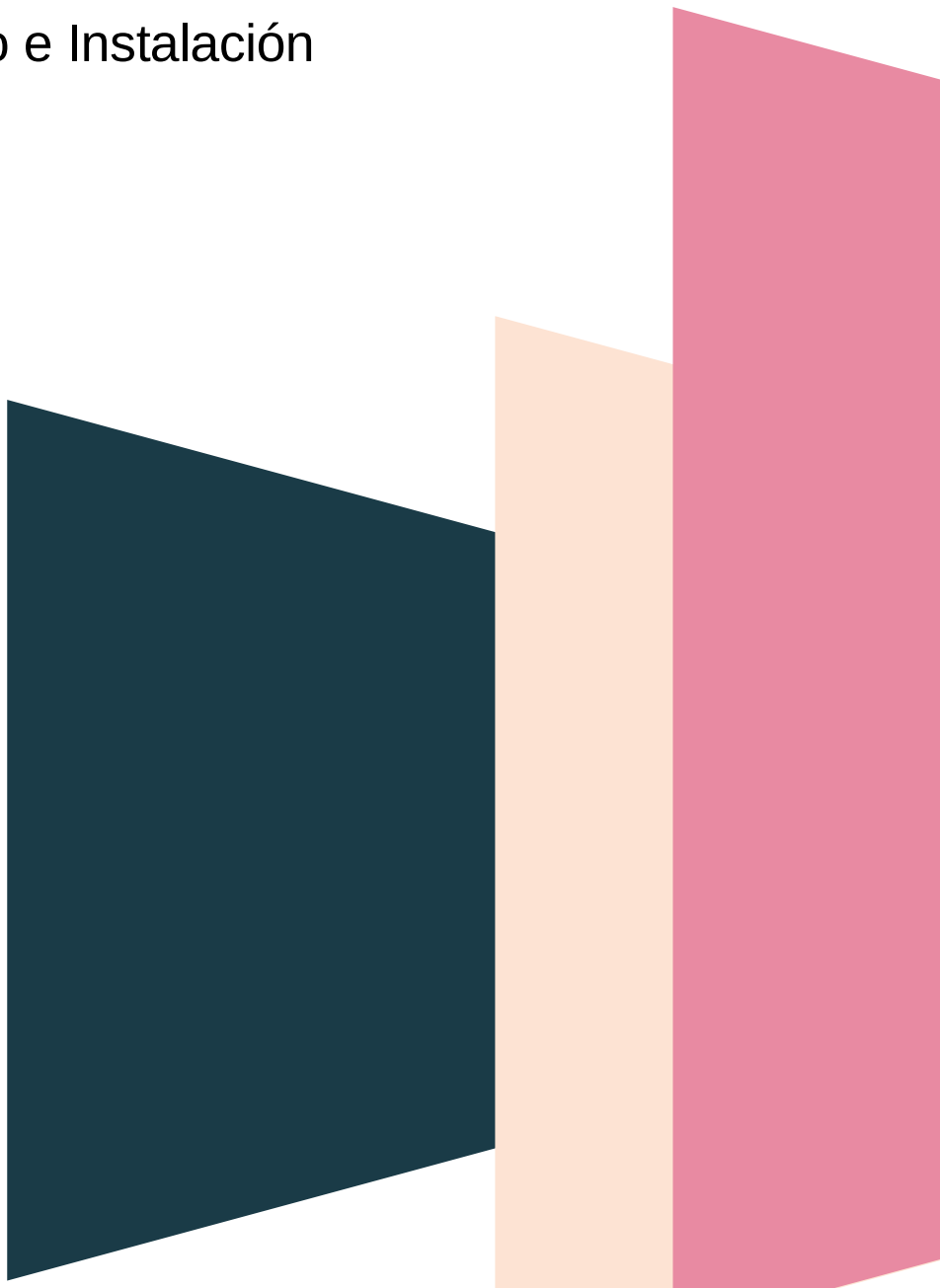
minsait

An Indra company

EDITRAN/XAdES

Firma XAdES-BES para TGSS
Unix
Manual de Usuario e Instalación

mayo de 2019



ecosystems
communications

1. INTRODUCCIÓN

1.1. OBJETIVO

El objetivo del documento es explicar la funcionalidad desarrollada en EDItran que permite recibir ficheros firmados de la TGSS, verificando su validez y extrayendo los datos que deben tratar las aplicaciones del cliente. Se detalla también la instalación y utilización del nuevo componente.

1.2. DESCRIPCIÓN

El protocolo definido por la TGSS para intercambiar los ficheros de pago R03 con las EEFF ha cambiado para incorporar la firma electrónica de los datos emitidos. Para adaptarse a esta nueva situación, EDITRAN ha implementado la posibilidad de verificar dicha firma en el extremo receptor.

Los pasos que describen este tipo de transmisiones se resume a continuación:

- ✓ TGSS firmará ficheros de aplicación norma 34 con 2 firmantes: uno del grupo de gestores y otro del grupo de interventores, ambos con certificados emitidos por la CA de la Seguridad Social. Las firmas son XAdES-BES con los datos *attached*.
- ✓ Las firmas se envían a las entidades receptoras vía EDITRAN con las especificaciones de sesión que indique la TGSS.
- ✓ En las EEFF, se definirá el grupo de tesoreros y el de gestores, cada uno de ellos con 3 posibles firmantes remotos (aunque el fichero sólo viene con una firma de cada uno de los grupos anteriores).
- ✓ Las EEFF en su proceso posterior a recepción, por cada fichero recibido en esa sesión:
    - Validará la firma, extrayendo los datos de aplicación asociados siempre que sea posible. En esta primera versión no se validará que los certificados hayan sido revocados.
    - En un segundo paso, sólo si la firma es correcta, se validará que los firmantes son los esperados. Es decir, que viene una firma de cada grupo y que son correctos los nombre y NIF de los firmantes.
    - Se genera un fichero de control asociado al recibido que contiene el resultado del proceso de verificación. Este fichero es el que deberán analizar las aplicaciones del cliente cuando generen el fichero de respuesta.

Las EEFF, con la información anterior, y con el tratamiento del fichero de aplicación, enviarán un fichero de confirmación (positivo o negativo), a la TGSS sin firmar.

2. INSTALACIÓN y REQUISITOS

2.1. Requisitos de instalación.

La instalación del nuevo módulo exige tener instalada una versión de EDITRAN 5.0 ó superior.

Por otra parte, debido a que alguna de las nuevas funcionalidades se han desarrollado en Java, hay requisitos adicionales a los de una instalación estándar de EDITRAN.

- Tener instalado correctamente Java 6 o superior. Si no lo tuviera puede descargarlo desde http://www.oracle.com/technetwork/java.
- Tener instalada la correspondiente extensión Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files.

