Seguridad a nivel de mensaje. Encriptación y envío de login

Índice

1. Encriptación de mensajes SOAP

Para encriptar los mensajes vamos a modificar el documento WSDL de los proyectos `SWSeguroCliente` y `SWSeguroServicio` (por supuesto, va a ser el mismo documento para ambos). Inmediatamente después del cierre de la etiqueta `</sp:SignedParts>` vamos a añadir:

```
<sp:EncryptedParts>
        <sp:Body />
</sp:EncryptedParts>
```

Con eso indicamos que queremos que se encripte el cuerpo. Opcionalmente (pero es muy deseable) podemos pedir que que se encripte antes de firmar. Eso es deseable para evitar que alguien pueda construir un "diccionario", de textos y firmas asociadas, y usarlo para así calcular el contenido de un mensaje a partir de su firma. Si encriptamos antes de firmar el mensaje, el resultado de la encriptación será distinto cada vez, y la firma se calculará a partir de dicho resultado encriptado, con lo cuál el problema de seguridad está resuelto. Para indicar que la encriptación se hace antes de firmar, insertamos la siguiente etiqueta después del cierre de `</sp:AlgorithmSuite>`, pero antes de cerrar la `</wsp:Policy>`:

```
    <sp:EncryptBeforeSigning/>
```

Nota: Existen otras alternativas para solucionar el problema de seguridad con las firmas y la encriptación. Una sería insertar un valor aleatorio en el texto, así la firma nunca sería igual para el mismo mensaje. Otra forma sería indicar en el WSDL que la firma también debe encriptarse. Para hacerlo, tenemos que añadir, después de la sección `<sp:EncryptedParts>`, otra a través de la cual se indica que también encripten aquellos elementos XML del mensaje SOAP, cuyo nombre sea "`Signature`":

```
<sp:EncryptedElements>
        <sp:XPath>
                //*[local-name()='Signature']
        </sp:XPath>
</sp:EncryptedElements>
```

Esta última alternativa no la vamos a utilizar porque puede dar problemas con Axis2.

Ahora podemos eliminar los paquetes `es.ua.jtech.seguro` de ambos proyectos (¡pero no su subpaquete `es.ua.jtech.seguro.custom`!, que es donde mantenemos el código que no queremos que se sobreescriba durante la generación). Volvemos a generar el código (en ambos) y los refrescamos. Las clases `Seguro_SeguroSOAP_Client` y

`Seguro_SeguroSOAP_Server` que se generan de nuevo en el paquete `es.ua.jtech.seguro`, las podemos eliminar, ya que tenemos las nuestras en `es.ua.jtech.seguro.custom`. Vamos a abrir las nuestras, para añadir las propiedades de encriptación, indicando el nombre de usuario para encriptar (`cliente1` en el caso del servidor, y `servidor1` en el caso del cliente), y una vez más, el archvio "`crypto.properties`".

Empezamos por `Seguro_SeguroSOAP_Server.java`, añadiendo las propiedades:

```
     properties.put(SecurityConstants.ENCRYPT_USERNAME,
"useReqSigCert");
     properties.put(SecurityConstants.ENCRYPT_PROPERTIES,
"crypto.properties");
```

En `Seguro_SeguroSOAP_Client` añadimos las siguientes:

```
     properties.put(SecurityConstants.ENCRYPT_USERNAME,
"servidor1");
     properties.put(SecurityConstants.ENCRYPT_PROPERTIES,
"crypto.properties");
```

Se puede observar que en el servidor, en lugar de indicar el nombre "`cliente1`" para el `ENCRYPT_USERNAME`, hemos asignado "`useReqSigCert`", valor que indica al servidor que use el certificado del cliente para encriptar el mensaje de respuesta. Recordemos que el certificado del cliente va incluido en el mensaje, y que dicho certificado incluye la clave pública del cliente. Si el servidor encripta con ella, el cliente podrá desencriptarlo con la clave privada, que nadie más conoce.

Podemos ejecutar servidor y cliente, con el TCPMonitor o sin él, y comprobar que la comunicación se establece correctamente.

