INSTALACION DE LETS ENCRYPT

NOTA: Se instaló letś encrypt en la plataforma de amazon AWS contratando el servicio de Virtual Private Server.

**Let's Encrypt**

<https://www.linux-party.com/57-seguridad/9979-generar-tu-certificado-ssl-gratis-manualmente-con-let-s-encrypt>

**# mkdir /script**

**# cd /script  
# git clone https://github.com/letsencrypt/letsencrypt  
# cd letsencrypt/**

Generar el certificado.

**#./letsencrypt-auto certonly -a manual -d emergencia.ml -d www.emergencia.ml --server https://acme-v01.api.letsencrypt.org/directory --agree-dev-preview**

**./letsencrypt-auto certonly -a manual -d sanborns.ml --server https://acme-v01.api.letsencrypt.org/directory --agree-dev-preview**

**Configuración well-known en Apache2**

Para descargar el certificado se utilizó apache2.

<https://esferas.org/msqlu/2016/04/07/el-protocolo-acme-empleado-por-lets-encrypt/>

Configuración de la verificación del certificado.

**.well-known/acme-challenge**

Edición y configuración.

/etc/apache2/sites-available

**/etc/apache2/conf-available/letsencrypt.conf**

Configuración del archivo **letsencrypt.conf**

**Alias /.well-known/acme-challenge /var/www/letsencrypt**  
  
**<Directory /var/www/letsencrypt>  
 Options None  
 AllowOverride None  
 Require all granted  
</Directory>**

Crear el directorio

**/var/www/letsencrypt/archivoconstringdetokengeneradorporletsencrtopaverificar**

Configuración de permisos de acceso.

**sudo install -d -o www-data -g www-data -m 0755 /var/www/letsencrypt**

Se habilitó el servidor virtual de apache2

**sudo a2ensite letsencrypt.conf**

A su vez se desabilito el servidor virtual por defecto.

**sudo a2dissite 000-default.conf**

Reinicio de los servicios de apache2

**service apache2 reload**

Realizado lo anterior presionar enter a lestencript para validar el token del archivo….

Revisar el directorio donde se generaron los certificados expedidos por let's Encrypt

**/etc/letsencrypt/live/emergencia.ml**



El siguiente paso es detener el servicio de apache2, esto para no colisionar con los puertos del servidor de Glassfish.

**sudo service apache2 stop**

**Configuración de Glassfish 4**

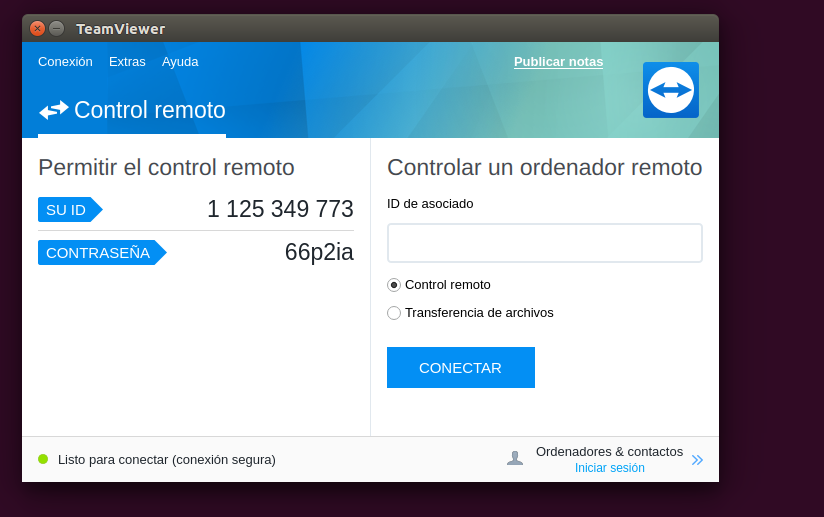
[**https://helpdesk.ssls.com/hc/en-us/articles/115001604071-How-to-install-a-SSL-certificate-on-GlassFish**](https://helpdesk.ssls.com/hc/en-us/articles/115001604071-How-to-install-a-SSL-certificate-on-GlassFish)

