]
StartLimitInterval=5
StartLimitBurst=10
ExecStart=/usr/lib/gitlab-runner/gitlab-runner "run" "--working-directory" "/home/gitlab-runner" "--config" "/etc/gitl
Restart=always
RestartSec=120
[Install]
WantedBy=multi-user.target
```

Compilación

Compilación PHP

```
# wget ...
# tar -zxvf php...
# cd php ...
# ./configure --with-apxs2=/www/bin/apxs --with-mysql --with-pdo-mysql –with-mysqli --prefix=/www/php --with-lib
```

Instalación de extensiones, ejemplos:

```
#cd /www/php/bin
# ./pecl install APCu
# ./pecl install mongodb
# ./pecl install opcache
```

Compilación Apache2

Actualmente la versión de apache instalada en el servidor es 2.4.23, esta se puede encontrar en:

```
https://archive.apache.org/dist/httpd/httpd-2.4.23.tar.bz2
```

```
cd /usr/local/src
wget https://archive.apache.org/dist/httpd/httpd-2.4.23.tar.bz2
tar -jxvf httpd-2.4.23.tar.bz2
cd httpd-2.4.23
```

Para compilar el apache con el APR bajar el código del apr y el apr-util (los dos se pueden entocontar en:

```
https://apr.apache.org/download.cgi) . Bajar por ejemplo los paquetes
http://apache.rediris.es//apr/apr-1.6.5.tar.gz y http://apache.rediris.es//apr/apr-util-1.6.1.tar.gz.
```

Descomprimirlos en srclib con el nombre de apr y apt-util respectivamente.

```
cd srclib
wget  http://apache.rediris.es//apr/apr-1.6.5.tar.gz
wget http://apache.rediris.es//apr/apr-util-1.6.1.tar.gz
tar -zxvf apr-1.6.5.tar.gz
tar -zxvf apr-util-1.6.1.tar.gz
mv apr-1.6.5. apr
mv apr-util-1.6.1 apr-util
```

Una vez hecho esto lanzar el configure, make y make install

```
# ./configure --prefix=/www --enable-so --enable-cgi --enable-info --enable-rewrite --enable-speling --enable-usertr
# make
# make install
```

Instalar el GeoIP, primero des librerías de desarrollo y luego el módulo (lo podemos encontrar en https://github.com/maxmind/geoip-api-mod_geoip2)

```
yum install GeoIP-devel
cd /usr/local/src
curl https://codeload.github.com/maxmind/geoip-api-mod_geoip2/zip/1.2.10 -o mod_geoip2.zip
unzip mod_geoip2.zip
cd geoip-api-mod_geoip2-1.2.10
```

```
/www/bin/apxs -i -a -L/usr/local/lib -I/usr/local/include -lGeoIP -c mod_geoip.c
```

Compilación de nginx

Para la instalación de nginx nos deberemos bajar la última versión y guardarla en la carpeta /install del servidor

```
wget https://nginx.org/download/nginx-1.14.0.zip
```

Una vez descargada descomprimimos el fichero en **/usr/local/src/**.

Ahora llega una parte importante, y es que como debemos compilar este servicio tendremos que obtener las opciones previas para la configuración de la compilación, por suerte tenemos varias servicios web instalados en diferentes máquinas para poder obtener estas opciones de manera rápida: Nos vamos un equipo que ya contenga el servicio instalado y lanzamos el siguiente comando:

```
/usr/local/nginx/sbin/nginx -V
```

De la salida de este comando podemos sacar la opciones de configure.

Nos colocamos en /usr/local/src/nginx y lanzamos el configure:

```
./configure –prefix=/usr/local/nginx/....
make
make install
```

Para compilar PHP debemos realizar las mismas operaciones que con Nginx, solo que para obtener las opciones de configuración:

```
/usr/local/src/PHP...../config.log
```

Ojo!!, la compilación de PHP puede tardar bastante, lanzar el ./configure... con un Screen. Del servidor que hemos obtenido las opciones de compilación vamos a obtener también las configuraciones:

```
/usr/local/nginx/conf en la misma carpeta de destino
/etc/php7 en la misma carpeta de destino
```

Después de hacer las copias de configuraciones deberemos aplicar los cambios oportunos, nuevos virtual-host, versiones de PHP, etc. Creamos enlaces simbólicos para nginx y php:

```
/etc
lrwxrwxrwx 1 root root 12 May 8 17:51 nginx ->nginx-1.14.0
```

```
lrwxrwxrwx 1 root root 4 May 9 10:57 php ->php7
```

También debemos copiar los script de inicio de servicios, corrigiendo lo que fuese necesario:

```
/etc/init.d/nginx o httpd o apache2
/etc/init.d/php-fpm
```

OJO! Con la configuración del puerto de escucha de php. Esto también estará especificado en el VirtualHost de Ningx y hay dos opciones posibles, que deben cuadrar tanto en el virtualhost como en la configuración de PHP.

```
/etc/php/fpm/pool.d/www.conf
```

Dejamos los servicios listos por si hubiese un reinicio:

```
chkconfig –list | more
chkconfig –add nginx
chkconfig –add php-fpm
chkconfig –level 35 php-fpm on
chkconfig –level 35 nginx on
```

Arrancamos los servicios desde /etc/init.d/