Servicio Nacional de Aduanas
Subdirección de Informática
Departamento de Sistemas

# Especificación Técnica del Web Service de Autorización y Autenticación

# WSAA

**Para el desarrollo de Clientes de Entidades Externas**

# Tabla de Contenidos

# Propósito

El presente documento describe los aspectos técnicos del Web Service de Autorización y Autenticación (WSAA), adoptado por el Departamento de Sistemas de la Subdirección de Informática del Servicio Nacional de Aduanas.

# Descripción general del servicio

El propósito del WSAA es controlar el acceso y permisos asociados de los Clientes de Entidades Externas (CEE) a los distintos Web Services de Negocio (WSN) del Servicio Nacional de Aduanas que apliquen este esquema de seguridad, basado en las recomendaciones de la WebService-Interoperability Organization (WS-I) para cubrir los aspectos de seguridad e interoperabilidad.

Para dicho propósito, se adoptó la autenticación de los computadores del CEE mediante certificados digitales X.509, emitidos por entidades certificadoras reconocidos por el Servicio Nacional de Aduanas.

La utilización de especificaciones y protocolos estándares (PKI, XML, CMS, WSDL y SOAP), permite que el cliente que consuma el servicio pueda ser desarrollado con cualquier lenguaje de programación moderno.

# Descripción de entidades, componentes y estándares

## *Entidades*

### WSAA

Web Service de Autenticación y Autorización (WSAA), es el componente que se encarga de validar al usuario y asignar los permisos de acceso a los servicios de negocio por medio de la emisión de un Token de acceso.

### WSN

Representa a un Web Service de Negocio (WSN). Son los servicios que contienen la lógica de negocio, estos son consumidos por clientes externos que denominaremos CEE.

### CEE

Los Clientes de Entidades Externas (CEE), corresponden a todos los clientes que se encuentran registrados por el SNA como clientes que pertenecen a una entidad reconocida por el SNA lo que les permite realizar solicitudes del tipo TAR a un WSAA de Aduana.

## *Componentes*

### TAR

El Token Access Request (TAR) corresponde a un XML para realizar una solicitud que se envía al WSAA para obtener el permiso de acceso para consumir un WSN. Cuando se emite esta solicitud, es encapsulada en un CMS para asegurar la integridad de los datos y al mismo tiempo permitir la confirmación de la fuente que realiza la solicitud.

## TA

El Token de Acceso (TA), corresponde a un XML que permite a los CEE consumir los WSN. Corresponde a la respuesta de un TAR donde se han cumplido todas las validaciones aplicadas al CEE.

# *Estándares*

## Web Services

Los Web Services utilizados cumplen con las especificaciones WSDL 1.1, SOAP 1.1 y XML 1.0., con el fin de incorporar las recomendaciones de la WS-I (Basic Profile 1.0) para asegurar la interoperabilidad.

## XML v1.0

Los mensajes electrónicos que fluyen entre los Web Services cliente-servidor, se basan en XML v 1.0 los cuales son descritos en XML Schema, siguiendo los estándares de la W3C y la propuesta de WS-I en Basic Profile 1.0.

## CMS

CMS describe una sintaxis de encapsulación para protección de datos. Este deriva del PKCS #7 v 1.5.

En la solución se utiliza un CMS del tipo “signed data”, el cual es codificado en Base64.

## X.509 v3.0

Los certificados digitales que se utilizan corresponden al estándar X509 v3.0. Estos son utilizados para comprobación de dominios y de Web Services. Estos certificados son por Entidades Certificadoras que deben estar reconocidas en Chile para realizar trámites electrónicos y firma electrónica avanzada.

## Codificación UTF8

La codificación de caracteres que utilizan los mensajes XML, utiliza codificación UTF8.

## Contenedor PKCS #12

Los certificados y claves privadas actualmente se guardan en formato PKCS #12. Este estándar permite guardar tanto la clave privada como pública con una clave de acceso que es de tipo simétrico.

