Danske Bank

Encryption, Signing and Compression in Financial Web Services

Details of how to call the Danske Bank financial web service

Table of Contents

Version history

Version	Date	Description of change	Author
2.2	5. January 2010	RFC 1952 added to clarify gzip. Section on future changes removed. Example XML added. Section on signatures added.	Mikkel T. Jensen
2.3	22. February 2010	RSA-OAEP replaced by RSA-v1.5. More requirements to signatures added.	Mikkel T. Jensen
2.4	16. April 2010	Section added on future changes to the financial WS specification. Introduction changed.	Mikkel T. Jensen
2.4.1	4. March 2010	Example XML corrected to use RSA 1.5.	Mikkel T. Jensen
2.4.2	18. April 2012	Corrected Introduction text	Christian Enevold
2.4.3	29. September 2012	Changes due to renaming of Sampo Pankki to Danske Bank.	Mikkel T. Jensen
2.4.4	4. December 2012	Changes to URL.	Lea Troels Møller Pedersen
2.4.5	20. August 2014	Corrected mistake with optional encryption	Mikkel T. Jensen
2.4.6	16. September 2014	Corrected diagram	Mikkel T. Jensen
2.4.7	12. May 2015	Error removed from transport signature example XML.	Mikkel T. Jensen
2.4.8	27. July 2017	Added references to EDI WS and PKI documents	Andreja Andric

Introduction

The Danske Bank Web Services solution is build on the Web Services specifications from the Federation of Finnish Financial services [1].

This document clarifies the use of XML encryption and compression and how this is implemented in Danske Bank. Principal aim of this document is to cover the details of the security elements in the EDI Web Service [2]. However, the requirements on XML encryption and enveloped signatures also apply to the PKI Service [3].

Regarding encryption

Our implementation of encryption

XML encryption is applied at the ApplicationRequest/ApplicationResponse level. This means that the ApplicationRequest (or ApplicationResponse) element is replaced by an EncryptedData element from the XML encryption standard. The EncryptedData contains the actual encrypted data (in a CipherData subelement) as well as sub-elements necessary for decryption (EncryptionMethod and KeyInfo). The reason this approach can work is that the ApplicationRequest/ApplicationResponse/EncryptedData element is base64 encoded when the actual web service call is made. This means that there is no schema-check of the ApplicationRequest/Response schema on the web service level. The schema-check of the

ApplicationRequest/Response should not be done until the actual XML has been reconstructed using base64 decoding, followed by decryption. The order of processing when receiving a web service call should be as follows:

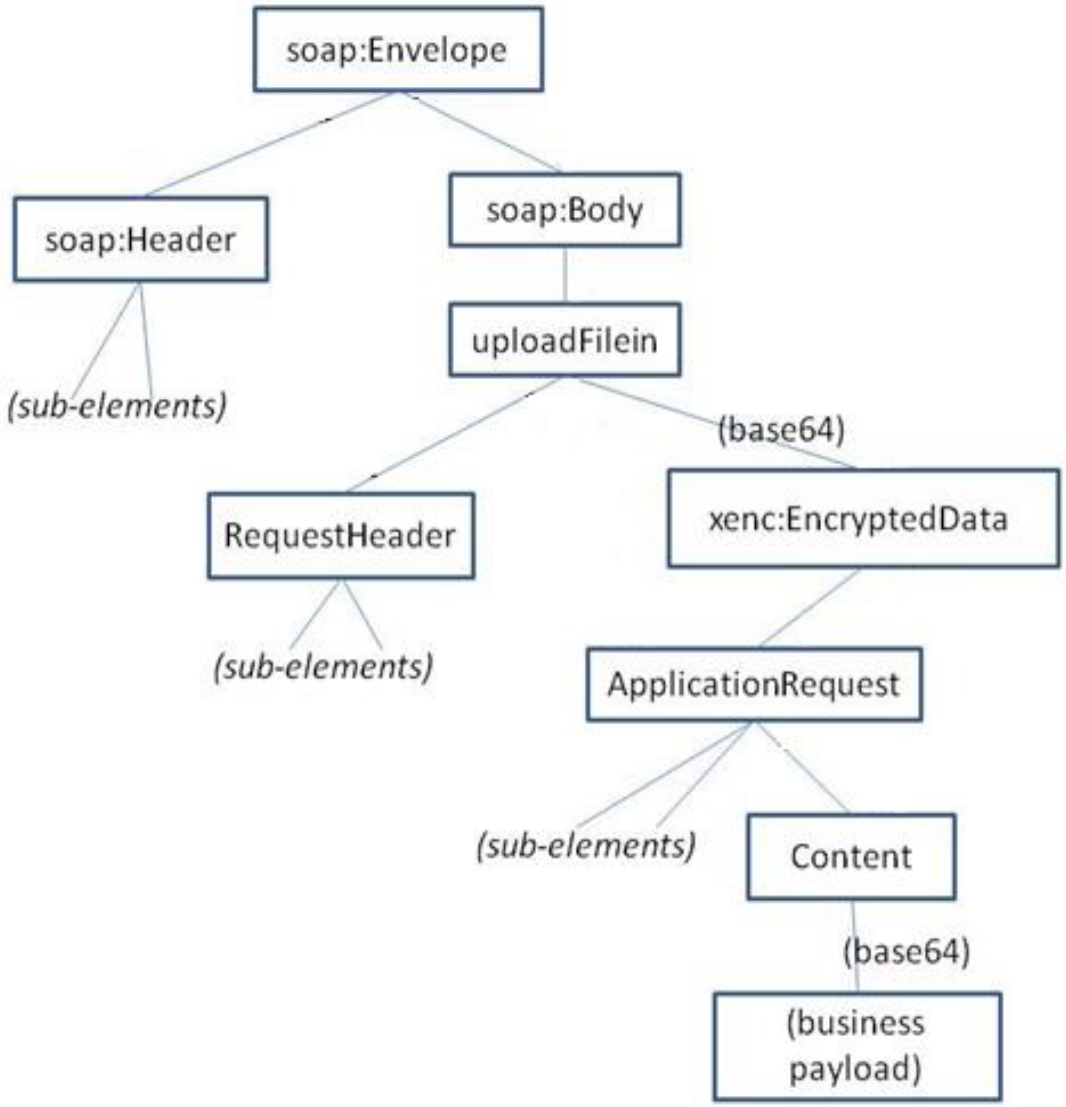
1. The web service call arrives at the server
2. As part of the web service call, the incoming message is checked against the WSDL. This does not include schema checking the ApplicationRequest, due to the base64 encoding.
3. The SOAP header and security information in it is processed, including the transport signature. If everything is not correct, an error message is returned and processing stops.
4. Processing based on RequestHeader can be done.
5. XML is constructed by base64 decoding the content of the ApplicationRequest in the incoming message (in the WSDL, it is defined to be of type xsd:base64Binary). The base64 decoding results in an EncryptedData element.
6. The EncryptedData element is decrypted using XML encryption functionality. This creates an ApplicationRequest element.
7. The ApplicationRequest element can be schema checked now.
8. Further processing of the ApplicationRequest.

The processing of a received ApplicationResponse on the client follows the same pattern. In the same way, the order of processing in the construction of XML prior to a web service call should be as follows:

1. The document for sending is compressed using Gzip
2. The Gzipped file is encoded in Base64 so it becomes printable
3. The result is enclosed in a <Content> tag.
4. The so made <Content> tag is placed inside an <ApplicationRequest> tag along with <CustomerId> and other information
5. The Compression is set to true
6. The ApplicationRequest is Digitally signed (business signatures).
7. The ApplicationRequest element is encrypted, producing an EncryptedData element.
8. The EncryptedData element is base64 encoded.
9. A RequestHeader and a soap Body are produced.
10. The base64 data produced in step 8 is inserted in the soap Body in the ApplicationRequest element.
11. The rest of the soap message is constructed (soap Header and soap Envelope)
12. The soap message is digitally signed (transport signature).
13. The webservice is called.

The construction of an ApplicationResponse on the server follows the same pattern. This description can be compared to the message lifecycle description on page 9 of the specification.

The layering of information in the web-service call is visualized in the following figure:



The figure shows the soap envelope containing the soap Header and soap Body. The soap Body contains an uploadFilein element (the name of this element depends on the operation called, it could also be getUserInfoin, downloadFileListin, downloadFilein, or deleteFilein). This element contains a RequestHeader, and an element containing base64Binary coded data. The base64Binary data can be decoded an EncryptedData element. The EncryptedData element will decrypt into an ApplicationRequest element.

The figure is closely related to the figures on pages 15 and 16 of the specification (version 1.05).

Regarding the Encryption and EncryptionMethod elements

The ApplicationRequest and ApplicationResponse elements contain subelements called *Encryption* (*Encrypted* in ApplicationResponse) and *EncryptionMethod*. These elements can give rise to confusion, since for instance the description of the Encryption elements states:

'If this element is present and the content is the string "true" (case-sensitive) it means that the Content is encrypted or the requested data should be encrypted by the bank'

The first part of the sentence is misleading, since if the Encryption element is visible (i.e. not itself in an encrypted form) then the ApplicationRequest and the Content are no longer encrypted (they may have been in a previous processing step). The only true use of the element is stated in the second part of the sentence: If the element contains "true", the response should be encrypted by the bank. This interpretation also implies that the Encrypted element in ApplicationResponse serves no purpose, since no reply is needed in response to the response.

Similarly, the EncryptionMethod element is only used to communicate the cipher to be used in the response. For this, one EncryptionMethod element is inadequate, since two encryption ciphers are needed: One for symmetric encryption to encrypt the actual data, and one for asymmetric encryption to encrypt the symmetric key. This problem should be solved in a future version of the specification. At present it should be up to the individual banks to decide how to handle this. Maybe the same encryption ciphers used in the request should be used in the response, or maybe the same ciphers should always be used.

The specification does not state a default behavior if the Encryption and EncryptionMethod elements are not present in ApplicationRequest.

