Manual de Desarrollador
Autenticación Automática

OI2007_AUTAUTOM_MDE_1.9

Subdirección Informática
Servicio Impuestos Internos

Fecha:18/11/2007

CONTROL DE VERSIONES

Versión	Fecha
1.0	21/01/2003
1.2	17/02/2003
✓ Se Modifico Introducción ✓ Se Agrego: Capitulo 3, Capitulo 4, Capitulo 5	
1.3	08/03/2004
Se modifico Url en Capitulo 7 (Guía para Pruebas, le faltaba la “s” al http). Donde decía: http://palena.sii.cl/DTEWS/CrSeed.jws?WSDL http://palena.sii.cl/DTEWS/GetTokenFromSeed.jws?WSDL Se cambio por: https://palena.sii.cl/DTEWS/CrSeed.jws?WSDL https://palena.sii.cl/DTEWS/GetTokenFromSeed.jws?WSDL	
1.4	08/04/2004
Se modifico texto de introducción (como acceder a los WS del SII)	
1.5	07/05/2004
Se modifico texto de los mensajes de salida	
1.6	31/05/2004
Se agrego Capitulo 8: Como Firmar una semilla.	
1.7	18/11/2005
Se agrego Error 21, en punto 5.2.2	
1.8	03/07/2006 en punto.
Se agrego error 12, en punto 5.2.2	
1.9	18/11/2007
Se modifico detalle del errores : 11, 12 y –3 (Capítulo 6).	

INTRODUCCIÓN

El método de autenticación automática (AUTAUTOM), es un chequeo del uso de la llave privada del certificado del cliente, mediante el uso de Web Services (WS).

Para cumplir su objetivo AUTAUTOM, entrega a las empresas dos Web services (WS) "*CrSeed y GetTokenFromSeed*", mediante los cuales se podrá obtener un Texto aleatorio o Semilla y un Token (requisitos de la autenticación), los que serán detallados más adelante.

Este documento está dirigido a quienes tengan la misión de utilizar y probar los WS mencionados anteriormente (CrSeed y GetTokenFromSeed).

Para acceder a los servicios que ofrece el SII, se debe utilizar WSDL(Web Services Definition Language).

WSDL es un lenguaje descriptor, basado en XML, que permite conocer en forma abstracta, la gramática de los componentes de un Web Service (ubicación, formato, tipos de datos, servicios, funciones, parámetros de entrada, salida, etc).

Para poder acceder a un WSDL, se debe conocer su ubicación, por ejemplo el WSDL de los WS entregados son:

https://palena.sii.cl/DTEWS/CrSeed.jws?WSDL.

https://palena.sii.cl/DTEWS/GetTokenFromSeed.jws?WSDL.

Cuando el cliente conoce el WSDL del servicio, puede construir un Request en formato SOAP (Simple Object Access Protocolo), para luego enviarlo hacia el proveedor de servicio.

Requisitos de uso.

Para poder utilizar este manual, es necesario tener previo conocimiento de XML, Web Services y Certificado Digital.

Recomendaciones: Se recomienda el uso de la herramienta *XMLSPY5* de la Altova GmbH http://www.altova.com

CAPÍTULO 1

ANÁLISIS DEL SISTEMA

Este sistema permite la implementación de la Autenticación Automática, mediante el uso de WS y Certificado Digital.

AUTAUTOM es un sistema implementado bajo la tecnología B2B, que permite que las aplicaciones se comuniquen entre sí con llamadas de programa a programa.

A grandes rasgos la utilización de esta aplicación, requiere que un cliente remoto se pueda autenticar en el SII mediante Certificado Digital. Para esto es necesario que dicho cliente solicite a la aplicación del SII un texto aleatorio llamado Semilla.

Una vez entregada la semilla al cliente, éste deberá firmarla y enviarla nuevamente hacia el sitio del SII, quien se encargará de validar la firma y la vigencia de dicho texto. Si la validación es OK, la aplicación le entrega al cliente un identificador de autenticación llamado Token. Dicho identificador, le permitirá al cliente navegar por las otras aplicaciones del SII, sin tener que autenticarse nuevamente.

1.1 Quienes pueden utilizar esta aplicación.

Esta aplicación puede ser utilizada por todas aquellas Personas o Empresas, que tengan registrada una clave secreta en las BD del SII.

Actualmente la aplicación solo permite autenticarse con Certificado Digital Válido para el SII.

1.2 Objetivos de la aplicación.

El objetivo de la aplicación es dar solución a la Autenticación Automática del SII.

1. 3 Característica de la aplicación

- ✓ Autenticación programa a programa
- ✓ Autenticación sin intervención de humanos por parte de servidor
- ✓ Desarrollo en base WS
- ✓ Actualmente sólo permite Autenticarse con Certificado Digital
- ✓ Cliente necesita estar registrado en las bases de datos del SII como un contribuyente habilitado para ingresar a las aplicaciones de Internet que requieren autenticación.