# Descripción de Funcionamiento

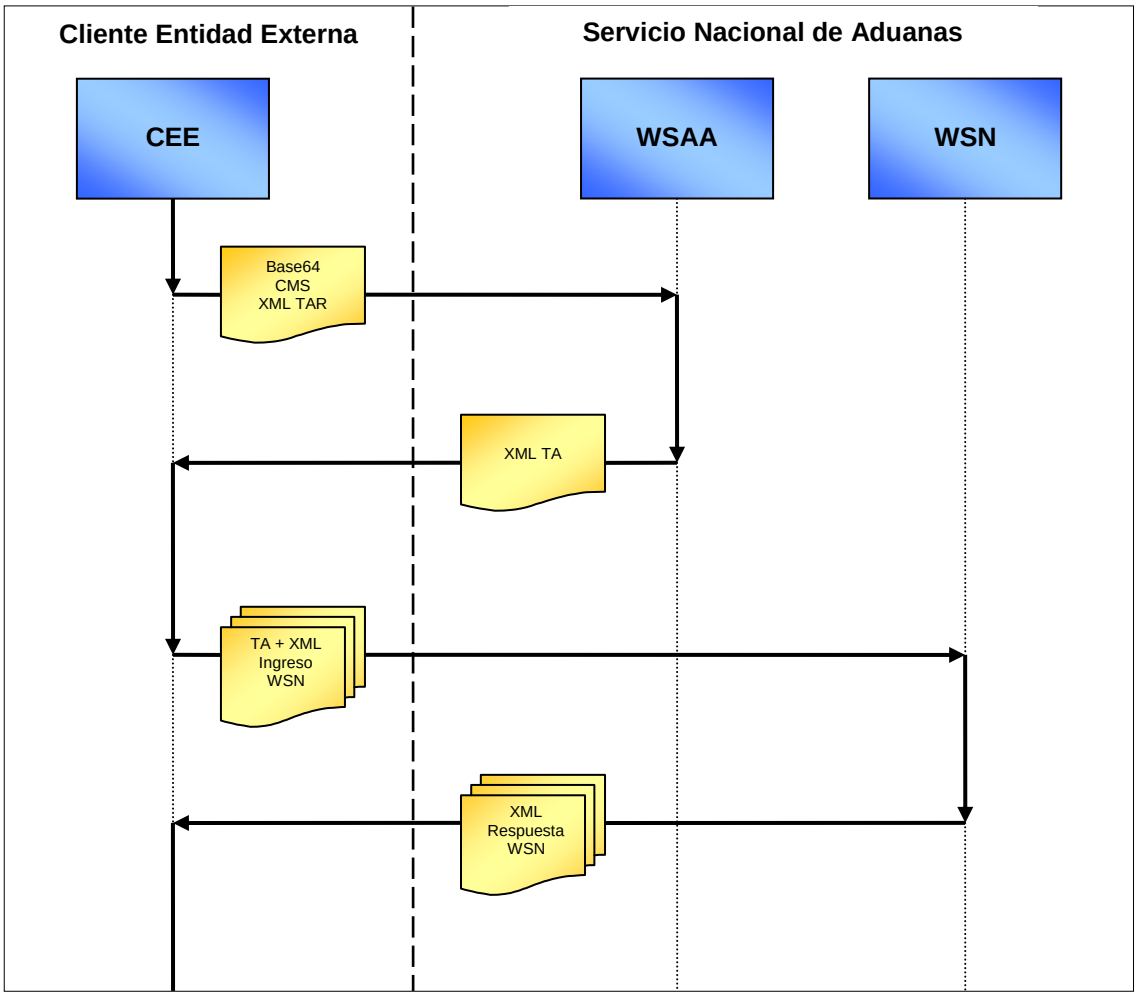
Para que un CEE esté autorizado a usar un WSN del Servicio Nacional de Aduanas que necesita autenticación mediante el esquema de seguridad WSAA, deberá realizar un trámite administrativo previo, el cual consiste en enviar la Huella digital SHA-1 de un certificado X.509 válido y vigente para firma electrónica avanzada que haya sido por una de las Autoridades Certificadoras (CA) reconocidas en Chile, con la cual Aduanas lo registrará como un CEE autorizado para consumir los WSN, otorgándole los perfiles correspondientes al WSN que desee consumir. Luego, el CEE deberá firmar las solicitudes TA con el mismo certificado con el cual fue registrado en Aduanas.