2.2. Instrucciones para la instalación.

El cliente recibirá o se descargará de la página de EDITRAN un fichero **tar.gz** con el paquete para su SO. Dejar el fichero en el directorio dónde se vaya a instalar. Puede crear un directorio nuevo o bien dejarlo en el directorio dónde tenga instalado EDITRAN. En cualquier caso el producto creará el subdirectorio XAdES con el nuevo software instalado.

Suponiendo que el SO es Solaris y que lo vamos a instalar en el mismo directorio que EDITRAN, por ejemplo en */opt/editran*, seguir los siguientes pasos:

1. Dejar el fichero en */opt/editran*.
2. Situarse en */opt/editran* y descomprimir y extraer el software, ejecutando:

```
> gunzip xades-sparc-sun-solaris2.10.V##-2012-##-##.tar.gz
> tar –xvf xades-sparc-sun-solaris2.10.V##-2012-##.##.tar
```

3. Si no hay error en el paso anterior, se habrá creado el directorio XAdES con la siguiente estructura:

```
|XAdES
   |--lib
      |---boot
   |--logs
   |--rsc
      |---truststore
   |--bin
   |--utils
```

4. Dejar el entorno del usuario preparado para ejecutar aplicaciones java. Si no lo tuviera ya definido, incluir en .profile o .chsrc (dependiendo de la shell) las variables de entorno:

√ **JAVA_HOME=** ruta de instalación de java
√ Añadir a **PATH** $(JAVA_HOME)/bin

2.3. Módulos instalados

Los módulos que integran el producto EDITRAN/XAdES se ordenan en el siguiente árbol de directorios:

1. **XAdES/lib**: Contiene las librerías java que implementan el servidor para la verificación de firmas XAdES y la interfaz gráfica.
2. **XAdES/logs**: En este directorio se crearán los ficheros de log del sistema.

3. **XAdES/rsc**: Es el directorio dónde residen los recursos que necesita el servidor, como por ejemplo esquemas para la validación del formato XML de las firmas.
4. **XAdES/rsc/truststore**: En la instalación se suministra un almacén con la CA de la TGSS ya incorporada. Este fichero tiene como contraseña *"password"*. Este almacén es de tipo PKCS12 pero no contiene claves, sólo certificados que son públicos por lo que tampoco son necesarias grandes medidas de seguridad. No obstante, por si el usuario quisiera crear su propio *keystore* se proporciona también el certificado de la CA de TGSS: **rsc/truststore/CA-xades-TGSS.cer**.
5. **XAdES/bin**: Contiene los scripts para arrancar y parar el servidor y el cliente que lanzará desde EDITRAN/G las peticiones de verificación de los ficheros recibidos.
6. **XAdES/utils**: Con la instalación se proporciona una herramienta gráfica para manejar almacenes, claves, certificados, etc. La aplicación ***Portecle*** es *free software* y se incluye la distribución para Windows.

2.4. Scripts de arranque y parada del servidor

El script *XAdES/bin/start.sh* arranca el servidor java ThreadPooledServer para la verificación de las firmas. Este proceso es el que aparece en el resto del documento referenciado como Servidor/XAdES. En el script se da valor a las siguientes variables que el usuario debe modificar para adaptarlas a su instalación:

PWD= ruta de instalación
MEM_INI= memoria inicial destinada, en Megas
MEM_MAX= memoria máxima destinada, en Megas
IP= IP en la que arranca el servidor
PORT= Puerto en el que escucha el servidor

Siguiendo con el ejemplo del apartado anterior podría quedar así:

```
PWD=/opt/editran/XAdES
MEM_INI=1024
MEM_MAX=1024
IP="127.0.0.1"
PORT="7775
```

Con el script XAdES/bin/stop.sh se detiene el servidor enviando la señal SIGTERM al proceso. De igual manera que en el de arranque, hay que editarlo para dar el valor correcto a la variable PWD. En nuestro ejemplo:

```
PWD=/opt/editran/XAdES
```

Al ejecutar start.sh, si el servidor arranca correctamente creará el fichero **ThreadPooledServer.pid** con el pid del proceso. Además se deja información de log que refleja el estado del proceso. Por ejemplo:

```
/opt/editran/XAdES: >cd logs
/opt/editran/XAdES/logs: >cat out.log
24/04/2012 14:37:49,564     0 INFO  [main] servers.ThreadPooledServer main -
Starting Server
 24/04/2012 14:37:49,610    46 INFO  [Thread-3] servers.ThreadPooledServer
openServerSocket - Port:7775
 24/04/2012 14:37:49,611    47 INFO  [Thread-3] servers.ThreadPooledServer
openServerSocket - LocalSocketAddress:/127.0.0.1:7775
 24/04/2012 14:37:49,613    49 INFO  [Thread-3] servers.ThreadPooledServer
openServerSocket - InetAddress:/127.0.0.1
```

En el caso de que ya estuviera arrancado obtenemos la siguiente respuesta:

```
/opt/editran/XAdES/bin: >./start.sh
```
Error: The Server is already started

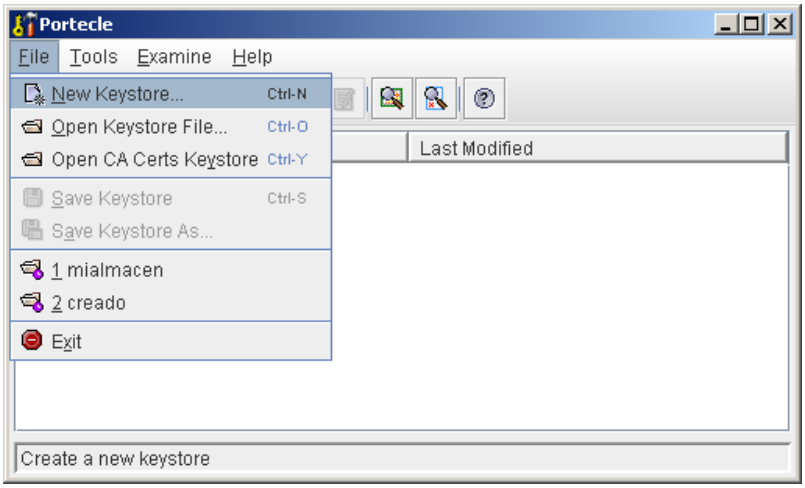
3. INSTALACIÓN CERTIFICADOS de CAs de TGSS

Si se decide crear un nuevo almacén hay que tener en cuenta que el tipo ha de ser PKCS12. En principio podría crearse con *keytool*, la utilidad que da Java para tratar *keystores*, pero si tiene problemas se recomienda instalar la herramienta Portecle que se proporciona con la instalación y seguir los pasos que se indican a continuación.