CAPÍTULO 2

VISIÓN GENERAL DEL SISTEMA

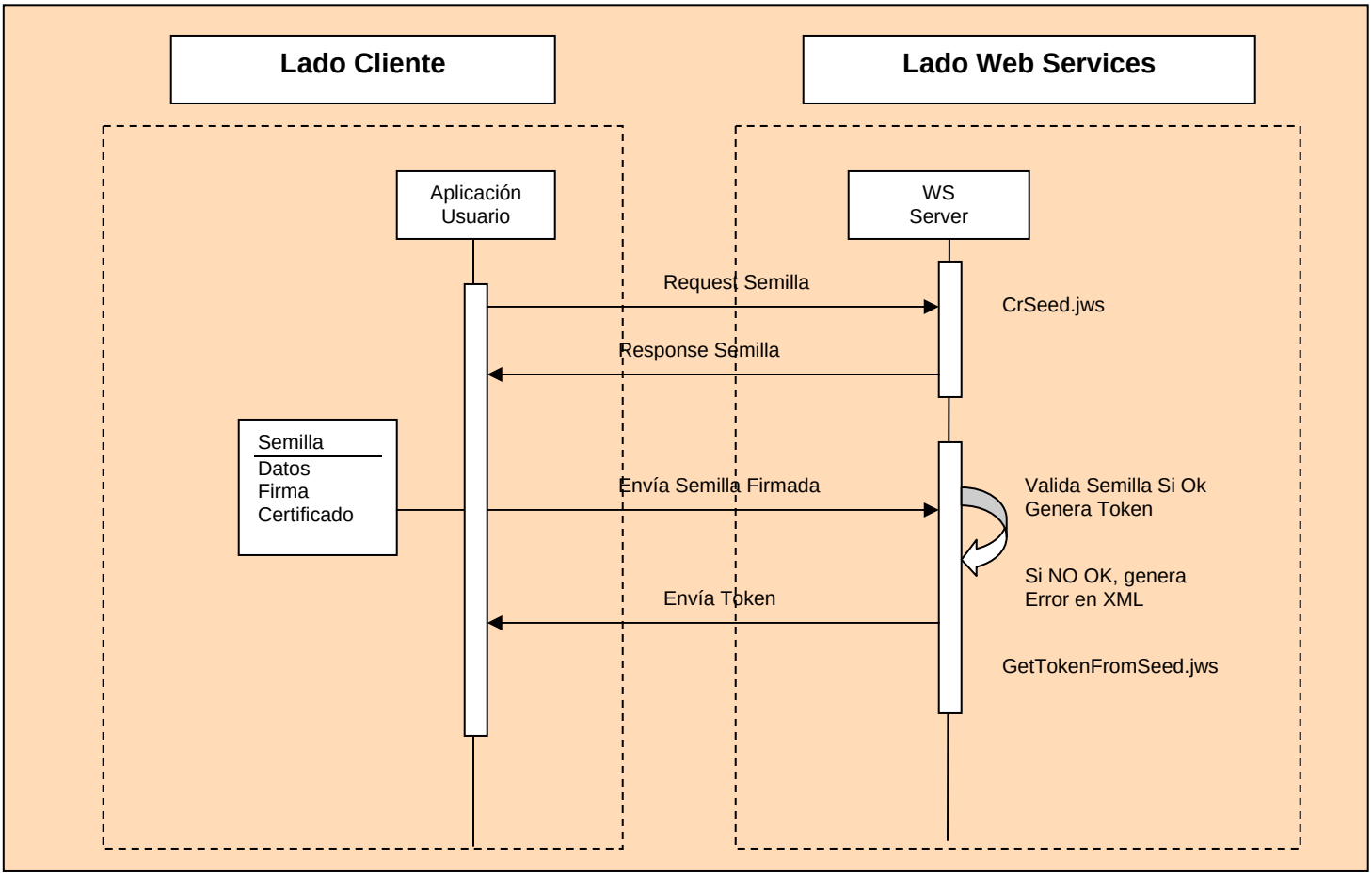

Figura 1.0

De acuerdo al diagrama superior (figura 2.0), para que un cliente se pueda autenticar, lo primero que debe hacer es solicitar una Semilla mediante un Request, hacia el WS CrSeed.jws.

Cuando el WS recibe el requerimiento, genera automáticamente una Semilla en formato XML.

Una vez que se ha generado una semilla, es almacenada en una base de datos y luego es enviada al cliente en el "Header" del "Response".

Una Semilla es un número único y aleatorio que sirve como identificador para la sesión de un cliente y que tiene un time out de 2 (dos) minutos.

Cuando el cliente recibe la Semilla, debe firmarla, para luego enviarla en formato estándar XML(definido por el SII), hacia nuestro sitio.

Una vez recibida la semilla firmada, se validará su firma y su vigencia.

Si la validación de la Semilla es OK, se genera automáticamente un Token, el cual es almacenado en una Base de Datos y luego es enviado hacia el cliente.

Un Token es un identificador único el cual es almacenado y enviado al cliente en el Body (Cuerpo) del **Response**.

La generación del Token la realiza el WS GetTokenFromSeed.jws

Cuando el cliente recibe el Token, ya está Autenticado y puede ingresar a cualquier aplicación del SII.

Nota : Ver Ejemplo de Token, ANEXO 1

Si la validación falló, el Web Services envía un mensaje de error en formato XML.

Ver ejemplo Mensaje Error en Punto 4.2

La información que debe contener el XML que Firma la Semilla (XML Entrada) es:

- ✓ Semilla
- ✓ Firma
- ✓ Módulo
- ✓ Certificado Digital

Ver Ejemplo Archivo XML Entrada (Semilla Firmada) en: Punto 3.2

La validación del XML, consiste en:

- ✓ Validar que su formato XML este OK (que cumpla formato solicitado por el SII).
- ✓ Validar que la Semilla este vigente (ya que la semilla tiene una duración de 2 min.).
- ✓ Validar su certificado Digital.
- ✓ Validar su firma.

CAPITULO 3

WSDL DE AUTENTICION AUTOMATICA

Tal como se menciono anteriormente la AUTAUTOM, entrega dos WS:

- ✓ CrSeed
- ✓ GetTokenFromSeed.jws

3.1.1 WSDL de CrSeed.jws

`CrSeed` , entrega un solo método "`getSeed`", el cual permite Obtener una Semilla.

