OFTP2
Interoperability Tests
Test Cases

History:

Version	Date	Description	Author
1.0	09.08.2006	Complete list of test cases	Melber/Hauth
1.1	01.09.2006	• Correction of Test Case 3.2: ”SFIDSIGN = Y” instead of “SFIDSIGN = N” • Spelling corrections (Test Cases 2.3 and 2.4)	Möller/Hauth
2.0	16.02.2007	Adding of chapter 4.6 Automatic Exchange of Certificates	Melber/Hauth
2.1	15.10.2007	Added chap. for exchanging certificates without request	Melber
2.2	01.12.2008	Rewrite the chapter with the automatic certificate exchange	Gloski/ von der Heydt/Bender
2.3	12.01.2009	Result of OFTP2 Group web meeting 04.12.2008: chapter 4.6: Adding test for exchanging certificates in simple scenarios with SSID == SFID.destination/originator, chapter 4.7: removing test of revoked CA by a CRL, adding tests for TSL changes	Bender
2.3.1	15.01.2009	Result of OFTP2 Group web meeting 12.01.2009: chapter 4.6 and 4.7: no use of revoked certificates in later test cases => renumbering certificates, remove of redundant test cases, the previously optional test cases are mandatory now	Bender
2.3.2	27.02.2009	Corrections in test cases, renumbering test cases	Bender
2.3.3	05.03.2009	Result of OFTP2 Group web meeting 02.03.2009: The use of all specified file formats are mandatory now, (especially file formats: T, F128 and V44), Corrections in test case 7.1	Bender
2.3.4	10.11.2009	Result of OFTP2 Group web meeting 06.11.2009: No different Certificates for the same function in one Station, remove unused Certificates CB02+CB03 (Test 6.3.3 and dependent test 6.3.5) Test case 3.1 + 3.2, correction “SFIDCIPH = 01” instead of “SFIDCIPH = 00” consistency: Certificate List (Page 29) CA10 is for Station A1 Clarifications and spelling	Bender
2.3.5	10.05.2010	Result of OFTP2 Group web meeting 01.04.2010: Test case 3.3, correction “SFIDCIPH = 01” instead of “SFIDCIPH = 00”	Bender
2.3.6	7.10.2010	Test case 6.3.5 deleted	Walther
2.3.7	3.12.2010	Test case 6.3.4 modified.	Walther
2.3.8	28.02.2013	Test case 1.3.1 “Creation of session with OFTP2 level (V 5) accepted” added	Gaschet

2.3.9	21.06.2013	Duplicate of test case 1.3.1 removed, chapter 4.6 text corrected (certificate binding to CLID instead of fqDHN)	Walther
2.3.10	16.01.2017	Test cases 1.1 and 1.2 removed Test cases 6.3.8 and 6.3.9 added	Walther
2.3.11	06.04.2018	Test case 3.1 changed for EERPHSH	Melber
2.3.12	01.06.2018	Test case 6.3.3 updated	Walther
2.4	26.08.2020	Amended: Test cases 4.4.1 and 4.5.1 Added: Test case 4.1.6 Restart file transfer functionality capability; Added: Section 4.6.4 TLS functionality Test case 6.4.1 Deny unsecure Diffie-Hellmann key exchange Test case 6.4.2 Validation of TLS client certificate functionality Test case 6.4.3 Rejection of an expired TLS certificate	Walther

Content:

1 Introduction

The document describes the test cases for the OFTP2 interoperability tests. OFTP2 is described in oftp2-internet-draft.

2 Overview

The tests are divided in two parts. Part one contains CMS package tests and file transmission without security features. Part two will include tests of file services, signed EERP/NERP and TLS.

The number of test cases is kept small. Therefore test cases like “exchange of CMS file” will contain tests in both directions, writing and reading or sending and receiving.

For each test case there will be one file suggested to perform the test with. The example files for the tests will be delivered to the participants of the test rally by CD.

The test cases define the use of special file formats. These file formats must be used in the tests. Every tested OFTP2 software must implement the file formats U,T,F128 and V44.

3 Test Objects

Local Station	
Software Vendor	
Software Name	
Software Version	
Contact/Tester	
Remote Station	
Software Vendor	
Software Name	
Software Version	
Contact/Tester	

4 Test Cases

4.1 CMS and OFTP2 Basic

4.1.1 Test deleted

4.1.2 Test deleted

4.1.3 Creation of session

Test Case 1.3.1	**Creation of session with OFTP2 level (V 5) accepted**		
Date:	Version:	Test File: not applicable	Test passed (Y/N):
Prerequisite: Network connection and OFTP SSID information had been exchanged between the test partners.			
Procedure: Initiate and respond to an OFTP session to the test partner with following SSID options: SSIDLEV = 5 SSIDSDEB = 2048 SSIDSR = B SSIDCMPR = N SSIDREST = Y SSIDCRED = 99 SSIDAUTH = N			
Expected Result: The test partner can successfully initialize an OFTP session (exchange of SSID) and end the session with ESID, reason='00' (normal session termination)			
Real Result:			

Test Case 1.3.2	**Creation of session with version downgrade in the answer**		
Date:	Version:	Test File: not applicable	Test passed (Y/N):

Prerequisite:

Network connection and OFTP SSID information had been exchanged between the test partners.

Procedure:

Initiate and respond to an OFTP session to the test partner with following SSID options:

Initiator	**Responder**
SSIDLEV = 5	SSIDLEV = 4 or less
SSIDSDEB = 2048	SSIDSDEB = 2048
SSIDSR = B	SSIDSR = B
SSIDCMPR = N	SSIDCMPR = N
SSIDREST = Y	SSIDREST = Y
SSIDCRED = 99	SSIDCRED = 99
SSIDAUTH = N	SSIDAUTH = N

Expected Result:

In response to an OFTP2 solicitation, the responder must be able to negotiate a session version between 1 and 4. The initiator may either accept or reject the negotiation with an 'end of session' code '10'.

Real Result:

4.1.4 Transmission of file (unstructured, no security features)

Test Case 1.4	**Transmission of file (unstructured, no security features)**		
Date:	Version:	Test File: U	Test passed (Y/N):
Prerequisite: OFTP session successfully established. (like in test case 1.3)			

Procedure:
 	
Send and receive an unstructured file to and from the test partner with following SFID options:
 	
`SFIDFMT = U`
`SFIDLRECL = 00000`
`SFIDSEC = 00`
`SFIDCIPH = 00`
`SFIDCOMP = 0`
`SFIDENV = 0`
`SFIDSIGN = N`
`SFIDDESCL = 0`			

Expected Result: The test partners can successfully transmit the files.			
Real Result:			

4.1.5 Transmission of file (unstructured, no security features) with EERP

Test Case 1.5	**Transmission of file (unstructured, no security features) with EERP**		
Date:	Version:	Test File: U	Test passed (Y/N):
Prerequisite: OFTP session successfully established. (like in test case 1.3)			

Procedure:
 	
Send and receive an unstructured file to and from the test partner with following SFID options:
 	
`SFIDFMT = U`
`SFIDLRECL = 00000`
`SFIDSEC = 00`
`SFIDCIPH = 00`
`SFIDCOMP = 0`
`SFIDENV = 0`
`SFIDSIGN = N`
`SFIDDESCL = 0`
 	
Receive and send the associated EERP to the sent and received file.			