Iniciar Glassfish4

**cd /opt/glassfish4/bin**



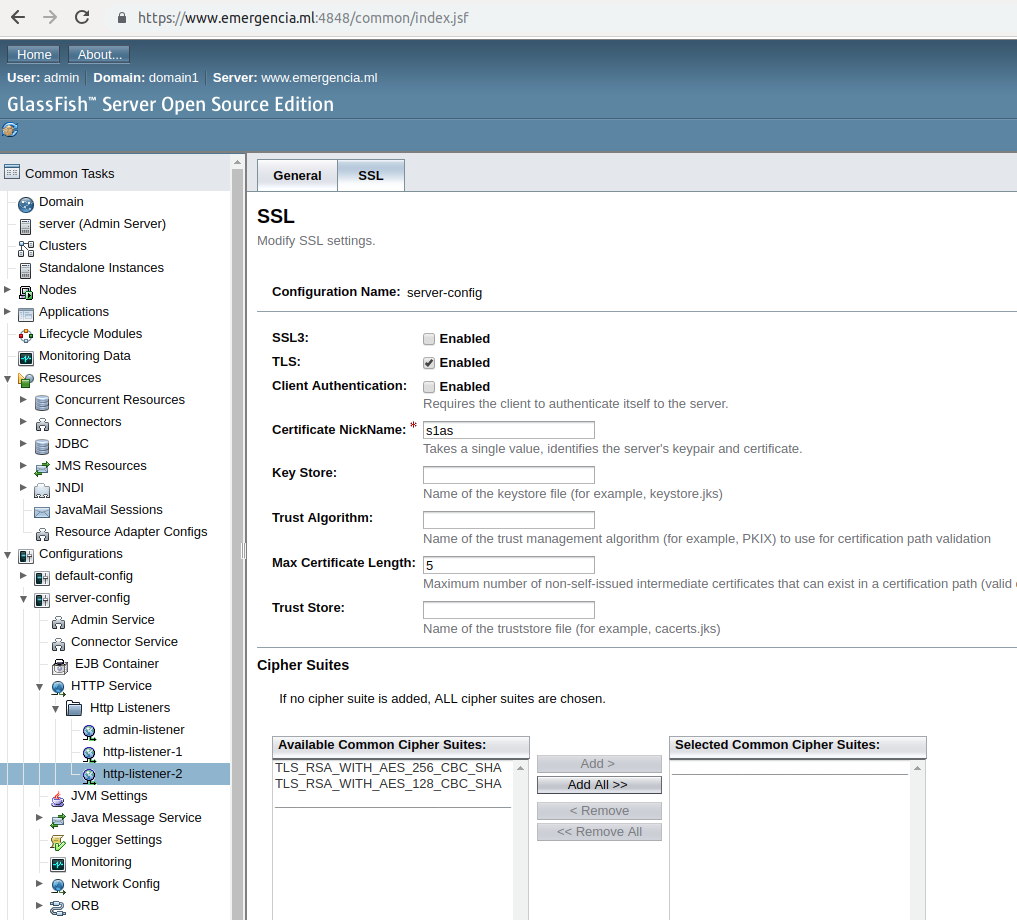
**./asadmin start-domain**



Dirigirse al navegador para autenticarse al administrador de glassfish.



**.well-known/acme-challenge**



Verificar el nombre del nickname del certificado.

