CryptosecOpenKey

Web Service de la RA

Manual de usuario

Rev. 2.0.2776

REALIA TECHNOLOGIES, SL 24 de mayo de 2022

Version	Fecha	Componentes	Breve descripcion del cambio
1.0.0	04/07/2014	Todo	Versión inicial
1.0.1	12/12/2018	Documentación	Cambio a formato LaTeX
1.0.2	11/02/2020	RA	Interfaz de usuarios y aprobacion de certificados
1.0.3	23/11/2020	TSA	Informes XLS y autenticación de usuarios
1.0.4	08/07/2021	WS	Web Services de la RA
1.0.5	10/11/2021	Todo	Restauración
1.0.5-2762	02/02/2022	RA	TLSv.x en la configuración SMTP
2.0.2774	17/02/2022	Todo	Archivar datos

Cuadro 1: Control de versiones

Índice

1. Introducción

Con el fin de integrar la operación de certificados con sistemas externos, se proporciona en la autoridad de registro (en adelante RA) un web Services seguro (en adelante WSS). El WSDL debe ser accesible en la siguiente URL

https://[IP_SERVER]/CryptosecOpenKey/wservices/WService?wsdl

Este WSS proporciona una serie de médos que harán posible la operación de certificados del mismo modo que un operador de certificados lo hace mediante la interfaz WEB. Únicamente no son compatibles con el WSS las políticas con aprobación web.

2. Configuración

El WSS de la RA es accesible mediante protocolo HTTP con autenticación de cliente. Por este motivo, se deben configurar los siguientes certificados para operar el WSS:

- Certificado CA de confianza: Para la consión segura SSL.
- Ooperador de certificado: para la autenticación de cliente incluyendo la clave privada correspondiente a certificado de operación de la RA.

Dependiendo de la tecnología usada en el lado cliente del WSS el modo en que se realiza esta configuración puede variar, en el caso de java, un modo de tratar el primer punto es añadir el certificado de CA al almacén de certificados de la JVM, por ejemplo con el comando keytool, como se muestra en (Fig. 1).

```
c:\>keytool.exe -import -trustcacerts -keystore C:\java\jdk8\jre\lib\security\cacerts -file C:\minicacertificate.cer
```

Figura 1: Añadir un certificado de CA al almacén de certificados de la JVM.

El segundo punto se puede lograr añadiendo la clave privada del operador como en la fig. 2.