Expected Result: The test partners can successfully transmit the files and the associated EERP.			
Real Result:			

4.1.6 Restart file transfer functionality capability.

Test Case 1.6	**Restart file transfer functionality capability.** Optional (only if both parties support this functionality)		
Date:	Version:	Test File: UL	Test passed (Y/N):
Prerequisite: Successfully finished 4.1.5 A file transfer session has been successfully established.			
Procedure: Interrupt the session during transfer. Retransmit with SFIDREST greater than 0.			
Expected Result: Automatic restart of data transfer. The receiver should also answer with SFPAACNT greater than 0. The file received should be valid.			
Real Result:			

4.2 File Services

4.2.1 Transmission of CMS file (3DES encrypted)

Test Case 2.1	**Transmission of CMS file (3DES encrypted)**		
Date:	Version:	Test File: T	Test passed (Y/N):

Prerequisite:

Certificates between the test partners had been exchanged.
OFTP session successfully established. (like in test case 1.3).

Procedure:

Creation of a CMS file that uses 3DES encryption.
Sending the CMS file via OFTP to the test partner and in exchange receiving a CMS file from the partner.

SFID options:

```
SFIDSEC   = 01
SFIDCIPH  = 01
SFIDCOMP  = 0
SFIDENV   = 1
SFIDSIGN  = N
SFIDDESCL = 0
```

The received file has to be decrypted.

Expected Result:

The result file equals the original file.
The test partner has the same result.

Real Result:

4.2.2 Transmission of CMS file (compressed)

Test Case 2.2	**Transmission of CMS file (compressed)**		
Date:	Version:	Test File: F128	Test passed (Y/N):
Prerequisite: OFTP session successfully established (like in test case 1.3).			

Procedure:
 	
Creation of a CMS file that uses compression.
Sending the CMS file via OFTP to the test partner and in exchange receiving a CMS file from the partner.
 	
SFID options:
`SFIDSEC = 00`
`SFIDCIPH = 00`
`SFIDCOMP = 1`
`SFIDENV = 1`
`SFIDSIGN = N`
`SFIDDESCL = 0`
 	
The received file has to be decompressed.			

Expected Result: The result file equals the original file. The test partner has the same result.			
Real Result:			

4.2.3 Transmission of CMS file (signed)

Test Case 2.3	**Transmission of CMS file (signed)**		
Date:	Version:	Test File: V44	Test passed (Y/N):

Prerequisite:

Certificates between the test partners had been exchanged.
OFTP session successfully established (like in test case 1.3).

Procedure:

Creation of a CMS file that uses signature.
Sending the CMS file via OFTP to the test partner and in exchange receiving a CMS file from the partner.

SFID options:

```
SFIDSEC   = 02
SFIDCIPH  = 01
SFIDCOMP  = 0
SFIDENV   = 1
SFIDSIGN  = N
SFIDDESCL = 0
```

The signature of the received file has to be verified.

Expected Result:

The signature of the received file is valid.
The test partner has the same result.

Real Result:

4.2.4 Transmission of CMS file (compressed, AES encrypted)

Test Case 2.4	**Transmission of CMS file (compressed, AES encrypted)**		
Date:	Version:	Test File: U	Test passed (Y/N):

Prerequisite:

Certificates between the test partners had been exchanged.
OFTP session successfully established (like in test case 1.3)

Procedure:

Creation of a CMS file that uses compression and AES encryption.
Sending the CMS file via OFTP to the test partner and in exchange receiving a CMS file from the partner.

SFID options:

```
SFIDSEC   = 01
SFIDCIPH  = 02
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = N
SFIDDESCL = 0
```

The received file has to be decrypted and decompressed.

Expected Result:

The result file equals the original file.
The test partner has the same result.

Real Result:

4.2.5 Transmission of CMS file (signed, compressed, 3DES encrypted)

Test Case 2.5	**Transmission of CMS file (signed, compressed, 3DES encrypted)**		
Date:	Version:	Test File: F128	Test passed (Y/N):
Prerequisite: Certificates between the test partners had been exchanged. OFTP session successfully established (like in test case 1.3).			

Procedure:
 	
Creation of a CMS file that uses signature, compression and 3DES encryption.
Sending the CMS file via OFTP to the test partner and in exchange receiving a CMS file from the partner.
 	
SFID options:
`SFIDSEC = 03`
`SFIDCIPH = 01`
`SFIDCOMP = 1`
`SFIDENV = 1`
`SFIDSIGN = N`
`SFIDDESCL = 0`
 	
The received file has to be decrypted, decompressed and the signature has to be verified.			

Expected Result: The result file equals the original file and the signature is valid. The test partner has the same result.			
Real Result:			

4.2.6 Transmission of file with file description

Test Case 2.6	**Transmission of file with file description**		
Date:	Version:	Test File: V44	Test passed (Y/N):
Prerequisite: OFTP session successfully established (like in test case 1.3).			

Procedure:
 	
Sending a file with UTF-8 encoded file description via OFTP to the test partner and in exchange receiving a file with UTF-8 encoded file description from the partner.
 	
SFID options:
`SFIDSEC = 00`
`SFIDCIPH = 00`
`SFIDCOMP = 0`
`SFIDENV = 0`
`SFIDSIGN = N`
`SFIDDESCL = <length of the file description>`			

Expected Result: The file description is of the given length and readable (at least valid UTF-8 encoded).			
Real Result:			

4.2.7 Transmission of large file

Test Case 2.7	**Transmission of large file**		
Date:	Version:	Test File: UL	Test passed (Y/N):
Prerequisite: OFTP session successfully established. (like in test case 1.3).			

Procedure:
 	
Sending a large file (>100MB) via OFTP to the test partner and in exchange receiving a large file from the partner.
 	
SFID options:
`SFIDSEC = 00`
`SFIDCIPH = 00`
`SFIDCOMP = 0`
`SFIDENV = 0`
`SFIDSIGN = N`
`SFIDDESCL = 0`			

Expected Result: The large file can be received.			
Real Result:			

4.2.8 Rejection of file transmission with answer reason text

Test Case 2.8	**Rejection of file transmission with answer reason text**		
Date:	Version:	Test File: not applicable	Test passed (Y/N):
Prerequisite: OFTP session successfully established (like in test case 1.3).			

Procedure:
 	
Sending a file via OFTP to the test partner using an invalid destination in the SFID and in exchange receiving a file from the partner that is addressed to an invalid destination. The file reception will be rejected by answering with SFNA. The answer reason text will be filled with an UTF-8 encoded explanation.
 	
SFNA values:
`SFNAREAS = 02`
`SFNARRTR = N`
`SFNAREASL = <length of the answer reason text>`			

Expected Result: The answer reason text is of the given length and readable (at least valid UTF-8 encoded).			
Real Result:			

4.3 Signed Responses

4.3.1 Transmission of file with signed EERP

Test Case 3.1	**Transmission of file with signed EERP**		
Date:	Version:	Test File: T	Test passed (Y/N):

Prerequisite:

Certificates between the test partners had been exchanged.
OFTP session successfully established (like in test case 1.3).

Procedure:

Sending a file via OFTP to the test partner and in exchange receiving a file from the partner, both requesting a signed response.

Hash of the transmitted Virtual File, i.e., not the hash of the original file and a signature has to be created and used in the EERP. The signature or the received file has to be verified.

SFID options:

```
SFIDSEC   = 00
SFIDCIPH  = 01
SFIDCOMP  = 0
SFIDENV   = 0
SFIDSIGN  = Y
SFIDDESCL = 0
```

Expected Result:

The field EERPHSH is filled with the hash of the transmitted data, field EERPSIG is filled with the signature and can be successfully verified.

File hash must be compared with the hash of the file sent and must be equal.

The test partner has the same result.

Real Result:

4.3.2 Transmission of file with signed NERP

Test Case 3.2	**Transmission of file with signed NERP**		
Date:	Version:	Test File: F128	Test passed (Y/N):

Prerequisite:

Certificates between the test partners had been exchanged.
OFTP session successfully established (like in test case 1.3).