Los TA son solicitados a través del WSAA, el cual se realiza enviando un XML TAR correspondiente al CEE, mediante mensajes SOAP a través de canal SSL.

El WSAA al recibir el XML del TAR, lo verifica y valida, y si el requerimiento es correcto, devuelve un mensaje XML que contiene el TA que habilita al CEE a utilizar los WSN correspondientes. El TA deberá ser utilizado por el CEE para acceder al WSN.

## *Diagrama de interacción*

A continuación se presenta el diagrama de interacción general que representa la comunicación entre los distintos componentes del esquema de seguridad. La comunicación termina cuando el Web Services de Negocio puede ser invocado correctamente, entregando una respuesta a la llamada.



## *Invocación del WSAA*

Para que un CEE realice la invocación del Web Service de Autorización y Autenticación, debe seguir los siguientes pasos:

1. Generar el mensaje XML del TAR (LoginTicketRequest.xml).
2. Generar el CMS que contenga el TAR, firma electrónica y el certificado X.509 del cliente (LoginTicketRequest.xml.cms).
3. Codificar en Base64 el CMS (LoginTicketRequest.xml.cms.base64).
4. Invocar el WSAA con el parámetro indicado en el punto anterior, donde recibirá como respuesta el XML con el TA (LoginTicketResponse.xml).
5. Extraer y validar la información del TA.

La definición del WSAA (WSDL), y los XSD del TAR y del TA se describen en el anexo 1 del presente documento.

## Generar XML TAR

El TAR es una estructura XML definida en el XSD del anexo 2.
La descripción de los tags contenidos son los siguientes:

- <source>: Indica el DN del certificado que será utilizado por WSAA para verificar la firma electrónica del TAR generado por el computador (CEE) que realiza el requerimiento. Deberá corresponder al del certificado de firma incluido en el CMS.
- <destination>: Indica el DN del WSAA. Corresponde al valor del DN del certificado de servidor generado por el CA del WSAA, el cual será uno para cada ambiente (desarrollo/testing/producción).
- <uniqueId>: Entero de 32 bits sin signo que junto con <generationTime> identifica el requerimiento.
- <generationTime>: Momento en que fue generado el requerimiento.
- <expirationTime>: Momento en el que expira la solicitud.
- <service>: Identificación del WSN para el cual se solicita el TA.