Antes de incorporar la encriptación, el cuerpo (`<Body>`) del mensaje que enviaba el cliente tenía el siguiente aspecto:

```
   <soap:Body xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/
     oasis-200401-wss-wssecurity-utility-1.0.xsd"
wsu:Id="Id-18227730">
      <ns2:saluda xmlns:ns2="http://jtech.ua.es/Seguro/">
         <nombre>Boyan</nombre>
         <apellido>Bonev</apellido>
      </ns2:saluda>
   </soap:Body>
```

y el de respuesta por parte del servidor,

```
   <soap:Body xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/
     oasis-200401-wss-wssecurity-utility-1.0.xsd"
wsu:Id="Id-33008558">
      <ns2:saludaResponse xmlns:ns2="http://jtech.ua.es/Seguro/">
```

```
         <saludo>¡Hola, B. Bonev!</saludo>
      </ns2:saludaResponse>
   </soap:Body>
```

Con la encriptación los cuerpos de los dos mensajes respectivamente pasan a ser:

```
   <soap:Body xmlns:wsu="http://docs.oasis-open.org/4d5wss/2004/01/
     oasis-200401-wss-wssecurity-utility-1.0.xsd"
wsu:Id="Id-16865950">
      <xenc:EncryptedData
xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
        Id="EncDataId-1"
Type="http://www.w3.org/2001/04/xmlenc#Content">
         <xenc:EncryptionMethod
Algorithm="http://www.w3.org/2001/04/
           xmlenc#tripledes-cbc" />
         <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
            <wsse:SecurityTokenReference
xmlns:wsse="http://docs.oasis-open.org/
wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
               <wsse:Reference
xmlns:wsse="http://docs.oasis-open.org/wss/2004/
                  01/oasis-200401-wss-wssecurity-secext-1.0.xsd"
                URI="#EncKeyId-42419DE053BAC3557812741332427872"
/>
            </wsse:SecurityTokenReference>
         </ds:KeyInfo>
         <xenc:CipherData>
<xenc:CipherValue>575x5QxJXVErqWnjzpc5aL7p1UqI/zIUSbnOz4pYDMXxD37IV
Qq2OLf0YYU1xwQXYrOQcdpQ8p/nJsEbFojhktOS+nn0u/V62tg+/MCGhz7FK3tu6h
EdXYZrASs9eOHwUga8aQebnHConML+lHlwo4Bxzhs/jv+SuTkKFCOpGhCAf6oled0
zd5ayTmh25y69EW95Gh+fw6a+mQQNEUVFUXAW43Fj9qBHuVmhVEZgN4Uo4Q6aUGf7
cbcBCDkTONND/NGhHP8l1vNvLm0Bq3rL6STD0hxXLmP7ksR4RFUcyisJZK3O0zcP+
TqubqyJG7Y2/GfBqKCAd1Fah7C8NER4oLq3LiQAPaNfsns/v+NtMUVm6kDwmbQ+mr
wFPYFI1i0yhHw9yN50nQjQb36DpEDau4J9FPKRQhBhx4IB79sFco4=
            </xenc:CipherValue>
         </xenc:CipherData>
      </xenc:EncryptedData>
   </soap:Body>
```

y:

```
   <soap:Body xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/
     oasis-200401-wss-wssecurity-utility-1.0.xsd"
wsu:Id="Id-25059489">
      <xenc:EncryptedData
xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
        Id="EncDataId-1"
Type="http://www.w3.org/2001/04/xmlenc#Content">
         <xenc:EncryptionMethod
Algorithm="http://www.w3.org/2001/04/
           xmlenc#tripledes-cbc" />
         <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
            <wsse:SecurityTokenReference
```

```
xmlns:wsse="http://docs.oasis-open.org/
wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
               <wsse:Reference
xmlns:wsse="http://docs.oasis-open.org/wss/2004/
                  01/oasis-200401-wss-wssecurity-secext-1.0.xsd"
                URI="#EncKeyId-E7426DF1A3141E447812741332453672"
/>
            </wsse:SecurityTokenReference>
         </ds:KeyInfo>
         <xenc:CipherData>
<xenc:CipherValue>hpXpVo8j3iuzuzbmTR8uHGA3hgBXP0p818qr5gE5/7Lzjn/no
nWLZGqjQjx9NzcX0olWmUl/6qdVcAyOXBVCTFMNKqJi1IpZu2ZGkei/wgPmyXkPu0
6WiWanBachCmfo2GNin2o0zKw9f6DuewRdCKtEnaL+/dL7P/eEOzRxUs6wR/peQWI
xeENTQw9C08N85bNMwhRdWgOVGydAw+eolximnM3Vwdr125wGQHlx9P73uq40jVAm
duGcnB9r7Z040NiC4p4tAF9ommRVPXX5LLf3X/Gm1cyzhq7ViuOm0MJJzTgG8m+Hd
xzE+Q5FoKlchNpMT99raE0jw7/RpwZ5ApAkipc5BH1Rzu2B01HzP66j8rogXO+l93
q0fuh9Qeu5W5eIZrm0+FgvdnuHLklD61090yYzgfUAb4vIKssLn7k=
            </xenc:CipherValue>
         </xenc:CipherData>
      </xenc:EncryptedData>
   </soap:Body>
```