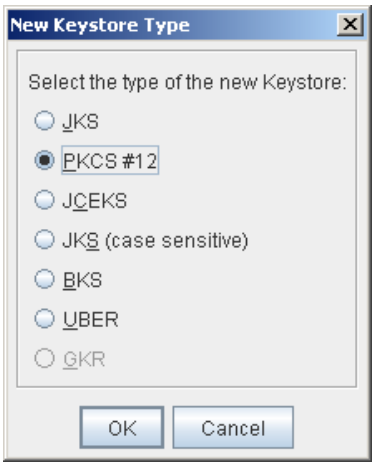
La utilidad Portecle se instalará en un equipo Windows y el fichero que guarda el almacén creado se llevará en binario a la máquina Unix de la instalación EDITRAN/XAdES.

3.1. Ejemplo de creación

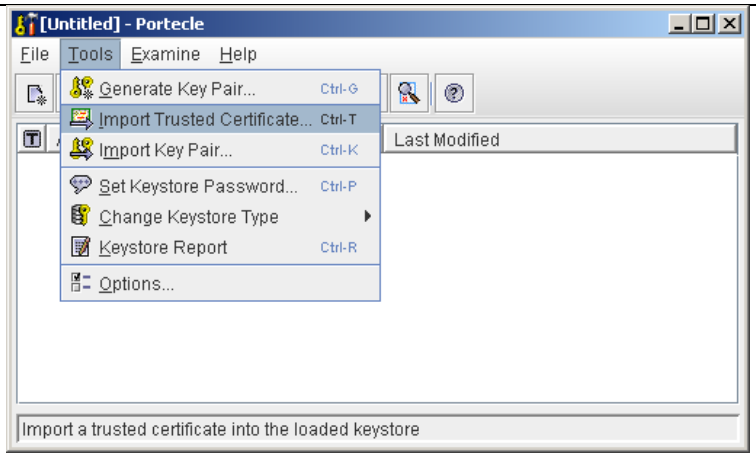
Una vez instalada la utilidad, ir a **Inicio>Programas>Portecle** y ejecutar Portecle. Aparecerá la siguiente ventana:



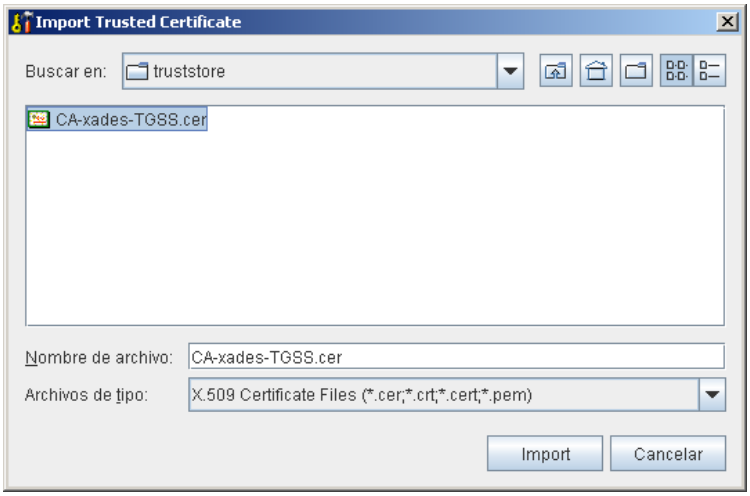
Seleccionar *New Keystore...* e introducir el tipo del nuevo almacén:



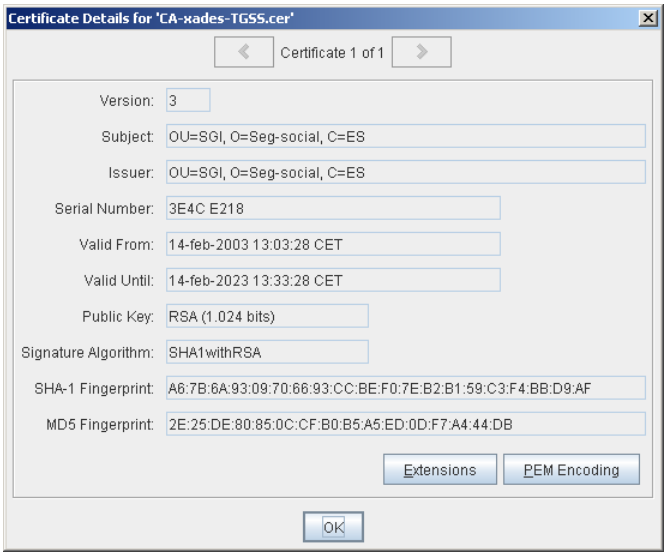
Una vez creado el almacén, incorporamos la CA de TGSS:



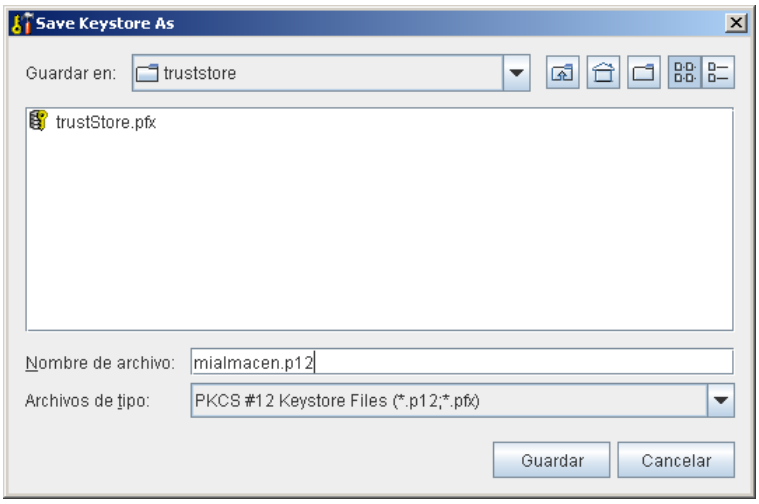
El certificado actual de la CA de la TGSS va con la instalación, si le dicen que debe incorporar alguna nueva, seleccionaría el fichero que haya recibido.



Al pulsar el botón *Import* le aparecerá la pantalla con los detalles del certificado que va a importar. Al pulsar *Ok* se le pide al usuario que confirme que es un certificado de confianza y que introduzca el alias que lo identifica, por ejemplo "catgss".