Un ejemplo del XML del TAR es el siguiente:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<loginTicketRequest version="1.0">
    <header>
        <source>SERIALNUMBER=CL123456789,  EMAILADDRESS=prueba@prueba.cl,  CN=Prueba,
        OU=Departamento de Prueba, O=Empresa de Prueba, L=Santiago, ST=Santiago,
        C=CL</source>
        <destination>C=CL,  O=Servicio  Nacional  de  Aduanas,  CN=wsaadesarrollo,
        OU=Departamento de Sistemas, DC=wldesarrollo</destination>
        <uniqueId>1280929280</uniqueId>
        <generationTime>2010-08-04T09:41:20-04:00</generationTime>
        <expirationTime>2010-08-04T10:41:20-04:00</expirationTime>
    </header>
    <service>swprueba</service>
</loginTicketRequest>
```

## Generar CMS conteniendo el XML TAR

Se deberá generar un mensaje CMS del tipo “SignedData” que contenga el mensaje anteriormente generado (LoginTicketRequest.xml) y su firma electrónica utilizando SHA1+RSA.

De esta forma, se obtiene el TAR (LoginTicketRequest.xml.cms).

## Codificar en Base64

Para poder enviar el TAR al WSAA, el mismo deberá ser codificado en Base64 (LoginTicketRequest.xml.cms.base64).

## Invocar el WSAA

Se debe invocar el método “loginCms” del WSAA. Dicho método recibe como parámetro una cadena correspondiente a la codificación en Base64 del TAR (LoginTicketRequest.xml.cms.base64) y devuelve una cadena denominada LoginTicketResponse.xml. De esta última se deberá extraer el Ticket de Acceso (TA).

## Extraer y validar la información del TA

El TA corresponde a un XML definida en el XSD del anexo 3.
La descripción de los tags contenidos son los siguientes:

- <source>: DN del WSAA. Corresponde al valor del DN del certificado de servidor generado por el CA del WSAA, el cual será uno para cada ambiente (desarrollo/testing/producción).
- <destination>: Corresponde al DN del CEE que realizó la solicitud a través del TAR.
- <uniqueId>: Entero de 32 bits sin signo que junto con <generationTime> identifica la generación del TA.
- <generationTime>: Momento en que fue generado el TA.
- <expirationTime>: Momento en el que expira el TA.
- <token> y <sign>: cadenas de caracteres que deben ser informadas al WSN. Las mismas componen el TA. El formato interno depende de la definición interna del Servicio Nacional de Aduanas, la cual será interpretada y validada por el WSN.

Un ejemplo de un TA generado conforme, es el siguiente:

```
<loginTicketResponse>
    <header>
        <source>C=CL,    O=Servicio    Nacional    de    Aduanas,    CN=wsaadesarrollo,
        OU=Departamento de Sistemas, DC=wldesarrollo</source>
        <destination>SERIALNUMBER=CL123456789,          EMAILADDRESS=prueba@prueba.cl,
        CN=Prueba,  OU=Departamento  de  Prueba,  O=Empresa  de  Prueba,  L=Santiago,
        ST=Santiago, C=CL</destination>
        <uniqueId>1280929383</uniqueId>
        <generationTime>2010-08-04T09:43:03.000-04:00</generationTime>
        <expirationTime>2010-08-05T09:43:03.000-04:00</expirationTime>
    </header>
    <credentials>
        <token>PFRva2VuPgogIDxjb2RpZ29DRUU+Q0wxMzc2MTg4MzQ8L2NvZGlnb0NFRT4KICA8c2VyaWF
        sPjM4ODc0NTE2ODg8L3NlcmlhbD4KPC9Ub2tlbj4K</token>
        <sign>zf3sGt0NcrfU7NsgkrlJNdZoCa7lcGITTjw4JrwEmk2DUdh0Ed9zARgqLhRk02UDCaTBXTTJ
        WWuZ1VqGcg4cBBgIPFmgjzzOWPP9lrP6P6iorF2BDq7sLUZHLtYpVE30TUMg/XgAzWbk29oJjxePEI
        8WTMTnxXpSon5k3SEQIY01Y60cg+NdIdhCp8Nbt+ozHISAIcW3LEH0lfjql79RMZsT2jsFTlA6287e
        gLc2TVJcliHBVzXCaLoXBjkUkNtEak3VKEmea/7qPKDKzorPHBfu9I1ErnQ1AKnaPeH20VE59xa+Va
        Z2Gq+68kacqizJeGIBpvStWZetWzxrFjIj/Q==</sign>
    </credentials>
</loginTicketResponse>
```

## *Errores lanzados por el WSAA al procesar un TAR*

En caso de encontrarse o producirse algún error, el WS responderá con un SoapFault conteniendo el código y descripción de los errores especificados en la siguiente tabla, la cual puede ir actualizándose según las necesidades del servicio.

| Errores | |
|---|---|
| **Código** | **Descripción** |
| 1.1 | No se puede decodificar parametro de entrada en Base64 |
| 1.2 | El CMS no es valido |
| 1.3 | Algoritmo de firma del CMS no soportado |
| 1.4 | Certificado del CMS expirado |
| 1.5 | Certificado del CMS no es valido |
| 1.6 | Error al rescatar Certificado del CMS |
| 1.7 | Error al validar Certificado del CMS con CA del WSAA de Aduanas |
| 1.8 | Error al validar Ruta del CA de Confianza del Certificado del CMS |
| 1.9 | Algoritmo invalido para validar Ruta del CA de Confianza del Certificado del CMS |
| 1.10 | Error al validar que el Certificado del CMS no este revocado |
| 1.11 | Certificado del CEE revocado por Aduanas |
| 2.1 | XML del TAR (LoginTicketRequest) mal formado |
| 2.2 | XML del TAR (LoginTicketRequest) invalido |
| 2.3 | Error al validar datos del TAR |
| 2.4 | Tag <source> del TAR no corresponde con DN del Certificado del CEE |
| 2.5 | Tag <destination> del TAR no corresponde con DN del WSAA de Aduanas |
| 2.6 | Fecha de generacion de TAR invalida |
| 2.7 | TAR expirado |
| 2.8 | CEE no registrado en Aduanas |
| 2.9 | WSN no registrado en Aduanas |
| 3.1 | Servicio WSAA no puede procesar el requerimiento |

# Referencias


- RFC-3852, Cryptographic Message Syntax (CMS)
- RFC-3548, The Base16, Base32 and Base64 Data Encoding
- RFC-2253, Lightweight Directory Access Protocol (v3): UTF-8 String Representation of Distinguished Names
- Web Service Security, Oasis 2003
- Basic Profile 1.0, WS-I

# Anexos

## *1. WSDL del WSAA*

A continuación, se muestra el XML del WSDL del WSAA en ambiente de desarrollo:

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:apachesoap="http://xml.apache.org/xml-soap"
xmlns:impl="http://www.aduana.cl/wsaa/servicio/WSAA"
xmlns:intf="http://www.aduana.cl/wsaa/servicio/WSAA"
xmlns:tns1="http://www.aduana.cl"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.aduana.cl/wsaa/servicio/WSAA">
 <wsdl:types>
  <schema xmlns="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" targetNamespace="http://www.aduana.cl">
   <import namespace="http://www.aduana.cl/wsaa/servicio/WSAA"/>
   <element name="loginCms">
    <complexType>
     <sequence>
      <element name="in0" type="xsd:string"/>
     </sequence>
    </complexType>
   </element>
   <element name="loginCmsResponse">
    <complexType>
     <sequence>
      <element name="loginCmsReturn" type="xsd:string"/>
     </sequence>
    </complexType>
   </element>
  </schema>
  <schema xmlns="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified"
targetNamespace="http://www.aduana.cl/wsaa/servicio/WSAA">
   <complexType name="LoginFault">
    <sequence/>
   </complexType>
   <element name="fault" type="impl:LoginFault"/>
  </schema>
 </wsdl:types>
   <wsdl:message name="loginCmsResponse">
      <wsdl:part element="tns1:loginCmsResponse" name="parameters"/>
   </wsdl:message>
   <wsdl:message name="LoginFault">
      <wsdl:part element="impl:fault" name="fault"/>
   </wsdl:message>
   <wsdl:message name="loginCmsRequest">
      <wsdl:part element="tns1:loginCms" name="parameters"/>
   </wsdl:message>
   <wsdl:portType name="LoginCMS">
      <wsdl:operation name="loginCms">
         <wsdl:input message="impl:loginCmsRequest"
name="loginCmsRequest"/>
         <wsdl:output message="impl:loginCmsResponse"
name="loginCmsResponse"/>
         <wsdl:fault message="impl:LoginFault" name="LoginFault"/>
```

```
      </wsdl:operation>
   </wsdl:portType>
   <wsdl:binding name="LoginCmsSoapBinding" type="impl:LoginCMS">
      <wsdlsoap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
      <wsdl:operation name="loginCms">
         <wsdlsoap:operation soapAction=""/>
         <wsdl:input name="loginCmsRequest">
            <wsdlsoap:body use="literal"/>
         </wsdl:input>
         <wsdl:output name="loginCmsResponse">
            <wsdlsoap:body use="literal"/>
         </wsdl:output>
         <wsdl:fault name="LoginFault">
            <wsdlsoap:fault name="LoginFault" use="literal"/>
         </wsdl:fault>
      </wsdl:operation>
   </wsdl:binding>
   <wsdl:service name="LoginCMSService">
      <wsdl:port binding="impl:LoginCmsSoapBinding" name="LoginCms">
         <wsdlsoap:address
location="http://200.72.133.28:7001/wsaa/servicio/WSAA.jws"/>
      </wsdl:port>
   </wsdl:service>
</wsdl:definitions>
```