Regarding the XML Encryption specification

An introduction to the XML Encryption specification is outside the scope of this text. It is recommended to read the specification, which can be found at http://www.w3.org/TR/xmlenc-core/. However, since the specification contains some degrees of freedom as to the XML elements and their content, a brief description of the expected content of an EncryptedData element (which is the root element of encrypted data) will be given here. The encryption used is in two layers:

1. The actual business data is encrypted using symmetric cryptography. The key used to perform this encryption is called the *ephemeral key.* A new ephemeral key is generated every time a message is encrypted. In the example below, the encrypted business data is contained in the EncryptedData/CipherData element. The symmetric cipher is declared in the EncryptedData/EncryptionMethod/@Algorithm attribute.
2. The ephemeral key is encrypted using asymmetric cryptography. The encrypted ephemeral key is contained in the EncryptedData/KeyInfo/EncryptedKey element. The asymmetric cipher is declared in the EncryptedData/KeyInfo/EncryptedKey/EncryptionMethod/@Algorithm attribute. The actual encrypted ephemeral key is contained in the EncryptedData/KeyInfo/EncryptedKey/CipherData element. The certificate used to perform the asymmetric encryption is contained in the EncryptedData/KeyInfo/EncryptedKey/KeyInfo/X509Data/X509Certificate element.

The expected structure of the EncryptedData element can be found in the abbreviated example below.

```
<xenc:EncryptedData xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
Type="http://www.w3.org/2001/04/xmlenc#Element">
      <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#tripledes-cbc"/>
      <dsig:KeyInfo xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
```

```
        <xenc:EncryptedKey>
            <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5"/>
            <dsig:KeyInfo>
                <dsig:X509Data>
                    <dsig:X509Certificate>MIIDyT...UijzrQQ==</dsig:X509Certificate>
                </dsig:X509Data>
            </dsig:KeyInfo>
            <xenc:CipherData>
                <xenc:CipherValue>NoO3acd...ggGQWA=</xenc:CipherValue>
            </xenc:CipherData>
        </xenc:EncryptedKey>
    </dsig:KeyInfo>
    <xenc:CipherData>
        <xenc:CipherValue>279oO2AZ6...okVnVDvI+KfG</xenc:CipherValue>
    </xenc:CipherData>
</xenc:EncryptedData>
```

Messages sent to the bank must obey the following rules:

- The EncryptedData/KeyInfo/EncryptedKey/KeyInfo/X509Data/X509Certificate element must be present, and it must contain the base64 encoded certificate used to perform the encryption. This is necessary since at one time the bank may have several valid encryption certificates.
- The actual business data must be encrypted using a symmetric cipher. It is not allowed to use asymmetric encryption to encrypt the business data directly. This is because encryption and decryption using asymmetric ciphers is much more CPU intensive than symmetric ciphers.

Clients should use standard software solutions (libraries) supporting the XML encryption specification rather than implementing it on their own.

Encryption Algorithms supported

The encryption algorithms supported by the bank are:

- Symmetric encryption: 3DES. http://www.w3.org/2001/04/xmlenc#tripledes-cbc
- Asymmetric encryption: RSA-v1.5. http://www.w3.org/2001/04/xmlenc#rsa-1_5

Other ciphers may be added later.

Our implementation of compression

Regarding compression, it should be applied at the Content element level. The Content element contains base64Binary coded data. Depending on the content of the Compression element, the base64Binary data can be in different formats:

1. If the Compression element contains ‘true’, the data is compressed. For decompression and decoding, the base64Binary data should be converted into binary data, which should be input to the decompression algorithm. The binary output of the decompression algorithm will be the legacy data or SEPA format XML.

2. If the Compression element contains 'false', the data is not compressed. For decoding, the base64Binary data should be base64 decoded. The binary output of this will be the legacy data or SEPA format XML.

When generating compressed data inside the Content element, the following procedure should be used:

The binary legacy data or the SEPA XML is used as input to the compression algorithm. The resulting binary data is base64 encoded and placed inside the Content element. Notice, that the input to the compression algorithm is the **binary** legacy or SEPA XML data, not a base64 encoded version of the data.

Regarding the compression algorithm, gzip (RFC1952) must be used [5]. The CompressionMethod element should contain the string "gzip".

Regarding signatures

Multiple business signatures

The Financial Messages specification allows for up to three enveloped business signatures in the ApplicationRequest element. However, if more than one business signature is to be used, care has to be taken.

If enveloped signatures based on the standard enveloped signature transform (http://www.w3.org/2000/09/xmldsig#enveloped-signature) are used, only the last signature will be verifiable. This is because the standard enveloped signature transform removes only the signature being verified, and leaves other signatures in the element still in place when the message digest is calculated. This effectively means that the message digest value of the signed element is changed every time a signature is added. Thus, the first signature will not be verifiable after the second signature has been added. If a third signature is added, the second signature will not be verifiable either.

The best solution to this problem probably is to not support multiple business signatures until a new version of the specification [1] and the corresponding schemas [4] has been made.

Supported algorithms

The following algorithms should be used in the digital signatures:

- For message digesting, SHA1 (http://www.w3.org/2000/09/xmldsig#sha1) should be used.
- For signing, RSA based on SHA1 (http://www.w3.org/2000/09/xmldsig#rsa-sha1) should be used.
- For canonicalization, exclusive canonicalization without comments (http://www.w3.org/2001/10/xml-exc-c14n#) should be used.

Requirements on the signatures

When reading the XMLDSIG standard, there is a lot of liberty in the way a signature can look. In order to be accepted by Danske Bank, every signature must contain the certificate used to create the signature.

For enveloped style signatures (business signatures), this means that a Signature/KeyInfo/X509Data/X509Certificate element must be found. The element must contain the signing certificate in base64 form.

For the WSSEC signatures (transport signatures), it means that a Security/BinarySecurityToken element must be found. The BinarySecurityToken must contain the signing certificate in base64 form, and it must be referenced from the Signature/KeyInfo/SecurityTokenReference element.

See the Appendix for examples of signatures satisfying these requirements.

Additionally, there is a time requirement on the transport signature. The transport signature contains a timestamp indicating at what time the signature was made. The signature will only be accepted if this timestamp is less than 60 minutes old. The interval in which the signature will be accepted is subject to change.

Future changes to the specification

Work has started to revise the financial web-service specification [1] and the corresponding schemas [4] to solve the problems described in this document. Care will be taken to make the changes backwards compatible whenever possible.

References

1. Security and Message Specification For Financial Messages Using Web Services. Nordea, OP-Pohjola Group, Sampo Bank. Version 1.05.
2. EDI Web Services, Danske Bank, [link]
3. PKI Service Description, Danske Bank, [link]
4. The WSDL and schema files for the financial services: BankCorporateFileService_20080616.wsdl, ApplicationRequest_20080918.xsd, ApplicationResponse_20080918.xsd.
5. RFC1952 - GZIP file format specification version 4.3. http://www.gzip.org/zlib/rfc-gzip.html

Appendix A: Example XML and SOAP

This appendix will go through the generation of an example signed and encrypted SOAP message. The XML shown here is delivered in a zip-file along with this document.

Generation of the ApplicationRequest

First a business content and ApplicationRequest element must be generated. This corresponds to step **Error! Reference source not found.** on the list on page **Error! Bookmark not defined.**. An example ApplicationRequest is found below:

```
<bxd:ApplicationRequest xmlns:bxd="http://bxd.fi/xmldata/">
	<bxd:CustomerId>ABC123</bxd:CustomerId>
	<bxd:Command>UploadFile</bxd:Command>
	<bxd:Timestamp>2009-12-17T09:30:47Z</bxd:Timestamp>
	<bxd:Environment>TEST</bxd:Environment>
	<bxd:Encryption>true</bxd:Encryption>
	<bxd:Compression>true</bxd:Compression>
	<bxd:CompressionMethod>gzip</bxd:CompressionMethod>
	<bxd:SoftwareId>CustomerSoftwareId</bxd:SoftwareId>
	<bxd:FileType>pain.001.001.02</bxd:FileType>
	<bxd:Content>UjBsR09EbGhjZ0dTQUxNQUFBUUNBRU1tQ1p0dU1GUXhEUzhi</bxd:Content>
</bxd:ApplicationRequest>
```

Note that the business content and the attributes may not make sense in this example.

Signing the ApplicationRequest (business signature)

A business signature is added to the ApplicationRequest. The signature is of the enveloped type. For this the signing certificate of the customer is used. This corresponds to step 1 on the list on page 4:

```
<bxd:ApplicationRequest xmlns:dpstate="http://danskebank.dk/AGENA/SEPA/dpstate"
xmlns:sig="http://danskebank.dk/AGENA/SEPA/SigningService" xmlns:bxd="http://bxd.fi/xmldata/">
	<bxd:CustomerId>ABC123</bxd:CustomerId>
	<bxd:Command>UploadFile</bxd:Command>
	<bxd:Timestamp>2009-12-17T09:30:47Z</bxd:Timestamp>
	<bxd:Environment>TEST</bxd:Environment>
	<bxd:Encryption>true</bxd:Encryption>
	<bxd:Compression>true</bxd:Compression>
	<bxd:CompressionMethod>gzip</bxd:CompressionMethod>
	<bxd:SoftwareId>CustomerSoftwareId</bxd:SoftwareId>
	<bxd:FileType>pain.001.001.02</bxd:FileType>
	<bxd:Content>UjBsR09EbGhjZ0dTQUxNQUFBUUNBRU1tQ1p0dU1GUXhEUzhi</bxd:Content>
<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="">
    <Transforms>
      <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
      <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
    </Transforms>
    <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
    <DigestValue>A3OH3jwrbbcYxX3Nv5+eFyEuTww=</DigestValue>
  </Reference>
```

```
</SignedInfo>

<SignatureValue>DneTCqbKEbQUE/TpaMlLpOKJmB0asSfWr5O53M1HVwwE4JOqTJ3XNeZtB21C/yoYKP
RKy3Iz8du+ObuhvuD0TiCASJZZw7FxkcpCm5+mAJDjCeB1KD+zPGjtGCo/B5Fcu6FfMcBT1/Q3oWMUuUP
GxfOKxWg1Lf1hUPpmjDtNF6Y=</SignatureValue><KeyInfo><X509Data><X509Certificate>MIIDIjCCAougA
wIBAgIEL4j5xzANBgkqhkiG9w0BAQUFADBpMQswCQYDVQQGEwJESzEQMA4GA1UEBxMHRGVubWFy
azEaMBgGA1UEChMRRGFuc2tlIEJhbmsgR3JvdXAxEDAOBgNVBAsTB0FwcGxTSW4xGjAYBgNVBAMUE
UZpbm5pc2hfU0VQQV90ZXN0MB4XDTA5MTAyNjEyMDMzMVoXDTEyMDIwMzEyMDMzMVowaTELMAkG
A1UEBhMCREsxEDAOBgNVBAcTB0Rlbm1hcmsxGjAYBgNVBAoTEURhbnNrZSBCYW5rIEdyb3VwMRAw
DgYDVQQLEwdBcHBsU0luMRowGAYDVQQDFBFGaW5uaXNoX1NFUEFfdGVzdDCBnzANBgkqhkiG9w0B
AQEFAAOBjQAwgYkCgYEAyKLcdWd6TvZ+AF3zAabivvf+s+QoTnxQ4+DOl8mDBMpo7Ka41i59u79qyHa24
3HuQgW4eCMist+CUfY/Ynu6DgsP/NFduuV2VZNMYdQIKwJMaytQ1zhYufMeiw8YdgGk8qVjZJOmwbCjnDI
1+da0CvEeEs1ZUZtVK9nRDDsnG9cCAwEAAaOB1jCB0zAMBgNVHRMEBTADAQH/MB0GA1UdDgQWBB
Ru/JyaNnOeCQC6jNdNZI/qfLNoYTCBlgYDVR0jBIGOMIGLgBRu/JyaNnOeCQC6jNdNZI/qfLNoYaFtpGswaT
ELMAkGA1UEBhMCREsxEDAOBgNVBAcTB0Rlbm1hcmsxGjAYBgNVBAoTEURhbnNrZSBCYW5rIEdyb3V
wMRAwDgYDVQQLEwdBcHBsU0luMRowGAYDVQQDFBFGaW5uaXNoX1NFUEFfdGVzdIIEL4j5xzALBgN
VHQ8EBAMCArwwDQYJKoZIhvcNAQEFBQADgYEAq6rHIg41XYLMjOLZ33toWKLos3uWJucHK0F6TKAXS
Np7Qo34hx04P0MidOXuj18mYZkU/usVHn/L6rkWSfJretpO0xWTzw50XSm83JUQIYeQyMiU/w7ByJSNlTa8
G1i4nKWjZ/kcl2Ur0oDUTTf/X8mIC24EkDCI9RQjGEYdUvE=</X509Certificate><X509IssuerSerial><X509Is
suerName>CN=Finnish_SEPA_test, OU=ApplSIn, O=Danske Bank Group, L=Denmark,
C=DK</X509IssuerName><X509SerialNumber>797505991</X509SerialNumber></X509IssuerSerial></X50
9Data></KeyInfo></Signature></bxd:ApplicationRequest>
```