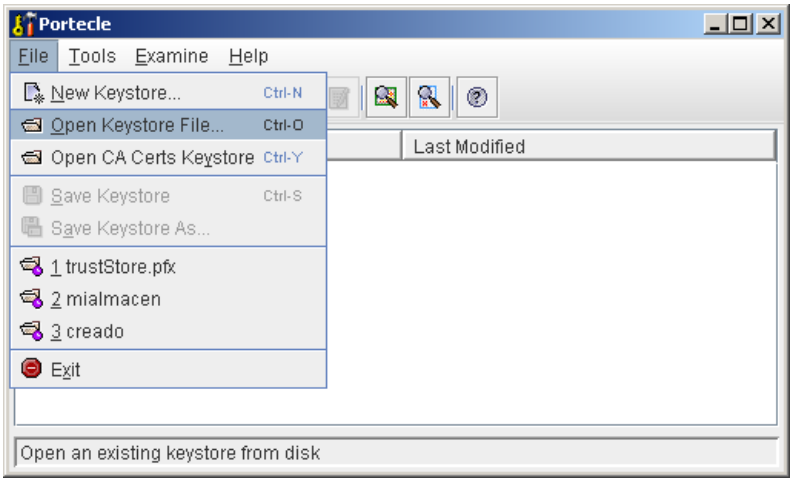
Ahora ya estaría incorporada. A continuación hay que guardar el almacén en disco: seleccione en el menú *File* de la ventana principal, la opción *Save Keystore As...*, se le pedirá la password y el path del archivo.



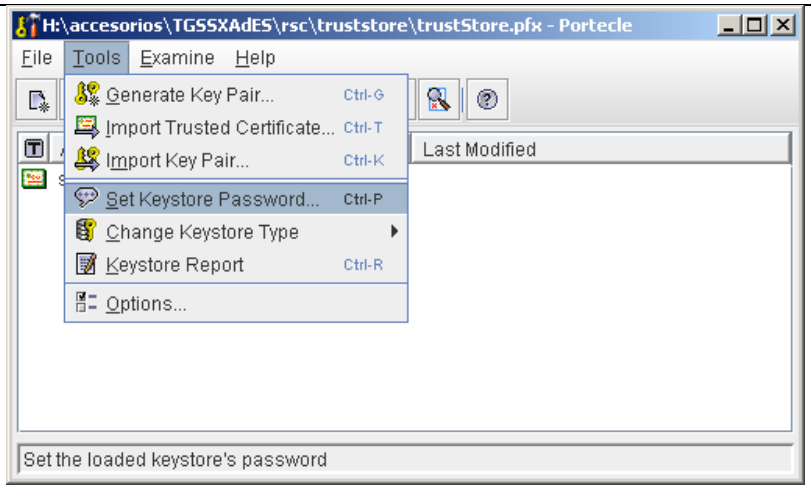
En este caso el fichero ***mialmacen.p12*** es el que habría que dejar en la máquina Unix, en el directorio ***XAdES/rsc/truststore***.

3.2. Ejemplo de cambio de password.

Una vez instalada la utilidad, ir a **Inicio>Programas>Portecle** y ejecutar Portecle. Aparecerá la siguiente ventana, en la que deberá seleccionar la opción *Open Keystore File...* del menú File.

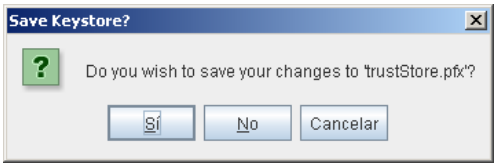


Abrir el almacén al que se quiere cambiar la password, para ello deberá conocer la password actual. Si este paso va bien, seleccionar la opción *Set Keystore Password* del menú Tools.



Se le pedirá que introduzca la nueva password:

Una vez modificada, podrá salvar los cambios con la opción Save Keystore del menú File. De todas formas, al hacer *Exit* si hay modificaciones pendientes, se le pedirá que confirme si quiere guardar los cambios:



4. INTERFAZ GRÁFICA

4.1. EDITRAN/G.

4.1.1. Sesión de presentación EDITRAN/G.

La TSGG establece algunos parámetros que deben seguir las sesiones EDITRAN para este tipo de transmisiones. A continuación se incluyen algunos otros que sólo afectan al lado receptor:

1. Destino de recepción. Este es el path dónde se dejaran los ficheros recibidos. EDITRAN/XAdES espera que los ficheros firmados tengan extensión **.xml**. Si el nombre en origen no la llevara, deberá poner en este campo algo similar al ejemplo:

   ./firmasTGSS/%o.xml

   De esta forma los ficheros se dejarían en el directorio firmasTGSS, con el nombre que tenían en origen y la extensión xml.

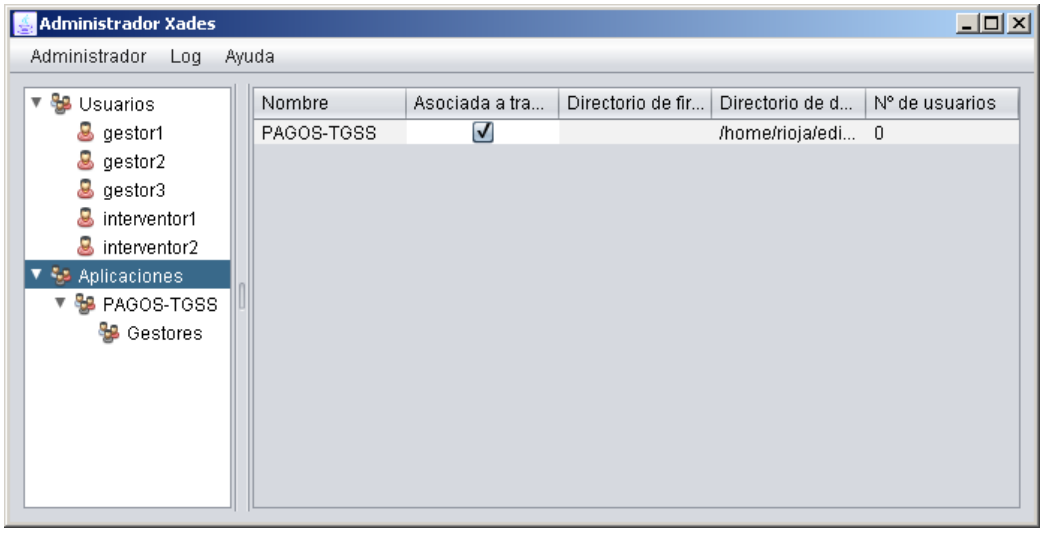
2. Procedimiento posterior a recepción: Indra proporcionará un procedimiento especial que realice la verificación de las firmas recibidas. Este procedimiento es un script por lo que en cualquier caso podrá ser modificado por las entidades si necesitan incluir otras operaciones.

4.2. EDITRAN/XAdES.

El objetivo principal de la interfaz gráfica de EDITRAN/XAdES es definir los firmantes posibles asociados a una Aplicación, entendiendo por Aplicación el concepto que identifica al conjunto de firmas de un mismo tipo procedentes de una misma entidad.

Los firmantes o usuarios pueden incorporarse a un grupo. El grupo se define cuando para dar por válida la firma nos sirve con que hayan firmado sólo algunos de sus integrantes.