## *2. XSD TAR*

A continuación se presenta el XSD que describe el XML del TAR:

```
<?xml version="1.0" encoding="UTF8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:annotation>
    <xsd:documentation xml:lang="es">Esquema de Ticket de pedido de
acceso a un WSN por parte de un CEE.</xsd:documentation>
  </xsd:annotation>
  <xsd:element name="loginTicketRequest" type="loginTicketRequest"/>
  <xsd:complexType name="loginTicketRequest">
    <xsd:sequence>
      <xsd:element name="header" type="headerType" minOccurs="1"
maxOccurs="1"/>
      <xsd:element name="service" type="serviceType" minOccurs="1"
maxOccurs="1"/>
    </xsd:sequence>
    <xsd:attribute name="version" type="xsd:decimal" use="optional"
default="1.0"/>
  </xsd:complexType>
  <xsd:simpleType name="serviceType">
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="[a-z][a-z,\-,_,0-9]*"/>
      <xsd:minLength value='3'/>
      <xsd:maxLength value='32'/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:complexType name="headerType">
    <xsd:sequence>
      <xsd:element name="source" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
      <xsd:element name="destination" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
      <xsd:element name="uniqueId" type="xsd:unsignedInt"
minOccurs="1" maxOccurs="1"/>
      <xsd:element name="generationTime" type="xsd:dateTime"
minOccurs="1" maxOccurs="1"/>
      <xsd:element name="expirationTime" type="xsd:dateTime"
minOccurs="1" maxOccurs="1"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

## *3. XSD TA*

A continuación se presenta el XSD que describe el XML del TA:

```
<?xml version="1.0" encoding="UTF8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:annotation>
    <xsd:documentation xml:lang="es">Esquema de Ticket de respuesta al
pedido de acceso a un WSN por parte de un CEE</xsd:documentation>
  </xsd:annotation>
  <xsd:element name="loginTicketResponse" type="loginTicketResponse"/>
  <xsd:complexType name="loginTicketResponse">
    <xsd:sequence>
      <xsd:element name="header" type="headerType" minOccurs="1"
maxOccurs="1"/>
      <xsd:element name="credentials" type="credentialsType"
minOccurs="1" maxOccurs="1"/>
    </xsd:sequence>
    <xsd:attribute name="version" type="xsd:decimal" use="optional"
default="1.0"/>
  </xsd:complexType>
  <xsd:complexType name="credentialsType">
    <xsd:sequence>
      <xsd:element name="token" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
      <xsd:element name="sign" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="headerType">
    <xsd:sequence>
      <xsd:element name="source" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
      <xsd:element name="destination" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
      <xsd:element name="uniqueId" type="xsd:unsignedInt"
minOccurs="1" maxOccurs="1"/>
      <xsd:element name="generationTime" type="xsd:dateTime"
minOccurs="1" maxOccurs="1"/>
      <xsd:element name="expirationTime" type="xsd:dateTime"
minOccurs="1" maxOccurs="1"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```