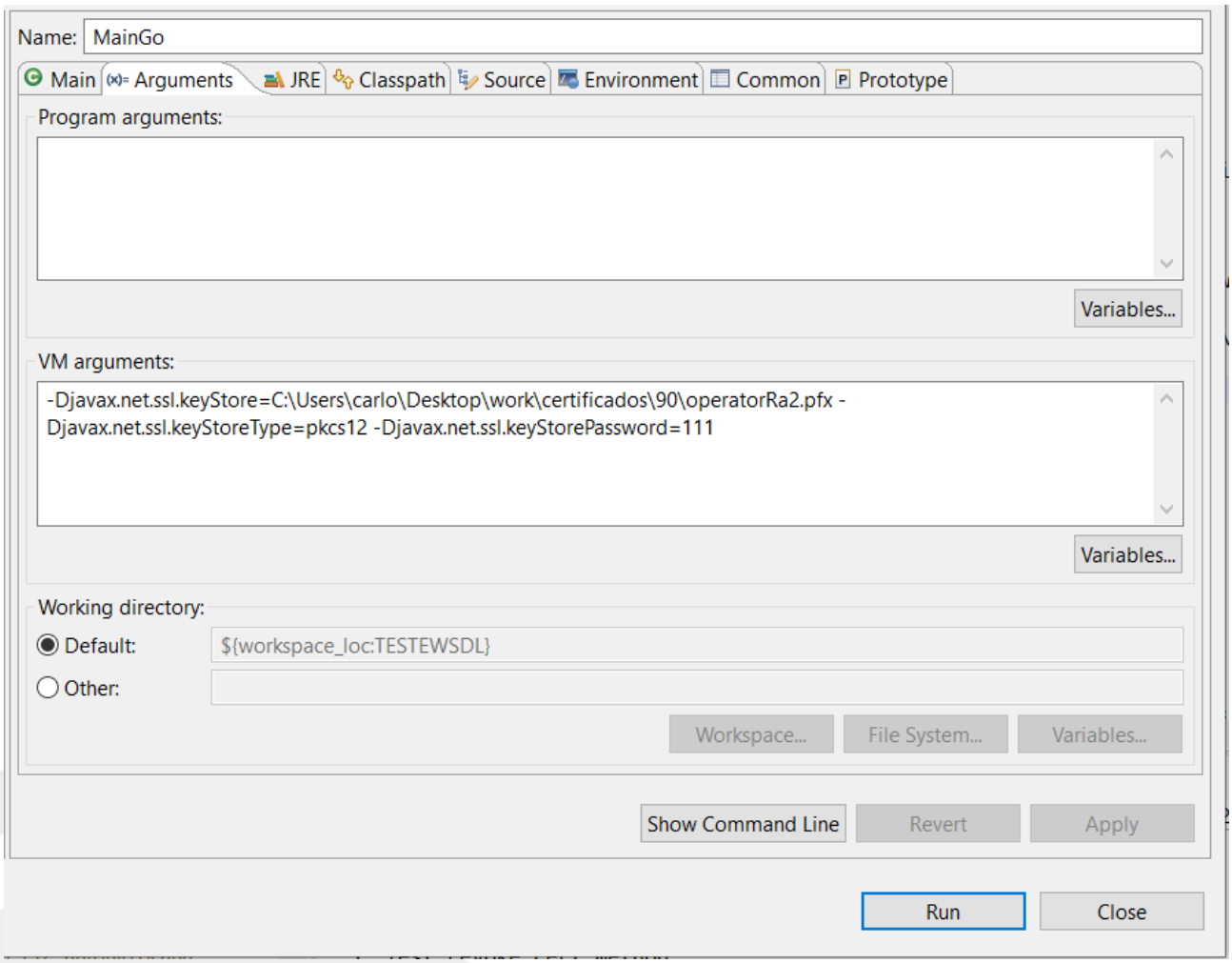

Figura 2: Añadiendo un certificado de operador a la JVM para configurar la autenticación de cliente.

El operador usado para la autenticación debe haber sido configurado por el adminsitrador de seguridad del mismo modo que cualquier operador de certificados de la RA: añadiendo roles dinámicos, tipos de usuarios, etc. De hecho, cualquier operador de certificados de la RA puede ser usado para autenticase con el WSS.

3. Métodos remotos

Los métodos que constituyen el WSS son

- **authorityList**: obtiene una lista de autoridades disponibles.
- **policyList**: obtiene una lista de políticas disponibles.
- **policy**: obtiene una lista detallada de las variables de una política seleccionada.
- **genCert**: generación de certificados.
- **statusCert**: obtiene un código de estado del certificado seleccionado.
- **revokeCert**: ejecuta una operación de revocación del certificado seleccionado.

- **suspendCert**: ejecuta una operación de suspensión del certificado seleccionado.
- **unsuspendCert**: ejecuta una operación de reabilitación de un certificado previamente suspendido.
- **signature**: calcula la firma de un hash (RSA).

Vamos a ver, en el resto de este documento cada uno de estos métodos en detalle.

3.1. authorityList

Este método no tiene ningún parámetro, puede usarse para obtener un array de string que representan las autoridades disponibles en la firma de un certificado.