Procedure:

Sending a file via OFTP to the test partner and in exchange receiving a file from the partner, both requesting a signed response.

A signature has to be created and used in the NERP. The signature or the received file has to be verified.

SFID options:

```
SFIDSEC   = 00
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
SFIDDESCL = 0
```

NERP values:

```
NERPREAS  = 34
NERPREASL = 0
```

Expected Result:

The signature of the received NERP is valid.
The test partner has the same result.
Note: The error has to be produced manually, e.g. by deliberately changing the received file after reception but before validation of the signature.

Real Result:

4.3.3 Transmission of file with signed NERP and reason text

Test Case 3.3	**Transmission of file with signed NERP and reason text**		
Date:	Version:	Test File: V44	Test passed (Y/N):

Prerequisite:

Certificates between the test partners had been exchanged.
OFTP session successfully established (like in test case 1.3).

Procedure:

Sending a file via OFTP to the test partner and in exchange receiving a file from the partner, both requesting a signed response.

A signature has to be created and used in the NERP as well as a reason text (UTF-8 encoded). The signature or the received file has to be verified.

SFID options:

```
SFIDSEC   = 00
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = N
SFIDDESCL = 0
```

NERP values:

```
NERPREAS  = 34
NERPREASL = <length of the reason text>
```

Expected Result:

The signature of the received NERP is valid, the reason text is of the given length and readable (at least valid UTF-8 encoded).
The test partner has the same result.

Real Result:

4.4 TLS

4.4.1 Creation of session and file transmission using TLS 1.2

Test Case 4.1	**Creation of session and file transmission using TLS 1.2**		
Date:	Version:	Test File: U	Test passed (Y/N):
Prerequisite: Certificates between the test partners had been exchanged.			
Procedure: Opening a session on the OFTP TLS port and sending a file to the test partner. In exchange the partner will open a session on the OFTP TLS port and will send a file as well. The TLS must use PFS ciphers. Check that PFS ciphers are supported.			
Expected Result: The OFTP session can be created using TLS 1.2, the file has been sent and the file received is identical to the original file. The test partner has the same result. If the server cannot use TLS 1.2, there should be a handshake to negotiate a lower version (1.1 or at least 1.0). If PFS ciphers are in the supported cipher list, a PFS cipher must be used.			
Real Result:			

4.5 OFTP Authentication

4.5.1 Creation of session and file transmission using OFTP authentication

Test Case 5.1	**Creation of session and file transmission using OFTP authentication**		
Date:	Version:	Test File: U	Test passed (Y/N):
Prerequisite: Certificates between the test partners had been exchanged or will be included in the AURP CMS signature package.			
Procedure: Opening a session with OFTP authentication and sending a file to the test partner. In exchange the partner will open a session with OFTP authentication and will send a file as well.			
Expected Result: The secure OFTP session can be created, the file been sent and the file is identical to the original file. The test partner has the same result. The test will be performed with different settings to check successful and failing scenarios: Unsuccessful cases: Sender sends an SSID with SSIDAUTH(N), receiver is configured to use SECAUTH → receiver should answer with an ESID(12). Receiver is configured to not use SECAUTH, sender sends SSID with SSIDAUTH(Y) → Receiver should answer with ESID(12). Sender is configured to use SECAUTH and sends an SSID with SSIDAUTH(Y), receiver answers with an SSID with Protocol Release Level < 5 (OFTP 1.4 or older) → sender should send an ESID(10). Also acceptable is the receiver answering with ESID(10). Receiver is configured to use SECAUTH, sender sends an SSID with Protocol Release Level < 5 (OFTP 1.4 or older) → receiver should answer with ESID(10).			
Real Result:			

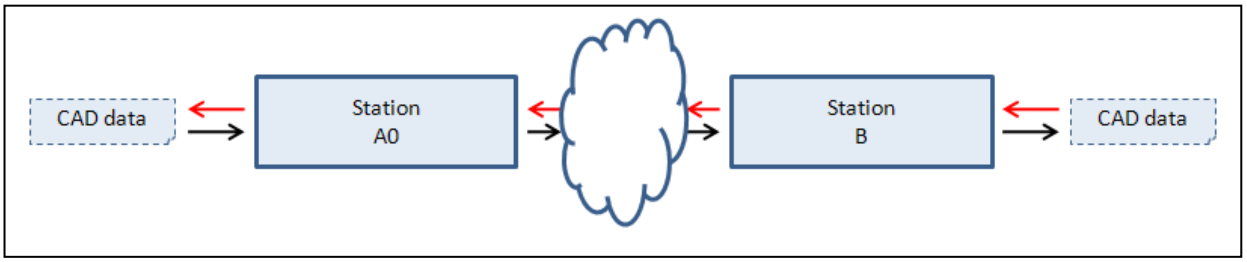
4.6 Automatic Exchange of Certificates

The following test scenario has to be tested twice. After the scenario has been successfully tested from test partner A to partner B the scenario has to be tested from partner B to partner A / partner C.

The certificate exchange test consists of two scenarios. In the simple one both partners have the SSID is equal to the destination/origination. In the second one partner have a different destination/origination as the SSID. Both scenarios must be tested successfully.

The content types (CAD data, INVOICE, ORDER) in the following test scenario are examples. No functionality is based on these types. These are only names indicating different configurations for different use cases.

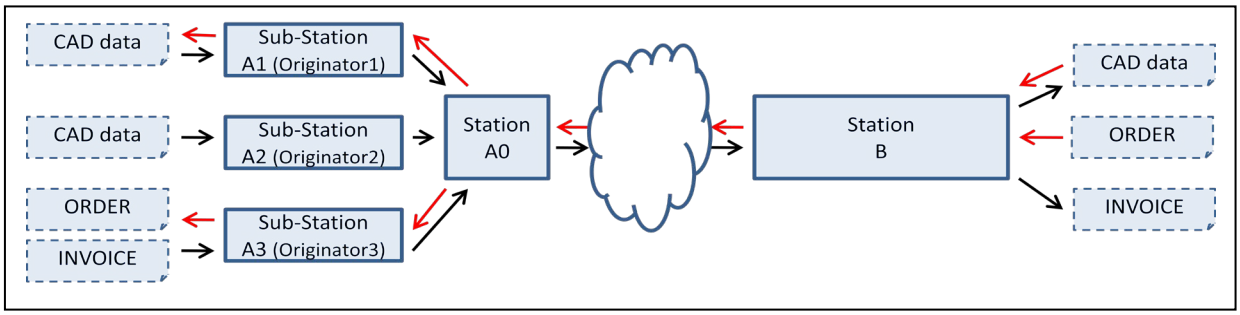
Simple test scenario:



Company A wants to send CAD data to company B and receive CAD data in the reverse direction. The destinations/originators of both partners are equal to the SSID of the station.

Extended test scenario:

In the next pictures there are entries called station und sub-station. Sub-station means that the sending/receiving points have a different SFID.destination/originator than the exchanged SSID. Station means that the SSID is the same as the SFID.destination/originator. The functionality station and sub-station can be performed in one software instance or in different instances. For an external communication partner the software/company should act in the same way.



1. Company A wants to send CAD data from station A1 to company B and station A1 and want to receive CAD data in the reverse direction. The data will be transferred via gateway/station A0.
2. Company A also wants to send CAD data from station A2 to company B.
3. Company B sends orders to station A3.
4. The invoice will be send from station A3 to company B.

This scenario will not occur often in practice. However, parts of this will do. The scenario should make sure that the interplay of the software products will also work in complex environments.