La ubicación del WSDL, para CrSeed.jws es:

https://palena.sii.cl/DTEWS/CrSeed.jws?WSDL

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions targetNamespace="https://palena.sii.cl/DTEWS/CrSeed.jws"
xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:apachesoap="http://xml.apache.org/xml-soap"
xmlns:impl="https://palena.sii.cl/DTEWS/CrSeed.jws"
xmlns:intf="https://palena.sii.cl/DTEWS/CrSeed.jws"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <wsdl:types/>
   <wsdl:message name="getSeedRequest">
  </wsdl:message>
   <wsdl:message name="getSeedResponse">
      <wsdl:part name="getSeedReturn" type="xsd:string"/>
   </wsdl:message>
   <wsdl:portType name="CrSeed">
      <wsdl:operation name="getSeed">
         <wsdl:input message="impl:getSeedRequest" name="getSeedRequest"/>
         <wsdl:output message="impl:getSeedResponse" name="getSeedResponse"/>
      </wsdl:operation>
   </wsdl:portType>
   <wsdl:binding name="CrSeedSoapBinding" type="impl:CrSeed">
      <wsdlsoap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
      <wsdl:operation name="getSeed">
         <wsdlsoap:operation soapAction=""/>
         <wsdl:input name="getSeedRequest">
            <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="https://palena.sii.cl/DTEWS/CrSeed.jws" use="encoded"/>
         </wsdl:input>
         <wsdl:output name="getSeedResponse">
            <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="https://palena.sii.cl/DTEWS/CrSeed.jws" use="encoded"/>
         </wsdl:output>
      </wsdl:operation>
   </wsdl:binding>
   <wsdl:service name="CrSeedService">
      <wsdl:port binding="impl:CrSeedSoapBinding" name="CrSeed">
         <wsdlsoap:address location="https://palena.sii.cl/DTEWS/CrSeed.jws"/>
      </wsdl:port>
   </wsdl:service>
</wsdl:definitions>
```

Diagrama 1-1 WSDL CrSeed.jws

3.1.2 WSDL de GetTokenFromSeed.jws

GetTokenFromSeed, entrega un solo servicio llamado "getToken", el cual permite Obtener un Token.

La ubicación del WSDL, para GetTokenFromSeed.jws:

https://palena.sii.cl/DTEWS/GetTokenFromSeed.jws?WSDL

```
<?xml version="1.0" encoding="UTF-8" ?>
- <wsdl:definitions targetNamespace="https://palena.sii.cl/DTEWS/GetTokenFromSeed.jws"
     xmlns="http://schemas.xmlsoap.org/wsdl/"
     xmlns:apachesoap="http://xml.apache.org/xml-soap"
     xmlns:impl="https://palena.sii.cl/DTEWS/GetTokenFromSeed.jws"
     xmlns:intf="https://palena.sii.cl/DTEWS/GetTokenFromSeed.jws"
     xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
     xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
     xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <wsdl:types />
  - <wsdl:message name="getTokenResponse">
      <wsdl:part name="getTokenReturn" type="xsd:string" />
    </wsdl:message>
  - <wsdl:message name="getVersionResponse">
      <wsdl:part name="getVersionReturn" type="xsd:string" />
    </wsdl:message>
    <wsdl:message name="getVersionRequest" />
  - <wsdl:message name="getTokenRequest">
      <wsdl:part name="pszXml" type="xsd:string" />
    </wsdl:message>
  - <wsdl:portType name="GetTokenFromSeed">
    - <wsdl:operation name="getVersion">
        <wsdl:input message="impl:getVersionRequest" name="getVersionRequest"
          />
        <wsdl:output message="impl:getVersionResponse"
          name="getVersionResponse" />
      </wsdl:operation>
    - <wsdl:operation name="getToken" parameterOrder="pszXml">
        <wsdl:input message="impl:getTokenRequest" name="getTokenRequest" />
        <wsdl:output message="impl:getTokenResponse" name="getTokenResponse" />
      </wsdl:operation>
    </wsdl:portType>
  - <wsdl:binding name="GetTokenFromSeedSoapBinding" type="impl:GetTokenFromSeed">
      <wsdlsoap:binding style="rpc"
        transport="http://schemas.xmlsoap.org/soap/http" />
    - <wsdl:operation name="getVersion">
        <wsdlsoap:operation soapAction="" />
      - <wsdl:input name="getVersionRequest">
          <wsdlsoap:body
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="https://palena.sii.cl/DTEWS/GetTokenFromSeed.jws"
            use="encoded" />
        </wsdl:input>
      - <wsdl:output name="getVersionResponse">
          <wsdlsoap:body
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="https://palena.sii.cl/DTEWS/GetTokenFromSeed.jws"
            use="encoded" />
        </wsdl:output>
      </wsdl:operation>
    - <wsdl:operation name="getToken">
        <wsdlsoap:operation soapAction="" />
      - <wsdl:input name="getTokenRequest">
          <wsdlsoap:body
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
```

```
            namespace="https://palena.sii.cl/DTEWS/GetTokenFromSeed.jws"
            use="encoded" />
        </wsdl:input>
      - <wsdl:output name="getTokenResponse">
          <wsdlsoap:body
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="https://palena.sii.cl/DTEWS/GetTokenFromSeed.jws"
            use="encoded" />
        </wsdl:output>
      </wsdl:operation>
    </wsdl:binding>
  - <wsdl:service name="GetTokenFromSeedService">
    - <wsdl:port binding="impl:GetTokenFromSeedSoapBinding"
        name="GetTokenFromSeed">
        <wsdlsoap:address
          location="https://palena.sii.cl/DTEWS/GetTokenFromSeed.jws" />
      </wsdl:port>
    </wsdl:service>
</wsdl:definitions>
```

Diagrama 1-2 WSDL GetTokenFromSeed.jws

CAPÍTULO 4

PARÁMETROS DE ENTRADA

4.1.1 Parámetros de Entrada para CrSeed.jws

CrSeed.jws no tiene parámetros de entrada, tal como se detalla en el ejemplo 3.1.2

4.1.2 Ejemplo Real Parámetros de Entrada Formato WSDL

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" SOAP-
ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
   <SOAP-ENV:Body>
      <m:getSeed xmlns:m="https://palena.sii.cl/DTEWS/CrSeed.jws"/>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

4.1.3 Parámetros de Entrada para GetTokenFromSeed.jws

Los parámetros de Entrada para GetTokenFromSeed.jws, corresponden a un String formado por los **campos del XML**, que enviara la Semilla Firmada.

4.1.4 Ejemplo Real Parámetros de Entrada Formato WSDL

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" SOAP-
ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
   <SOAP-ENV:Body>
      <m:getToken xmlns:m="https://palena.sii.cl/DTEWS/GetTokenFromSeed.jws">
         <pszXml xsi:type="xsd:string">String</pszXml>
      </m:getToken>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Los parámetros de entrada que deben formar el XML para el envío de la Semilla Firmada son:

Parámetros de Entrada XML:

✓ Etiqueta de inicio XML	<getToken>
✓ Semilla, es número único, generado por nuestro Web Server.	<Semilla>**1234567890123**</Semilla>
✓ Url's Signature	<SignedInfo></SignedInfo>
✓ Corresponde a la Firma.	<SignatureValue>gfnpbQ8vKMQzAJF/nuQxC/Gg=</SignatureValue>
✓ Módulo de la llave Pública.	<Modulus>**pvzPlABsnc9V4M2Wc+QcI8=**</Modulus>
✓ Exponente de la llave Pública.	<Exponent>**AQAB**</Exponent>
✓ Certificado.X509	<X509Certificate>MIBR08xDjAMBgNVBAcTBUNIRUxFMQww CgYDVQQtvAfDCCQxMeLAtNJKWJDCN199bO5CUiA3iTlr5BEtu DjmnF5dg6L0z03pXOfoaF9bD3zsgPjMRxYAZP33uj/prVHUv0E9g U8d/xvdWE21d6AGKGtklmQGSuW8wKogWokKkP UfKDlmcWkaSAv056hkvzPlABsnc9V4M2Wc+QcI8CAwEAATANB gkqhkiG9w0BAakRk8i3bCCAakRk8i==</X509Certificate>

Figura 1.3

4.1.5 Ejemplo Formato XML de Entrada

Este es un ejemplo del formato XML para el envió de Semilla firmada de acuerdo al Estándar XML Digital Signature. **Los demás nombres y etiquetas son Obligatorios.**