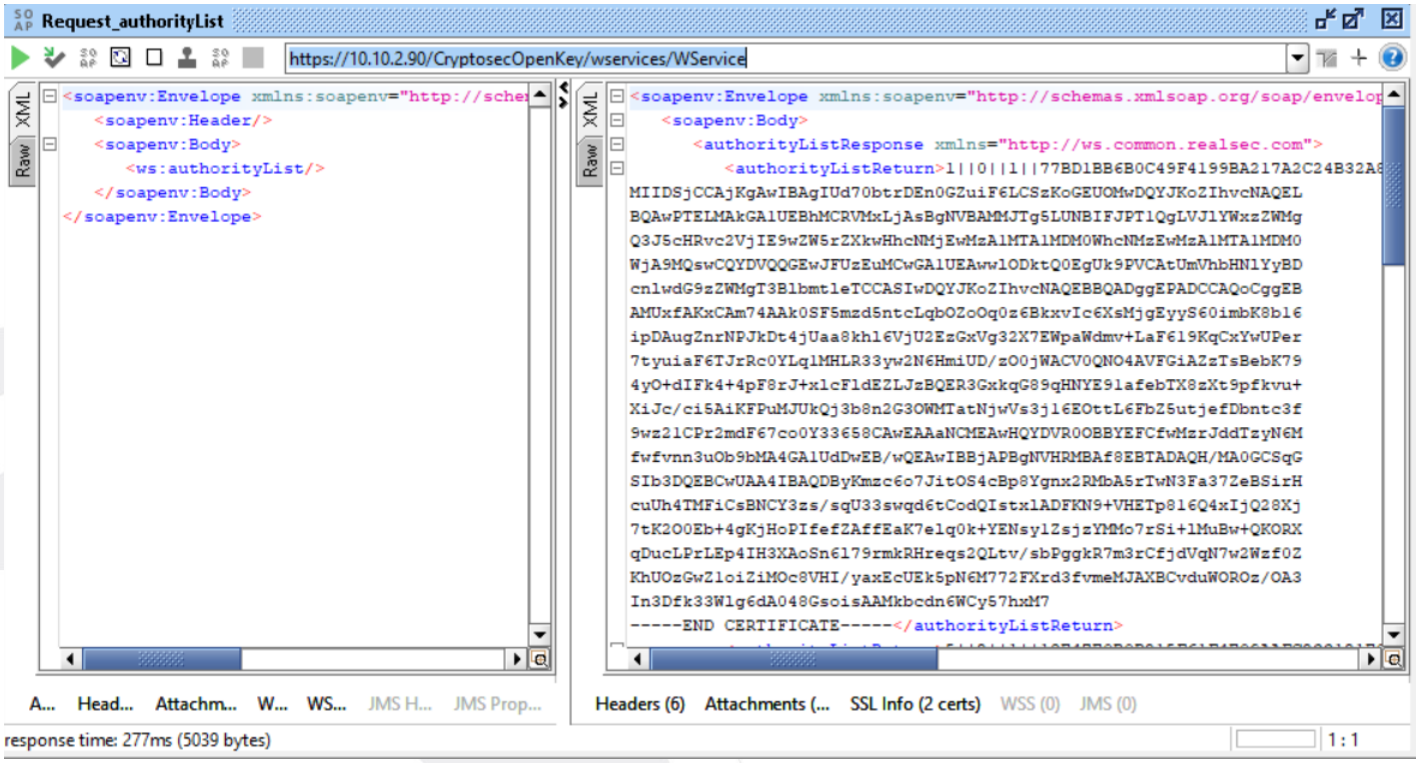

Figura 3: Ejemplo de petición authorityList

El valor retornado es un array de string con la representación de cada autoridad: id, índice, certificado y otras propiedades del objeto authoridad.

3.2. policyList

Este método no tiene parámetros, se usa para obtener un array de string representando los perfiles de certificado disponibles para el operador.

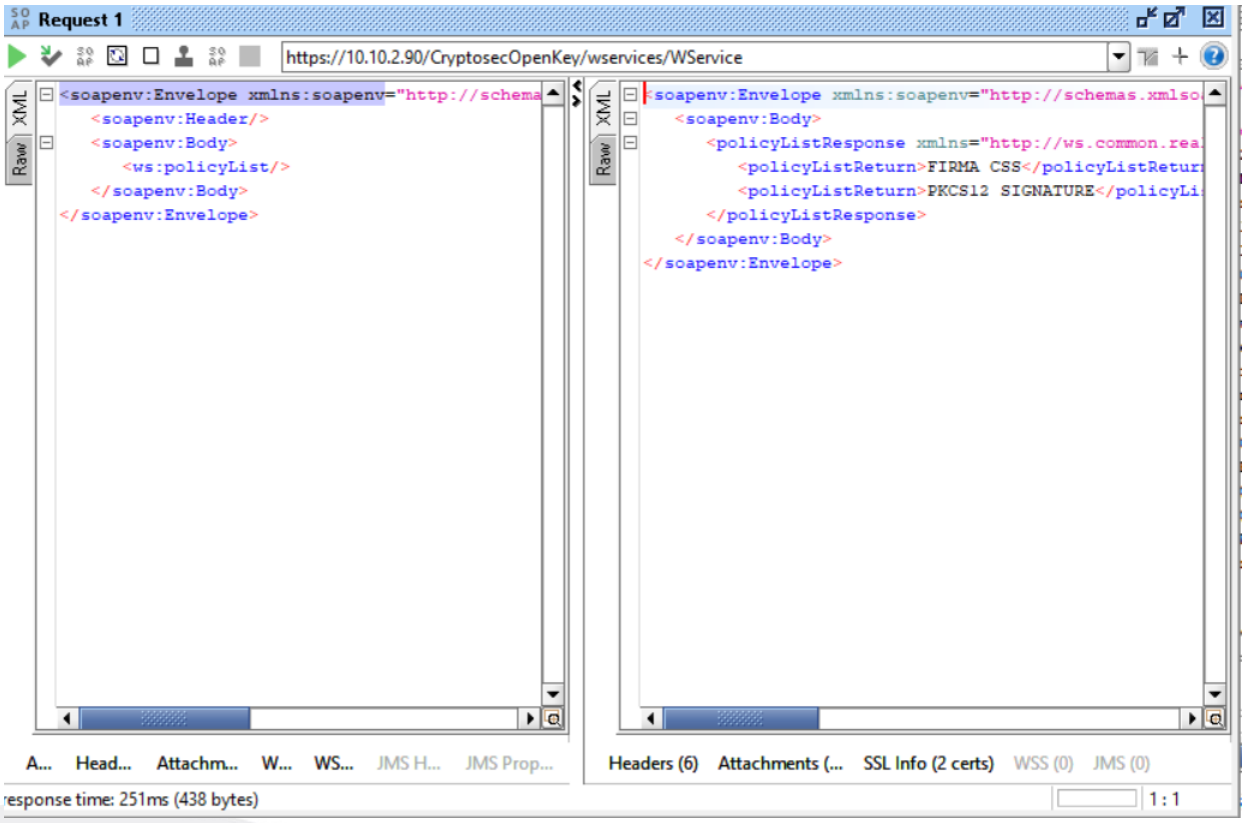

Figura 4: Ejemplo de petición policytyList

El valor retornado consiste en un array de nombre para cada política disponible.