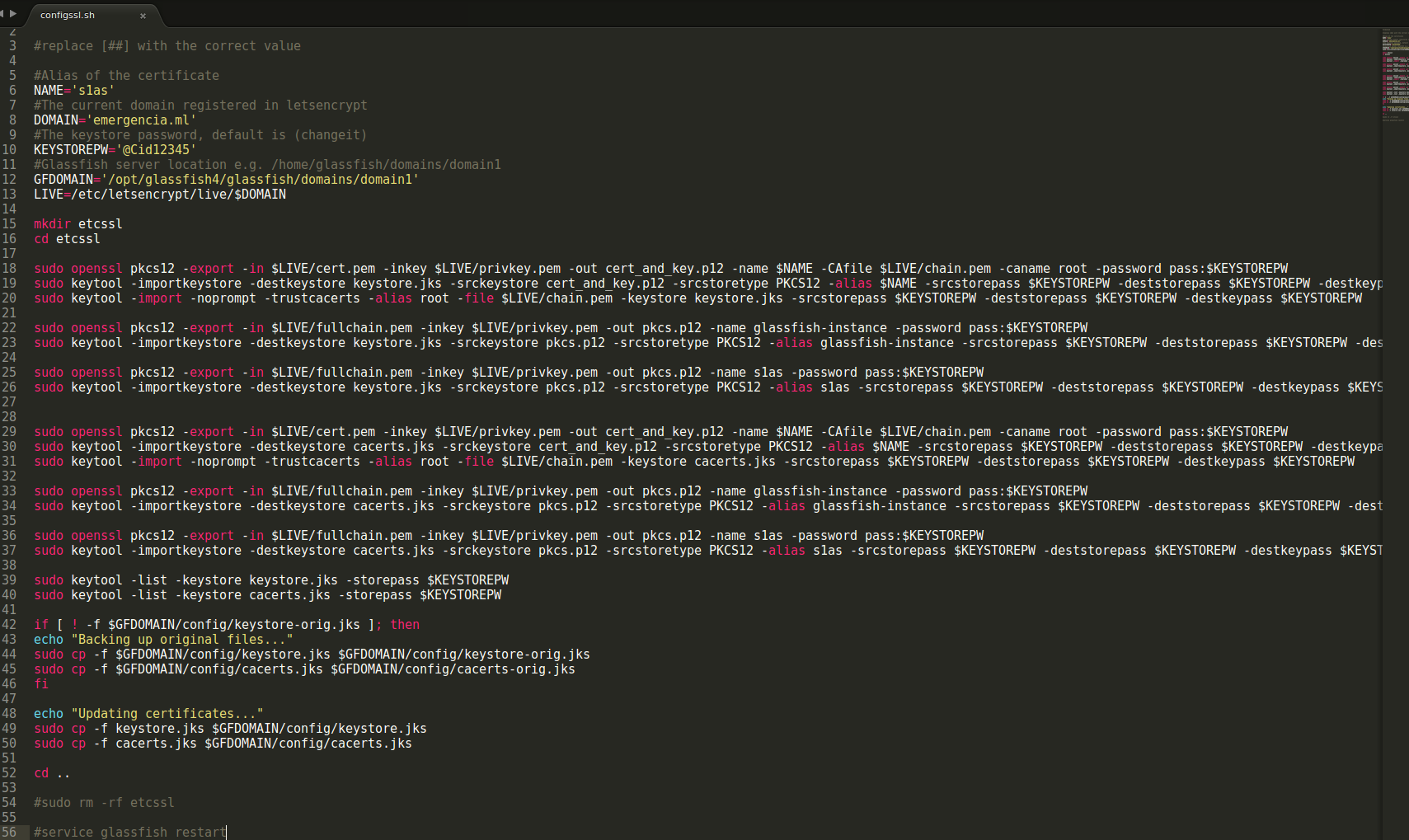
Como ya se tienen generado los certificados en apache

**/etc/letsencrypt/live**



Lo siguiente en realizar va a hacer la conversión de los certificados **.pem a .jks** de java ya que glassfish y tomcat utilizan este tipo de certificado.

Para ello editaremos un script en bash para que realice esta conversión **configssl.sh.**