The test should include:

1. Access to the TSL
2. The use of up to 5 certificates.
3. Scenarios where SSID != SFID.originator/destination.
4. Certificate replacement with binding to the CLID
5. The initial exchange of certificates.
6. The addition certificates (for roll over situations)
7. The requests of certificates
8. The validation of certificates
9. The replacement of certificates
10. The mapping of certificates to different types of documents and destinations/originators in certificate exchange scenarios, i.e. a specific certificate is only valid for a specified document / file type (e.g. invoices, CAD data, …)
11. The TSL trust hierarchy
12. The revocation of certificates via CRL

Prerequisites for all tests:

- The OFTP2 software must allow changing the Odette TSL URL to the Odette Test-TSL list. http://www.odette.org/TSL/TSL_Test.xml
- The software must allow putting the caching times for TSL access and CRL access down. For the maximum value of 15 days a test would take a long time!
- Each test partner creates a Root CA certificate and a CA certificate (OFTP2-CA-trust certificate) that is signed by the Root CA. It is possible to include one or more intermediate certificates in this chain. The OFTP2 CA certificate must follow the technical requirements of the OFTP2 policy. In addition, all CA certificates must have a valid URI-CRL, which is accessible from the Internet via HTTP for all test partners.
- The CA Root certificate, the possible intermediate CA certificates and the trusted CA certificates must be sent to the Odette for including in the TSL test list.
- At the beginning the following certificates should be created. All should be signed by the Trusted OFTP2 CA certificate, except certificate CB05 and CB06. The certificates must by compliant to the OFTP2 certificate policy.

Certificates from Company A:

CA01: TLS certificate from A0, that also supports the signing of CAD data from station A1 and for the encryption of CAD data that will be sent back to A1 and A2. Initially, it is also used for the OFTP authentication and EERP signing of station A0.

Key usage: **Digital Signature** and **Key Encipherment**

Extended key usage: **TLS server Authentication** and **TLS client Authentication**

CA02: Rollover certificate for CA01, with a later expiry date, but it should be also valid at the moment. So it has the same issuer, subject, key usage and extended key usage as CA01.

Key usage: **Digital Signature** and **Key Encipherment**

Extended key usage: **TLS server Authentication** and **TLS client Authentication**

CA03: Replacement for certificate for CA01. So it has the same issuer, subject, key usage and extended key usage as CA01.

Key usage: **Digital Signature** and **Key Encipherment**

Extended key usage: **TLS server Authentication** and **TLS client Authentication**

CA04: TLS certificate from A0, that also supports the signing of CAD data from station A1 and for the encryption of CAD data that will be sent back to A1, A2 and order encryption for A3. Initially, it is also used for the OFTP authentication and EERP signing of station A0. The subject should be distinguished from CA01.

Key usage: **Digital Signature** and **Key Encipherment**

Extended key usage: **TLS server Authentication** and **TLS client Authentication**

CA05: Signing Certificate for Invoices from station A3 with the same issuer and subject as CA04.

Key usage: **Digital Signature**

CA06: Certificate for encrypting orders which will be sent to station A3 with the same issuer and subject as CA04.

Key usage: **Key Encipherment**

CA07: Rollover certificate for CA04, with a later expiry date, but it should be also valid at the moment. So it has the same issuer, subject, key usage and extended key usage as CA04.

Key usage: **Digital Signature** and **Key Encipherment**

Extended key usage: **TLS server Authentication** and **TLS client Authentication**

CA08: Replacement for certificate for CA04. So it has the same issuer, subject, key usage and extended key usage as CA04.

Key usage: **Digital Signature** and **Key Encipherment**

Extended key usage: **TLS server Authentication** and **TLS client Authentication**

CA09: Encrypting Certificate for OFTP Authentication for station A0 sent to station B with the same subject as CA08.

Key usage: **Key Encipherment**

CA10: Signing Certificate for EERP from station A1 with a different subject as CA09.

Key usage: **Digital Signature**

CA11: Signing Certificate for file signing from station A1 with a different subject as CA10.

Key usage: **Digital Signature**

CA12: Encryption Certificate for file encryption from station A1 with the same subject and issuer as CA10.

Key usage: **Key Encipherment**

CA13: Replacement for certificate for CA04. It may have a **different issuer** and must have a **different subject**, but the **same** key usage and extended key usage as CA04.

Key usage: **Digital Signature** and **Key Encipherment**

Extended key usage: **TLS server Authentication** and **TLS client Authentication**

CA14: Replacement for certificate for CA13. It may have a **different issuer** and must have a **different subject**, but the **same** key usage and extended key usage as CA13.

Key usage: **Digital Signature** and **Key Encipherment**

Extended key usage: **TLS server Authentication** and **TLS client Authentication**

Certificates from Company B:

CB01: TLS certificate from B, that also supports the signing of CAD data from station B and for the encryption of CAD data that will be sent back to B. Initially, it is also used for the OFTP authentication and EERP signing of station B.

Key usage: **Digital Signature** and **Key Encipherment**

Extended key usage: **TLS server Authentication** and **TLS client Authentication**

CB04: Replacement for certificate for CB01. So it has the same issuer, subject, key usage and extended key usage as CB01.

Key usage: **Digital Signature** and **Key Encipherment**

Extended key usage: **TLS server Authentication** and **TLS client Authentication**

CB05: A new certificate. The issuer is the root CA, subject is the same as CB01, key usage and extended key usage as CB01.

Key usage: **Digital Signature** and **Key Encipherment**

Extended key usage: **TLS server Authentication** and **TLS client Authentication**

CB06: Certificate from a CA, who is not on the Odette OFTP2. It has the same subject, key usage and extended key usage as CB01, but another issuer.

Key usage: **Digital Signature** and **Key Encipherment**

Extended key usage: **TLS server Authentication** and **TLS client Authentication**

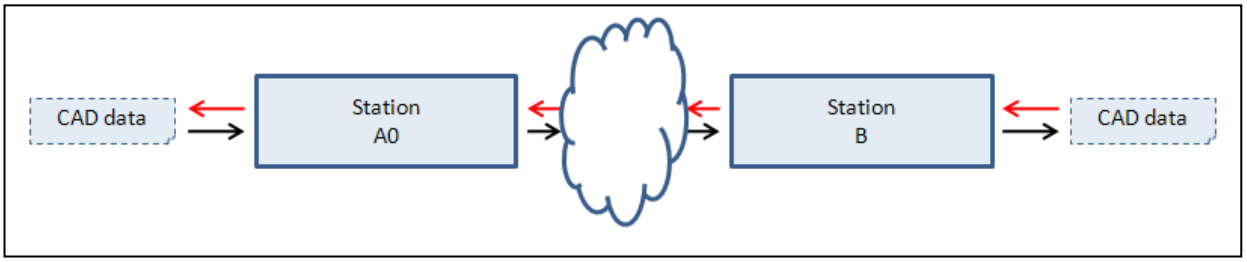
- The values of issuer, subject, key usage and extended key usage are unique for each certificate.
- The Test-TSL from Odette contains the root certificates of Partner A and B.
- The productive TSL from Odette does not contain the root certificates of Partner A and B.
- Each certificate contains a link to an existing CRL.
- The partners are authenticated by the OFTP2 certificate authentication.