Encrypting the ApplicationRequest

The signed ApplicationRequest element is encrypted using XML encryption. For this, the encryption certificate of the bank is used. This corresponds to step 7 on the list on page 4:

```
<xenc:EncryptedData xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
Type="http://www.w3.org/2001/04/xmlenc#Element">
      <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#tripledes-cbc"/>
      <dsig:KeyInfo xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
            <xenc:EncryptedKey>
                  <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5"/>
                  <dsig:KeyInfo>
                        <dsig:X509Data>

      <dsig:X509Certificate>MIIDyTCCArGgAwIBAgIKNDQ0NDQzMDAwMjANBgkqhkiG9w0BAQsFADCBxD
EQMA4GA1UEAxMHREJHUk9PVDELMAkGA1UEBhMCREsxEzARBgNVBAcTCkNvcGVuaGFnZW4xEDA
OBgNVBAgTB0Rlbm1hcmsxGjAYBgNVBAoTEURhbnNrZSBCYW5rIEdyb3VwMRowGAYDVQQLExFEYW5
za2UgQmFuayBHcm91cDEYMBYGA1UEBRMPNjExMjYyMjgxMTMwMDAxMQkwBwYDVQQEEwAxCTAHB
gNVBCoTADEJMAcGA1UEDBMAMQkwBwYDVQQREwAwHhcNMDkxMDMwMTIwMDAwWhcNMTExMDM
wMTIwMDAwWjCBxTERMA8GA1UEAxMIREJHQ1JZUFQxCzAJBgNVBAYTAkRLMRMwEQYDVQQHEwpD
b3BlbmhhZ2VuMRAwDgYDVQQIEwdEZW5tYXJrMRowGAYDVQQKExFEYW5za2UgQmFuayBHcm91cDE
aMBgGA1UECxMRRGFuc2tlIEJhbmsgR3JvdXAxGDAWBgNVBAUTDzYxMTI2MjI4NDQzMDAwMjEJMAcG
A1UEBBMAMQkwBwYDVQQqEwAxCTAHBgNVBAwTADEJMAcGA1UEERMAMIGdMA0GCSqGSIb3DQEB
AQUAA4GLADCBhwKBgQCpae2tNbCmPRni74t4pXpzUwsUTRBeazPpchZHjn+HBpfQAH66kCzOcxLHOrL
71CFYIj6B4WrqXZx4SUldQpKu67vSh5ZsLyyXPZniCo+DtX2zOYGEigzq6hIylR5XKnXkZfJSxo/TgGX3rkLxF
6cbaP6mxN7LcCv9dUj+eRnngQIBA6NAMD4wEQYDVR0OBAoECERCR0NSWVBUMBkGA1UdIwQSMBC
ADsLGxMP29sL29/X39cXEMA4GA1UdDwEB/wQEAwIEMDANBgkqhkiG9w0BAQsFAAOCAQEAD8+BpnM/
K10m5iKO4BTcvGi77oSp0sul5utP6ZFEveuqjlMx4xxh+1SVRSMXQttd49/08BIUOaYrIjvP+dJT5J6D+UeFRSj
E6mevjRE1qqIpCsHoCWVE9WkiuF3QLaouD9PH5FPsk53QVdd4Yp+vIKDVYQbpc12A8wAEu8hkngqxN2/E
pd8u/3LQgsTmY6mn5ek2EFNI6o2dQa503DpzjoXGURIFOoAaWB7ZUJhe7nxG1Xjn1OnttreoP6twCTrG8IoL
fGKDkLVQ3G+7vAqXgB+JwFLUoULRaw0eISPgE7nMkz0la0lu+aJlaHRJZq/F9eMkDXFqRSMGHHNUijzrQ
```

```
Q==</dsig:X509Certificate>
                        </dsig:X509Data>
                    </dsig:KeyInfo>
                    <xenc:CipherData>

        <xenc:CipherValue>NoO3acdWBuH0nduRMdLZtOhN645erE7JjABu4waNg7GYbEvxrj6m1mdhpj32BZ4
eI+cp5r5UIhEYxt4sHIj8tMxJqcogxrJEiD6Ig/zzc1mDV61fQidJWgcIkWOUA2deb1Wz4OeS1oMe1F6M1wmvS
PWNpSZbEmEG/4zv9ggGQWA=</xenc:CipherValue>
                    </xenc:CipherData>
                </xenc:EncryptedKey>
        </dsig:KeyInfo>
        <xenc:CipherData>

        <xenc:CipherValue>279oO2AZ63oVYSFjMT+mW/YSUBbWJmNiGeg1OlendoFEhhpHYcpTbGXG7kO/
SXtewRr1aZp19L4/1qqNJW0BQeXluwVd7Qz9odekyzIGN4gBPuvp3ZH46yPT7kPfc3umwUIVZ5R3GPhEYG
m4AUhK8Ygju/FZPZy+iU84bUG5iSV8ORYtEck+UQpGbbfCyYPX5U96zkV0hj6HL+Cwelg2Q9Gp6chu0IpBL
/WRF0n1Kn1J/Z7LkVbKB9cROm3B2v77vJkjrhq4FwwT0wIkZZbq4yX5w7qeMlgvAUZ2OdPDwhEGSj+TM6ai
dGYddNDhQ0SDfbLKETWg8Lh6vuRhFk925uVUoiLJgw1MBf22t3H1Tz5z30leOJHdsEJt6XfHP0P2WUpSw1
EAAqFq4RBl4hOgrzoRqVfZM9u1MPML8bYhLDI4O8cKCDvo4iwd1Mv0niu0veD7Asyo43nA4hVh/BsUskEZe
/SaStOo+WXzeL7cdJmeW9AkTkpsiidF8iEERY2iANdJpDNGS7B7jQ3uDZiVOVYBoCG2YrecMSrlcpfxzNap
DJKUshyxRigfPSGtN9YkMZsgc/whhkIVN1VnV2A541butR9SG3m8ITbm/fgcvt36DPXcmlV4cyHWTiX2lv9LF
+IAmG+WVBnu6d4LMCXWMa4gzTnrgoXvXpAKNI51UmH6jiYOn1Or8Ls5YOzDeZRcVdgRFT1K2sHrZ4W
WYeSUQfsxfZo9qzngTQ4BvAomPrL6Xh4IYAPMQxJZio1yo8/C7lyZiwv/XAeNuZ4/tG7tF3vKi/B6mjHm9iqN6
OtAuQ8iHCZt8ayBMhbgHb6vORxv77A1+OKxCkZQfcxyHesCnjj0ONEvnoJi3Z/2TjupjAMYNgJ+O20Wt4Q8J
hWGfuCOl13H4oy2Z/Wy+lsKqRyJ4ihlXrz0T5LsFwncdE7ho3MhyT2C3NAkrosgpVRIyB99DLJpWhWR4s3tx
pfc47DRihny3gq7iepcl60ANTL1tBePk0PRD/y/tf59aZjhWSMBWQpvwJaVAHJgpBGtCUYkGXLAkUICZ+cY4
osHeTyIivNshX4F/O8AvDjXvlQUKYss+Bz7i+raQMFVvQNp0Lr3BLc3FPAQCgZDFB9xuaVOu245hxtWakaq5
ZC/jg09KkA+Vwj13nkK0dY0YE3IDIxTIEVvuN61uIRJEBwh0JWU96fPycVfmhdvhlQF6lcWzvbsCKhAMZu83v
0Wb941LsssZzIKHa7Qj/2U4maTHgI03iZhdETQtVyvrd1FS+PK86V62jDhy53VSF1bqP9eH/16bqLXSSyvyCE
lLEO4LhZCbS33fUpKM0sej5gp7AQhFTA4kVrv0K6D9/7R0gR72GZF9/+HXwKq4VHg+s8380xen3F5O9ZWe
2nX9y3yIeH2Us/PoKjyYjKwV3DWYUi0Jtvw93uobzxAMLFhqFPX8+bHFbnyN1yqEjJFB5BZkX8Xa1G8Sp6ig
kGtXZltX0IaxBzrWvbpyuZW2LU/1igkQkA20cuLFPrEax1gZzVZyBP9zEKdqCGRUGc+HFuirgtG0ncR+4KKbf
VtN7r3xkVNfaihxXVeF45T1/TBE+mdjRd5kMICLEY29uT3IFZbZsPUoSSOEa6wNLYtQTSt7DSJiX4abhxUDx
QeHgF0iLbKJw99h/OjopfOCPyTIIxglayLlT6TrDsiwHnxuwbRkeRmh7gXS79qBZIyBpi1g6ZMTB4FDfP0+1QI
FzT5wmoOlzEZ1SgOT8JgGd5c/xeLZ3NqLvlO4ZxefO6KV+l8IadOipV5utV3mOhgAI9xDzOnqYb7UZy9OcZR
p9hWX6KxzokyomQGwUqjAio/k3wqpwGBuCNgLd+piWUBkot2RRlQeVFeYjaa0hTsNmAmpbvgoj2HhERFfg
RQk6/IU4XbBSHmSX3Va+NK56uDEBqPhEg57G6Ok1ozqXQPRJayob87GZaEzEgUDW12ieXYIjXXZIz92Ie
w04/2bhXc8dtQEyxOcNkFugfWEd7EeNyFULSAljdsQfXzaouOKsWL3XzQ71+sIIJkFU0KfKitxlFaukmwUfLx
W/ZwGglQcOchs7E5Tu5ZGiI+ih5wOm1py+M+cgBk3wyFGlcjD9IJZ5oM9IQAjJn78NwTQ8LG8VrFzAe0ObM
7/8sWZbr+c1lMmDvAI3qAARray6rNTMamqgu0cc1hcdyEyn3CmZjy1x9VWbbNXdqk/lXu6c6Kgda8rMrVyV9
B9KyWutw9ht9u8bK9ny31zDnIlzEm3sVaY4XzYbXqvX7ZpVvcC607KiKE7m9HWcuPXsqA7eFPbPrZl6Jue9
F8OkZIIH4vsOY+FVo0ju9JufJSoAA+eHctUZ4c4biYPJIxa82THQT9XkE47X8eWF2ucY6upGJ8La8H27wavM
cuKJNYxMnJKKpdpMlmBMgTZuyvyAzfoyO/CiaSggm3wanYl586FNe9zXnDqtsR+4RTNrBNd28fqwyneLoeEi
3Om/dOB3WLGQHkCyburz3HIn8ZPOr2dDMuRQzs8WBMr2KUohdQ6jXonYEEhKFeJUx2HFamPgjKl+Mcp
+6a3as3L8I10JihBdgrM6vKe7fHc6dPhzNVz3ep8AYll/6sr01v5uAK42Oy9Q9TNxZsL47zecMd4CwRJdHQMS
4DKfiHZBOLE3Cep7Ca5bKhYIkTK+nkCw5twkMSJdqzEqaVu0WPFwLQv4ysibrOPtN5E+alzWYQPnGB2kcq
3vH0mfqnxcP1aLxJBtR0u5MPWehfCem9i/lO/few4XnbplMMcf8YjzH1brpp+P667c9O9qVDAsj9KFVIVZlpv2Z
0DU3pS49oZkU+vVQWOahFqXYYNCbW5kJhXwAjHFjHTjQnRQPcGqkjJ4fEiLvYOQ5yE1hW/LwNdua8NcC
nzuDg6P7SpOmIZ8p3GENITvPRUxZvTcDK5efdsoVb8FXVuyftVxwUXmnZ9FCmdjwr57tt3iWb8S064LHJYO
6X4HvcfR4VtPBVjVO5qCE57hdy+XYOvyIT9bhCWLvdll354ni6L6lgxx0Y/fiBNZlmT7FeVl91PIw6lW0qvzHR2
Pkz+abzvqU5pNlTyJRt20+5hezvwnL//VTLyZmBENtiG6FL4yi1NL+PWcAvVw86kQBlkTvlSuMvaZwQ1kRIrw
1gu/uEGcU5COacYwvKTuL/GMh10g5e0GYcYEdHCtjTE5CCWCOYt/tTLgNkeV2Q7QdY5Xk9Zsc3zS1Krwr4
Dbs9Hk4WCWHhoM4ZUn35F08bs2p2N0bdeag1BQ1Wn/tHdJ3l2ZbpAqJGkZYj4BcEW+3oI9VmaQOI+eIGN
Odv4IydlOtoz8bL50LqxaCMGIAu5BCQUHRZKOm95SvcLZocG+UCPKnrfVXkCPz20nhwMXh6ZPK/vBCoE
Yf2ICSMJfpvEQRFKeM/j4SYSjm/C0C3QneeCtOjxmCrwqinKqrzkmMpNTYXZjUua3WtGjq4H4nvwVHKGSL/
ZSs6OifBO3YBFEYcHqC58ZtYGBafFH+aaTdNQqMpd8X0U37rgtqfeg06NSt+QT9Ezj35SL7yjs+4+rvX2Rpyk
FD3s4S2I+3aVBxqY5vTTlKP7iDZH1ADKjn7rSxeiuJYhxqJnNzQyMMsgGmYbW3rnjlsTqKU/XD6twOcEb17K
SII9+2Jp3MGjcLPdYdWKsiobdamdU9Cj6mIhG5d1DThNajwdTXjVzP7aPsmjJtNaMPFRr2l62wSiUFtLSlLJm
```