**Codigo del script configssl.sh**

**#!/bin/sh**

**#replace [##] with the correct value**

**#Alias of the certificate**

**NAME='s1as'**

**#The current domain registered in letsencrypt**

**DOMAIN='emergencia.ml'**

**#The keystore password, default is (changeit)**

**KEYSTOREPW='@Cid12345'IMPORTANT NOTES:**

**- Congratulations! Your certificate and chain have been saved at:**

**/etc/letsencrypt/live/globalkurento.ml/fullchain.pem**

**Your key file has been saved at:**

**/etc/letsencrypt/live/globalkurento.ml/privkey.pem**

**Your cert will expire on 2019-06-13. To obtain a new or tweaked**

**version of this certificate in the future, simply run**

**letsencrypt-auto again. To non-interactively renew \*all\* of your**

**certificates, run "letsencrypt-auto renew"**

**- Your account credentials have been saved in your Certbot**

**configuration directory at /etc/letsencrypt. You should make a**

**secure backup of this folder now. This configuration directory will**

**also contain certificates and private keys obtained by Certbot so**

**making regular backups of this folder is ideal.**

**- If you like Certbot, please consider supporting our work by:**

**Donating to ISRG / Let's Encrypt: https://letsencrypt.org/donate**

**Donating to EFF: https://eff.org/donate-le**

**#Glassfish server location e.g. /home/glassfish/domains/domain1**

**GFDOMAIN='/opt/glassfish4/glassfish/domains/domain1'**

**LIVE=/etc/letsencrypt/live/$DOMAIN**

**mkdir etcssl**

**cd etcssl**

**sudo openssl pkcs12 -export -in $LIVE/cert.pem -inkey $LIVE/privkey.pem -out cert\_and\_key.p12 -name $NAME -CAfile $LIVE/chain.pem -caname root -password pass:$KEYSTOREPW**

**sudo keytool -importkeystore -destkeystore keystore.jks -srckeystore cert\_and\_key.p12 -srcstoretype PKCS12 -alias $NAME -srcstorepass $KEYSTOREPW -deststorepass $KEYSTOREPW -destkeypass $KEYSTOREPW**

**sudo keytool -import -noprompt -trustcacerts -alias root -file $LIVE/chain.pem -keystore keystore.jks -srcstorepass $KEYSTOREPW -deststorepass $KEYSTOREPW -destkeypass $KEYSTOREPW**

**sudo openssl pkcs12 -export -in $LIVE/fullchain.pem -inkey $LIVE/privkey.pem -out pkcs.p12 -name glassfish-instance -password pass:$KEYSTOREPW**

**sudo keytool -importkeystore -destkeystore keystore.jks -srckeystore pkcs.p12 -srcstoretype PKCS12 -alias glassfish-instance -srcstorepass $KEYSTOREPW -deststorepass $KEYSTOREPW -destkeypass $KEYSTOREPW**

**sudo openssl pkcs12 -export -in $LIVE/fullchain.pem -inkey $LIVE/privkey.pem -out pkcs.p12 -name s1as -password pass:$KEYSTOREPW**

**sudo keytool -importkeystore -destkeystore keystore.jks -srckeystore pkcs.p12 -srcstoretype PKCS12 -alias s1as -srcstorepass $KEYSTOREPW -deststorepass $KEYSTOREPW -destkeypass $KEYSTOREPW**

**sudo openssl pkcs12 -export -in $LIVE/cert.pem -inkey $LIVE/privkey.pem -out cert\_and\_key.p12 -name $NAME -CAfile $LIVE/chain.pem -caname root -password pass:$KEYSTOREPW**

**sudo keytool -importkeystore -destkeystore cacerts.jks -srckeystore cert\_and\_key.p12 -srcstoretype PKCS12 -alias $NAME -srcstorepass $KEYSTOREPW -deststorepass $KEYSTOREPW -destkeypass $KEYSTOREPW**

**sudo keytool -import -noprompt -trustcacerts -alias root -file $LIVE/chain.pem -keystore cacerts.jks -srcstorepass $KEYSTOREPW -deststorepass $KEYSTOREPW -destkeypass $KEYSTOREPW**

**sudo openssl pkcs12 -export -in $LIVE/fullchain.pem -inkey $LIVE/privkey.pem -out pkcs.p12 -name glassfish-instance -password pass:$KEYSTOREPW**

**sudo keytool -importkeystore -destkeystore cacerts.jks -srckeystore pkcs.p12 -srcstoretype PKCS12 -alias glassfish-instance -srcstorepass $KEYSTOREPW -deststorepass $KEYSTOREPW -destkeypass $KEYSTOREPW**

**sudo openssl pkcs12 -export -in $LIVE/fullchain.pem -inkey $LIVE/privkey.pem -out pkcs.p12 -name s1as -password pass:$KEYSTOREPW**

**sudo keytool -importkeystore -destkeystore cacerts.jks -srckeystore pkcs.p12 -srcstoretype PKCS12 -alias s1as -srcstorepass $KEYSTOREPW -deststorepass $KEYSTOREPW -destkeypass $KEYSTOREPW**

