SAMLINK

SAMLINK CERTIFICATE SERVICE

SERVICE DESCRIPTION FOR SOFTWARE COMPANIES

31 January 2018

Table of contents

1 Introduction

The WSDL describes what kinds of messages can be sent and received in Samlink's certificate retrieval service. The WSDL describes three operations:

- getCertificate
- getServiceCertificates
- revokeCertificate

Samlink's system solution only supports the getCertificate operation.

The SOAP message of Samlink's certificate retrieval service does not use encryption or compression. Samlink's system solution does not support compression, since the messages passing through the system are typically small. In the WSDL, ApplicationRequest includes a Base64-coded CertApplicationRequest, the schema of which is described in CertApplicationRequest_20090422.xsd (Appendix 2). In the response message, ApplicationResponse includes a Base64-coded CertApplicationResponse, the schema of which is described in CertApplicationResponse_20090422.xsd (Appendix 3).

This document describes the getCertificate operation of Samlink's certificate retrieval service, and the methods of use of the operation's CertApplicationRequest and CertApplicationResponse elements.

2 Use cases of the customer software's certificate retrieval

During the first time, a PKI key pair must be created in the customer software. A certificate request (CSR) is generated using this key pair. The CSR is sent to Samlink via the WS channel, so that during the first time, the customer uses a one-off password (TransferKey) for identification, and the valid certificate during any following times.

The same CSR may be used for retrieving a valid certificate that has previously been retrieved. When the certificate is valid for fewer than 60 days, the customer may retrieve a new certificate as long as the certificate request has been created with a new key pair. An old key pair cannot be used for retrieving a new certificate.

A new certificate may only be retrieved if the certificate has not yet been retrieved, or the current certificate will expire in less than 60 days. If an attempt is made to renew the certificate in violation of the aforementioned conditions, that results in an error according to the table below:

key	**user**	**validity**	**description**
new	new	no certificate	Normal creation of a new one
new	old	less than 60	Normal renewal
new	old	over 60	ERROR: The certificate cannot yet be renewed
old	new	less than 60	ERROR: Defective user information
old	new	over 60	ERROR: Defective user information
old	old	less than 60	ERROR: The private key must be re-generated
old	old	over 60	The old structure is returned

3 getCertificate operation

The elements of the SOAP message are explained below in more detail.

3.1 SenderId

An identifier and username that is used to identify the sender of a message. This was given to the company when signing the agreement. The value is the same as in the CustomerId element of the CertApplicationRequest element described below and in the Surname (SN) field of the certificate request.

3.2 RequestId

A delivery identifier, aiming to make it easier to resolve problems. Samlink will not check if the identifier has been previously used.

3.3 Timestamp

A time stamp indicating when Application Request Header has been created.

3.4 ApplicationRequest

Includes a Base64-coded CertApplicationRequest.

4 CertApplicationRequest

4.1 CustomerId

An identifier and username that is used to identify the party creating a signature request for the certificate. This was given to the company when signing the agreement.
The value is the same as in the SenderId element described above, and in the certificate request's Surname (SN) field.

4.2 Timestamp

Compulsory in a scheme. Value is not used.

4.3 Environment

Value: PRODUCTION The test feature is not used.

4.4 SoftwareId

The software's precise identifier aims to make it easier to resolve problems.

4.5 Command

Value: GetCertificate

4.6 ExecutionSerial

Not in use

4.7 Encryption

Not in use

4.8 EncryptionMethod

Not in use

4.9 Compression

Not in use

4.10 CompressionMethod

Not in use

4.11 Service

Compulsory in a scheme. The default value ISSUER is used.

4.12 Content

Base64-coded signature request (PKCS#10) of the certificate.

4.13 TransferKey

A one-off password (8+8) used on the first time. The first segment is given to the company when signing the agreement; the second segment will be delivered separately by mail. Once the customer has received the signed certificate, the certificate (Signature element) will be used for identifying the customer, in which case the TransferKey element must no longer appear. In the WS channel, the one-off password is marked as "used" when the signed certificate is supplied to the customer.

4.14 SerialNumber

Not in use

4.15 Signature

On the first time, the customer identifies themselves by using the one-off password (TransferKey element). After that, the TransferKey element is no longer used, but the identification takes place by means of an XML signature.

5 CertApplicationResponse

5.1 CustomerId

The identifier the customer uses in the CertApplicationRequest.

5.2 Timestamp

The time stamp indicates when CertApplicationResponse has been created.

5.3 ResponseCode and ResponseText

ResponseCode	ResponseText
0	OK
5	UNKNOWN APPLICATION REQUEST
6	THE CERTIFICATE CANNOT YET BE RENEWED
7	DEFECTIVE USER INFORMATION
8	THE PRIVATE KEY MUST BE RE-GENERATED
12	THE FORMAL CHECK OF THE MATERIAL FAILED
26	TECHNICAL ERROR
30	IDENTIFICATION FAILED

5.4 ExecutionSerial

Not in use

5.5 Encrypted

Not in use

5.6 EncryptionMethod

Not in use

5.7 Compressed

Not in use

5.8 CompressionMethod

Not in use

5.9 CustomerExtension

Not in use

5.10 Certificates

Includes one Certificate element.

