CSFXFER – Transfer Encryption keys to a foreign CKDS

1. Purpose

When encrypted data needs to be transferred from one sysplex to another, and those sysplexes have differing master keys, we need to transfer the encryption key. This can be accomplished using a common transport key.

2. Method

A common transport key will be defined in both the source CKDS and the target CKDS. The source will need the EXPORTER key, and the target will need the IMPORTER key. The clear key values of these two keys will be identical.

At the source site, the key under which the data is encrypted is EXPORTED so that it is encrypted under the EXPORTER key. It is stored in a file, which is transported to the target site. There the key is IMPORTED using the corresponding IMPORTER key.

This program is designed to work with the following types of keys.

CIPHER, CIPHERXI, CIPHERXL, CIPHERXO, DATA, DATAC, DATAM, DATAMV, DATAXLAT, DECIPHER, ENCIPHER, EXPORTER, IKEYXLAT, IMPORTER, IPINENC, MAC, MACD, MACVER, OKEYXLAT, OPINENC, PINGEN, PINVER.

3. How to use CSFXFER

CSFXFER is easy to use for those who have familiarity with using JCL to run REXX programs.

3.1 JCL Examples

CSFXFER is a REXX program which can be executed using the following JCL examples.

```
Example 1: EXPORTing keys
//EXPORT   EXEC  PGM=IKJEFT01,PARM='%CSFXFER'
//SYSPROC   DD   DISP=SHR,DSN=your.clist.library
//SYSTSPRT  DD   SYSOUT=*
//SYSTSIN   DD   DUMMY
//REPORT    DD   SYSOUT=*
//CONTROL   DD   *
EXPORTER TSGLB.TEST1.EXPORTER
EXPORT   LENNIE.TEST1.DESKEY01  TYPE DATA
EXPORT   LENNIE.TEST1.DESKEY02  TYPE DATA
//KEYFILE   DD   DISP=OLD,DSN=your.keyfile,
//          UNIT=SYSALLDA,SPACE=(TRK,(1,1)),
//          DCB=(LRECL=296,BLKSIZE=0,RECFM=FB)
```

```
Example 2: IMPORTing keys
//IMPORT   EXEC  PGM=IKJEFT01,PARM='%CSFXFER'
//SYSPROC   DD   DISP=SHR,DSN= your.clist.library
//SYSTSPRT  DD   SYSOUT=*
//SYSTSIN   DD   DUMMY
//REPORT    DD   SYSOUT=*
//CONTROL   DD   *
IMPORTER TSGLB.TEST1.IMPORTER
```

```
IMPORT   LENNIE.TEST1.DESKEY01  TYPE DATA +
 NEWNAME TSGLB.TEST1.DESKEY01
IMPORT   LENNIE.TEST1.DESKEY02  TYPE DATA +
 NEWNAME TSGLB.TEST1.DESKEY02
//KEYFILE   DD   DISP=OLD,DSN= your.keyfile
```

```
Example 3: PRINT the keyfile
//REPORT   EXEC  PGM=IKJEFT01,PARM='%CSFXFER'
//SYSPROC   DD   DISP=SHR,DSN= your.clist.library
//SYSTSPRT  DD   SYSOUT=*
//SYSTSIN   DD   DUMMY
//REPORT    DD   SYSOUT=*
//CONTROL   DD   *
 REPORT
//KEYFILE   DD   DISP=OLD,DSN= your.keyfile
```

3.2 DDnames

The DDnames used are the normal ones for execution under IKJEFT01 (i.e. SYSTSPRT and SYSTSIN) and CONTROL, REPORT and KEYFILE.

3.2.1 CONTROL

CONTROL defines the control statements used to define what is to be done. Statements should be supplied in a card image file (LRECL=80, RECFM=FB). Each statement can be on one or two lines. If two lines are used the statement should be broken at a blank. The first of two lines should end with a "+" sign. The statements that can be used are,

Statement	***Parameters***	***Notes***
EXPORTER	*exportkeyname*	This is used to specify the EXPORTER key that has previously been defined (probably using CSFKGUP).
EXPORT	*keyname* TYPE *keytype*	This is used to specify a key to be exported under an EXPORTER key. The *keytype* should correspond to one of the key types specified above.
IMPORTER	*Importkeyname*	This is used to specify the IMPORTER key that has previously been defined (probably using CSFKGUP).
IMPORT	*Keyname* TYPE *keytype* NEWNAME *newkeyname*	This is used to specify a key to be imported using an IMPORTER key. The NEWNAME keyword is optional. If not specified the key will be stored under its original name. The *keytype* should correspond to one of the key types specified above, and should match the key type used at EXPORT time.
REPORT	<none>	Produce a report of the contents of the KEYFILE.