**sudo keytool -list -keystore keystore.jks -storepass $KEYSTOREPW**

**sudo keytool -list -keystore cacerts.jks -storepass $KEYSTOREPW**

**if [ ! -f $GFDOMAIN/config/keystore-orig.jks ]; then**

**echo "Backing up original files..."**

**sudo cp -f $GFDOMAIN/config/keystore.jks $GFDOMAIN/config/keystore-orig.jks**

**sudo cp -f $GFDOMAIN/config/cacerts.jks $GFDOMAIN/config/cacerts-orig.jks**

**fi**

**echo "Updating certificates..."**

**sudo cp -f keystore.jks $GFDOMAIN/config/keystore.jks**

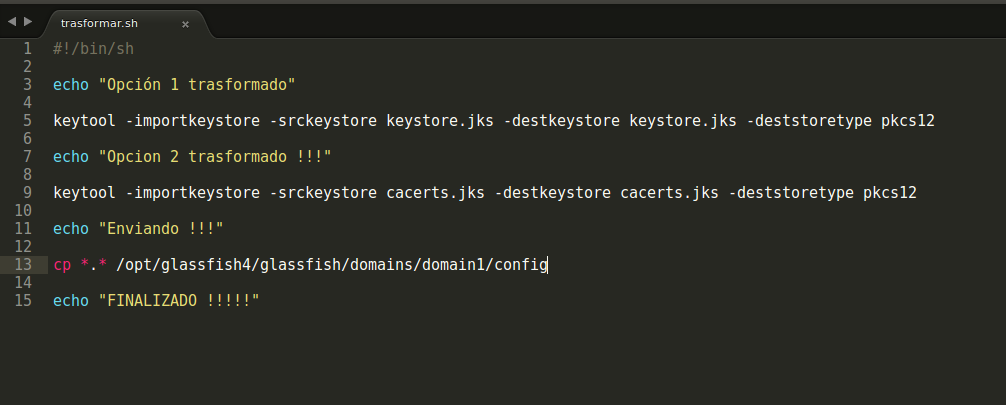
**sudo cp -f cacerts.jks $GFDOMAIN/config/cacerts.jks**

**cd ..**

**#sudo rm -rf etcssl**

**#service glassfish restart**

Crear otro script para transformar los **archivos.jks** al estar de **keytool** más reciente, el script se llama **transformar.sh**.



**Código fuente de** **transformar.sh**

**#!/bin/sh**

**echo "Opción 1 trasformado"**

**keytool -importkeystore -srckeystore keystore.jks -destkeystore keystore.jks -deststoretype pkcs12**

**echo "Opcion 2 trasformado !!!"**

**keytool -importkeystore -srckeystore cacerts.jks -destkeystore cacerts.jks -deststoretype pkcs12**

**echo "Enviando !!!"**

**cp \*.\* /opt/glassfish4/glassfish/domains/domain1/config**

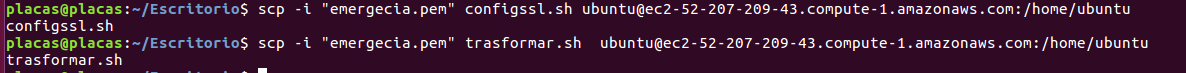
**echo "FINALIZADO !!!!!"**

**----------------------------------------------------**

Una vez preparados los scripts **.sh**  los enviaremos al servidor de amazon donde los vamos a ejecutar.

**scp -i "emergecia.pem" configssl.sh ubuntu@ec2-52-207-209-43.compute-1.amazonaws.com:/home/ubuntu**

**scp -i "emergecia.pem" trasformar.sh ubuntu@ec2-52-207-209-43.compute-1.amazonaws.com:/home/ubuntu**



Conectamos al servidor de amazon por ssh.

**ssh -i "emergecia.pem" ubuntu@ec2-52-207-209-43.compute-1.amazonaws.com**



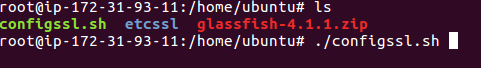
Una vez en el servidor proporcionamos permisos al script.

**sudo chmod 777 configssl.sh**



Ejecutamos el script

./**configssl.sh**



Entrar a directorio donde se generaron los certificados **.jks**

**cd /home/ubuntu/etcssl**

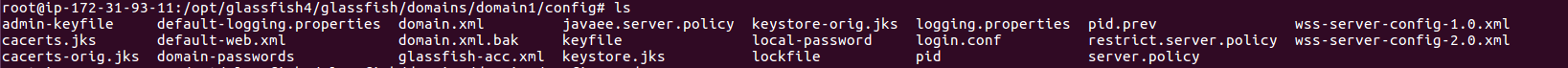


En el directorio iniciamos el script de **transforma.sh**.