3.3. policy

Obtiene una lista detallada de las variables de una política seleccionada. El único parámetro que admite este método es un string con el nombre de la política consultada.

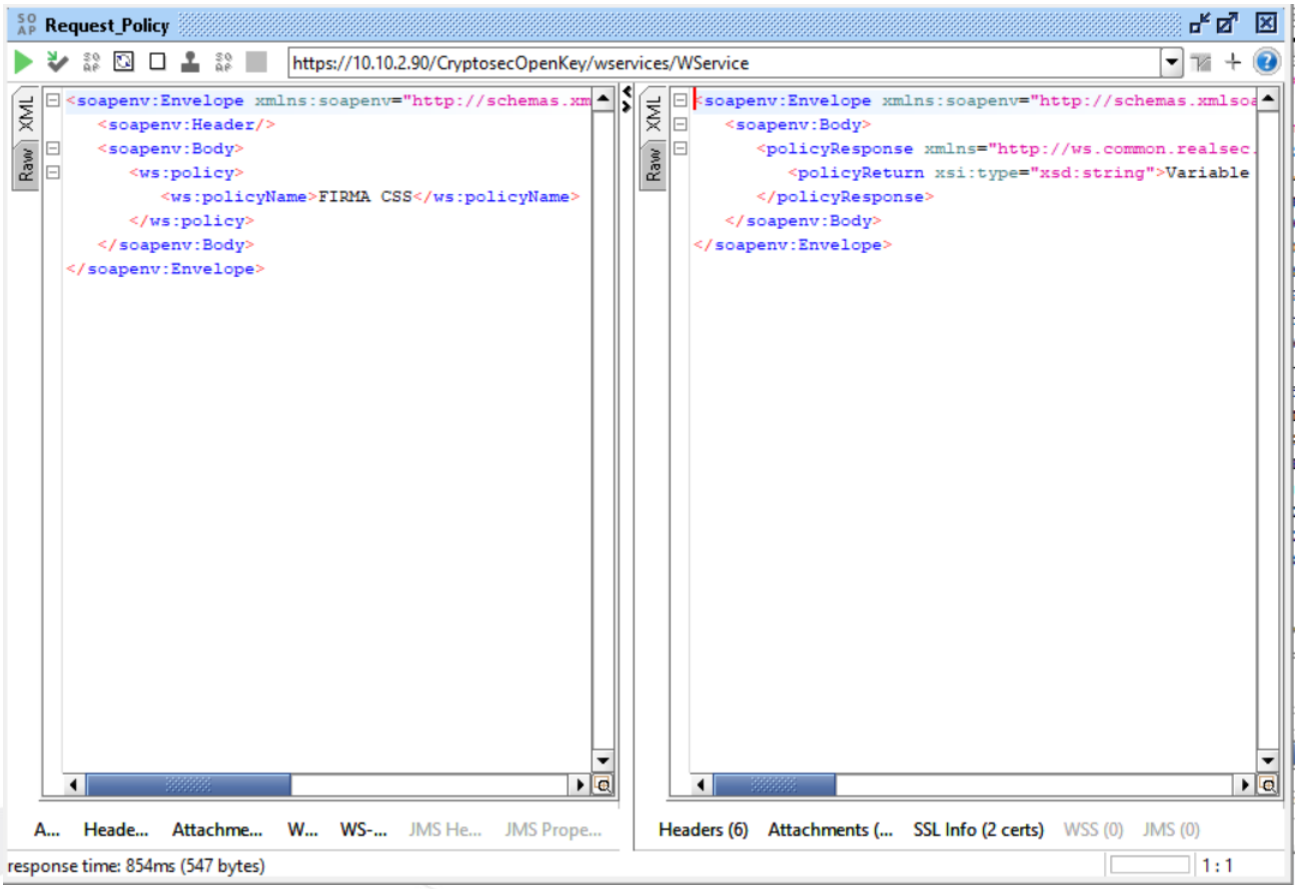

Figura 5: Ejemplo de petición policy

El valor retornado consiste en un array de string, cada uno representando un parámetro de la política.

3.4. genCert

Este método se usa para la generación de certificados. Los parámetros que admite son:

- **pkcs10**: string codificado en Base64 con CSR en PKCS10 .Puede ser null si la política de generación de clave no es PKCS10.
- **policyName**: string con el nombre de la política seleccionada.
- **params**: string representandp la lista de parámetros, codificados en la siguiente manera key1=value1||key2=value2||key3=value3||...
- **authorityIndex**: String con el nombre de la CA firmante.

Hay tres parámetros con nombres predefinidos que pueden usarse dependiendo de la request:

- **USER_TYPE**: El tipo de usuario definido en la RA.
- **USER_EMAIL**: El email del usuario para recibir notificaciones.
- **PKCS12**: El password del pkcs12 cunado la política tiene configurado PKCS12 como método de generación de clave.