2. Login en el mensaje

Para enviar información de login, es decir, nombre de usuario y password, existe un token dedicado a ese fin. Para incluirlo debemos modificar el WSDL de ambos proyectos, añadiendo, después de la sección `</sp:EncryptedParts>`, el token de nombre de usuario, firmado:

```
    <sp:SignedSupportingTokens>
            <wsp:Policy>
                  <sp:UsernameToken
sp:IncludeToken="http://docs.oasis-open.org/ws-sx/
ws-securitypolicy/200702/IncludeToken/AlwaysToRecipient"/>
               </wsp:Policy>
    </sp:SignedSupportingTokens>
```

El documento `Seguro.wsdl` completo habrá quedado así:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<wsdl:definitions
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://jtech.ua.es/Seguro/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702"
xmlns:wsp="http://www.w3.org/2006/07/ws-policy"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/
  oasis-200401-wss-wssecurity-utility-1.0.xsd"
name="Servicio" targetNamespace="http://jtech.ua.es/Seguro/">
  <wsp:Policy wsu:Id="p1">
```

```
    <sp:AsymmetricBinding>
      <wsp:Policy>
        <sp:InitiatorToken>
          <wsp:Policy>
                  <sp:X509Token
sp:IncludeToken="http://docs.oasis-open.org/ws-sx/
ws-securitypolicy/200702/IncludeToken/AlwaysToRecipient">
                    <wsp:Policy>
                            <sp:WssX509V3Token10 />
                    </wsp:Policy>
                  </sp:X509Token>
          </wsp:Policy>
        </sp:InitiatorToken>
        <sp:RecipientToken>
          <wsp:Policy>
                  <sp:X509Token
sp:IncludeToken="http://docs.oasis-open.org/ws-sx/
                    ws-securitypolicy/200702/IncludeToken/Never">
                    <wsp:Policy>
                            <sp:WssX509V3Token10 />
                    </wsp:Policy>
                  </sp:X509Token>
          </wsp:Policy>
        </sp:RecipientToken>
        <sp:AlgorithmSuite>
          <wsp:Policy>
            <sp:TripleDesRsa15 />
          </wsp:Policy>
        </sp:AlgorithmSuite>
        <sp:EncryptBeforeSigning/>
      </wsp:Policy>
    </sp:AsymmetricBinding>
    <sp:Wss10>
      <wsp:Policy>
        <sp:MustSupportRefEmbeddedToken />
        <sp:MustSupportRefIssuerSerial />
      </wsp:Policy>
    </sp:Wss10>
    <sp:SignedParts>
            <sp:Body />
    </sp:SignedParts>
    <sp:EncryptedParts>
            <sp:Body />
    </sp:EncryptedParts>
    <sp:SignedSupportingTokens>
            <wsp:Policy>
                    <sp:UsernameToken
sp:IncludeToken="http://docs.oasis-open.org/ws-sx/
ws-securitypolicy/200702/IncludeToken/AlwaysToRecipient"/>
                  </wsp:Policy>
    </sp:SignedSupportingTokens>
  </wsp:Policy>
  <wsdl:types>
    <xsd:schema targetNamespace="http://jtech.ua.es/Seguro/">
      <xsd:element name="saluda">
              <xsd:complexType>
                      <xsd:sequence>
                              <xsd:element name="nombre"
type="xsd:string" />
```

```
                                <xsd:element name="apellido"
type="xsd:string"></xsd:element>
                        </xsd:sequence>
                </xsd:complexType>
      </xsd:element>
      <xsd:element name="saludaResponse">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="saludo" type="xsd:string"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
    </xsd:schema>
  </wsdl:types>
  <wsdl:message name="saludaRequest">
    <wsdl:part element="tns:saluda" name="parameters"/>
  </wsdl:message>
  <wsdl:message name="saludaResponse">
    <wsdl:part element="tns:saludaResponse" name="parameters"/>
  </wsdl:message>
  <wsdl:portType name="Seguro">
    <wsdl:operation name="saluda">
      <wsdl:input message="tns:saludaRequest"/>
      <wsdl:output message="tns:saludaResponse"/>
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="SeguroSOAP" type="tns:Seguro">
    <soap:binding style="document"
      transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="saluda">
      <wsp:PolicyReference URI="#p1" wsdl:required="true"/>
      <soap:operation
soapAction="http://jtech.ua.es/Servicio/saluda"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="Seguro">
    <wsdl:port binding="tns:SeguroSOAP" name="SeguroSOAP">
      <soap:address location="http://localhost:8080/"/>
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

Eliminamos los paquetes `es.ua.jtech.seguro` que llevan .java autogenerados, y ejecutamos las clases `GeneraCodigo` de ambos proyectos, refrescándolos después. De las clases que se han generado, eliminamos las clases innecesarias (`es.ua.jtech.seguro.Seguro_SeguroSOAP_Server` y `Client`) que ya tenemos implementadas (en `es.ua.jtech.seguro.custom.*`).