4.6.1 Odette TSL access

Test Case 6.1	**Odette TSL access and import of CA certificates**		
Date:	Version:	Test File: U	Test passed (Y/N):
Prerequisite: Configure OFTP2 to have access to the Odette Test TSL List: http://www.odette.org/TSL/TSL_Test.xml			
Procedure: The Oftp2 Software must import the Test CA Certificates. This can be triggered by time or by a user.			
Expected Result: The Odette TSL Certificates are imported in the local Certificate store and can be used for verification.			
Real Result:			

4.6.2 Simple test scenario (SSID equals SFID.destination/originator)

See description 4.6



4.6.2.1 Initial Certificate Exchange via OFTP2 TLS with ODETTE_CERTIFICATE_DELIVER

Test Case 6.2.1	**Initial certificate exchange via OFTP2 TLS with an ODETTE_CERTIFICATE_DELIVER**		
Date:	Version:	Test File: U	Test passed (Y/N):

Prerequisite:

Test case 6.1. and Initial Exchange of OFTP2 parameters (see the OFTP PARAMETERS datasheet in the IGL) for the CAD data exchange between A0 and B in both directions with one Certificate CA01/CB01 for all (TLS, OFTP authentication, EERP signing, file encryption and file signing).

Procedure:

SFID options for the ODETTE_CERTIFICATE_DELIVER from A0 to B:

```
SFIDORIG  = {A0}
SFIDDEST  = {B}
Content   = {CA01}
```

SFID options for the ODETTE_CERTIFICATE_DELIVER from B to A0:

```
SFIDORIG  = {B}
SFIDDEST  = {A0}
Content   = {CB01}
```

After this session the OFTP authentication can be used.

Test partner A0 sends a CAD file to B and B send this file back to A0 with the following:

SFID options for data exchange:

```
SFIDDSN   = {CADDATA}
SFIDORIG  = {A0}
SFIDDEST  = {B}
SFIDSEC   = 03
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

SFID for the way back:

```
SFIDDSN   = {CADDATA}
```

```
SFIDORIG  = {B}
SFIDDEST  = {A0}
SFIDSEC   = 03
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

Expected Result:

A session can be established without exchanging certificates before.
The Partners can be authenticated (via Odette TSL CAs).
After the ODETTE_CERTIFICATE_DELIVER from B to A0 the OFTP authentication can be used.
The test partners can successfully transmit the files. The file and the EERP signatures can be successfully verified.

Real Result:

4.6.2.2 Initial Certificate Exchange initiated by an ODETTE_CERTIFICATE_REQUEST

Test Case 6.2.2	**Initial Exchange of certificates with ODETTE_CERTIFICATE_REQUEST and ODETTE_CERTIFICATE_DELIVER**		
Date:	Version:	Test File: U	Test passed (Y/N):

Prerequisite:

Successfully finish **Test Case 6.1**
Exchange of OFTP2 parameters for the CAD data like Test Case 6.2.1. **Remove the exchanged certificates from test case 6.2.1**. One Certificate CA01/CB01 is used for all (TLS, OFTP authentication, EERP signing, file encryption and file signing) like test case 6.2.1.

Procedure:

SFID options for the ODETTE_CERTIFICATE_REQUEST from B to A0:

```
SFIDORIG  = {B}
SFIDDEST  = {A0}
Content   = {CB01}
```

SFID options for the ODETTE_CERTIFICATE_DELIVER from A0 to B:

```
SFIDORIG  = {A0}
SFIDDEST  = {B}
Content   = {CA01}
```

After this session the OFTP authentication can be used.

Test partner A0 sends a CAD file to B and B send this file back to A0 with the following:

SFID options for data exchange:

```
SFIDDSN   = {CADDATA}
SFIDORIG  = {A0}
SFIDDEST  = {B}
SFIDSEC   = 03
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

SFID for the way back:

```
SFIDDSN   = {CADDATA}
SFIDORIG  = {B}
SFIDDEST  = {A0}
SFIDSEC   = 03
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

Expected Result:

A session can be established without exchanging certificates before.
The Partners can be authenticated (via Odette TSL CAs).
After the ODETTE_CERTIFICATE_DELIVER from B to A0 the OFTP authentication can be used.
The test partners can successfully transmit the files. The file and the EERP signatures can be successfully verified.

Real Result:

4.6.2.3 Add a new certificate in a rollover situation

Test Case 6.2.3	**Add the new certificate CA02 as a future replacement for certificate CA01 (rollover) for signing CAD data from A0 to B, encrypting CAD data from B to A0, EERP signing of A0 and OFTP Authentication and TLS of A0. So when the CA01 expired CA02 can be used and the data exchange can go on.**		
Date:	Version:	Test File: U	Test passed (Y/N):

Prerequisite:

Successfully finish **Test Case 6.2**

Procedure:

A0 sends the Certificate CA02 with an **ODETTE_CERTIFICATE_DELIVER** to B.

SFID options for the ODETTE_CERTIFICATE_DELIVER:

```
SFIDORIG  = {A0}
SFIDDEST  = {B}
Content   = {CA02}
```

A0 send a CAD data file to B with the following SFID options:

```
SFIDDSN   = {CADDATA}
SFIDORIG  = {A0}
SFIDDEST  = {B}
SFIDSEC   = 03
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

B send the same CAD file back to A0 with the following SFID options:

```
SFIDDSN   = {CADDATA}
SFIDORIG  = {B}
SFIDDEST  = {A0}
SFIDSEC   = 03
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

Expected Result:

The Certificate CA02 is added as a possible certificate to the configuration of station B for verifying signed CAD data and encrypt CAD data which are sent to A0. Also the EERP signing, OFTP Authentication and TLS certificate of A0 is changed.
B must successfully verify the signature. Both certificates CA01 and CA02 must be accepted to sign the data, as long as they are valid.
A0 can do the same with the CAD data from B (decrypt).

Real Result:

4.6.2.4 Replace a compromised certificate with an ODETTE_CERTIFICATE_REPLACE

Test Case 6.2.4	**Compromised certificate will be replaced with an ODETTE_CERTIFICATE_REPLACE**		
Date:	Version:	Test File: U	Test passed (Y/N):

Prerequisite:

Successfully finish **Test Case 6.2.3**

Procedure:

The certificate CA02 is compromised. A0 wants to change it against CA03 and send an **ODETTE_CERTIFICATE_REPLACE** to B.

SFID options for the ODETTE_CERTIFICATE_REPLACE from A0 to B:

```
SFIDORIG  = {A0}
SFIDDEST  = {B}
Content   = {CA03}
```

Test the successful replace by sending CAD data from A0 to B and back to A0.

SFID options for this:

```
SFIDDSN   = {CADDATA}
SFIDORIG  = {A0}
SFIDDEST  = {B}
SFIDSEC   = 03
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

SFID options direction back:

```
SFIDDSN   = {CADDATA}
SFIDORIG  = {B}
SFIDDEST  = {A0}
SFIDSEC   = 03
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

Expected Result:

The certificate is replaced. The old certificates CA02 and also CA01 (same identification data) can't be used any longer.
The files can be successfully exchanged.

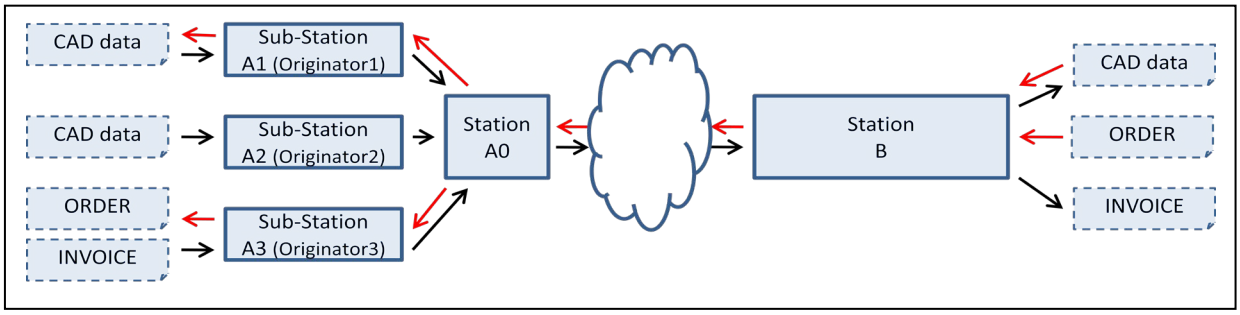
Real Result:

4.6.3 Extended test scenario (SSID unequal SFID.destination/originator)

See description 4.6

Remove the configuration from chapter 4.6.2. and start from scratch.