pF9W98umQyWpfWXJ9yOx1zCR674X4JEiT6waP1cGV7ZGklmYbr/Q3XSfjJAMKZvqOpnVLpzZwOGVq4oQ
W9DRFEL6wy3Ax2yHywykdqUeA2TFY8eo47X3WAO9IyX9ZPAS2vaJLB/XHlavOHc3p3OWT3IGpq6ngsO2
1FokVnVDvI+KfG</xenc:CipherValue>
</xenc:CipherData>
</xenc:EncryptedData>

This step should be performed by a piece of standard software.

Generation of the SOAP message

The next step is to generate a SOAP message (including the RequestHeader), base64 encode the XML from the previous step and insert it in a the ApplicationRequest element of the SOAP. This corresponds to steps 8 to 11 on the list on page 4:

```
<soapenv:Envelope xmlns:sign="http://danskebank.dk/AGENA/SEPA/SigningService"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:cor="http://bxd.fi/CorporateFileService"
xmlns:mod="http://model.bxd.fi" xmlns:dpstate="http://danskebank.dk/AGENA/SEPA/dpstate">
    <soapenv:Header/>
    <soapenv:Body>
        <cor:deleteFilein>
     <mod:RequestHeader>
       <mod:SenderId>ABC123</mod:SenderId>
       <mod:RequestId>33</mod:RequestId>
       <mod:Timestamp>2009-12-17T09:30:47Z</mod:Timestamp>
       <mod:Language>EN</mod:Language>
       <mod:UserAgent>CustomerSoftwareId</mod:UserAgent>
       <mod:ReceiverId>DABAFIHH</mod:ReceiverId>
     </mod:RequestHeader>

    <mod:ApplicationRequest>PHhlbmM6RW5jcnlwdGVkRGF0YSB4bWxuczp4ZW5jPSJodHRwOi8vd3d3
LnczLm9yZy8yMDAxLzA0L3htbGVuYyMiIFR5cGU9Imh0dHA6Ly93d3cudzMub3JnLzIwMDEvMDQveG1sZ
W5jI0VsZW1lbnQiPjx4ZW5jOkVuY3J5cHRpb25NZXRob2QgQWxnb3JpdGhtPSJodHRwOi8vd3d3LnczLm9y
Zy8yMDAxLzA0L3htbGVuYyN0cmlwbGVkZXMtY2JjIi8+PGRzaWc6S2V5SW5mbyB4bWxuczpkc2lnPSJodH
RwOi8vd3d3LnczLm9yZy8yMDAwLzA5L3htbGRzaWcjIj48eGVuYzpFbmNyeXB0ZWRLZXk+PHhlbmM6RW
5jcnlwdGlvbk1ldGhvZCBBbGdvcml0aG09Imh0dHA6Ly93d3cudzMub3JnLzIwMDEvMDQveG1sZW5jI3JzYS
0xXzUiLz48ZHNpZzpLZXlJbmZvPjxkc2lnOlg1MDlEYXRhPjxkc2lnOlg1MDlDZXJ0aWZpY2F0ZT5NSUlEeVR
DQ0FyR2dBd0lCQWdJS05EUTBORFF6TURBd01qQU5CZ2txaGtpRzl3MEJBUXNGQURDQnhERVFNQTR
HQTFVRUF4TUhSRUpIVWs5UFZERUxNQWtHQTFVRUJoTUNSRXN4RXpBUkJnTlZCQWNUQ2tOdmNHV
nVhR0ZuWlc0eEVEQU9CZ05WQkFnVEIwUmxibTFoY21zeEdqQVlCZ05WQkFvVEVUmhibk5yWlNCQ1lX
NXJJRWR5YjNWd01Sb3dHQVlEVlFRTEV4RkVZVzV6YTJVZ1FtRnVheUJIY205MWNERVlNQllHQTFVRUJ
STVBOakV4TWpZeU1qZ3hNVE13TURBeE1Ra3dCd1lEVlFRRUV3QXhDVEFIQmdOVkJDb1RBREVKTUFj
R0ExVUVEQk1BTVFrd0J3WURWUVFSRXdBd0hoY05NRGt4TURNd01USXdNREF3V2hjTk1URXhNRE13
TVRJd01EQXdXakNCeFRFUk1BOEdBMVVFQXhNSVJFSkhRMUpaVUZReEN6QUpCZ05WQkFZVEFrUkx
NUk13RVFZRFZRUUhFd3BEYjNCbGJtaGhaMlZ1TVJBd0RnWURWUVFJRXdkRVpXNXRZWEpyTVJvd0d
BWURWUVFLRXhGRVlXNXphMlVnUW1GdWF5QkhjbTkxY0RFYU1CZ0dBMVVFQ3hNUlJHRnVjMnRsSU
VKaGJtc2dSM0p2ZFhBeEdEQVdCZ05WQkFVVER6WXhNVEkyTWpJNE5EUXpNREF3TWpFSk1BY0dBM
VVFQkJNQU1Ra3dCd1lEVlFRcUV3QXhDVEFIQmdOVkJBd1RBREVKTUFjR0ExVUVFUk1BTUlHZE1BME
dDU3FHU0liM0RRRUJBUVVBQTRHTEFEQ0Job0tCZ1FDcGFlMnROYkNtUFJuaTc0dDRwWHB6VXdzVVR
SQmVhelBwY2haSGpuK0hCcGZRQUg2NmtDek9jeExIT3JMNzFDRllJajZCNFdycVhaeDRTVWxkUXBLdTY
3dlNoNVpzTHl5WFBabmlDbytEdFgyek9ZR0VpZ3pxNmhJeWxSNVhLblhrWmZKU3hvL1RnR1gzcmtMeEY2
Y2JhUDZteE43TGNDdjlkVWorZVJubmdRSUJBNk5BTUQ0d0VRWURWUjBPQkFvRUNFUkNSME5TV1ZC
VU1Ca0dBMVVkSXdRU01CQ0FEc0xHeE1QMjlzTDI5L1gzOWNYRU1BNEdBMVVkRHdFQi93UUVBd0lFT
URBTkJna3Foa2lHOXcwQkFRc0ZBQU9DQVFFQUQ4K0Jwbk0vSzEwbTVpS080QlRjdkdpNzdvU3Awc3Vs
NXV0UDZaRkV2ZXVxamxNeDR4eGgrMVNWUlNNWFF0dGQ0OS8wOEJJVU9hWXJJanZQK2RKVDVKNk
QrVWVGUlNqRTZtZXZqUkUxcXFJcENzSG9DV1ZFOVdraXVGM1FMYW91RDlQSDVGUHNrNTNRVmRkN
FlwK3ZJS0RWWVFicGMxMkE4d0FFdThoa25ncXhOMi9FcGQ4dS8zTFFnc1RtWTZtbjVlazJFRk5JNm8yZF
```