El valor retornado por este método es un string en todos los casos, puede variar dependiendo de si el método de generación de clave es PKCS10 o PKCS12 como se muestra, respectivamente en Fig. 6 y Fig. 7.

en el caso de un PKCS10, el valor de retorno es como hemos dicho un string conteniendo el certificado codificado como DER.

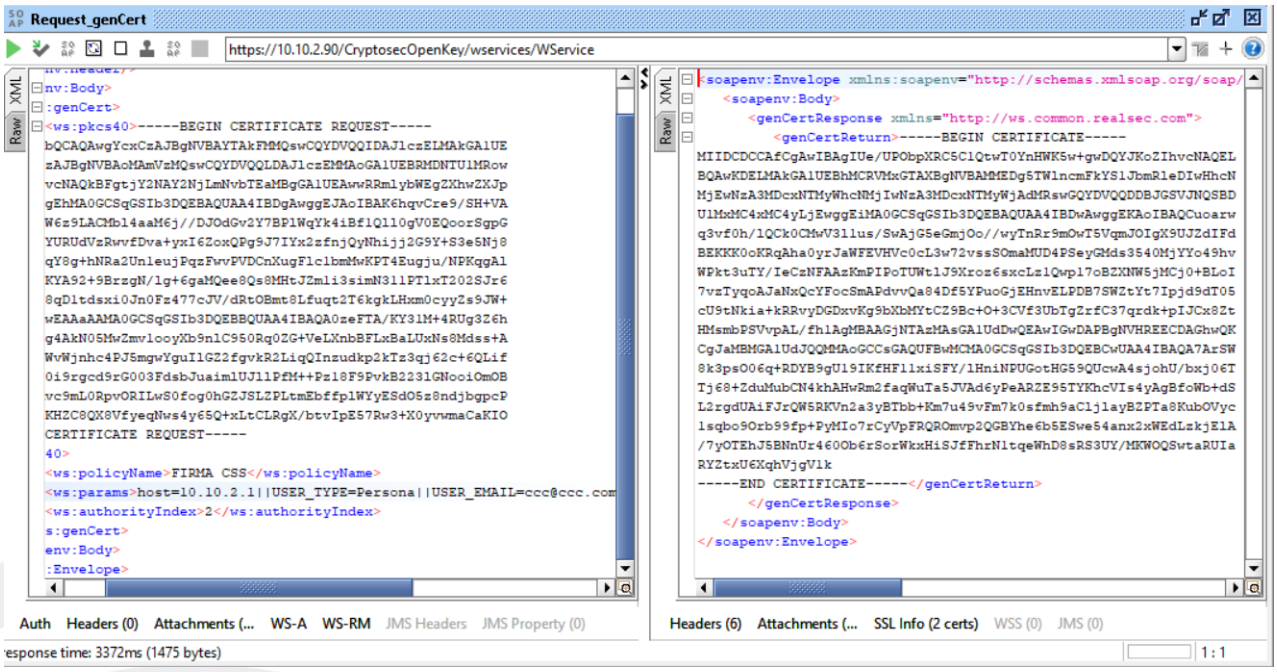

Figura 6: genCert petición PKCS10

en el caso de un PKCS12, el valor retornado es un string conteniendo la representación binaria del array de bytes que constituye el archivo PKCS12.