3.2.2 KEYFILE

KEYFILE is a sequential file that is use to hold the values of the keys which are exported. It is the transport container for the keys. This file should have the following DCB.

LRECL=296, RECFM=FB, DSORG=PS, BLKSIZE=0

KEYFILE will contain both binary and EBCDIC data. When it is moved from one site to another it should be treated as binary data.

3.2.3 REPORT

The REPORT file defines a message set which indicates what has happened during execution.

3.3 Diagnostics

CSFXFER can be run with a DIAG parameter. This will produce extra undocumented diagnostic messages.

3.4 Security

No extra security is defined for this program other than those rules in place in the two installations for ICSF keys and APIs. You may get RACF violations if you do not have the required level of access to the CSFKEYS and CSKSERV resources. The resource names of the keys will be defined locally. If you have granular access controls active you will need CONTROL access to the key being written as specified in an IMPORT operation on the KEY or NEWNAME parameters.

The only ICSF services used by these programs are as follows,

3.4.1 IMPORT

CSNBKIM, CSNBKRC, CSNBKRW

3.4.2 EXPORT

CSNBKEX

The user specified for this job will need READ access to the resources, CSFKIM, CSFKRC, CSFKRW to import keys, and CSFKEX to export keys.

4. Reports

Two reports can be produced during execution of CSFXFER.

When a key is exported the following report will be produced. This report is also used when the REPORT control statement is used to report the contents on a KEYFILE entry.

```
-------------------------------------------------------------------------------------------
Key Label          :  TSGLB.TEST1.DESKEY01
Exporter Key       :  TSGLB.TEST1.EXPORTER
Key Type           :  DATA
From System        :  RSMP
From CKDS          :  USER.CSFCKDS
Exported by job    :  TSGLBMX
Exported by Userid :  TSGLB
Exported Date      :  15 Jun 2014
Exported Time      :  20:02:34
Export Return Code :  00000000
Export Reason Code :  00000000
Key Value          :  02000000 0100C000 00000000 00000000 F7666693 83C03D26 3EE22059 1B1004D6
                   :  00000000 00000000 00000000 00000000 00000000 00000000 00000010 D81988F8
-------------------------------------------------------------------------------------------
```

For most of the fields in this report should be quite clear. The value reported as being the CKDS is the value taken from the CCVT at the time the EXPORT operation was carried out. Under exceptional

circumstances it is possible this is not the correct CKDS. However if no operation are underway to re-encrypt the CKDS while the export is carried out, then this will be the correct CKDS.

The values for the Return Code should always be zero. However it is possible that the reason code will be a non-zero value. If needed this can be looked up in the manual,

ICSF Application Programmer's Guide SC14-7508 in Appendix A.

When a key is imported the following report is produced.

```
-------------------------------------------------------------------------------------------------
Source key label   :  TSGLB.TEST1.DESKEY01
Target key label   :  TSGLB.TEST2.DESKEY01
Exporter Key       :  TSGLB.TEST1.EXPORTER
Importer Key       :  TSGLB.TEST1.IMPORTER
Key Type           :  DATA
From System        :  RSMP
To System          :  SOW1
From CKDS          :  USER.CSFCKDS
To CKDS            :  CSFUSER.CSFCKDS
Exported by job    :  TSGLBMX
Exported by Userid :  TSGLB
Exported Date      :  15 Jun 2014
Exported Time      :  20:02:34
Imported by job    :  TSGLBMX
Imported by Userid :  LENNIE
Imported Date      :  15 Jun 2014
Imported Time      :  20:02:34
Exported Key Value :  02000000 0100C000 00000000 00000000 F7666693 83C03D26 3EE22059 1B1004D6
                   :  00000000 00000000 00000000 00000000 00000000 00000000 00000010 D81988F8
Imported Key Value :  01000000 0100C000 1B6B4EE3 024054B8 544A9A35 14FB872D 82198ACB C73A622F
                   :  00000000 00000000 00000000 00000000 00000000 00000000 00000010 D2467207
-------------------------------------------------------------------------------------------------
```

Again, most of these values should be self-explanatory. The target CKDS specification is subject to the same constraints as the source CKDS, as noted above.