El script *XAdES/bin/start_admin.sh* arranca la interfaz gráfica de EDITRAN/XAdES, realizada en Java. La pantalla principal de la interfaz es:



En la interfaz podemos ver en la parte izquierda un árbol con los perfiles que tenemos: los perfiles de usuarios y los perfiles de las aplicaciones, junto con los usuarios y grupos asociados a cada aplicación.

En la parte de la derecha podemos ver una serie de tablas que nos dan más información sobre los distintos perfiles.

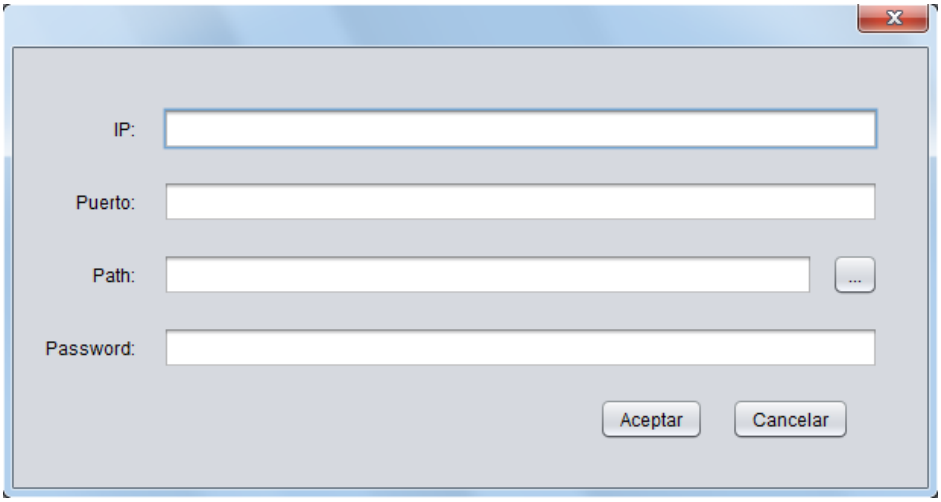
En el menú desplegable tenemos una serie de opciones distintas:

- Administrador: Podemos guardar o cargar los perfiles (en formato XML), y configurar los parámetros necesarios para comunicarse con el Servidor/XAdES.

- Log: Podemos consultar el log del VerificaXAdES.

4.2.1. Configuración Servidor/XAdES

En esta opción se introducen los valores configurables del servidor java que realiza la verificación de las firmas XAdES. Se despliega el menú en el que debe introducir los siguientes valores:



- **IP:** Es la IP del equipo donde reside el servicio java de verificación de firmas. En principio ésta aplicación se debe ejecutar en el mismo entorno que EDITRAN. Por lo tanto en este campo podrá poner o bien la dirección IP de su equipo o bien **127.0.0.1**.

- **Puerto:** Puerto de escucha del servicio java para la verificación de firmas. Este parámetro deberá coincidir con el valor dado a la variable *PORT* en el script *start.sh*.

- **Path del almacén de certificados:** Es la ruta del fichero keystore donde se guarda el certificado de la TGSS (CA).

- **Password:** Deberá introducir la password del keystore.

Estos datos quedan guardados en un fichero de texto plano llamado "xades.cfg".

4.2.2. Edición de perfiles.

Seleccionando en el árbol un elemento y presionando el botón derecho nos aparecerá un menú contextual donde podemos editar los perfiles. Para cada elemento, tenemos las siguientes opciones:

- **Usuarios:** Podemos añadir un usuario nuevo.
- **Usuario seleccionado:** Podemos editar ese usuario o borrar su perfil.
- **Aplicaciones:** Crear nueva aplicación.
- **Aplicación seleccionada:** Las opciones son asignar un usuario a esa aplicación, asignarle un grupo, editar aplicación o eliminar la aplicación.
- **Grupo seleccionado:** Se puede editar el grupo, añadir usuario a ese grupo o eliminar grupo.

4.2.2.1.Usuarios EDITRAN/XAdES.

Los usuarios posibles de cada grupo, los identificamos por un alias que lo identifica a nivel local y por el campo CN extraído del campo Distingued Name (DN) del certificado del firmante. Este campo es el que se utilizará en el proceso de verificación para comprobar que vienen los firmantes esperados.

En el caso de TGSS el CN tiene textualmente los dos apellidos y el nombre del firmante, luego hay un espacio, un guión, otro espacio y el número de DNI de la persona escrito siempre con 9 caracteres (8 dígitos + letra)

TGSS indica que en caso de que no vengan bien los firmantes, se envíe un fichero de confirmación con código de control 16 (Firmante(s) no autorizado(s) en la cuenta). Los CN normalizados de los firmantes los debería proporcionar TGSS.

Para crear un usuario, seleccionamos en el árbol el elemento "Usuarios" y en el menú contextual que aparece al clicar con el botón derecho, seleccionamos "Nuevo usuario" y veremos la siguiente interfaz, en la que podemos dar de alta:

Esta misma interfaz nos aparece si seleccionamos un usuario y en el menú contextual seleccionamos "Editar usuario".

4.2.2.2.Aplicación de EDITRAN/XAdES.

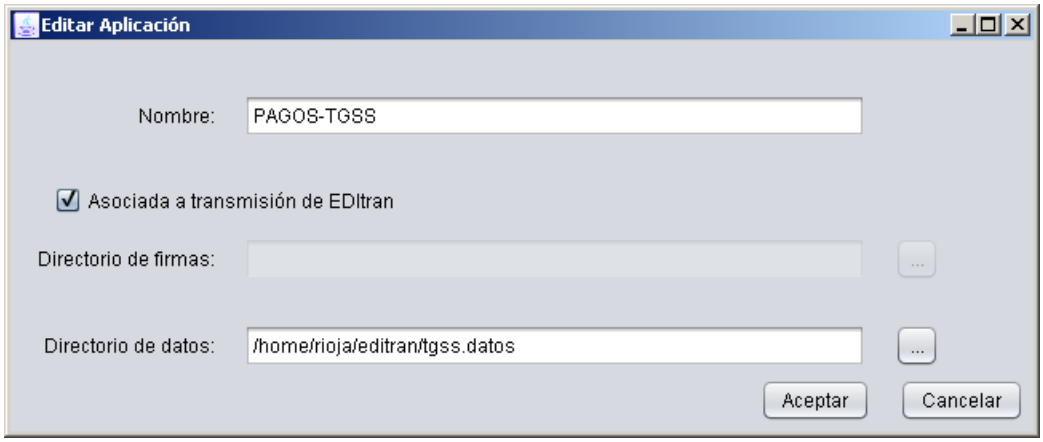
TGSS podrá emitir en cada transmisión, n ficheros a tratar. Cada uno de los XML transmitidos contiene 2 firmas autorizadas. No puede contener ni más firmas ni menos (excepto que TGSS indique lo contrario). Una de ellas, pertenece siempre al grupo de Gestión de TGSS. La otra, pertenece siempre al grupo de Intervención de TGSS. A su vez, hay 3 posibles firmantes en cada uno de los grupos descritos.