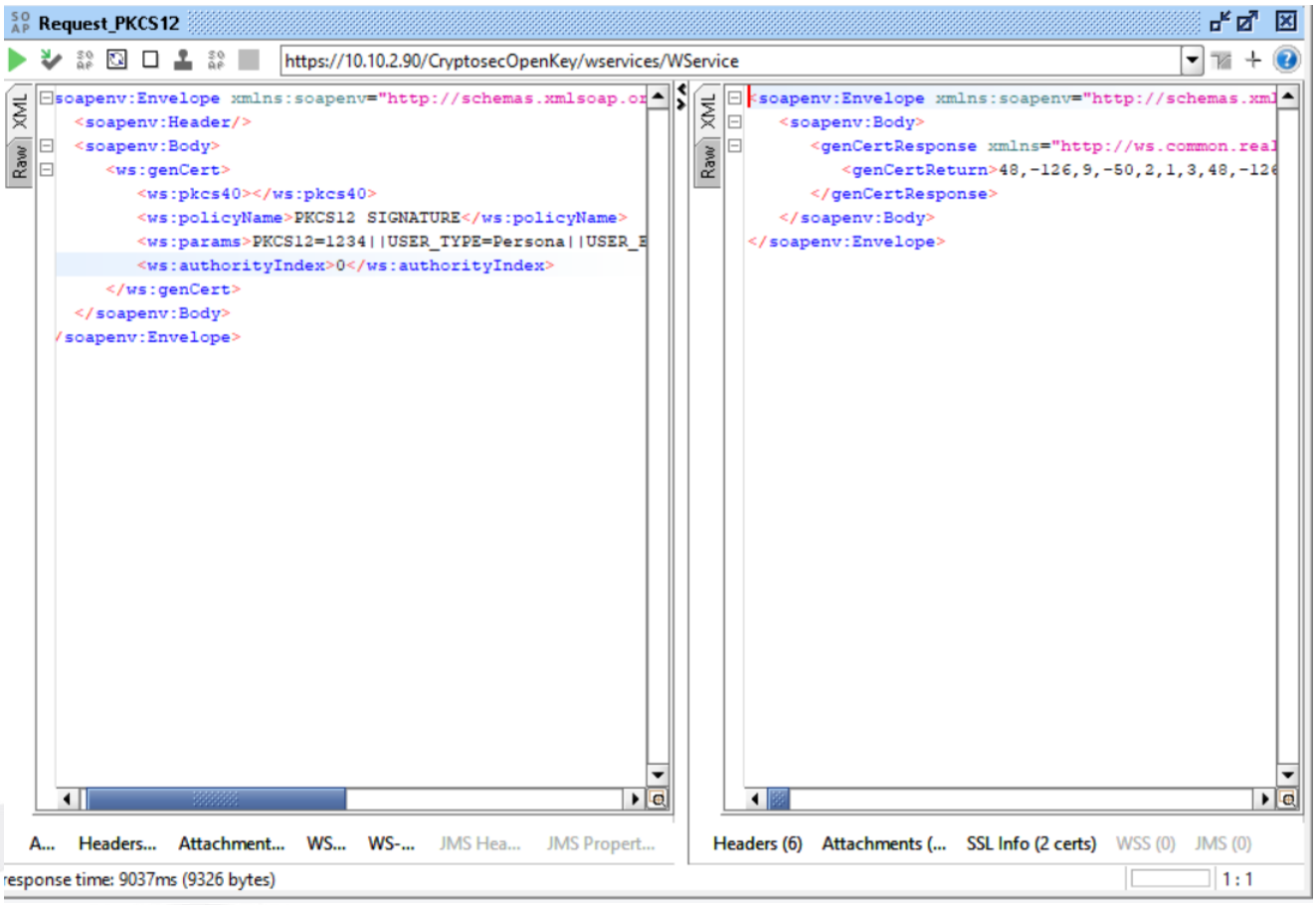

Figura 7: genCert petición PKCS12

3.5. statusCert

Obtiene un código de estado del estado de un certificado. El único parámetro es el serial number del certificado a consultar, en Base 16.

El valor retornado es un string cuyos posibles valores son:

1: Estado emitido.

2: Estado revocado.

3: Estado suspendido.

Error: certificado no encontrado.

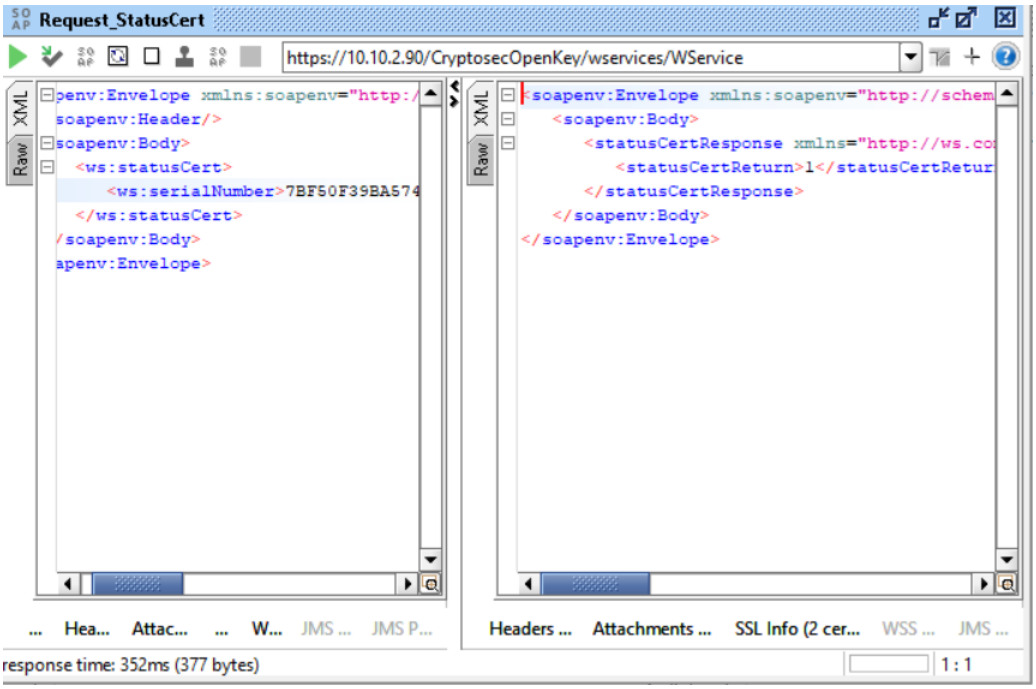

Figura 8: Ejemplo de petición statusCert

3.6. revokeCert

Ejecuta una operación de revocación de un certificado seleccionado. La lista de parámetros consiste en dos strings:

- **serialNumber**: Número de serie del certificado.
- **revokeReason**: String con el motivo de revocación.