5. Creating the IMPORTER and EXPORTER keys

The following example JCL can be used to create the transport keys. The values to use for the keys can be obtained through the random number generation process of the ICSF panels. The job below would need to be run on both the source and target systems. This would allow key transfer in both directions. Alternatively the IMPORTER key could be defined only at the target site and the EXPORTER key only at the source site. This would allow one-way transport only. The important point is that the key parts would need to be the same in all cases. Transport or agreement of those key parts is outside the scope of this document.

Once in the ICSF main panel, choose option 5, "UTILITY" and the select option 3, "RANDOM". Once in that panel select ODD parity numbers.

```
//KGUPPROC EXEC PGM=CSFKGUP,PARM=('SSM')
//CSFCKDS  DD   DSN=USER.CSFCKDS,DISP=OLD
//CSFIN    DD   *
  ADD    LABEL(TSGLB.TEST1.EXPORTER) TYPE(EXPORTER) CLEAR,
  KEY(2A1AE0AB8089E9BF,91CE61A2BA264FB5)
  ADD    LABEL(TSGLB.TEST1.IMPORTER) TYPE(IMPORTER) CLEAR,
  KEY(2A1AE0AB8089E9BF,91CE61A2BA264FB5)
//CSFDIAG  DD   SYSOUT=*                          FB,133
//CSFKEYS  DD   DSN=userid.ICSF.KEYS,DISP=OLD  FB,208
//CSFSTMNT DD   DSN=userid.ICSF.STMT,DISP=OLD  FB,080
//*
//CSFEUTIL EXEC PGM=CSFEUTIL,PARM=('USER.CSFCKDS,REFRESH')
```

Further details of the running of the CSFKGUP utility can be found in the manual,

ICSF Administrator's Guide SA22-7521-17

6. Overview of steps to transfer keys

1. Agree key values for the IMPORTER and EXPORTER keys.
2. Create the IMPORTER and EXPORTER keys using the JCL above in section 5 above.
3. Export the keys you wish to transport using the JCL in Example 1 in section 3.1 above.
4. Transfer the KEYFILE to the target site using whatever method is appropriate, such as FTP, email, or some other appropriate transport mechanism.
5. Import the required keys using the JCL in Example 2 in section 3.1 above.

7. Installation

This program can be run from any suitable CLIST or REXX library. No APF authorisation issues exist.

The delivery materials consist of this document and two other files called,

```
K.CLIST.XMI
K.CNTL.XMI
```

These two files are binary files and must be transmitted as such if they are moved. They have LRECL=80 and RECFM=FB.

These two files should be uploaded to your z/OS system and then a TSO RECEIVE command issued against each of them, thus,

```
RECEIVE INDA(K.CLIST.XMI)
RECEIVE INDA(K.CNTL.XMI)
```

This will result is the file being re-constituted as a PDS on the z/OS system.

The contents of these two PDS libraries are as shown in the following table

Library	*Member*	*Description*
CLIST	CSFXFER	This is the REXX program which is described in this document
CLIST	DEC	Sample REXX program used in testing. Performs encryption.
CLIST	ENC	Sample REXX program used in testing. Performs decryption.
CLIST	ENCDEC	Sample REXX program used in testing. Performs encryption and decryption.
CNTL	DEC	Sample JCL to perform decryption.
CNTL	DEFKEY	Sample JCL to define IMPORTER and EXPORTER keys.
CNTL	ENC	Sample JCL to perform encryption.
CNTL	ENCDEC	Sample JCL to perform encryption and decryption.
CNTL	RUNXFER	Sample JCL to run CSFXFER EXPORT and IMPORT.
CNTL	RUNXFER2	Sample JCL to run CSFXFER IMPORT

8. Messages

The following messages may be produced during execution.

Message id	*Message Text*	*Explanation and Action*
XFER0010E	Unable to open REPORT file	Failure with the REPORT file. Examine JCL for errors.
XFER0020E	Unable to open CONTROL file	Failure with the CONTROL file. Examine JCL for errors.