En base a lo anterior, mientras TGSS no notifique lo contrario:

- No puede haber 1 única firma en el fichero XML, ni 3. Sólo puede haber 2.
- No puede haber 2 firmas en el fichero XML que pertenezcan por ejemplo a intervención ó a cualquier otro grupo distinto. Deben pertenecer una a Gestión y la otra a Intervención.
- No puede haber firmantes que no conozcamos entre los 3 posibles de cada grupo que haya indicado TGSS.

Desde la interfaz gráfica de EDITRAN/XAdES, en este caso se deberá configurar la aplicación para que se ajuste a los requisitos anteriores.

Para dar de alta una aplicación, debemos elegir en el elemento "Aplicaciones" la opción "Nueva aplicación" de su menú contextual.



El campo Nombre es el alias local con que identificamos la aplicación. Si la marcamos como asociada a transmisión de EDITRAN, el valor dado en Nombre debe coincidir con el nombre de la Presentación en EDITRAN/G. Además, no tendremos que seleccionar un directorio de firmas puesto que los ficheros tratados serán los recibidos por EDITRAN.

El Directorio de firmas, en caso de no marcar la aplicación como Asociada a una transmisión de EDITRAN, es un directorio en el que se verificarán todos los ficheros que se encuentren en él, como si fueran archivos firmados.

El Directorio de datos es aquel en el que el usuario indicará el path donde se dejan los ficheros de datos extraídos de las firmas validadas.

A cada aplicación le podemos asignar los usuarios y grupos que pueden firmar la aplicación.

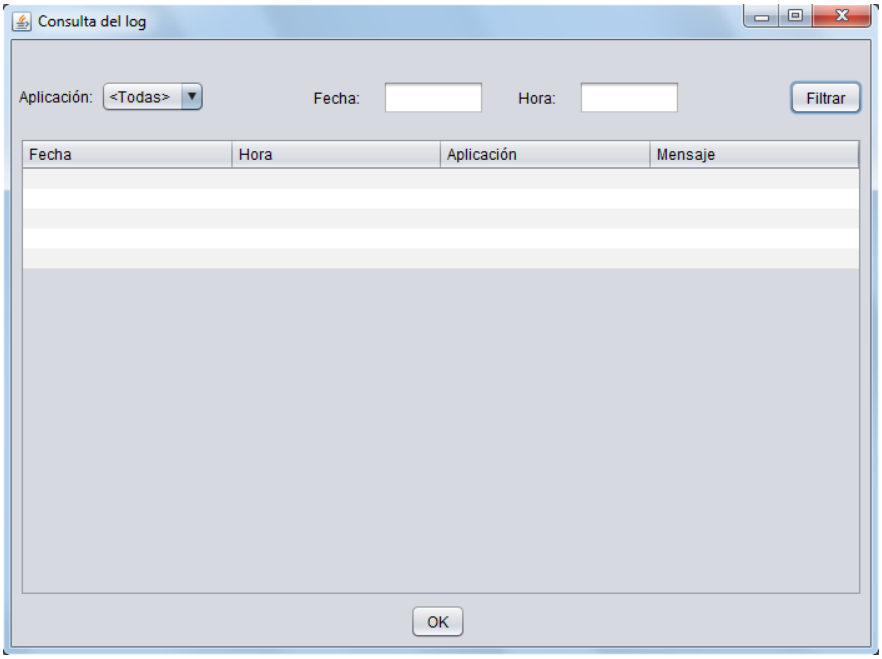
4.2.2.3.Grupos asociados a aplicación de EDITRAN/XAdES

Seleccionando una aplicación, podemos asignarle un grupo, en el que debemos elegir los siguientes parámetros:

- **Grupo:** Nombre del grupo.
- **Firmas obligatorias:** Número mínimo de firmas obligatorias.
- **Firmas permitidas:** Número máximo de firmas permitidas.

4.2.3. Consulta de Log

En la pestaña de log, podemos hacer una consulta del log de VerificaXAdES. La interfaz es la siguiente:



Podemos filtrar el log por aplicación, fecha y hora. Además, haciendo doble click en cada fila podemos ver en más detalle el mensaje de log.

5. PROCEDIMIENTO POSTERIOR A RECEPCION

La incorporación de la validación de las firmas XAdES al proceso de descarga de EDITRAN/G se hará en el proceso de usuario posterior a recepción. Para facilitar su integración en los clientes, se les facilitará un script estándar que realice las tareas obligatorias. No obstante, el cliente podrá modificarlo y adaptarlo a su instalación para añadir otros pasos. El procedimiento posterior a recepción a medida para estas transmisiones con TGSS se llama ***PSTRCV_TGSS.bash*** y tiene los siguientes pasos:

- Registrar en el fichero de salida de EDITRAN/G información general del resultado de la descarga de ficheros.

```
echo "##############################################################################"
echo "SCRIPT POSTERIOR A RECEPCION TGSS: $0."
echo "`date`"
echo "PRESENTACION : $2-$1 ($4-$5-$6)"
echo "RESULTADO    : $2-$3"
echo "##############################################################################"
```

- Si la descarga ha finalizado correctamente, ejecutar la aplicación ***VerificaXades.exe*** que válida las firmas de todos los ficheros recibidos. Al comando se le pasarán como argumentos: el nombre de la Aplicación que en este caso deberá coincidir con el nombre de la Presentación y el path del fichero que contiene la lista de ficheros recibidos (opcionalmente se puede utilizar además el parámetro -DNn para omitir la validación de los DNs de los certificados). La sintaxis y funcionalidad detallada de este proceso se documenta en el siguiente apartado.

```
if [ "$3" != "0000" ]; then
    exit 0
fi
flist=`pwd`/${4:0:8}.${4:8:1}/${5:0:8}.${5:8:1}/$6$2.ficheros

xadesp=`dirname $0`
if [ -n $xadesp ]; then
    cd $xadesp
fi

VerificaXades $1 $flist > VerificaXades.out
```

- El proceso VerificaXades va dejando en la salida estándar el path de los ficheros de control generados con el resultado de la firma del fichero de datos asociado. En nuestro script de ejemplo, lo que hacemos es registrar el contenido de cada fichero de la lista en la salida de EDITRAN/G y borrarlo. Este es el punto que posiblemente deberá modificar el cliente para enlazar con su proceso de generación del fichero de respuesta para TGSS. En el apartado 5.2 se describe el formato de los ficheros de control generados por cada firma recibida.

```
#funcion para procesar fichero de control
processFile() {
    # Esto es un ejemplo, el cliente necesitara procesar
    # el fichero
    echo $1:
    cat $1
    rm $1
}

exec 3<VerificaXades.out
while read linea <&3; do
    processFile $linea
done
rm VerificaXades.out
```