Ahora debemos incluir el nombre de usuario y la contraseña en la llamada que realiza el cliente.

```
        properties.put(SecurityConstants.USERNAME, "boyan");
        properties.put(SecurityConstants.PASSWORD, "boyan-pass");
```

Nota: estos nombre de usuario y contraseña no tienen nada que ver con los del alias de los certificados ni la contraseña del almacén o del certificado. Tampoco tienen nada que ver con el nombre y apellido pasados como parámetros a la operación "`saluda`".

Ahora en el servidor se puede comprobar esta información de login. Vamos a introducir una función que devuelva verdadero si el login es correcto, y falso en caso contrario. En aplicaciones reales se trataría de que el servidor consultara a un servidor LDAP o hiciera una consulta en una base de datos para comprobar que el usuario es válido, y para obtener su rol o sus permisos. Vamos a hacer los cambios en la clase que implementa la operación, para así comprobar el usuario antes de saludar. La clase `es.ua.jtech.seguro.custom.SeguroImpl` queda así:

```
package es.ua.jtech.seguro.custom;

import java.util.Vector;

import javax.annotation.Resource;
import javax.jws.WebService;
import javax.xml.ws.WebServiceContext;

import org.apache.ws.security.WSConstants;
import org.apache.ws.security.WSSecurityEngineResult;
import org.apache.ws.security.WSUsernameTokenPrincipal;
import org.apache.ws.security.handler.WSHandlerConstants;
import org.apache.ws.security.handler.WSHandlerResult;

import es.ua.jtech.seguro.Seguro;

@WebService(
                endpointInterface="es.ua.jtech.seguro.Seguro",
                wsdlLocation="file:src/main/resources/Seguro.wsdl",
                targetNamespace="http://jtech.ua.es/Seguro/",
                serviceName="Seguro")
public class SeguroImpl implements Seguro {

        @Resource
        WebServiceContext wsContext;

        @Override
        public String saluda(String nombre, String apellido) {
                if(loggedIn()){
                        return "¡Hola, "+nombre.substring(0,1) + ".
" + apellido + "!";
                }else{
                        return "Login incorrecto, le retiro el
saludo.";
                }
        }

        @SuppressWarnings("unchecked")
```

```
        private boolean loggedIn(){
                Vector<WSHandlerResult> handlerResults =
(Vector<WSHandlerResult>)
wsContext.getMessageContext().get(WSHandlerConstants.RECV_RESULTS);
                for(WSHandlerResult handlerResult :
handlerResults){
                        for(WSSecurityEngineResult handler :
(Vector<WSSecurityEngineResult>)handlerResult.getResults()){
                                int action =
((Integer)handler.get(WSSecurityEngineResult.
                                  TAG_ACTION)).intValue();
                                if(action == WSConstants.UT){
                                        WSUsernameTokenPrincipal
utp = (WSUsernameTokenPrincipal)
handler.get(WSSecurityEngineResult.TAG_PRINCIPAL);
                                        if(utp!=null){
if(utp.getName().equals("boyan") && utp.getPassword().
equals("boyan-pass")){
                                                        return
true;
                                                }
                                        }
                                }
                        }
                }
                return false;
        }

}
```

Ejecutamos servidor y cliente, y obtenemos la salida esperada:

```
saluda.result=¡Hola, B. Bonev!
```

Si ahora cambiamos la contraseña del login que tenemos en el cliente a otra diferente,

```
        properties.put(SecurityConstants.PASSWORD, "otra-pass");
```

y ejecutamos el cliente, obtenemos:

```
saluda.result=Login incorrecto, le retiro el saludo.
```

3. Ejercicios

3.1. Encriptar la firma

Desarróllese un servicio web seguro con CXF (y su cliente) que realice la suma de dos números enteros y devuelva el resultado en otro número entero. El servicio deberá ir firmado por el cliente y encriptado. Para evitar exponerse a que un hacker cree un diccionario y desencripte el mensaje, la firma también deberá ir firmada. En lugar de hacerlo con:

```
        <sp:EncryptBeforeSigning/>,
```

hágase con:

```
<sp:EncryptedElements>
        <sp:XPath>
                //*[local-name()='Signature']
        </sp:XPath>
</sp:EncryptedElements>
```

3.2. Saldo

En este ejercicio se pide implementar el servicio Saldo con una operación obtenerSaldo que devolverá:

- 1000 si el usuario es "pepe" (contraseña "1234")
- 20 si el usuario es "juan" (contraseña "abcd")
- -1 si el usuario no se ha logueado correctamente