En este directorio es donde se generan las claves de glassfish, en este caso solo se remplazaron.

**/opt/glassfish4/glassfish/domains/domain1/config**



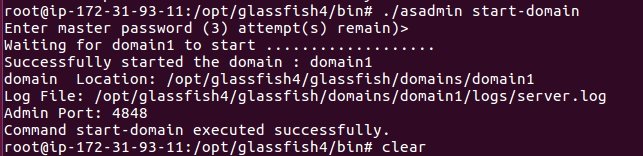
Ir al directorio bin de glassfish4, **cd /opt/glassfish4/bin**



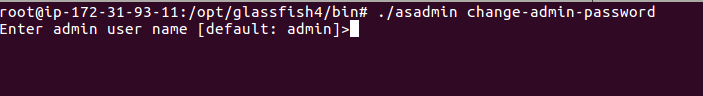
Detenemos el servidor de Glassfish4, **./asadmin stop-domain**



Iniciamos nuevamente el servidor Glassfish4, **./asadmin start-domain** nos pedira la contrasena de admin es la contrsseña que generamos en los certificados de conversion de **.pem** a **.jks**  en este caso es el siguiente **@Cid12345**



Actualizamos la credenciales de usuario de glassfish con los nuevos accesos encriptados.



**CAMBIAR CONTRASEÑA**

**./asadmin change-admin-password**

**admin /\*usuario por defecto\*/**

**admin12345 /\*enter ya que no existe una contraseña\*/**

**admin12345 /\*nueva contraseña\*/**

**admin12345 /\*confirmar nueva contraseña\*/**

**Activar login remoto**

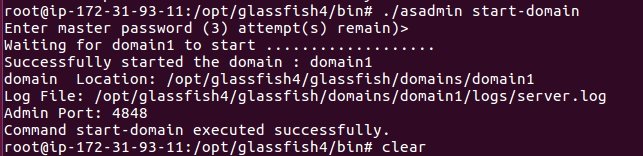
**./asadmin --host localhost --port 4848 enable-secure-admin**

Volvemos a reiniciar el servidor de Glassfish4

Detenemos el servidor de Glassfish4, **./asadmin stop-domain**



Iniciamos nuevamente el servidor Glassfish4, **./asadmin start-domain** nos pedira la contrasena de admin es la contrsseña que generamos en los certificados de conversion de **.pem** a **.jks**  en este caso es el siguiente **@Cid12345**