```
<?xml version="1.0"?>
<getToken>
   <item>
      <Semilla>10</Semilla>
   </item>
   <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
      <SignedInfo>
         <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
         <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
         <Reference URI="">
            <Transforms>
               <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
            </Transforms>
            <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
            <DigestValue>8slcL05kmrM8NGw4I9NSfRqYA9E=</DigestValue>
         </Reference>
      </SignedInfo>
   <SignatureValue>jlbzatIIBLW8AjH++5uVTTrGIMVwGButuoAR88y/hvSc1+6/eW1K864fK3cKi76oArqk7lAM4pP
okoXme0JT/hRXXGo6ecuKzO18z2WfPWwgnN0f3ac03TDu7PwfqiDG9mhQpYfIkNp6GNJIiqlg9PG2w1fOJ1QoypsrQmKq6
YU=</SignatureValue>
   <KeyInfo>
         <KeyValue>
            <RSAKeyValue>
               <Modulus>2Pb4kEB19m7NmOUYew9f36325yrTLTPMU7qzYG2A0/BsubxDdgQw2Op0x6zXvOVX
sYI9KkPXtD5orKJMjwxYRv9wUWdyiE776Rv4ljfJO7EQhIK1fDQDnPt0HefBS06Xzg2QLBvLR+pe1vc6C02Dr99v+lnLA8
mnZiJlRHndhNU=
</Modulus>
      <Exponent>AQAB</Exponent>
      </RSAKeyValue>
         </KeyValue>
         <X509Data>
<X509Certificate>MIIF1DCCBLygAwIBAgIDAQNtMA0GCSqGSIb3DQEBBQUAMIHGMQswCQYDVQQG
EwJDTDEYMBYGA1UEChMPQWNlcHRhLmNvbSBTLkEuMTgwNgYDVQQLEy9BdXRv
cmlkYWQgY2VydGlmaWNhZG9yYSBDbGFzZSAzIHBlcnNvbmEgbmF0dXJhbDFD
MEEGA1UEAxM6QWNlcHRhLmNvbSBBdXRvcmlkYWQgY2VydGlmaWNhZG9yYSBD
bGFzZSAzIHBlcnNvbmEgbmF0dXJhbDEeMBwGCSqGSIb3DQEJARYPaW5mb0Bh
Y2VwdGEuY29tMB4XDTAxMDkyNTIxMDgxMloXDTAyMDkyNTIxMDgxMlowgZ8x
CzAJBgNVBAYTAkNMMRgwFgYDVQQKEw9BY2VwdGEuY29tIFMuQS4xLDAqBgNV
BAsTI0NlcnRpZmljYWRvIENsYXNlIDMgUGVyc29uYSBOYXR1cmFsMRwwGgYJ
KoZIhvcNAQkBFg1uY2hlbGVAc2lpLmNsMSowKAYDVQQDEyFOSUNPTEFTIFpB
UFJJQU4gQ0hFTEVCSUZTS0kgQkFFWkEwgZ8wDQYJKoZIhvcNAQEBBQADgY0A
MIGJAoGBANj2+JBAdfZuzZjlGHsPX9+t9ucq0y0zzFO6s2BtgNPwbLm8Q3YE
MNjqdMes17zlV7GCPSpD17Q+aKyiTI8MWEb/cFFncohO++kb+JY3yTuxEISC
tXw0A5z7dB3nwUtOl84NkCwby0fqXtb3OgtNg6/fb/pZywPJp2YiZUR53YTV
AgMBAAGjggJyMIICbjAdBggrBgEEAbVrDwQRFg9BY2VwdGEuY29tIFMuQS4w
JQYDVR0RBB4wHKAaBggrBgEEAcEBAaAOFgwxMC40MTEuODcxLTIwDwYIKwYB
Jh0z1DR3Pl3xOiaFIjSXsQO2PSzcA3wZXYF+KDrMul8e5lAF2NNiLmMVtXEx
ZykMaTGGWS0ZETDhJmBwEZGpP4+lt/JhgwF1Sb6wdrXp7MFCJUc1Tj+/5JqH
1kP0E63/hVElrcP0g8Zn8Z+vr/PMGW1kKgE0IyS4iJ8eIhNSK5phFyKJUn0l
BmIZX7u89d5u7X8=</X509Certificate>
         </X509Data>
      </KeyInfo>
   </Signature>
</getToken>
```

Figura 1.4

CAPÍTULO 5

PARÁMETROS DE SALIDA

5.1.1 Parámetros de Salida

La salida de los Servicios corresponden a un “string” XML codificado según estándar XML, por lo tanto es necesario que el programa cliente sea capaz de decodificar el “string” y llevarlo a formato original - decodificado, los campos de retorno son: ESTADO, GLOSA, DATOS(SEED o TOKEN)

Donde

Campo	Tipo	Largo	Detalle	Obligatorios
ESTADO	String Alfanum	1-8	Código Estado	S
GLOSA	String Alfanum.	1-1	Detalle Estado	S
DATOS	String numérico	1-5	El nombre de este tag, varia dependiendo de los datos solicitados(Token, Seed), por ejemplo, si estamos solicitando Token el tag de datos se llamaría <TOKEN>, lo mismo para el Seed.	S

Figura 1.5

5.2.1 Estados de Salida

Los Estados de Salida se detallaran dependiendo del WS que corresponda.

5.2.2 Estados de Salida de CrSeed son:

Estado	Detalle Estado
00	OK genera Semilla
-1	No se registro línea en el Archivo de Configuración
-2	ERROR: RETORNO. ✓ "ERROR RETORNO" ✓ "NO PUEDO CREAR O ACT. TOKEN"

Figura 1.6

5.2.2 Los Estados de Salida de GetTokenFromSeed son:

Estado	Detalle Estado
00	Token Creado
01	XML Inválido (IOException), función valSignedXml
02	XML Inválido (SAXException), función valSignedXml
03	XML Inválido (ParserConfigurationException), función valSignedXml
04	XML Inválido, elemento “Signature” no existe, función valSignedXml
05	XML Inválido, firma invalida, función valSignedXml
06	XML Inválido, elemento “Semilla” no existe, función getSeed
07	ERROR (MessageException).
08	ERROR RETORNO : ✓ "PARAMETROS INCORRECTOS" ✓ "TIME-OUT DEL SEED" ✓ "NO GENERA TOKEN func:CreaToken" ✓ "NO PUEDO ACT. SEED CON TOKEN" ✓ "TIME-OUT del SEED" ✓ "NO Existe SEED"
09	ERROR (MessageException).
10	ERROR RETORNO: ✓ "ERROR RETORNO DATOS" ✓ "NO PUEDO CREAR O ACT. TOKEN"
11	XML Inválido, elemento “Certificate” no existe, función getCertificado
12	ERROR (12) (MessageException)
21	Firma invalida(La llave pública no coincide con la del certificado).
-3	Error en Autenticación
-07	Error (12) parse ERROR en Validación del RUT (verificar que el usuario se encuentre registrado en el SII con la opción de autenticación mediante “Certificado Digital”

Figura 1.7

5.2.3 Ejemplos de Salida

A continuación se mostrará una serie de ejemplos de salida en ambos formatos Codificado y Decodificado.

5.2.3.1 Ejemplo Parámetros de Salida WSDL Codificado CrSeed.jws

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
   <soapenv:Body>
      <ns1:getSeedResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="https://palena.sii.cl/DTEWS/CrSeed.jws">
         <getSeedReturn xsi:type="xsd:string">&lt;?xml version=&quot;1.0&quot;
encoding=&quot;UTF-8&quot;?&gt;
&lt;SII:RESPUESTA
xmlns:SII=&quot;http://www.sii.cl/XMLSchema&quot;&gt;&lt;SII:RESP_HDR&gt;&lt;ESTADO&gt;00
&lt;/ESTADO&gt;&lt;/SII:RESP_HDR&gt;&lt;SII:RESP_BODY&gt;&lt;SEMILLA&gt;000000000078&lt;/
S  EMILLA&gt;&lt;/SII:RESP_BODY&gt;&lt;/SII:RESPUESTA&gt;</getSeedReturn>
      </ns1:getSeedResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

Formato Decodificado de los Parámetros de Salida CrSeed.jws