FhNTAzRHB6am9YR1VSSUZPb0FhV0I3WlVKaGU3bnhHMVhqbjFPbnR0cmVvUDZ0d0NUckc4SW9MZkdL
RGtMVlEzRys3dkFxWGdCK0p3RkxVb1VMUmF3MGVJU1BnRTduTWt6MGxhMGx1K2FKbGFIUkpacS9GO
WVNa0RYRnFSU01HSEhOVWlqenJRUT09PC9kc2lnOlg1MDlDZXJ0aWZpY2F0ZT48L2RzaWc6WDUwOU
RhdGE+PC9kc2lnOktleUluZm8+PHhlbmM6Q2lwaGVyRGF0YT48eGVuYzpDaXBoZXJWYWx1ZT5Ob08zY
WNkV0J1SDBuZHVSTWRMWnRPaE42NDVlckU3SmpBQnU0d2FOZzdHWWJFdnhyajZtMW1kaHBqMzJC
WjRlSStjcDVyNVVJaEVZeHQ0c0hJajh0TXhKcWNvZ3hySkVpRDZJZy96emMxbURWNjFmUWlkSldnY0lrV0
9VQTJkZWIxV3o0T2VTMW9NZTFGNk0xd212U1BXTnBTWmJFbUVHLzR6djlnZ0dRV0E9PC94ZW5jOkNpc
GhlclZhbHVlPjwveGVuYzpDaXBoZXJEYXRhPjwveGVuYzpFbmNyeXB0ZWRLZXk+PC9kc2lnOktleUluZm8+
PHhlbmM6Q2lwaGVyRGF0YT48eGVuYzpDaXBoZXJWYWx1ZT4yNzlvTzJBWjYzb1ZZU0ZqTVQrbVcvWVN
VQmJXSm1OaUdlZzFPbGVuZG9GRWhocEhZY3BUYkdYRzdrTy9TWHRld1JyMWFacDE5TDQvMXFxTkpX
MEJRZVhsdXdWZDdRejlvZGVreXpJR040Z0JQdXZwM1pINDZ5UFQ3a1BmYzN1bXdVSVZaNVIzR1BoRVl
HbTRBVWhLOFlnanUvRlpQWnkraVU4NGJVRzVpU1Y4T1JZdEVjaytVUXBHYmJmQ3lZUFg1VTk2emtWM
GhqNkhMK0N3ZWxnMlE5R3A2Y2h1MElwQkwvV1JGMG4xS24xSi9aN0xrVmJLQjljUk9tM0Iydjc3dkpranJoc
TRGd3dUMHdJa1paYnE0eVg1dzdxZU1sZ3ZBVVoyT2RQRHdoRUdTaitUTTZhaWRHWWRkTkRoUTBTRG
ZiTEtFVFdnOExoNnZ1UmhGazkyNXVWVW9pTEpndzFNQmYyMnQzSDFUejV6MzBsZU9KSGRzRUp0Nlh
mSFAwUDJXVXBTdzFFQUFxRnE0UkJsNGhPZ3J6b1JxVmZaTTl1MU1QTUw4YlloTERJNE84Y0tDRHZvN
Gl3ZDFNdjBuaXUwdmVEN0FzeW80M25BNGhWaC9Cc1Vza0VaZS9TYVN0T28rV1h6ZUw3Y2RKbWVXO
UFrVGtwc2lpZEY4aUVFUlkyaUFOZEpwRE5HUzdCN2pRM3VEWmlWT1ZZQm9DRzJZcmVjTVNybGNwZn
h6TmFwREpLVXNoeXhSaWdmUFNHdE45WWtNWnNnYy93aGhrSVZOMVZuVjJBNTQxYnV0UjlTRzNtOEl
UYm0vZmdjdnQzNkRQWGNtbFY0Y3lIV1RpWDJsdjlMRitJQW1HK1dWQm51NmQ0TE1DWFdNYTRnelRuc
mdvWHZYcEFLTkk1MVVtSDZqaVlPbjFPcjhMczVZT3pEZVpSY1ZkZ1JGVDFLMnNIclo0V1dZZVNVUWZze
GZabzlxem5nVFE0QnZBb21Qckw2WGg0SVlBUE1ReEpaaW8xeW84L0M3bHlaaXd2L1hBZU51WjQvdEc3
dEYzdktpL0I2bWpIbTlpcU42T3RBdVE4aUhDWnQ4YXlCTWhiZ0hiNnZPUnh2NzdBMStPS3hDa1pRZmN4e
Uhlc0NuamowT05Fdm5vSmkzWi8yVGp1cGpBTVlOZ0orTzIwV3Q0UThKaFdHZnVDT2wxM0g0b3kyWi9Xe
Stsc0txUnlKNGlobFhyejBUNUxzRnduY2RFN2hvM01oeVQyQzNOQWtyb3NncFZSSXlCOTlETEpwV2hXUjR
zM3R4cGZjNDdEUmlobnkzZ3E3aWVwY2w2MEFOVEwxdEJlUGswUFJEL3kvdGY1OWFaamhXU01CV1Fw
dndKYVZBSEpncEJHdENVWWtHWExBa1VJQ1orY1k0b3NIZVR5SWl2TnNoWDRGL084QXZEalh2bFFVS1
lzcytCejdpK3JhUU1GVnZRTnAwTHIzQkxjM0ZQQVFDZ1pERkI5eHVhVk91MjQ1aHh0V2FrYXE1WkMvamc
wOUtrQStWd2oxM25rSzBkWTBZRTNJREl4VElFVnZ1TjYxdUlSSkVCd2gwSldVOTZmUHljVmZtaGR2aGxR
RjZsY1d6dmJzQ0toQU1adTgzdjBXYjk0MUxzc3NaeklLSGE3UWovMlU0bWFUSGdJMDNpWmhkRVRRdFZ
5dnJkMUZTK1BLODZWNjJqRGh5NTNWU0YxYnFQOWVILzE2YnFMWFNTeXZ5Q0VsTEVPNExoWkNiUz
MzZlVwS00wc2VqNWdwN0FRaEZUQTRrVnJ2MEs2RDkvN1IwZ1I3MkdaRjkvK0hYd0txNFZIZytzODM4MH
hlbjNGNU85WldlMm5YOXkzeUllSDJVcy9Qb0tqeVlqS3dWM0RXWVVpMEp0dnc5M3VvYnp4QU1MRmhxRl
BYOCtiSEZibnlOMXlxRWpKRkI1QlprWDhYYTFHOFNwNmlna0d0WFpsdFgwSWF4QnpyV3ZicHl1WlcyTFU
vMWlna1FrQTIwY3VMRlByRWF4MWdaelZaeUJQOXpFS2RxQ0dSVUdjK0hGdWlyZ3RHMG5jUis0S0tiZlZ0
TjdyM3hrVk5mYWloeFhWZUY0NVQxL1RCRSttZGpSZDVrTUlDTEVZMjl1VDNJRlpiWnNQVW9TU09FYTZ3
TkxZdFFUU3Q3RFNKaVg0YWJoeFVEeFFlSGdGMGlMYktdzk5aC9Pam9wZk9DUHlUSUl4Z2xheUxsVDZ
UckRzaXdIbnh1d2JSa2VSbWg3Z1hTNzlxQlpJeUJwaTFnNlpNVEI0RkRmUDArMVFJRnpUNXdtb09sekVaM
VNnT1Q4SmdHZDVjL3hlTFozTnFMdmxPNFp4ZWZPNktWK2w4SWFkT2lwVjV1dFYzbU9oZ0FJOXhEek9u
cVliN1VaeTlPY1pScDloV1g2S3h6b2t5b21RR3dVcWpBaW8vazN3cXB3R0J1Q05nTGQrcGlXVUJrb3QyUlJs
UWVWRmVZamFhMGhUc05tQW1wYnZnb2oySGhFUkZmZ1JRazYvSVU0WGJCU0htU1gzVmErTks1NnVE
RUJxUGhFZzU3RzZPazFvenFYUVBSSmF5b2I4N0daYUV6RWdVRFcxMmllWFlJalhYWkl6OTJJZXcwNC8y
YmhYYzhkdFFFeXhPY05rRnVnZldFZDdFZU55RlVMU0FsamRzUWZYemFvdU9Lc1dMM1h6UTcxK3NJSU
prRlUwS2ZLaXR4bEZhdWttd1VmTHhXL1p3R2dsUWNPY2hzN0U1VHU1WkdpSStpaDV3T20xcHkrTStjZ0Jr
M3d5RkdsY2pEOUlKWjVvTTlJUUFqSm43OE53VFE4TEc4VnJGekFlME9iTTcvOHNXWmJyK2MxbE1tRHZ
BSTNxQUFScmF5NnJOVE1hbXFndTBjYzFoY2R5RXluM0NtWmp5MXg5VldiYk5YZHFrL2xYdTZjNktnZGE4
ck1yVnlWOUI5S3lXdXR3OWh0OXU4Yks5bnkzMXpEbklsekVtM3NWYVk0WHpZYlhxdlg3WnBWdmNDNjA3
S2lLRTdtOUhXY3VQWHNxQTdlRlBiUHJabDZKdWU5RjhPa1pJSUg0dnNPWStGVm8wanU5SnVmSlNvQU
ErZUhjdFVaNGM0YmlZUEpJeGE4MlRIUVQ5WGtFNDdYOGVXRjJ1Y1k2dXBHSjhMYThIMjd3YXZNY3VLS
k5ZeE1uSktLcGRwTWxtQk1nVFp1eXZ5QXpmb3lPL0NpYVNnZ20zd2FuWWw1ODZGTmU5elhuRHF0c1Ir
NFJUTnJCTmQyOGZxd3luZUxvZUVpM09tL2RPQjNXTEdRSGtDeWJ1cnozSEluOFpQT3IyZERNdVJRenM
4V0JNcjJLVW9oZFE2alhvbllFRWhLRmVKVXgySEZhbVBnaktsK01jcCs2YTNhczNMOEkxMEppaEJkZ3JNN
nZLZTdmSGM2ZFBoek5WejNlcDhBWWxsLzZzcjAxdjV1QUs0Mk95OVE5VE54WnNMNDd6ZWNNZDRDd1
JKZEhRTVM0REtmaUhaQk9MRTNDZXA3Q2E1YktoWUlrVEsrbmtDdzV0d2tNU0pkcXpFcWFWdTBXUEZ3
TFF2NHlzaWJyT1B0TjVFK2FseldZUVBuR0Iya2NxM3ZIMG1mcW54Y1AxYUx4SkJ0UjB1NU1QV2VoZkNlb
TlpL2xPL2ZldzRYbmJwbE1NY2Y4WWp6SDFicnBwK1A2NjdjOU85cVZEQXNqOUtGVklWWmxwdjJaMERV
M3BTNDlvWmtVK3ZWUVdPYWhGcVhZWU5DYlc1a0poWHdBakhGakhUalFuUlFQY0dxa2pKNGZFaUx2W
U9RNXlFMWhXL0x3TmR1YThOY0NuenVEZzZQN1NwT21JWjhwM0dFTklUdlBSVXhadlRjREs1ZWZkc29W