5.1. Proceso VerificaXades

5.1.1. Sintaxis

```
Uso: VerificaXades <aplicación_xades> [<lista_firmas>] [-DNn]
```

5.1.2. Descripción

Es un proceso cliente que lanza peticiones al proceso Java Servidor/XAdES para verificar la firma de cada fichero descargado. En el caso de que la *<aplicación_xades>* no estuviese asociada a la transmisión de EDITRAN no sería necesario pasar el argumento *<lista_firmas>* puesto que en este caso el usuario introduce en el perfil el directorio dónde buscar las firmas. El parámetro -DNn es opcional y sirve para omitir la validación de los DNs encontrados en el fichero contra los DNs de los usuarios que están en los perfiles. En este caso se mantiene, no obstante, la validación respecto al número de firmas máximas y mínimas que deben encontrarse en los ficheros verificados, según lo indicado en dichos perfiles.

Por cada firma XAdES se generan como ficheros de salida:

- El de datos siempre que se hayan podido extraer de la firma. Se dejarán en el directorio de datos que el usuario puso en el perfil de la *<aplicación_xades>* y se nombrará igual que la firma pero sin la extensión **.xml**.
- Un fichero de control asociado con el resultado del proceso de verificación. Este fichero se deja en el mismo directorio que los datos y se nombra igual que el fichero de firma añadiendo la extensión **.ctrl**.

El proceso de verificación irá dejando la traza de los ficheros que se verifican y del resultado en un fichero de Log que podrá ser consultado desde la interfaz gráfica. Ver apartado 4.2.3.

5.1.3. Códigos de retorno

Valor	Descripción
0	Proceso finaliza correctamente. Todas las firmas son válidas.
1	Error de sintaxis en la llamada al comando.
2	Error de acceso al perfil de la aplicación XAdES.
3	Error de licencia.
4	No hay firmas para validar.
5	Alguna de las firmas recibidas es incorrecta o los firmantes no son los esperados.

5.2. Fichero de resultado de la verificación

El fichero de control se dejará siempre que se haya obtenido alguna respuesta del servidor Java. Es el fichero que analizarán los procesos del cliente para saber cual ha sido el resultado de la correspondiente firma. Es un fichero de texto que tiene la estructura siguiente:

- La primera línea lleva el código de resultado y la descripción del error si el proceso ha fallado. En el apartado Anexo B se da la lista de los posibles errores.
- Las siguientes líneas son los DNs de los firmantes. El primer carácter de cada línea será una 'S' o 'N' para indicar si el usuario era uno de los admitidos según el

perfil que se definió para la Aplicación/XAdES. Cuando se ha utilizado el parámetro -DNn en la llamada, ese carácter será 'X', significando que no se ha hecho comprobación alguna con dichos DNs.

A continuación, se listan dos ficheros de ejemplo cuando el proceso ha finalizado:

a) Correctamente:

```
00
S CN=PEREZ FILEMON ANTONIO - 12345678B, OU=31, OU=PERSONAL EXTERNO, OU=ACGISS, O=Seg-social,
C=ES
S CN=GARCIA GARCILASO MARIA - 87654321A, OU=74, OU=PERSONAL EXTERNO, OU=ACGISS, O=Seg-social,
C=ES
```

b) Con error:

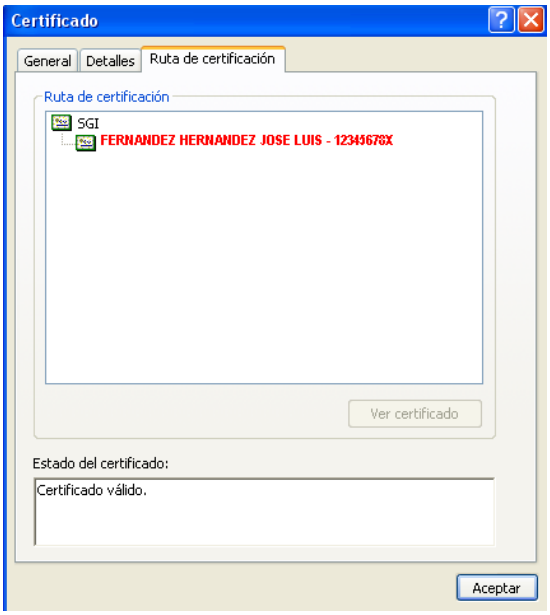
```
47 Error: Función peticionaria no reconocida
```

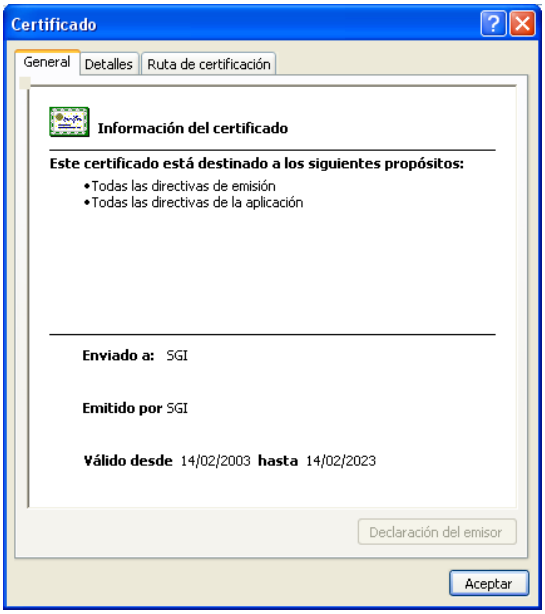
6. ANEXO A

6.1. Extraer certificado de la CA

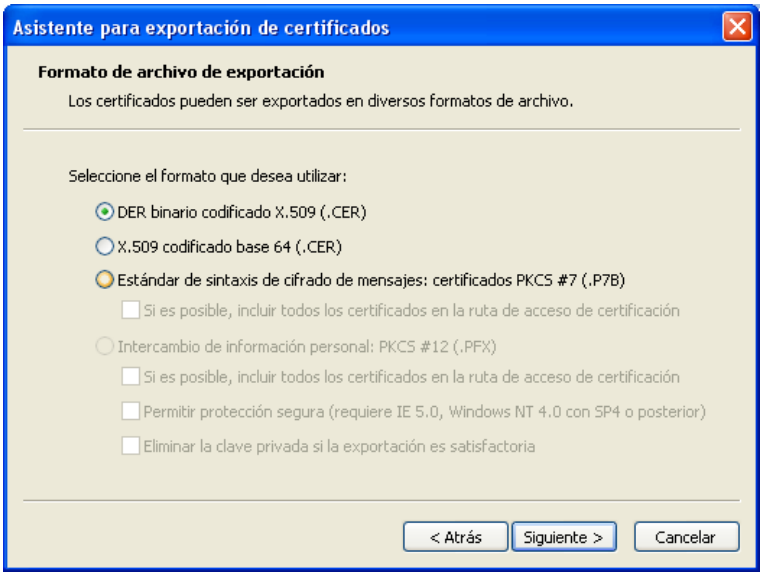
El propio paquete de instalación, lleva el certificado de la CA de la TGSS, por lo tanto, no es necesario realizar este proceso. Sin embargo, se describe el mismo, por si TGSS cambia de certificado.