Message id	*Message Text*	*Explanation and Action*
XFER0030E	Return code *ret1* reading CONTROL file	Error encountered on the CONTROL file. Look up *ret1* in the return codes from the REXX EXECIO interface
XFER0040E	*nnn* bad CONTROL data *baddata*	There is bad data supplied on a control file record. Examine the statements and correct them. The value of *nnn* is for internal diagnostic purposes. The field *baddata* contains the values which cannot be parsed.
XFER0050E	duplicate EXPORTER statement	Only one EXPORTER statement is allowed for each execution of CSFXFER.
XFER0060E	EXPORTER and IMPORTER statements	An EXPORTER statement has been found after an IMPORTER statement. Only one of EXPORTER and IMPORTER is allowed.
XFER0070E	Key label length error	An EXPORT statement has been processed. The key label specified must not be longer than 64 characters.
XFER0080E	Invalid key type *keytype*	An EXPORT statement has been processed. The value of *keytype* specified for key type is not supported.
XFER0090E	Unable to EXPORT. No EXPORTER	An EXPORT statement has been encountered before an EXPORTER statement has been found.
XFER0100E	duplicate IMPORTER statement	Only one IMPORTER statement is allowed for each execution of CSFXFER.
XFER0110E	EXPORTER and IMPORTER statements	An IMPORTER statement has been found after an EXPORTER statement. Only one of EXPORTER and IMPORTER is allowed.
XFER0120E	Key label length error	An IMPORT statement has been processed. The key label specified must not be longer than 64 characters.
XFER0130E	Invalid key type *keytype*	An IMPORT statement has been processed. The value of *keytype* specified for key type is not supported.
XFER0150E	Newname label length error	An IMPORT statement is being processed. The NEWNAME specifies a key label which is longer than 64 characters.
XFER0160E	Unable to IMPORT. Missing IMPORTER key. *key*	An IMPORT statement is being processed. It cannot be actioned as there is no IMPORTER key specified.
XFER0170E	Record not found in keyfile.	An IMPORT statement has been found but the key specified is not in the specified KEYFILE.
XFER0180E	Invalid control statement	This control statement cannot be recognised.
XFER0200I	IMPORTER key stored.	An IMPORTER statement has been successfully processed.
XFER0210I	EXPORTER key stored.	An EXPORTER statement has been successfully processed.
XFER0220I	REPORT request detected.	A REPORT request has been processed.
XFER0300E	Unable to open KEYFILE for INPUT.	During an IMPORT operation the KEYFILE cannot be processed. Examine JCL for errors.
XFER0300E	Unable to open KEYFILE for OUTPUT.	During an EXPORT operation the KEYFILE cannot be processed. Examine JCL for errors.
XFER0310E	Import failed: RC=*ret*,Reas=*reas*	An IMPORT operation (CSNBKIM) has failed with return code *ret* and reason code *reas.* Examine the return code and reason code in the manual ICSF Application Programmer's Guide SC14-7508 in Appendix A.

Message id	*Message Text*	*Explanation and Action*
XFER0320E	Record create failed: RC=*ret*,Reas=*reas*	An operation to create a CKDS record (CSNBKRC) has failed with return code *ret* and reason code *reas.* Examine the return code and reason code in the manual ICSF Application Programmer's Guide SC14-7508 in Appendix A.
XFER0330E	Duplicate key: newname	An attempt to create a record in the target CKDS has failed because a record of the same name already exists.
XFER0340E	Record write failed: RC=*ret*,Reas=*reas*	An operation to write a CKDS record (CSNBKRC) has failed with return code *ret* and reason code *reas.* Examine the return code and reason code in the manual ICSF Application Programmer's Guide SC14-7508 in Appendix A.
XFER0350E	Export failed: RC=*ret*,Reas=*reas*	An EXPORT operation (CSNBKEX) has failed with return code *ret* and reason code *reas.* Examine the return code and reason code in the manual ICSF Application Programmer's Guide SC14-7508 in Appendix A.
XFER0360E	Key not found *key*	An EXPORT operation has failed as the source key does not exist in the source CKDS.
XFER0400I	Key *key* of type *keytype* successfully imported.	An IMPORT operation has succeeded.
XFER0410I	Exporting key = *key*	An EXPORT operation is being attempted for key *key*.
XFER0900I	- End of program	The program has ended.

9. Disclaimer

The code supplied has not been subjected to any formal test and is distributed on an "AS IS" basis without any warranty either express or implied. The implementation of any of the techniques described or used herein is a customer responsibility and depends on the customers' operational environment. While each item may have been reviewed for accuracy in a specific situation and may run in a specific situation and may run in a specific environment, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Lennie Dymoke-Bradshaw, RSM Partners
19 June 2014