```
<SII:RESPUESTA xmlns:SII="http://www.sii.cl/XMLSchema">
  - <SII:RESP_HDR>
      <ESTADO>00</ESTADO>
    </SII:RESP_HDR>
  - <SII:RESP_BODY>
      <SEMILLA>000000000078</SEMILLA>
    </SII:RESP_BODY>
</SII:RESPUESTA>
```

Figura 1.8

5.2.3.2 Ejemplo Parámetros de Salida WSDL Codificado GetTokenFromSeed.jws

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
   <soapenv:Body>
      <ns1:getTokenResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://palena.sii.cl/DTEWS/GetTokenFromSeed.jws">
         <getTokenReturn xsi:type="xsd:string">&lt;?xml version=&quot;1.0&quot;
encoding=&quot;UTF-8&quot;?&gt;
&lt;SII:RESPUESTA
xmlns:SII=&quot;http://www.sii.cl/XMLSchema&quot;&gt;&lt;SII:RESP_HDR&gt;&lt;ESTADO&gt;00
&lt;/ESTADO&gt;&lt;GLOSA&gt;Token
Creado&lt;/GLOSA&gt;&lt;/SII:RESP_HDR&gt;&lt;SII:RESP_BODY&gt;&lt;TOKEN&gt;XAuSbYXiNh9Ik&
lt;/TOKEN&gt;&lt;/SII:RESP_BODY&gt;&lt;/SII:RESPUESTA&gt;</getTokenReturn>
      </ns1:getTokenResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

5.2.3.4 Ejemplo Parámetros de Salida WSDL Decodificado GetTokenFromSeed.jws

```
<SII:RESPUESTA xmlns:SII="http://www.sii.cl/XMLSchema">
  - <SII:RESP_HDR>
      <ESTADO>00</ESTADO>
      <GLOSA>Token Creado</GLOSA>
    </SII:RESP_HDR>
  - <SII:RESP_BODY>
      <TOKEN>XAuSbYXiNhIk</TOKEN>
    </SII:RESP_BODY>
  </SII:RESPUESTA>
```

Figura 1.9

CAPÍTULO 6

EJEMPLOS XML DE RESPUESTA

6.1 Ejemplo Salida genera Semilla:

6.1.1 Ejemplo de Salida, Estado 00 (genera Semilla)