Solo un valor de retorno "OK" nor dirá que la operación se ha realizado correctemente.

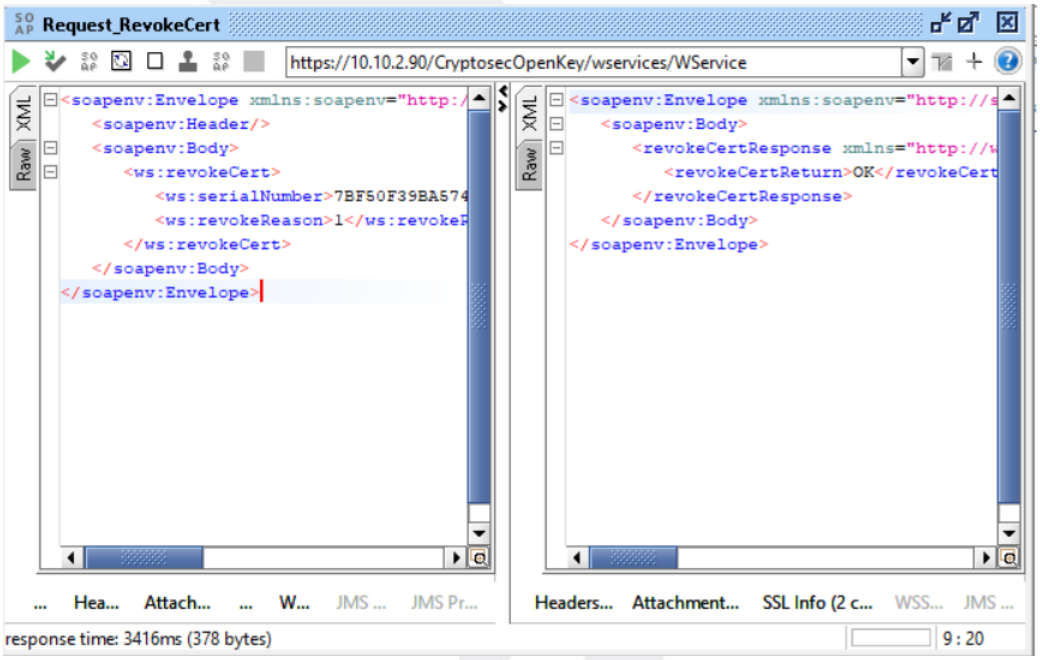

Figura 9: Ejemplo de petición revokeCert

3.7. suspendCert, unsuspendCert

La operación suspendCert cambia el estado de un certificado de 1 (issued) a 3 (suspended). recíprocamente, la operación unsuspendCert operation cambia el estdo de un certificado previamente suspendido de 3 (suspended) a 1 (issued). Both operations has only one parameter: the serialNumber as base16 encoded string.