Extended test scenario:



4.6.3.1 Initial Certificate Exchange via OFTP2 TLS with ODETTE_CERTIFICATE_DELIVER

Test Case 6.3.1	**Initial certificate exchange via OFTP2 TLS with an ODETTE_CERTIFICATE_DELIVER**		
Date:	Version:	Test File: U	Test passed (Y/N):

Prerequisite:

Test case 6.1. and Initial Exchange of OFTP2 parameters (see the OFTP PARAMETERS datasheet in the IGL) for the CAD data exchange between A1 and B in both directions with one Certificate CA04/CB01 for all (TLS, OFTP authentication, EERP signing, file encryption and file signing).

Procedure:

SFID options for the ODETTE_CERTIFICATE_DELIVER from A0 to B:

```
SFIDORIG  = {A0}
SFIDDEST  = {B}
Content   = {CA04}
```

SFID options for the ODETTE_CERTIFICATE_DELIVER from A1 to B:

```
SFIDORIG  = {A1}
SFIDDEST  = {B}
Content   = {CA04}
```

SFID options for the ODETTE_CERTIFICATE_DELIVER from B to A0:

```
SFIDORIG  = {B}
SFIDDEST  = {A0}
Content   = {CB01}
```

After this session the OFTP authentication can be used.

SFID options for the ODETTE_CERTIFICATE_DELIVER from B to A1:

```
SFIDORIG  = {B}
SFIDDEST  = {A1}
```

```
Content   = {CB01}
```

Test partner A1 sends a CAD file to B and B send this file back to A1 with following

SFID options for data exchange:

```
SFIDDSN   = {CADDATA}
SFIDORIG  = {A1}
SFIDDEST  = {B}
SFIDSEC   = 03
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

SFID for the way back:

```
SFIDDSN   = {CADDATA}
SFIDORIG  = {B}
SFIDDEST  = {A1}
SFIDSEC   = 03
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

Expected Result:

A session can be established without exchanging certificates before.
The Partners can be authenticated (via Odette TSL CAs).
After the ODETTE_CERTIFICATE_DELIVER from B to A0 the OFTP authentication can be used.
The test partners can successfully transmit the files. The file and the EERP signatures can be successfully verified.

Real Result:

4.6.3.2 Exchange Certificates initiated by an ODETTE_CERTIFICATE_REQUEST

Test Case 6.3.2	**Exchange of certificates with ODETTE_CERTIFICATE_REQUEST and ODETTE_CERTIFICATE_DELIVER**		
Date:	Version:	Test File: U	Test passed (Y/N):

Prerequisite:

Successfully finish **Test Case 6.1**
Exchange of OFTP2 parameters for the CAD data exchange between A2 and B (Certificate CA04/CB01 for all TLS, OFTP authentication, EERP signing, file encryption and file signing).

Procedure:

B sends an **ODETTE_CERTIFICATE_REQUEST** to A2 and A2 answers with an **ODETTE_CERTIFICATE_DELIVER** transferring certificate CA04 to station B.
After this A2 sends an **ODETTE_CERTIFICATE_REQUEST** to B and B sends also his certificate with an **ODETTE_CERTIFICATE_DELIVER** which includes the CB01.

SFID options for the ODETTE_CERTIFICATE_REQUEST from B to A2:

```
SFIDORIG  = {B}
SFIDDEST  = {A2}
Content   = {CB01}
```

SFID options for the ODETTE_CERTIFICATE_DELIVER from A2 to B:

```
SFIDORIG  = {A2}
SFIDDEST  = {B}
Content   = {CA04}
```

After that we test the successful certificate exchange and binding.
The data can be transferred from A2 to B with following SFID options:

SFID options for data exchange:

```
SFIDDSN   = {CADDATA}
SFIDORIG  = {A2}
SFIDDEST  = {B}
SFIDSEC   = 03
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

Expected Result:

The test partner B can successfully receive the file, extract it and successfully verify the signature. Also Station A2 receive the EERP can successfully verify the EERP.

Real Result:

4.6.3.3 Using of different file certificates at one station

Test Case 6.3.3	**Using of different file certificates at one station. Request certificates by an ODETTE_CERTIFICATE_REQUEST and answers with ODETTE_CERTIFICATE_DELIVER.**		
Date:	Version:	Test File: U	Test passed (Y/N):

Prerequisite:

Successfully finished **Test Case 6.3.2**
Exchange of OFTP2 parameters for INVOICE and ORDER exchange between A3 and B.
Used certificates: TLS (CA04/CB01), OFTP authentication (CA04/CB01), EERP signing (CA04/CB01), file signing for invoice (CA05/---) and file encryption for order (CA06/---)

Procedure:

A2 and B request certificates with an **ODETTE_CERTIFICATE_REQUEST** and get answers with **ODETTE_CERTIFICATE_DELIVER** for certificates CA04/CB01, CA05/CB02 and CA06/CB03.

Note: each **ODETTE_CERTIFICATE_DELIVER** contains one and only one certificate.

SFID options for the ODETTE_CERTIFICATE_REQUEST from B to A3:
```
SFIDORIG  = {B}
SFIDDEST  = {A3}
Content   = {CB01}
```

SFID options for the ODETTE_CERTIFICATE_DELIVER from A3 to B:
```
SFIDORIG  = {A3}
SFIDDEST  = {B}
Content   = {CA04}
Next ODETTE_CERTIFICATE_DELIVER
SFIDORIG  = {A3}
SFIDDEST  = {B}
Content   = {CA05}
Next ODETTE_CERTIFICATE_DELIVER
SFIDORIG  = {A3}
SFIDDEST  = {B}
Content   = {CA06}
```

SFID options for the ODETTE_CERTIFICATE_REQUEST from A3 to B:
```
SFIDORIG  = {A3}
SFIDDEST  = {B}
Content   = {CA06}      (CA04 and CA05 are also possible with separate
ODETTE_CERTIFICATE_DELIVER)
```

SFID options for the ODETTE_CERTIFICATE_DELIVER from B to A3:
```
SFIDORIG  = {B}
SFIDDEST  = {A3}
Content   = {CB01}
```

SFID options for order exchange (encrypted):
```
SFIDDSN   = {order}
SFIDORIG  = {B}
```

```
SFIDDEST  = {A3}
SFIDSEC   = 01
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

SFID options for invoice exchange (signed):

```
SFIDDSN   = {invoice}
SFIDORIG  = {A3}
SFIDDEST  = {B}
SFIDSEC   = 02
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

Expected Result:

The test partner A3 can successfully receive the order from B and encrypt it successfully. Also Station B receives the EERP to this order and can successfully verify the EERP.
And in the other way:
The test partner B can successfully receive the invoice from A3 and successfully verify the signature. Also Station A receives the EERP to this invoice and can successfully verify the EERP.

Real Result:

4.6.3.4 Add a new certificate in a rollover situation

Test Case 6.3.4	**Add the new certificate CA07 as a future replacement for certificate CA04 (rollover) for signing CAD data from A1 to B, encrypting CAD data from B to A1 and EERP signing of A1.** **So when the CA04 expired CA07 can be used and the data exchange can go on.**		
Date:	Version:	Test File: U	Test passed (Y/N):

Prerequisite:

Successfully finish **Test Case 6.3.1**

Procedure:

A1 sends the Certificate CA07 with an **ODETTE_CERTIFICATE_DELIVER** to B.

SFID options for the ODETTE_CERTIFICATE_DELIVER:

```
SFIDORIG  = {A1}
SFIDDEST  = {B}
Content   = {CA07}
```

A1 send a CAD data file to B with the following SFID options:

```
SFIDDSN   = {CADDATA}
SFIDORIG  = {A1}
SFIDDEST  = {B}
SFIDSEC   = 03
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

B send the same CAD file back to A1 with the following SFID options:

```
SFIDDSN   = {CADDATA}
SFIDORIG  = {B}
SFIDDEST  = {A1}
SFIDSEC   = 03
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

Expected Result:

The Certificate CA07 is added as a possible certificate to the configuration of station B for verifying signed CAD data and encrypt CAD data which are sent by/to A1. Also the EERP signing certificate of A1 is changed.
All other configuration/certificates of B should not be changed. This relates especially to the TLS and the OFTP Auth. certificates.
B must successfully verify the signature. Both certificates CA04 and CA07 must be accepted to sign the data, as long as they are valid.
A1 can do the same with the CAD data from B (decrypt).

Note:
If stations A2 and A3 used the same certificate CA04, the change applies also to station A2 and A3.
If any station wants to use a certificate independently of other local stations, it must use a different (individual) certificate.

Real Result:

4.6.3.5 Test deleted

4.6.3.6 Replace a compromised certificate with an ODETTE_CERTIFICATE_REPLACE

Test Case 6.3.6	**Compromised certificate will be replaced with an ODETTE_CERTIFICATE_REPLACE**		
Date:	Version:	Test File: U	Test passed (Y/N):

Prerequisite:

Successfully finish **Test Case 6.3.5**

Procedure:

The certificate CA07 is compromised. A0 and A1 want to change it against CA08 and send an **ODETTE_CERTIFICATE_REPLACE** to B. (Normally all station, that are using the CA07, send this deliver, but this is not explained here)

SFID options for the ODETTE_CERTIFICATE_REPLACE from A0 to B:

```
SFIDORIG  = {A0}
SFIDDEST  = {B}
Content   = {CA08}
```

SFID options for the ODETTE_CERTIFICATE_REPLACE from A1 to B:

```
SFIDORIG  = {A1}
SFIDDEST  = {B}
Content   = {CA08}
```

Other ODETTE_CERTIFICATE_REPLACE for A2 and A3 possible.

Test the successful replace by sending CAD data from A1 to B and back to A1.

SFID options for this:

```
SFIDDSN   = {CADDATA}
SFIDORIG  = {A1}
SFIDDEST  = {B}
SFIDSEC   = 03
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

SFID options direction back:

```
SFIDDSN   = {CADDATA}
SFIDORIG  = {B}
SFIDDEST  = {A1}
SFIDSEC   = 03
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

Expected Result: The certificates are replaced. The old certificates CA07 and also CA04 (same identification data) can't be used any longer in the involved configurations (A0+A1). The files can be successfully exchanged.
Real Result:

4.6.3.7 Use up to 5 certificate types

Test Case 6.3.7	**Use up to 5 certificate types**		
Date:	Version:	Test File: U	Test passed (Y/N):
Prerequisite: Station A0 replaces the certificate for OFTP authentication to certificate CA09, A1 replaces the certificate for EERP signing to CA10, A1 replaces the certificate for file signing to CA11 and A1 replaces the certificate for file encryption to CA12. Company B must be informed and change the configuration for Partner A0 and A1 in the same way (other CLIDs).			

Procedure:
 	
A0 sends the Certificate CA09 with an **ODETTE_CERTIFICATE_DELIVER** to B.
 	
SFID options for the ODETTE_CERTIFICATE_DELIVER:
`SFIDORIG = {A0}`
`SFIDDEST = {B}`
`Content = {CA09}`
 	
A1 sends the Certificate CA10, CA11, CA12 with an **ODETTE_CERTIFICATE_DELIVER** to B.
 	
SFID options for the ODETTE_CERTIFICATE_DELIVER:
`SFIDORIG = {A1}`
`SFIDDEST = {B}`
`Content = {CA10}`
 	
SFID options for the ODETTE_CERTIFICATE_DELIVER:
`SFIDORIG = {A1}`
`SFIDDEST = {B}`
`Content = {CA11}`
 	
SFID options for the ODETTE_CERTIFICATE_DELIVER:
`SFIDORIG = {A1}`
`SFIDDEST = {B}`
`Content = {CA12}`			

Station B sends CAD data to A1 and sends the data back to B.

SFID options for CADDATA exchange from B to A1:

```
SFIDDSN   = {CADDATA}
SFIDORIG  = {B}
SFIDDEST  = {A1}
SFIDSEC   = 03
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

SFID options for CADDATA exchange from A1 to B:

```
SFIDDSN   = {CADDATA}
SFIDORIG  = {A1}
SFIDDEST  = {B}
SFIDSEC   = 03
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

Expected Result:

The test partner A1 can successfully receive the CAD data from B, extract it and successfully verify the signature. Also Station B receives the EERP to this and can successfully verify the EERP.
And in the other way:
The test partner B can successfully receive the CAD data from A1, extract it and successfully verify the signature. Also Station A receives the EERP to this data and can successfully verify the EERP.

Real Result:

4.6.3.8 Add a new certificate in a rollover situation including SFIDDESC

Test Case 6.3.8	**Add the new certificate CA13 as a future replacement for certificate CA04 (rollover) for signing CAD data from A2 to B, encrypting CAD data from B to A2, EERP signing of A2. So when the CA04 expired CA13 can be used and the data exchange can go on.**		
Date:	Version:	Test File: U	Test passed (Y/N):

Prerequisite:

Successfully finish **Test Case 6.3.2**

Procedure:

A2 sends the Certificate CA13 with an **ODETTE_CERTIFICATE_DELIVER** to B.

SFID options for the ODETTE_CERTIFICATE_DELIVER:

```
SFIDORIG  = {A2}
SFIDDEST  = {B}
SFIDDESC  = {999 bytes (UTF-8): add the Issuer (m), serial number (m),
authority key identifier (m), basic key usage (m), extended key usage (m),
subject (m) up to 999 bytes of the certificate CA04}
Content   = {CA13}
```

A2 send a CAD data file to B with the following SFID options:

```
SFIDDSN   = {CADDATA}
SFIDORIG  = {A2}
SFIDDEST  = {B}
SFIDSEC   = 03
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

B send the same CAD file back to A2 with the following SFID options:

```
SFIDDSN   = {CADDATA}
SFIDORIG  = {B}
SFIDDEST  = {A2}
SFIDSEC   = 03
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

Expected Result: The Certificate CA13 is added as a possible certificate to the configuration of station B for verifying signed CAD data and encrypt CAD data which are sent to A2. Also the EERP signing of A2 is changed. B must successfully verify the signature. Both certificates CA04 and CA13 must be accepted to sign the data, as long as they are valid**.** A2 can do the same with the CAD data from B (decrypt).
Real Result:

4.6.3.9 Replace a compromised certificate with an ODETTE_CERTIFICATE_REPLACE including SFIDDESC

Test Case 6.3.9	**Compromised certificate will be replaced with an ODETTE_CERTIFICATE_REPLACE**		
Date:	Version:	Test File: U	Test passed (Y/N):

Prerequisite:

Successfully finish **Test Case 6.3.8**

Procedure:

The certificate CA13 is compromised. A2 wants to change it against CA14 and send an **ODETTE_CERTIFICATE_REPLACE** to B.

SFID options for the ODETTE_CERTIFICATE_REPLACE from A2 to B:

```
SFIDORIG  = {A2}
SFIDDEST  = {B}
SFIDDESC  = {999 bytes (UTF-8): add the Issuer (m), serial number (m), authority key identifier (m), basic key usage (m), extended key usage (m), subject (m) up to 999 bytes of the certificate CA13}
Content   = {CA14}
```

Test the successful replace by sending CAD data from A2 to B and back to A2.

SFID options for this:

```
SFIDDSN   = {CADDATA}
SFIDORIG  = {A2}
SFIDDEST  = {B}
SFIDSEC   = 03
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

SFID options direction back:

```
SFIDDSN   = {CADDATA}
SFIDORIG  = {B}
SFIDDEST  = {A2}
SFIDSEC   = 03
```

SFIDCIPH = 01 SFIDCOMP = 1 SFIDENV = 1 SFIDSIGN = Y
Expected Result: The certificate is replaced. The old certificates CA13 can't be used any longer. The files can be successfully exchanged.
Real Result:

4.6.4 TLS functionality

4.6.4.1 Deny unsecure Diffie-Hellman key exchange

Test Case 6.4.1	**Deny unsecure Diffie-Hellman key exchange (with keys less than 1024bits)**		
Date:	Version:	Test File: N/A	Test passed (Y/N):
Note: Odette provides a server with keys less than 1024. Please request server details to info@odette.org.			
Prerequisites: Successfully finished test 6.3.9			
Procedure: Initiate a TLS 1.x OFTP2 session to the test server with following SSID options: SSIDLEV = 5 SSIDSDEB = 2048 SSIDSR = B SSIDCMPR = N SSIDREST = Y SSIDCRED = 99 SSIDAUTH = N SSIDCODE = REMOTE SSIDPSWD = REMOTE If the TLS and OFTP2 handshake succeeds, the server responds with the following SSID options: SSIDLEV = 5 SSIDSDEB = 2048 SSIDSR = B SSIDCMPR = N SSIDREST = Y SSIDCRED = 99 SSIDAUTH = N SSIDCODE = LOCAL SSIDPSWD = LOCAL			

Expected Result: - A TLS connection cannot be established. - The TLS client must refuse to use this TLS server due to the very small DH key size - Optionally, in addition: The TLS server receives a TLS error message during session takedown
Real Result:

4.6.4.2 Validation of TLS client certification functionality

Test Case 6.4.2	**Validation of TLS client certificate functionality**		
Date:	Version:	Test File:N/A	Test passed (Y/N):
Prerequisite: Successfully finish test 4.6.3.9			
Procedure: Connect to the remote partner using a certificate with the X509v3 extended key usage “TLS web client authentification” as a TLS client certificate (i.e. CA01) Connect to the remote partner using a certificate without the X509v3 extended key usage “TLS web client authentification” as a TLS client certificate (i.e. CA05 or CA06) Connect to the remote partner without a TLS client certificate. The TLS server must be configured to require a TLS client certificate. Connect to the remote partner without a TLS client certificate. The TLS server must be configured to not require a TLS client certificate.			
Expected Result: The client must be able to authenticate itself in a TLS connection. The TLS server verifies the correctness of the TLS client certificate. The certificate must include client authentication in the extended key usage. If not a warning/error message must occur. The remote TLS server must verify the correctness of this certificate and close the TLS session. Optionally, the TLS servers send a TLS closure alert number 43 (“unsupported certificate”). The TLS server must close the TLS session due to the fact that no TLS client certificate is being transmitted. Optionally, the TLS servers send a TLS closure alert number 41 (“no certificate”). The TLS server must allow the TLS session without a TLS client certificate.			
Real Result:			

4.6.4.3 Rejection of an expired TLS certificate [added]

Test Case 6.4.3	**Rejection of an expired TLS certificate**		
Date:	Version:	Test File:N/A	Test passed (Y/N):
Prerequisite: Client authentication is switched on on the TLS server.			
Procedure: Connect to the remote party with a TLS client certificate with a “not after” date in the past (certificate is not valid anymore). Connect to the remote party with a non-expired TLS client certificate. The TLS server uses an TLS server certificate with a “not after” date in the past (certificate is not valid anymore).			
Expected Result: The TLS server identifies that the certificate has expired and closes the TLS session. Optionally, the TLS servers send a TLS closure alert number 45 (“certificate expired”). The TLS client identifies that the certificate has expired and closes the TLS session. Optionally, the TLS client send a TLS closure alert number 45 (“certificate expired”).			
Real Result:			

4.7 CRL access and TSL hierarchy

4.7.1 Revoke a Certificate by the issuer CRL

Test Case 7.1	**Revoke Certificate by the issuer CRL**		
Date:	Version:	Test File: U	Test passed (Y/N):

Prerequisite:

Successfully finish **Test Case 6.3.7**

Procedure:

The certificate CA12 is revoked by the issuing CA.
Do the following data exchange continuously until B reads the CRL!
After this, B can't encrypt the order, because there is no valid certificate.

SFID options for the data exchange:

```
SFIDDSN   = {CADDATA}
SFIDORIG  = {B}
SFIDDEST  = {A1}
SFIDSEC   = 03
SFIDCIPH  = 01
SFIDCOMP  = 1
SFIDENV   = 1
SFIDSIGN  = Y
```

Expected Result:

The certificates CA12 can't be used any longer.
The CAD data files to A1 can't be exchanged, because the encryption certificate is no longer a trusted certificate.
Even if the sender would use the revoked certificate to encrypt the file, on the receiver's side the file should be refused (NERPREAS 33: file decryption failed), because a revoked certificate has been used.

Real Result:

4.7.2 Try to use a certificate, signed by a CA that is not in the Odette Test TSL

Test Case 7.2	**B wants to use CB06 as a TLS trusted certificate.**		
Date:	Version:	Test File: U	Test passed (Y/N):
Prerequisite: Successfully finish **Test Case 6.3.7.** It is possible that Company B must put the CA Certificates of certificate CB06 to their key store to use CB06.			
Procedure: B tries to establish a session to A0.			
Expected Result: The session cannot be established, because the certificate CB06 is not a valid Odette OFTP2 certificate. CB06 is not valid because the issuer is not an allowed OFTP CA issuer.			
Real Result:			

4.7.3 Try to use an non-trusted Odette OFTP2 CA (from an intermediate CA)

Test Case 7.3	**B wants to use CB05 as a TLS certificate.**		
Date:	Version:	Test File: U	Test passed (Y/N):
Prerequisite: Successfully finish **Test Case 6.3.8**			
Procedure: B tries to establish a session to A0.			
Expected Result: The session cannot be established, because the certificate CB05 is not a valid Odette OFTP2 certificate. CB05 is not valid because the issuer (root CA) is not an allowed OFTP CA issuer. The root CA certificate is only for verification reason in the TSL list.			
Real Result:			

4.7.4 Try to use a certificate, which issuer is removed from Odette OFTP2 TSL

Test Case 7.4	**Issuer CB05 is removed from the Odette TSL Test list.** **B wants to use CB01 as a TLS certificate.**		
Date:	Version:	Test File: U	Test passed (Y/N):
Prerequisite: Successfully finish **Test Case 6.3.7**			
Procedure: A establish a session to B. This procedure is continued until A reads the TSL list the next time. Then the session creation should fail.			
Expected Result: After A reads the TSL, the session cannot be established, because the TLS certificate CB01 is no longer a valid Odette OFTP2 certificate.			
Real Result:			