```
<?xml version="1.0" encoding="UTF-8" ?>
- <SII:RESPUESTA xmlns:SII="http://www.sii.cl/xxx">
- <SII:RESP_HDR>
    <ESTADO>00</ESTADO>
  </SII:RESP_HDR>
- <SII:RESP_BODY>
    <SEED>000000000064</SEED>
  </SII:RESP_BODY>
</SII:RESPUESTA>
```

6.1.2 Ejemplo de Salida, Estado -1 (Error No genera Semilla)

```
<?xml version="1.0" encoding="UTF-8" ?>
- <SII:RESPUESTA xmlns:SII="http://www.sii.cl/xxx">
- <SII:RESP_HDR>
    <ESTADO>-1</ESTADO>
    <GLOSA>Error : (Message Exception) </GLOSA>
  </SII:RESP_HDR>
</SII:RESPUESTA>
```

6.1.3 Ejemplo de Salida, Estado -2 (Error : BD)

```
<?xml version="1.0" encoding="UTF-8" ?>
- <SII:RESPUESTA xmlns:SII="http://www.sii.cl/xxx">
- <SII:RESP_HDR>
    <ESTADO>-2</ESTADO>
    <GLOSA>ERROR RETORNO</GLOSA>
  </SII:RESP_HDR>
</SII:RESPUESTA>
```

Nota: El estado -2, tiene asociado dos errores, detallados en la tabla “Estados de Salida Genera Semilla”. Aquí se hace mención a sólo uno de ellos a modo de ejemplo.

6.2 Ejemplo Salida genera Token

6.2.1 Ejemplo de Salida, Estado 00 (genera Token)

```
<SII:RESPUESTA xmlns:SII="http://www.sii.cl/xxx">
  - <SII:RESP_HDR>
      <ESTADO>00</ESTADO>
     <GLOSA>Token Creado</GLOSA>
    </SII:RESP_HDR>
  - <SII:RESP_BODY>
      <TOKEN /> AB82001ABRT<TOKEN />
    </SII:RESP_BODY>
  </SII:RESPUESTA>
```

6.2.2 Ejemplo de Salida, Estado 01 (Error:XML invalido (IOException), función valSignedXml)

```
<SII:RESPUESTA xmlns:SII="http://www.sii.cl/xxx">
  - <SII:RESP_HDR>
      <ESTADO>01</ESTADO>
  <GLOSA> XML invalido (IOException), función valSignedXml</GLOSA>
    </SII:RESP_HDR>
  </SII:RESPUESTA>
```

6.2.3 Ejemplo de Salida, Estado 02 (Error: XML invalido, (SAXException), función valSignedXml)

```
<SII:RESPUESTA xmlns:SII="http://www.sii.cl/xxx">
  - <SII:RESP_HDR>
      <ESTADO>02</ESTADO>
      <GLOSA> XML Invalido (SAXException), funcion valSignedXml</GLOSA>
    </SII:RESP_HDR>
  </SII:RESPUESTA>
```

6.2.4 Ejemplo de Salida, Estado 03 (Error: XML Invalido ParserConfigurationException), funcion valSignedXml)

```
<SII:RESPUESTA xmlns:SII="http://www.sii.cl/xxx">
  - <SII:RESP_HDR>
      <ESTADO>03</ESTADO>
      <GLOSA> XML Invalido (ParserConfigurationException), funcion valSignedXml </GLOSA>
    </SII:RESP_HDR>
  </SII:RESPUESTA>
```

6.2.5 Ejemplo de Salida, Estado 04 (Error: XML Invalido, elemento “Signature” no existe, funcion valSignedXml)

```
<SII:RESPUESTA xmlns:SII="http://www.sii.cl/xxx">
  - <SII:RESP_HDR>
      <ESTADO>04</ESTADO>
      <GLOSA> XML Invalido, elemento “Signature” no existe, función valSignedXml </GLOSA>
    </SII:RESP_HDR>
  </SII:RESPUESTA>
```

6.2.6 Ejemplo de Salida, Estado 05 (Error: XML Invalido, firma invalida, función valSignedXml)

```
<SII:RESPUESTA xmlns:SII="http://www.sii.cl/xxx">
  - <SII:RESP_HDR>
      <ESTADO>05</ESTADO>
      <GLOSA> XML Invalido, firma invalida, funcion valSignedXml
        </GLOSA>
    </SII:RESP_HDR>
  </SII:RESPUESTA>
```

6.2.7 Ejemplo de Salida, Estado 06 (Error: XML Invalido, elemento “Semilla” no existe, función getSeed)

```
<?xml version="1.0" encoding="UTF-8" ?>
- <SII:RESPUESTA xmlns:SII="http://www.sii.cl/xxx">
  - <SII:RESP_HDR>
      <ESTADO>06</ESTADO>
      <GLOSA> XML Invalido, elemento “Semilla” no existe, función
      getSeed</GLOSA>
  </SII:RESP_HDR>
</SII:RESPUESTA>
```

6.2.8 Ejemplo de Salida, Estado 07 (ERROR (MessageException)

```
<?xml version="1.0" encoding="UTF-8" ?>
- <SII:RESPUESTA xmlns:SII="http://www.sii.cl/xxx">
  - <SII:RESP_HDR>
      <ESTADO>07</ESTADO>
      <GLOSA> ERROR (MessageException)</GLOSA>
  </SII:RESP_HDR>
</SII:RESPUESTA>
```

6.2.9 Ejemplo de Salida, Estado 08 (ERROR :Retorno)