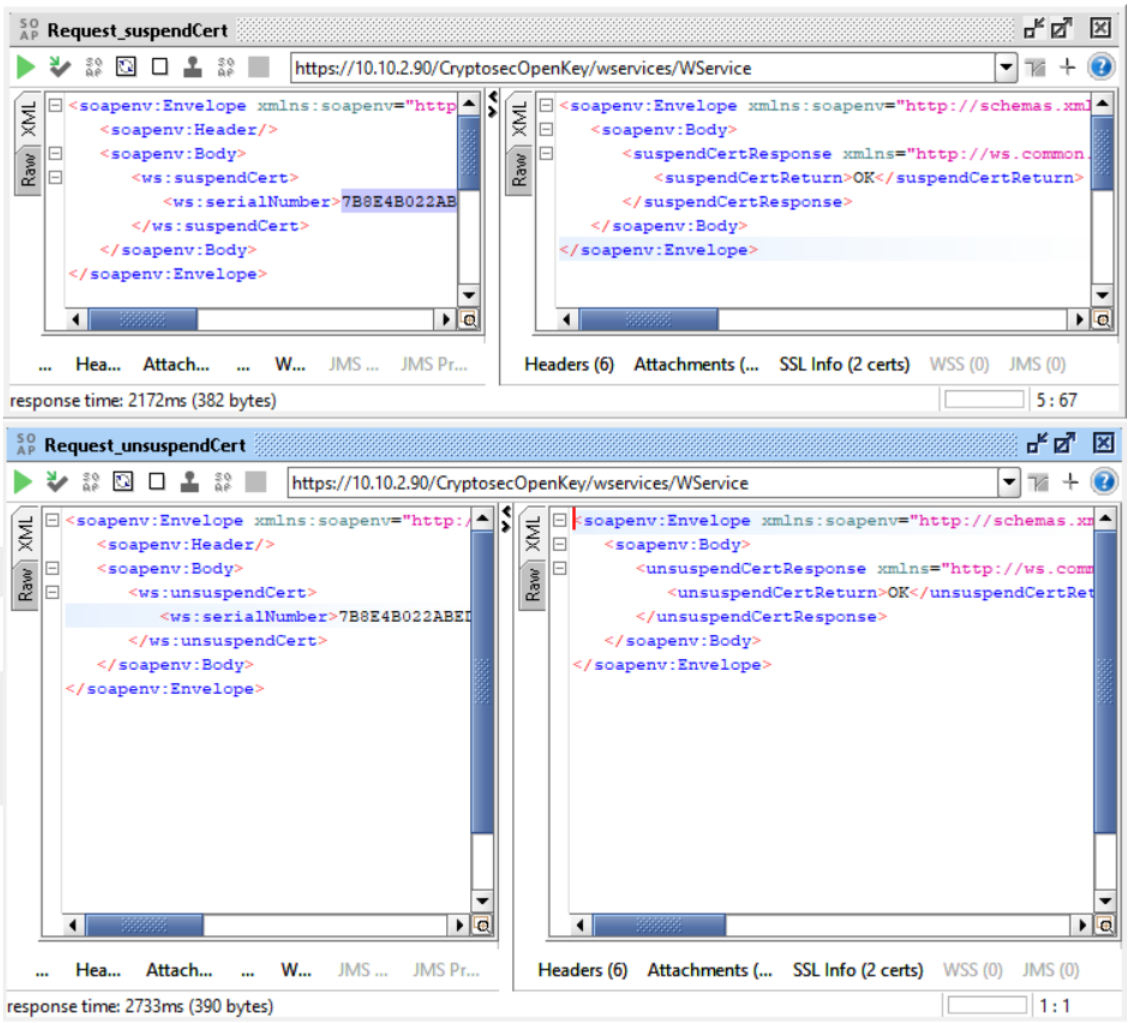

Figura 10: suspendCert, unsuspendCert ejemplos

De nuevo, solo un valor de retorno "OK" nor dirá que la operación se ha realizado correctemente

3.8. signature

Este médodo puede ser usado para calcular una firma RSA de un hash mediante el certificado de CA.

Los parámtros son dos strings:

- **dataToSign**: hash de los datos a firmar, codificados en base16.
- **authorityIndex**: Un string con el índice index de la autoridad seleccionada.

El valor de retorno esperado es un string codificado en base 16.

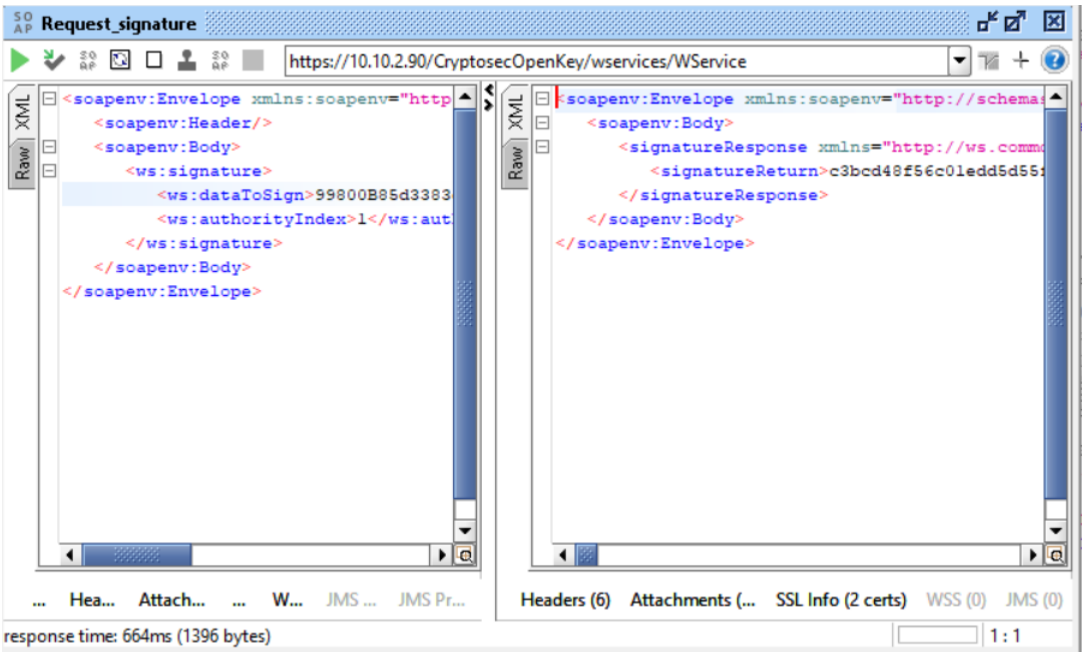

Figura 11: Ejemplo de una petición de firma.