YjhGWFZ1eWZ0Vnh3VVhtblo5RkNtZGp3cjU3dHQzaVdiOFMwNjRMSEpZTzZYNEh2Y2ZSNFZ0UEJWalZP
NXFDRTU3aGR5K1hZT3Z5SVQ5YmhDV0x2ZGxJMzU0bmk2TDZsZ3h4MFkvZmlCTlpsbVQ3RmVWbDkxU
El3NklXMHF2ekhSMlBreithYnp2cVU1cE5sVHlKUnQyMCs1aGV6dnduTC8vVlRMeVptQkVOdGlHNkZMNHl
pMU5MK1BXY0F2Vnc4NmtRQmxrVHZsU3VNdmFad1Exa1JJcncxZ3UvdUVHY1U1Q09hY1l3dktUdUwvR0
1oMTBnNWUwR1ljWUVkSEN0alRFNUNDV0NPWXQvdFRMZ05rZVYyUTdRZFk1WGs5WnNjM3pTMUtyd3l
0RGJzOUhrNFdDV0hob000WlVuMzVGMDhiczJwMk4wYmRlYWcxQlExV24vdEhkSjNsMlpicEFxSkdrWllqN
EJjRVcrM29JOVZtYVFPSStlSUdOT2R2NEl5ZGxPdG96OGJMNTBMcXhhQ01HSUF1NUJDUVVIUlpLT205
NVN2Y0xab2NHK1VDUEtucmZWWGtDUHoyMG5od01YaDZaUEsvdkJDb0VZZjJJQ1NNSmZwdkVRUkZLZ
U0vajRTWVNqbS9DMEMzUW5lZUN0T2p4bUNyd3FpbktxcnprbU1wTlRZWFpqVXVhM1d0R2pxNEg0bnZ3
VkhLR1NML1pTczZPaWZCTzNZQkZFWWNIcUM1OFp0WUdCYWZGSCthYVRkTlFxTXBkOFgwVTM3cmd
0cWZlZzA2TlN0K1FUOUV6ajM1U0w3eWpzKzQrcnZYMlJweWtGRDNzNFMySSszYVZCeHFZNXZUVGxLU
DdpRFpIMUFES2puN3JTeGVpdUpZaHhxSm5OelF5TU1zZ0dtWWJXM3JuamxzVHFLVS9YRDZ0d09jRWIx
N0tTSUk5KzJKcDNNR2pjTFBkWWRXS3Npb2JkYW1kVTlDajZtSWhHNWQxRFRoTmFqd2RUWGpWelA3Y
VBzbWpKdE5hTVBGUnIybDYyd1NpVUZ0TFNsTEptcEY5Vzk4dW1ReVdwZldYSjl5T3gxekNSNjc0WDRKR
WlUNndhUDFjR1Y3WkdrbG1ZYnIvUTNYU2ZqSkFNS1p2cU9wblZMcHpad09HVnE0b1FXOURSRkVMNnd5
M0F4MnlIeXd5a2RxVWVBMlRGWThlbzQ3WDNXQU85SXlYOVpQQVMydmFKTEIvWEhsYXZPSGMzcDNP
V1QzSUdwcTZuZ3NPMjFGb2tWblZEdkkrS2ZHPC94ZW5jOkNpcGhlclZhbHVlPjwveGVuYzpDaXBoZXJEYX
RhPjwveGVuYzpFbmNyeXB0ZWREYXRhPg==</mod:ApplicationRequest>
            </cor:deleteFilein>
        </soapenv:Body>
</soapenv:Envelope>

Signing the SOAP message (transport signature)

The last step is to create the transport signature on the SOAP message. The signature is based on a customer signing certificate, and the style of the signature is detached. This corresponds to step 12 on the list on page 4:

```
<soapenv:Envelope xmlns:mod="http://model.bxd.fi" xmlns:cor="http://bxd.fi/CorporateFileService"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sign="http://danskebank.dk/AGENA/SEPA/SigningService">
    <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 xml:id="Timestamp-5293c3c7-b69f-48e2-866e-9fc9cd35a422"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
                <wsu:Created>2010-05-07T11:26:49Z</wsu:Created>
                <wsu:Expires>2010-05-07T11:31:49Z</wsu:Expires>
            </wsu:Timestamp>
            <wsse:BinarySecurityToken xml:id="SecurityToken-192f1eb9-6c10-482c-8719-
76754a128e84" 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">MIIDnzCCAoegAwIBAgIHDSQ4jQY0sTANBgkqhkiG9w0BAQsFADCBwjEQMA4GA1UEAxMH
REJHQ0FEQjELMAkGA1UEBhMCREsxEzARBgNVBAcTCkNvcGVuaGFnZW4xEDAOBgNVBAgTB0Rlbm1
hcmsxGjAYBgNVBAoTEURhbnNrZSBCYW5rIEdyb3VwMRgwFgYDVQQLEw9EYW5za2UgQmFuayBBL1M
xGDAWBgNVBAUTDzYxMTI2MjI4MjIzMDEwMTEJMAcGA1UEBBMAMQkwBwYDVQQqEwAxCTAHBgNVB
AwTADEJMAcGA1UEERMAMB4XDTEwMDUwNzExMDIwM1oXDTEyMDUwNjExMDIwM1owgaYxIDAeBgN
VBAMTF0JFTkdUU1NPTiBPRyBGUi4gSkVOU0VOMQswCQYDVQQGEwJESzEcMBoGA1UEChMTREJUU
yBERU1PIDEuICgzNEFLKTEeMBwGA1UECxMVQ09SUE9SQVRFIERFVkVMT1BNRU5UMTcwNQYDVQQ
FEy5TRS1LRVJIVi9EQUJBOjAwOTIxMDA4NDEtQUdSOjA2MTA0OC1VU1I6MDYxMTMzMIGfMA0GCSqG
SIb3DQEBAQUAA4GNADCBiQKBgQCVi1Bq4B6VNK8x6ISrNQlPBl3izM/RNcyGHAE/9FTdqFjMOhZygT0m
0K5SQjLY0AftdoHmLDtEbnzOzGC1x+y/pM8DbfmmDRXV6oC4jzq97AjFTUYI0tJrAe1Fc10mh4SnstWVYaA
hsZu8LxSPyC4JBB+66msvK4RgkLauTZw3/QIDAQABozgwNjAJBgNVHQ4EAgQAMBkGA1UdIwQSMBCAD
```

```
sLGxMP29vnBwsTCxsTDMA4GA1UdDwEB/wQEAwIGwDANBgkqhkiG9w0BAQsFAAOCAQEAREM7TjPXuf
RxrYM90hCP373oRBwrpFqZaXsEo0BDegbZ5ASks5YXZL6OAjU2Ax93wJwpKn1yDBHCkhdZcC426vhDOQ
mjdSiL6mr6OQ8+YSCmITcYM7xZxYYHAW0MBOJCfNZ0rx32OoJV9gSENq408pu23j4nPrTpIdXUovLsLBV
aGpeR6mGUnjRYUwFXt3N73kWx+z6U0SFS0rPA2dmzt3WFxzodxdyvj4KWpLEOrsVjAYTMLCNqTycYiUYZ
eDFGDrwyAfVIh4zhCYtrrPTPhDmOokhzQTv9B2W/8tNWNFNudHGHmm4DsJozPuj1aeWe70nTfgojpVC/hI
WI1KQFbg==</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-5293c3c7-b69f-48e2-866e-9fc9cd35a422">
                            <Transforms>
                                <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
                            </Transforms>
                            <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                            <DigestValue>DUOvvFgg7x0JG9iTrzVHXb4DdQo=</DigestValue>
                        </Reference>
                        <Reference URI="#Body-89bd1994-b8b8-4b7b-a58b-593dcfd671a5">
                            <Transforms>
                                <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
                            </Transforms>
                            <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                            <DigestValue>4Qd2BBARXSg4zRgI0AhwU+CiqB4=</DigestValue>
                        </Reference>
                    </SignedInfo>

        <SignatureValue>HCRlddnNZTElpF1avxIFu0EkCb7tizEsdCmPFdCH2wFTVW5pW6/5l2aDkDobu5rlHF
r5trH8I42+3sv0lji1DcTEfsMLkdhQY1n0TAFX6LBZSP11inP9ydGoWsOm3pnGwC8nXm4zXlmhJFczCSQRG
70b+pCF2Ogc34Eh3FUT8MA=</SignatureValue>
                    <KeyInfo>
                        <wsse:SecurityTokenReference xmlns="">
                            <wsse:Reference URI="#SecurityToken-192f1eb9-6c10-482c-8719-
76754a128e84" 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 xml:id="Body-89bd1994-b8b8-4b7b-a58b-593dcfd671a5">
            <cor:uploadFilein>
                <mod:RequestHeader>
                    <mod:SenderId>061133</mod:SenderId>
                    <mod:RequestId>33</mod:RequestId>
                    <mod:Timestamp>2009-12-17T09:30:47Z</mod:Timestamp>
                    <mod:Language>EN</mod:Language>
                    <mod:UserAgent>CustomerSoftwareId</mod:UserAgent>
                    <mod:ReceiverId>DABAFIHH</mod:ReceiverId>
                </mod:RequestHeader>
        <mod:ApplicationRequest>PHhlbmM6RW5jcnlwdGVkRGF0YSBUeXBlPSJodHRwOi8vd3d3LnczLm9y
Zy8yMDAxLzA0L3htbGVuYyNFbGVtZW50IiB4bWxuczp4ZW5jPSJodHRwOi8vd3d3LnczLm9yZy8yMDAxLz
A0L3htbGVuYyMiPjx4ZW5jOkVuY3J5cHRpb25NZXRob2QgQWxnb3JpdGhtPSJodHRwOi8vd3d3LnczLm9y
Zy8yMDAxLzA0L3htbGVuYyN0cmlwbGVkZXMtY2JjIi8+PGRzaWc6S2V5SW5mbyB4bWxuczpkc2lnPSJodH
RwOi8vd3d3LnczLm9yZy8yMDAwLzA5L3htbGRzaWcjIj48eGVuYzpFbmNyeXB0ZWRLZXk+PHhlbmM6RW
5jcnlwdGlvbk1ldGhvZCBBbGdvcml0aG09Imh0dHA6Ly93d3cudzMub3JnLzIwMDEvMDQveG1sZW5jI3JzYS
0xXzUiLz48ZHNpZzpLZXlJbmZvPjxkc2lnOlg1MDlEYXRhPjxkc2lnOlg1MDlDZXJ0aWZpY2F0ZT5NSUlEeVR
DQ0FyR2dBd0lCQWdJS05EUTBORFF6TURBd01qQU5CZ2txaGtpRzl3MEJBUXNGQURDQnhERVFNQTR
HQTFVRUF4TUhSRUpIVWs5UFZERUxNQWtHQTFVRUJoTUNSRXN4RXpBUkJnTlZCQWNUQ2tOdmNHV
```