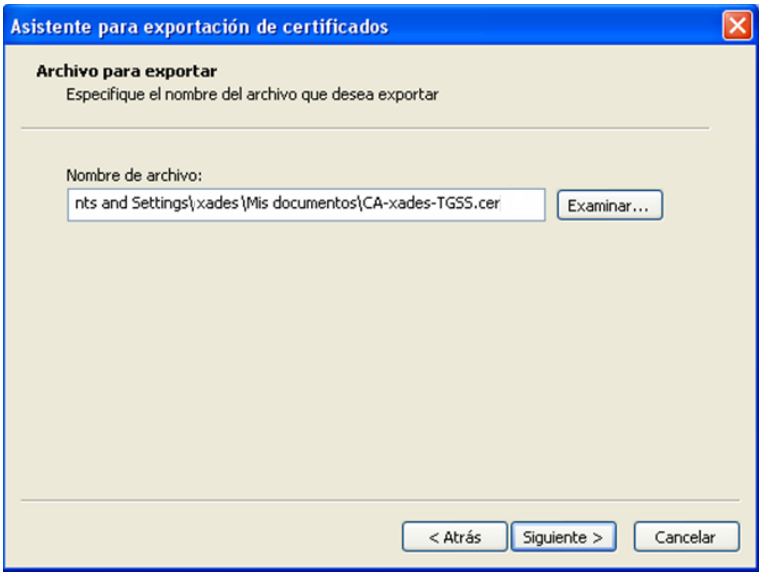
En caso de **no disponer** del certificado de la CA, lo podemos conseguir, cogiendo uno de los certificados de los firmantes, lo abrimos y vamos a la pestaña Ruta de certificación. Marcamos SGI (CA) y pulsamos doble click



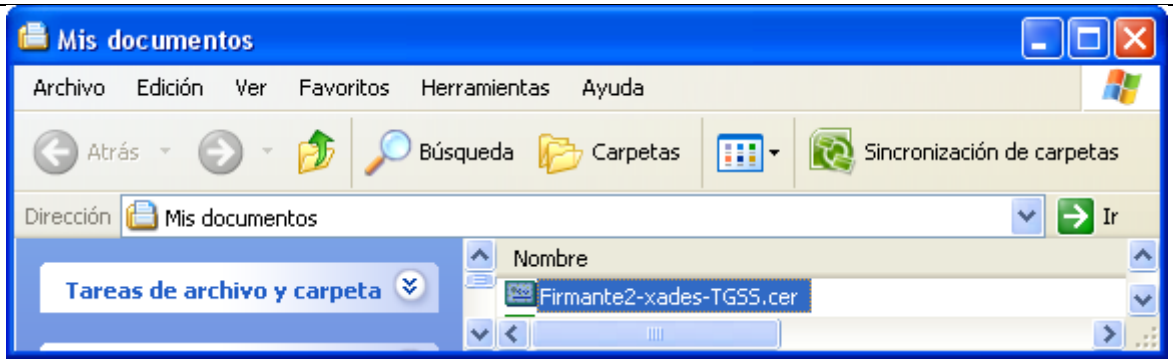


Vamos a Detalles, copiar en archivo y se abre otra ventana. Seleccionamos "DER binario codificado X.509 (.CER)" y lo guardamos





Ahora ya tenemos todos los datos necesarios



En caso que hubiera otras CA (parece que en esta primera fase no es así), repetimos la operación de exportar el certificado de otras CA. Si los 2 firmantes firman bajo la misma CA, no es necesario exportar la CA del segundo firmante.

Ya tenemos los certificados de la CA, c:\b2.cer, (c:\b3.cer si hubiera 2 CA...).
Una vez tengamos la CA, sea porque nos la envió TGSS o la extrajimos de un certificado, la incorporaríamos al almacén cómo se comentó en el apartado 3.

7. ANEXO B

7.1. Códigos de Resultado del servidor

00- Verificación de la firma correctamente
12- Autoridad certificadora no válida
13- Certificado(s) revocado(s). No implementado aún
14- Certificado(s) caducado(2)
15- Documento modificado
16- Firmante(s) no autorizados en la cuenta: Esto es una validación posterior de la firma contra los perfiles de EDITRAN/XAdES
47- Error inesperado en proceso servidor. Las causas por las que el proceso servidor puede fallar son las que se listan a continuación.

- Petición no construida correctamente.
- Error al leer del buffer de lectura la longitud de la trama.
- Error al leer del buffer de lectura la petición.
- Función peticionaria no reconocida.
- Solamente se validan firmas Attach.
- Lenguaje máquina origen no reconocido.
- Valor de validación CRLS no reconocido.
- Valor de zip no reconocido.
- Valor de cifrado no reconocido.
- Valor de codificación base64 no reconocido.
- Identificador fichero a validar no reconocido.
- Identificador fichero salida no reconocido.
- Identificador clave keystore no reconocido.
- Identificador del path del key storage no reconocido.
- La ruta del fichero a validar debe ser mayor que 0.
- La longitud de la ruta del fichero de salida debe ser mayor que 0.
- La longitud de la ruta del fichero de salida de DN debe ser mayor que 0.
- La longitud de la PASSWORD del keystore debe ser mayor que 0.
- La longitud de la ruta del keystore a validar debe ser mayor que 0.
- Error en la lectura del fichero origen.
- Error al analizar el xml.
- No se encuentra el elemento Signature en el documento.
- Error en el análisis del elemento signature.
- Error en el proceso de validacion de la firma.
- Error obteniendo la información de firma no válida.
- No se utiliza.

- No se ha encontrado el certificado con el que se firmo dentro del documento.
- Error en la escritura del fichero destino.
- Error en la escritura del fichero dn.
- Error inesperado. + descripcion del error standard java exception
- Error leyendo el keystore.

Error OutOfMemoryError. + descripcion del error standard java exception: *Cuando se produzca este error, se debe aumentar los parámetros de asignación de memoria que están en el fichero* ***start.sh*** *de la instalación del producto.*

minsait

An Indra company

Contacto
editran@indra.es
T +34 91 480 80 80

Avda. de Bruselas 35
28108 Alcobendas,
Madrid, España
T +34 91 480 50 00
F +34 91 480 50 80

www.minsait.com