5.10.1 Certificate

Includes Name, Certificate, and CertificateFormat elements.

5.10.1.1 Name

The certificate's subject, such as
SURNAME=9923233, CN=COMPANY LTD, O=Materialservices-Samlink, C=FI

5.10.1.2 Certificate

Base64-coded signed certificate (X509v3).

5.10.1.3 CertificateFormat

Value: X509

5.11 Signature

Samlink will add an XML signature to all messages. The customer is responsible for ensuring that the signature is correct.

6 APPENDICES

Appendices can be downloaded from Samlink's web pages.

- WSDL documentation for the certificate service
- CertApplicationRequest.xsd schema
- CertApplicationResponse.xsd schema

7 Example messages

7.1 Request message

```
<soapenv:Envelope xmlns:opc="http://mlp.op.fi/OPCertificateService" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
    <soapenv:Header>
    </soapenv:Header>
    <soapenv:Body wsu:Id="id-3" xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
        <opc:getCertificatein>
            <opc:RequestHeader>
                <opc:SenderId>99415033</opc:SenderId>
                <opc:RequestId>12345678</opc:RequestId>
                <opc:Timestamp>2016-05-04T08:12:26.30</opc:Timestamp>
            </opc:RequestHeader>
            <opc:ApplicationRequest>PD94bWwgdmVyc2lvbj0iMS4wIj8+...
            WNhdGlvblJlcXVlc3Q+</opc:ApplicationRequest>
        </opc:getCertificatein>
    </soapenv:Body>
</soapenv:Envelope>
```

7.2 Response message

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
       xmlns:cer="http://mlp.op.fi/OPCertificateService">
       <soapenv:Header>
              <wsse:Security soapenv:mustUnderstand="1"
                     xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
                     <wsu:Timestamp wsu:Id="Timestamp-c8599f50-342d-4fb5-9031-53d5cdb045eb"
                            xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
                            <wsu:Created>2016-05-04T05:12:27Z</wsu:Created>
                            <wsu:Expires>2016-05-04T05:17:27Z</wsu:Expires>
                     </wsu:Timestamp>
                     <wsse:BinarySecurityToken
                            wsu:Id="SecurityToken-2f291284-dfcf-4ebc-a192-aa2ca14d720e"
                            EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0#Base64Binary"
                            ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3"
                            xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">MIIFETCCAvmgAwIBAgIQUHFBQ1tAv...IQW1tDtk=
                     </wsse:BinarySecurityToken>
                     <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
                            <SignedInfo>
                                   <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
                                   <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
                                   <Reference URI="#Timestamp-c8599f50-342d-4fb5-9031-53d5cdb045eb">
                                          <Transforms>
                                                 <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
                                          </Transforms>
                                          <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
                                          <DigestValue>57KNPysmumFct7yiRUosf3uyfJc=</DigestValue>
                                   </Reference>
                                   <Reference URI="#Body-b1b9dbf7-9980-4867-9fda-f9479a73441c">
                                          <Transforms>
                                                 <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
                                          </Transforms>
                                          <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
                                          <DigestValue>8ipUWG+6e0J6i94gydmIiR31Wyo=</DigestValue>
                                   </Reference>
                            </SignedInfo>
                            <SignatureValue>DvB+0cy21aidmg2THNx5jF5HiL05TH27AqLoLI+ZGm...r9BqgSIg==
                            </SignatureValue>
                            <KeyInfo>
                                   <wsse:SecurityTokenReference xmlns="">
                                          <wsse:Reference
                                                 URI="#SecurityToken-2f291284-dfcf-4ebc-a192-aa2ca14d720e"
                                                 ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3" />
                                   </wsse:SecurityTokenReference>
                            </KeyInfo>
                     </Signature>
              </wsse:Security>
       </soapenv:Header>
       <soapenv:Body wsu:Id="Body-b1b9dbf7-9980-4867-9fda-f9479a73441c"
              xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
              <cer:getCertificateout>
                     <cer:ResponseHeader>
                            <cer:SenderId>99415033</cer:SenderId>
                            <cer:RequestId>12345678</cer:RequestId>
                            <cer:Timestamp>2016-05-04T08:12:27+03:00</cer:Timestamp>
                            <cer:ResponseCode>0</cer:ResponseCode>
                            <cer:ResponseText>OK</cer:ResponseText>
                     </cer:ResponseHeader>
                     <cer:ApplicationResponse>PD94bWwgdmVyc2lvbj0iMS4...BvbnNlPg==
                     </cer:ApplicationResponse>
              </cer:getCertificateout>
       </soapenv:Body>
</soapenv:Envelope>
```