nVhR0ZuWlc0eEVEQU9CZ05WQkFnVElwUmxibTFoY21zeEdqQVlCZ05WQkFvVEVVUmhibk5yWlNCQ1lX
NXJJRWR5YjNWd01Sb3dHQVlEVlFRTEV4RkVZVzV6YTJVZ1FtRnVheUJIY205MWNERVlNQllHQTFVRUJ
STVBOakV4TWpZeU1qZ3hNVE13TURBeE1Ra3dCd1lEVlFRRUV3QXhDVEFIQmdOVkJDb1RBREVKTUFj
R0ExVUVEQk1BTVFrd0J3WURWUVFSRXdBd0hoY05NRGt4TURNd01USXdNREF3V2hjTk1URXhNRE13
TVRJd01EQXdXakNCeFRFUk1BOEdBMVVFQXhNSVJFSkhRMUpaVUZReEN6QUpCZ05WQkFZVEFrUkx
NUk13RVFZRFZRUUhFd3BEYjNCbGJtaGhaMlZ1TVJBd0RnWURWUVFJRXdkRVpXNXRZWEpyTVJvd0d
BWURWUVFLRXhGRVlXNXphMlVnUW1GdWF5QkhjbTkxY0RFYU1CZ0dBMVVFQ3hNUlJHRnVjMnRsSU
VKaGJtc2dSM0p2ZFhBeEdEQVdCZ05WQkFVVER6WXhNVEkyTWpJNE5EUXpNREF3TWpFSk1BY0dBM
VVFQkJNQU1Ra3dCd1lEVlFRcUV3QXhDVEFIQmdOVkJBd1RBREVKTUFjR0ExVUVFUk1BTUlHZE1BME
dDU3FHU0liM0RRRUJBUVVBQTRHTEFEQ0Jod0tCZ1FDcGFlMnROYkNtUFJuaTc0dDRwWHB6VXdzVVR
SQmVhelBwY2haSGpuK0hCcGZRQUg2NmtDek9jeExIT3JMNzFDRllJajZCNFdycVhaeDRTVWxkUXBLdTY
3dlNoNVpzTHl5WFBabmlDbytEdFgyek9ZR0VpZ3pxNmhJeWxSNVhLblhrWmZKU3hvL1RnR1gzcmtMeEY2
Y2JhUDZteE43TGNDdjlkVWorZVJubmdRSUJBNk5BTUQ0d0VRWURWUjBPQkFvRUNFUkNSME5TV1ZC
VU1Ca0dBMVVkSXdRU01CQ0FEc0xHeE1QMjlzTDI5L1gzOWNYRU1BNEdBMVVkRHdFQi93UUVBd0lFT
URBTkJna3Foa2lHOXcwQkFRc0ZBQU9DQVFFQUQ4K0Jwbk0vSzEwbTVpS080QlRjdkdpNzdvU3Awc3Vs
NXV0UDZaRkV2ZXVxamxNeDR4eGgrMVNWUlNNWFF0dGQ0OS8wOEJJVU9hWXJJanZQK2RKVDVKNk
QrVWVGUlNqRTZtZXZqUkUxcXFJcENzSG9DV1ZFOVdraXVGM1FMYW91RDlQSDVGUHNrNTNRVmRkN
FlwK3ZJS0RWWVFicGMxMkE4d0FFdThoa25ncXhOMi9FcGQ4dS8zTFFnc1RtWTZtbjVlazJFRk5JNm8yZF
FhNTAzRHB6am9YR1VSSUZPb0FhV0I3WlVKaGU3bnhHMVhqbjFPbnR0cmVvUDZ0d0NUckc4SW9MZkdL
RGtMVlEzRys3dkFxWGdCK0p3RkxVb1VMUmF3MGVJU1BnRTduTWt6MGxhMGx1K2FKbGFIUkpacS9GO
WVNa0RYRnFSU01HSEhOVWlqenJRUT09PC9kc2lnOlg1MDlDZXJ0aWZpY2F0ZT48L2RzaWc6WDUwOU
RhdGE+PC9kc2lnOktleUluZm8+PHhlbmM6Q2lwaGVyRGF0YT48eGVuYzpDaXBoZXJWYWx1ZT5CYXhDS
XUyWEVtWHJDSEdmQU1NSC8vVGxma1F5UXIzNnlQUmR0UTc4UTQvMjhTQ2k3Slh1S2hxN09EWFh5O
Wo0bDRVU0d0Y09TcTVsSU1QUFE3SnVnVkNRaVhaVkRCVEVLNlRqN3VNL0NQUlljb0Fmb3ZyMkw4QTB
telk3MDR0YzhDMVZIaXRaUDcyZVZrOUs4aFN3ZTRXK25QaS8wUlVSaUxoMHBhcVNQdjg9PC94ZW5jOk
NpcGhlclZhbHVlPjwveGVuYzpDaXBoZXJEYXRhPjwveGVuYzpFbmNyeXB0ZWRLZXk+PC9kc2lnOktleUluZ
m8+PHhlbmM6Q2lwaGVyRGF0YT48eGVuYzpDaXBoZXJWYWx1ZT5MY3pPcHlrbHQ2bXV3dGFjSmdyMk9
mdlNMczZKMXV5bUZaR28xT2x3NExveG9hWjhZdkMvZm0wa04xOEViQUxtZU8xS2p0cTZ2VWp3MEc4SFl
qaGlLanROUmpJSTJWbkRoTndZMGQ0cWU0QlNONjNIbFFBZzZabTNmd003MjdyVSs1WC9SZy9wSE9qU
DZwamFPenR1STZEY000Y1YzZmtBTnBGaGRFekhQWlQ5ek9ET0F6Y0wyRDdFam5hL0pnS1NybEVOdE
ZscWd3NG1BOVB0c3UrQndWQnprQzA2U3N3VXRsTG9ZYnB1TzZZL2RqMjc2S1Era05Nd1EzZ2FKTUVN
ZG0xcDBzRStkU0s2V1U0WHhrNTMzZkc1a21wRU9zWEtjbk1BanJlT25KdFVlUXRiamFLM2Vud2xIRjZpNEI
wQ0IvZmFqY0QyUFZBVmdlRXpxQ29DRmN1cEczVWlvR0EzNXRBRTZ0emhjRHNzVUNEaWhFV2NUVlNZ
VjFMMWowR2RDYXhHUTBRdHRhZWRYTVU3OUZoUXg3M2NjcFU0WnU0KzliNEVqRTRkanZMUXBJckd
VRnh3Z0I4NDl5Y0JzdmkrSVYvVG9haEZ3QXNwTHk3dlZaTUh6TnRGcm5uVGxUTFZ1ZVRmZmROZXZOT
XdaUGtKcFJwSUE0ZllmcDMwY2pEY2JqRmkvdlRsRmJJR3lzMnRIZHh4M0JPWkVXOThFeHBYTFZoRGd2
SDhMNzF5MDJaMVlBTCtRTDZxTFZrcDhXWnlROS9nSEo1dTVHZ1BiZFdsNWhKb2w3dTVOaHRGdUROU
k95aStQTDhOUXZLeXFaa1RoNVFWYnFBbjVBK2QxMEdqeW4vZ0JHbUpvRkxRbHFqM0pCamJaNktYYU9
QSy8xSEMrMTkrNUZmQ1NmNnB0eG04RjlSMXVNMzhEdHUvSlBVN1RKM3h1a2tiU1pZUGxQbTFKZnlKR
GFlSlYrcHIvMGVuNHNDMGhDcFRleXJzWHpCblRyTG5tOHFnbkkrMktLK2hFY01kQzRxSVdZeGJReEVOa
1NCNXlaWDdlTXUrbThndDNzYnZoT2ozOXUyK09HK2FLKzkwa1VuT3ZMYTJqUGtLSWw5SWEzTXRZemd
qbFdZSk1GQzdacWdwUnRNL0dCckFrYjRxU0IzbTdIVlRYS3NwYzBXT3BaWTZHY1Y2WFcxL21zZCt0N3Z
XUUlwQitDcHhZTEJXNVJHazU3TTNzY2gxeGFIbWwxVW4ycVNaQWxIYS85SjY1QlZkMG1SaDR1ZzNpaW
RWdEZIWUFickMzQWNyNEM3WmIraVl2MW5LWElpekwzRWk1cGZPcHo3YWNpTVk2dDBxQ3hjMllQajR4
WUtWdU9FUHgzUFNDaEVOTStjQlpYMWxWYk52cC9ncTJ4YmdDWXZJemtnYmcxQWxVZitCNnhPVlZ6cz
F3SnVOMk1ldnJtVDVMblNuMU1HaVBzMFVWTDl2VTJYT043QjBMN090dis4SVBYL0Myb1NTanZCS1h4Yk
g3RFFENTNmWTV0N2lNcHZydEFhVzl5cC9ySW5BYm9zN3R5SGpBRXAxeEU5bVdlSk96Vjc0VnpydDVIa
Gs1dU9xYzk0Nm4rWERuR21RMDV0SUlqSW50QnhlWFVDY0VoNTRhTXB0SjdZNjBCTmlHSU1EUWpVR
XVXU0pZZEpHeHFvYjA4Z3IrNVB5UjE0RUJFOWN6S2kvZnpsdlR1enp4RDBwdXFZQnZnNlRyeVB0WVJpN
2s4Nk42ZlR0OU4wYUhHeWhZL25OVWcxcndhZ2xwRThhdkFtREZIQjlzazRJNnZUMXc4OEdxeUpCaHRvb
GZOVHdnWDJjVGpjbjJBTUdkRytzWXdnSmF3VkRpU09GUlB1WnVKaXlEUzd6d2J4Wm13WlE3MjhHNHJK
OVFEaFc2U3R5MXcybzVxL2RZdjJiS0hHaDhZb0NMb0c1RFZoTUp2TnJjUk9aczB4RHNKelVRaVhDMWZ4
Tm91WllZRjU5OHc2N2dMWnZTNFh0Y1VVZUI2UkNoMTNONjZsaFZvUEdZZHFFSEJ2V3lpaVZYbHVjSmJI
YkJKOGtzanhMM242WTVtcjAxRDJLNXFKQW9RMDI3QVRFL282bXY3U3BCQ2EyOE9QMGlajg3b283SnB
jVmdrSDViZG9XOHBsNEVYY1BnbVZkTzcwTWgzTjAzSHNCc1NDY0dRWGFnQThmK0hod3JucnZWTENR
REVaNStBczBsbDRFVFRvTi9uSGhHMkRUYUI4ZkZWWk40YU02eW95SmVub0NmNk4vSXZsSi9pWHo1N
UtXMkFoZ3JMNk9TTXFuZkZoRVVSclVJdjJBcC9nanQyYjVtVEh2KytnTTdXQWFwRTB6QkR0R3NQYmFrZ
WN0VjNEVkpjZzFFZmFvNEw4WGV3N3c3L0U3MXJGOE4xTzFxd3ZLVzIwZ1J6TDZhekpaajhqRFZabW5lW