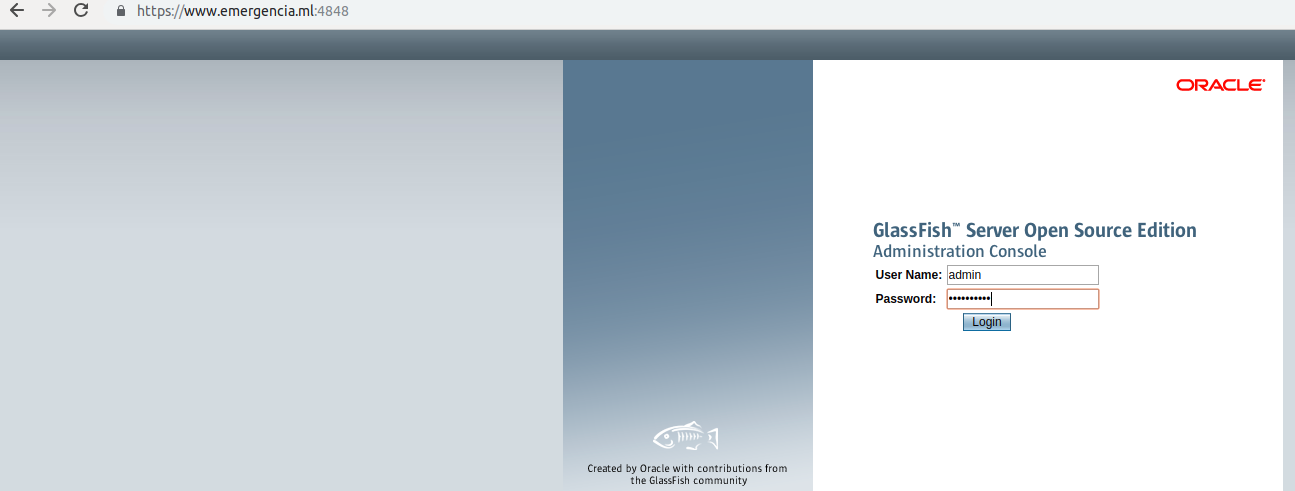


Habrimos el navegador y apuntamos al sevidor

[https://www.emergencia.ml:4848](https://www.emergencia.ml:4848/common/index.jsf)