```
<?xml version="1.0" encoding="UTF-8" ?>
- <SII:RESPUESTA xmlns:SII="http://www.sii.cl/xxx">
  - <SII:RESP_HDR>
      <ESTADO>08</ESTADO>
      <GLOSA> TIME-OUT DEL SEED </GLOSA>
  </SII:RESP_HDR>
</SII:RESPUESTA>
```

Nota : El estado 08, tiene varios errores asociados, los que se detallan en la tabla “Estados de Salida de Genera Token”. Aquí se hace mención a sólo uno de ellos a modo de ejemplo.

6.2.10 Ejemplo de Salida, Estado 09 (ERROR (MessageException))

```
<?xml version="1.0" encoding="UTF-8" ?>
- <SII:RESPUESTA xmlns:SII="http://www.sii.cl/xxx">
  - <SII:RESP_HDR>
      <ESTADO>09</ESTADO>
      <GLOSA> ERROR (MessageException) </GLOSA>
  </SII:RESP_HDR>
</SII:RESPUESTA>
```

6.2.11 Ejemplo de Salida, Estado 10 (ERROR: RETORNO DATOS)

```
<?xml version="1.0" encoding="UTF-8" ?>
- <SII:RESPUESTA xmlns:SII="http://www.sii.cl/xxx">
  - <SII:RESP_HDR>
      <ESTADO>10</ESTADO>
      <GLOSA> ERROR RETORNO DATOS</GLOSA>
  </SII:RESP_HDR>
</SII:RESPUESTA>
```

6.2.12 Ejemplo de Salida, Estado 11 (ERROR: XML Inválido, elemento “Certificate” no existe, función getCertificado)

```
<?xml version="1.0" encoding="UTF-8" ?>
- <SII:RESPUESTA xmlns:SII="http://www.sii.cl/xxx">
  <SII:RESP_HDR>
    <ESTADO>11</ESTADO>
   <GLOSA>XML Invalido, elemento “Certificate” no existe, función getCertificado
     </GLOSA>
  </SII:RESP_HDR>
</SII:RESPUESTA>
```

6.2.13 Ejemplo de Salida, Estado 12 (ERROR (12) (MessageException))

```
<?xml version="1.0" encoding="UTF-8" ?>
- <SII:RESPUESTA xmlns:SII="http://www.sii.cl/xxx">
  <SII:RESP_HDR>
    <ESTADO>12</ESTADO>
   <GLOSA>ERROR (12) (MessageException)</GLOSA>
   </SII:RESP_HDR>
 </SII:RESPUESTA>
```

6.2.14 Ejemplo de Salida, Estado -3 (Error en Autenticación)

```
<?xml version="1.0" encoding="UTF-8" ?>
- <SII:RESPUESTA xmlns:SII="http://www.sii.cl/xxx">
  <SII:RESP_HDR>
    <ESTADO>-3</ESTADO>
   <GLOSA>Error en Autenticación </GLOSA>
   </SII:RESP_HDR>
 </SII:RESPUESTA>
```

CAPITULO 7

GUIA PARA REALIZAR PRUEBAS

Para probar los WS de Autenticación Automática, se deben seguir los siguientes pasos:

1.- Para obtener una Semilla, se debe invocar el servicio:

https://palena.sii.cl/DTEWS/CrSeed.jws?WSDL

2.- Firmar Semilla Mediante un Cliente (ver Formato XML Entrada)

3.- Invocando el servicio GetTokenFromSeed.jws, para el envió del XML con las Semilla Firmada.

https://palena.sii.cl/DTEWS/GetTokenFromSeed.jws?WSDL

4.- Se Obtiene Token

Nota: Si bien en este manual se detalla la autenticación automática para el ambiente de producción, el procedimiento es el mismo para el ambiente de certificación, solo se debe cambiar el nombre del servidor, reemplazando a palena.sii.cl por maullin.sii.cl.

Por ejemplo:

1.- Para obtener una Semilla en certificación:

https://maullin.sii.cl/DTEWS/CrSeed.jws?WSDL

2.- Generar un Token

https://maullin.sii.cl/DTEWS/GetTokenFromSeed.jws?WSDL

CAPITULO 8

Como firmar una Semilla

Para firmar una semilla, se deben seguir los siguientes pasos:

- ✓ Obtener una semilla (invocando al WS CrSeed.jws de certificación o producción)

  La salida de CrSeed.jws corresponde al siguiente XML:

```
<?xml version="1.0" encoding="UTF-8" ?>
 <SII:RESPUESTA xmlns:SII="http://www.sii.cl/XMLSchema">
   <SII:RESP_BODY>
      <SEMILLA>000002360958</SEMILLA>
    </SII:RESP_BODY>
   <SII:RESP_HDR>
      <ESTADO>00</ESTADO>
    </SII:RESP_HDR>
  </SII:RESPUESTA>
```

- ✓ Una vez obtenido el xml, que incluye la semilla, se debe rescatar el campo a firmar, el campo a firmar corresponde a:

```
<SEMILLA>000002360958</SEMILLA>
```

- ✓ Una vez determinado el campo a firmar, este debe ser entregado al objeto (getToken), tal como se muestra en la figura 1.10.

```
<getToken>
     <item>
     <Semilla>000002360958</Semilla>
  </item>

</getToken>
```

Figura 1.10

Una vez integrado el objeto getToken con la semilla, se deben realizar los siguientes pasos.

- ✓ Aplicar la transformación y la canonicalización a este objeto.(Corresponde a una función interna propia de la librería de firma).
- ✓ Calcular el hash al objeto, para luego crear el elemento DigestValue (Corresponde a una función interna propia de la librería de firma).
- ✓ Crear el elemento SignedInfo .(Corresponde a una función interna propia de la librería de firma).
- ✓ Canonical izar y calcular la firma .(Corresponde a una función interna propia de la librería de firma)
- ✓ Crear el elemento SignatureValue con el valor de la firma (Corresponde a una función interna propia de la librería de firma)-
- ✓ Generar la información de claves (elemento keyInfo). .(Corresponde a una función interna propia de la librería de firma).

Nota: Estos pasos pueden ser realizados en forma manual o mediante el uso de una librería)

Por último se debe construir el elemento `Signature` que incluye los elementos `SignedInfo, SignatureValue y keyInfo`, tal como se puede observar en la figura 1.11, el cual comienza con `<?xml version="1.0"?>` (todo el elemento debe estar codificado ejemplo: < por < etc).