TNnc1lQckQ1Y3BReHZFMS82dDczdVQ0UHVkRDhBQVF1V05hMlpaZ0QxbTZyMFZINEJDbXZERWVKZG
NLcmRNYmdsdDR1RlVGWFl2OXgwWWY4STZSb3N5ekpYcUpUQ21Ba0V3UUFwUEJ3bE50LzRtaVBIaVN
lSldWMjg0cUl4L3gzMTJVVnR1YTdjNm1GZEVMUzlSazlyM0hSVTR6cXM3Z2RDczlUM3A5cDJpNGRINDIrc
zJBalZyNkVDWUVsOVZETTI1REZzaHI4WnJEVUNCWlBGcjFLbnhiNDBwR0I0d2JoeTRvNE5mVnIzZlhTMlJ
lRXNrSXA2SlloYUFLNUFHR01xblNyUVQ4dWdPbDRrcldXbWJvL0ZiQldxMmtuQVM5LzdicE9ZSE9XVHEy
M2RLT3dWT000TVZKVGlxckRITXNmOUhOdlp6dkNwckxJRGRFTWRFYUZXSWgxbGdQMk8yQkd6ZUR0c
FdhR0cxbHh3N0g5aGdaR2sxbENnSFhSWHd2R25LVUZLUWhCblZxaXk5VEp5M1dZbmYxOGJQRnlVblE3
QjZuditpRjJEMFprZGExMEhGd0kyMFpnWWIzcFpOakxkelVHOXdKT28xaWo1WkVDUlpDZTh2TzB6R2xZel
JTTC9iTElYUFZjV3FaS1lnbHBMVzd5R0JkdDVRR0t6TlBpM1d3MjNkZEVDcWFyKzZ4SlFYZG1RRk9EWmd
6Nk9ZMnc5bEpaOXhVdG9nMHFNVlNCM1lzREZsOXdNNHovSGJLUTFnK0VKRW5YV0J0V0ZWbWlnenB6
VjZuR0RSbUViNlJQRzF2bmFabmRYbnJZci9yc1JDcU0wVzB2Q2hrRjdDbXpwWWtDUHFWRWNLdnEvc2tC
V0lUT1lFdENUQmZzbXJaQUtodDdCckdSZVNaalp6U0ZoSXNKM212UnV6MFJaakhPSnJFTDJMYlgzT1R1
YmZML0JBSXo2Z2xLdWZTLzRmdnR0cnpPZzVBV0FROHVlRi9KaUprYXNVcWFnalhtZzVnakFkOHFTQmt
mMFY5cjI3cVFmM1poZEhXZ2xzNXlJZE1yTm5oN2ZRc2hzbGhaYzVYTTd4QVlkRHVuOVoxMkRvK3pJY3p
EeDFiOThEV0dTVkpnaDVGVUx4emwxalhhcEJkRFlCSzJ3WXBiMnVNcTVCQXovbk5SN0Nka3RFdGVTTE
4xWGFOL3ZSYnp5TlU1ckthRDV3VmFrQ0RiYmJvSXBZOUFEOUlvQXlZajAwOEthZTJ6SXNCcW5KNGhSV
TllS0ZzWUJWY0g5MStSZkZtVS9Cb0hZZkw2dUdCTHhHU29FK3J5QlVOMVlCWDlVQkg0R2FHc2lGUTlFc
Gk0cFJQUFdkNjJ6SGhxRzd1QUh1bGFkTGNVSjZnWU9Yb292WlQ5S2xPa2ZEZW1QdkpZeU9GWHp5K3V
PejMwSGgvYkVGNjNna25JMUZWY1kwZ0pyRnl3MUFTU2JkVWwrSDBZUHdYNnY0K1pOMjd1S3IyV0dSS
GVqUlBrTUtOTnArVmRuTy9YQWdIbjZFRnd1N0xUMnpVd2JONEt2amVLYTlRM0s1ZWtoVmRyQXUvTHhq
SWFHVGJmS0RqMGg2cDNlSFNvWHFpdE5BaUNXcFRTQXR2Q3BiNnJ5UmxkWkxvOG1sMTgvQkpYaWJ
DZU5DY3hESjFjNEpMemtleWJoR1IzMUFnQTZEZUhuLzArYzFMNS9sZ3RhRHNnbjRpV21TcTZnSzBCUml
pNWZsWHFWcGxUTVNVU0RhNDNVMmJEdkZiand2OGJCWk1yWjBwUkJ0Z2QwT1ZPdzFObnIvV1dUS3B
sTzl0d1NaRG1SKzBZOXhjNGVBejk0ZWhuTHd2SHU4UHFJb3Y1NHc3cnNUM3JJWXNRU052WVBiWDl5Q
1liRFNKbTFUSUx2Yk9sS3RFa3dLNVZzYXdha3NPRml3clcwVEpqa282MFB1YlNQRWc2alZZaElwNkNidXp
EN1I3cm9YWitVOC83MDZyczF2QnY5ZVdsdjhzK2JUdUtRVW1GUWkwZTNoOHdUVHVsUzFIRXd5eGt1am
E1aUd2RTRNZTcraEdaSlh0elFRZ3lCb3hsZFhHbVJ5RmVId29mOHk2VlRWSXRPU0dBSldJVmx6WnRXM0
Uyajc2WCsvNUlVWG95Zmt4bXZvWFdmdTBQOEwrazMwYkRGNlcweC84MEIzQ3Y1dUN4SnA3NlJnQUhk
RWZtZ2RUL0RqTnpvTzR5MTluNlJIUi8vYVJ4ZEVFdHpYbDNybDE5YWJXdXFvZjNjTkwxV0pvT1ZqZEZNd
U83N3pJWHNUdVVmSm1ZenAxditNNFhCcko4aGxwek9kTzFtMVhVM0JtWEo4T1krOGtscFhyT0FyRS9mS
kFjWjdHTUVJdVBmSFBVM3FOMEdpSTFaUWdDZG1vSDdiclhSL0NxOWhFdW04ZXVjYVlscW95UUhvQ2h
ValJ5aldhNmVaOG1VUzY2ZmRrbWdyWTh5UDhNWlEyME1UUGdFdkdaalkwVTJKdWx3bTZ4Q1crSGs3Vn
NsbExsNkJIQ2pISHpIRC9sOUlidmIxYy9McnNRMWgxUEdyRGhvZWllSnFpbnB1R0MvM3d3akR6dnFBRncv
RGM5MS81ajdEVitYdEhtaGxvV3V3VWs5UmM5Z05pOTAzSHRiVERud2hETWprVVFaZmw0cXBmVy80OU
M2VTVCaWtSQytCWVllVjVHR296Z21MMjBTMHNXeGJVYUV1SmZTN0pZNGw3WHRFTjVacUxYcVYxNW
Z2QXNVQ2Z1VVYrS0lFZ3VRd28wbnZEU25kWVAyRE1XQXVlZGpqaklhQ3BKSDJpZlBFdnNXMWlpbldsSlJ
zNmhGazVxSVUxcFE0dkJUT2d3S09BOWZLcVI1TVRCL2E0YWErQ0pub1ZMZVcya3dzTkk2MU5GZUJvQz
MrUGF4ZTY3b0lRUXBVSjJBRG96ejRmcEZ6cE8yUnU3ZkxvNGpSVlE9PTwveGVuYzpDaXBoZXJWYWx1Z
T48L3hlbmM6Q2lwaGVyRGF0YT48L3hlbmM6RW5jcnlwdGVkRGF0YT4=</mod:ApplicationRequest>
            </cor:uploadFilein>
        </soapenv:Body>
</soapenv:Envelope>

The XML examples are distributed along with this document.

Appendix B: Web service URL

The URL of the web service is:

https://businessws.danskebank.com/financialservice/edifileservice.asmx

It is not possible to fetch the web service wsdl by appending '?wsdl' to the URL. Instead, the wsdl must be retrieved from the bank homepage.