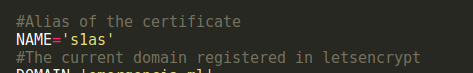
usuario:admin

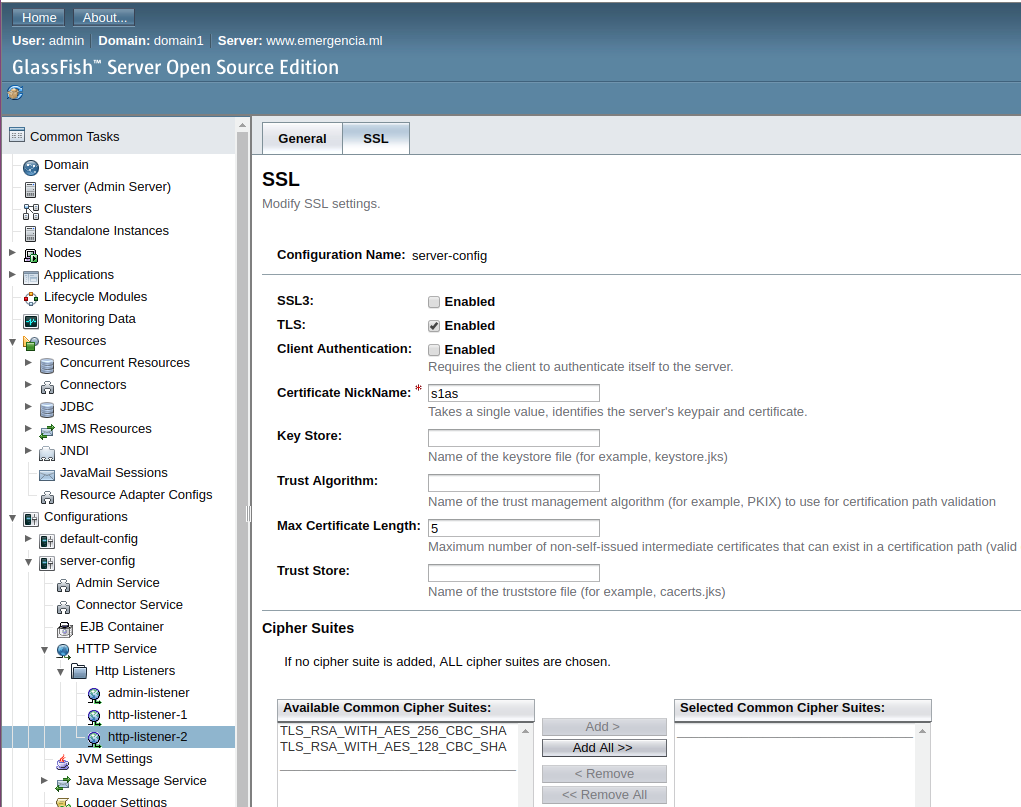
pwd:admin12345



@D3sarrollo321

Verificamos que el nick name se el mismo que generamos en el script

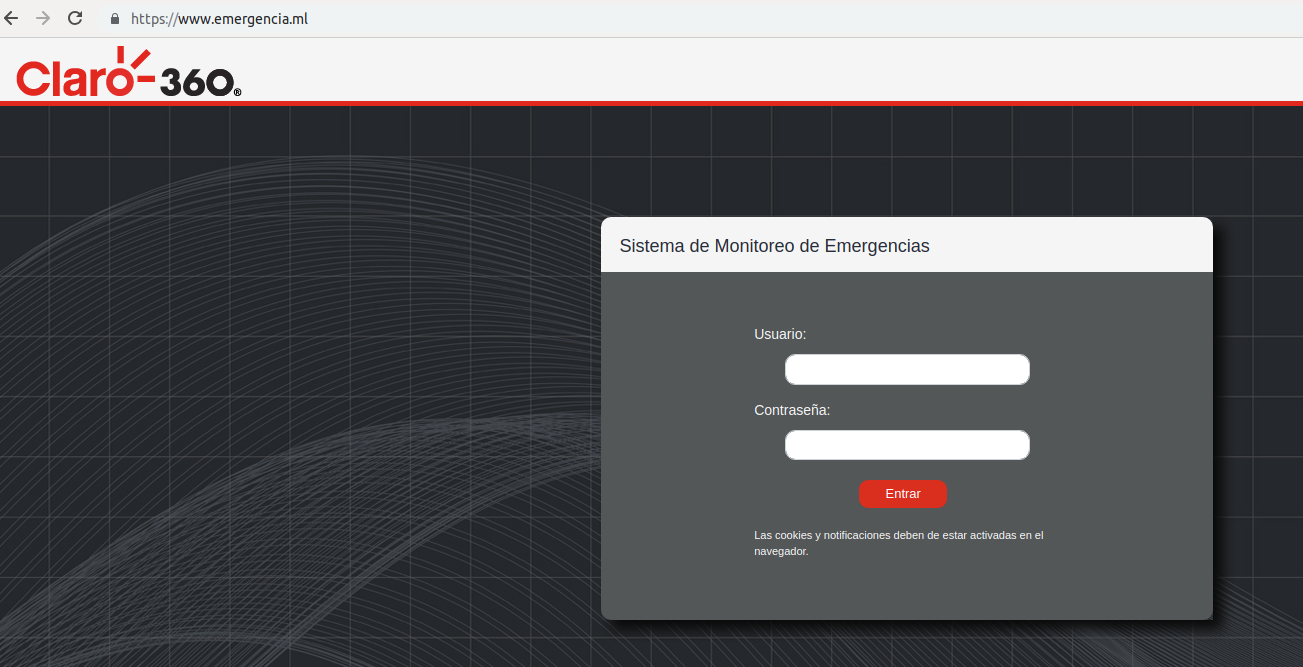


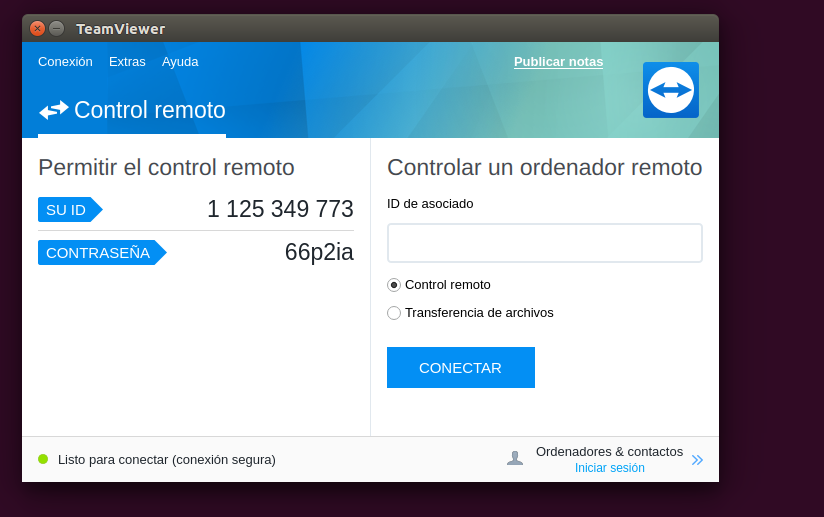


Verificamos el **https** valido.



Revisamos la aplicación de emergencias.





**apache**

<https://desarrolloweb.com/articulos/instalar-ssl-letsencrypt-ubuntu-apache.html>

NGINX

[https://medium.com/espanol/a%C3%B1adiendo-ssl-a-tu-sitio-con-lets-encrypt-en-nginx-8c3ffe66aad2](https://medium.com/espanol/añadiendo-ssl-a-tu-sitio-con-lets-encrypt-en-nginx-8c3ffe66aad2)

KURENTO

<https://www.taylorpetrick.com/blog/post/https-nodejs-letsencrypt>