```
<?xml version="1.0"?>
<getToken>
  <item>
  <Semilla>000002248802</Semilla></item>
<Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
  <SignedInfo>
    <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
    <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
    <Reference URI="">
      <Transforms>
        <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
      </Transforms>
      <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
      <DigestValue>kZvDbarenZxZPbWY7gNLxOan/NI=</DigestValue>
    </Reference>
  </SignedInfo>
<SignatureValue>czuCSQX5uoHzOOIS0V3bRe5WK8MNMzL6pm2dEpRVLDDAqj8fGtfOjPBAOzoY9MHtB9O1Ml4lpjRYEJ
o+9uAI+g/mC6PT20wFcOMr0J2SmJmlf+6MkNhoHbfVkGJ4zxGvCx1ZvtNLAkJovFqBlFaaoJ08Rvkd2FrSXjRIf+NqUYo=
</SignatureValue>
    <KeyInfo>
      <KeyValue>
<RSAKeyValue>
<Modulus>
wFgMvA/vy1BXOBXOWI5fW/n45OHf4g1WYWLvBd68A6vpFlv6bEapsMabeyaQjwa/
UCAt75dNQdfjSTgLxMeKvjuatItAv4Sq4ncAe5POHRVwu9eziU+9+LQBa5FemDEM
7pVHjGR1heSAgeIuPBv7j1TKwv+kRE+iUcYFiKwXH9M=
</Modulus>
<Exponent>AQAB</Exponent>
</RSAKeyValue>
</KeyValue>
        <X509Data>
<X509Certificate>MIIEbDCCA9egAwIBAgIDAgSAMAsGCSqGSIb3DQEBBDCBsTEdMBsGA1UECBQU
UmVnaW9uIE1ldHJvcG9saXRhbmExETAPBgNVBAcUCFNhbnRpYWdvMSIwIAYD
VQQDFBlFLUNlcnRjaGlsZSBDQSBJbnRlcm1lZGlhMTYwNAYDVQQLFC1FbXBy
ZXNhIE5hY2lvbmFsIGRlIENlcnRpZmljYWNpb24gRWxlY3Ryb25pY2ExFDAS
BgNVBAoUC0UtQ0VSVENISUxFMQswCQYDVQQGEwJDTDAeFw0wMzA3MTUxODQx
NTJaFw0wNDA3MTQwMDAwMDBaMIG6MQswCQYDVQQGEwJDTDEWMBQGA1UECBQN
TWV0cm9wb2xpdGFuYTERMA8GA1UEBxQIU2FudGlhZ28xKDAmBgNVBAoUH1Nl
cnZpY2lvcyBkZSBJbXB1ZXN0b3MgSW50ZXJub3MxGTAXBgNVBAsUEE9maWNp
bmEgSW50ZXJuZXQxHDAaBgNVBAMUE1p1bGVtYSBPbGd1aW4gVHJhcm8xHTAb
BgkqhkiG9w0BCQEWDnpvbGd1aW5Ac2lpLmNsMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQDAWAy8D+/LUFc4Fc5Yjl9b+fjk4d/iDVZhYu8F3rwDq+kW
W/psRqmwxpt7JpCPBr9QIC3vl01B1+NJOAvEx4q+O5q0i0C/hKridwB7k84d
FXC717OJT734tAFrkV6YMQzulUeMZHWF5ICB4i48G/uPVMrC/6RET6JRxgWI
rBcf0wIDAQABo4IBiTCCAYUwIwYDVR0RBBwwGqAYBggrBgEEAcEBAaAMFgox
MDQ1MDM1NC0zMAwGA1UdEwEB/wQCMAAwPAYDVR0fBDUwMzAxoC+gLYYraHR0
cDovL2NybC5lLWNlcnRjaGlsZS5jbC9FY2VydGNoaWxlQ0FJLmNybDAjBgNV
HRIEHDAaoBgGCCsGAQQBwQECoAwWCjk2OTI4MTgwLTUwgd8GA1UdIASB1zCB
1DCB0QYIKwYBBAHDUgUwgcQwLwYIKwYBBQUHAgEWI2h0dHA6Ly93d3cuZS1j
ZXJ0Y2hpbGUuY2wvMjAwMC9DUFMvMIGQBggrBgEFBQcCAjCBgxqBgEVsIHRp
dHVsYXIgaGEgc2lkbyB2YWxpZGFkbyBlbiBmb3JtYSBwcmVzZW5jaWFsLCBx
dWVkYW5kbyBoYWJpbGl0YWRvIGVsIENlcnRpZmljYWRvIHBhcmEgdXNvIHRy
aWJ1dGFyaW8sIHBhZ29zLCBjb21lcmNpbyB5IG90cm9zMAsGA1UdDwQEAwIE
8DALBgkqhkiG9w0BAQQDgYEAka3Y5VbyjbHwF9sew2+6ZRaL4zIQgv0Cnd9p
VGYqSVFQz2YK/AEyasFoWm2evdlo5QJ8TjKqd+QlI674tvAumNIARksCZeUW
hpjdD/vLp7exQUoVKOCInQVQQ6LUDAf6v9vgIB9Mwf6yTbnxwdvv1EeiLQEBd
2Af9oF7fVsXKLsY=</X509Certificate>
</X509Data>
    </KeyInfo>
</Signature>
</getToken>
```

Figura 1.11

El elemento “Signature”, indicado en la figura 1.11, corresponde al parámetro de entrada requerido por el WS GetTokenFromSeed.jws, que permite generar un token.

Ver punto: 4.1.3 del manual.

https://palena.sii.cl/DTEWS/GetTokenFromSeed.jws?WSDL.

ANEXO 1

1.- Ejemplo de TOKEN

En este anexo de muestra un ejemplo de un Token

```
TOKEN=gd43dh6sfE34Kd3
```