MULTIKEY Application

The MULTIKEY application is designed specifically to facilitate the secure entry of DATA type keys from a Trusted Key Entry work station using one or more key parts (which may imply multiple custody of these key parts). Neither ICSF nor the TKE has a facility to enter DATA type keys as key parts. In fact, the TKE does not support entry of DATA key type.

ICSF supplied panels only offer the capability to enter DATA keys in the clear as a single part via an ISPF panel. There is another 'ASIS' application addresses the ICSF panel limitation. For details go to http://www-1.ibm.com/support/techdocs/atsmastr.nsf , search All Documents using the keyword "CRYPTO", and look for "ICSF Key Part Entry Sample Panel Application".

MULTIKEY can also be used to input encrypted key values into the CKDS and to generate or verify CVV/CVC values. The application is invoked via JCL with special input control cards. Within this document all three usage types will be shown in examples. The main discussion is on how you can use the TKE for the entry of a clear 8-byte data key via TKE even though DATA keytype is not supported on the TKE Workstation.

The TKE can be used (without changes) to enter other OPERATIONAL keytypes (specifically a Key Encrypting Key of a type called EXPORTER). These keys can be entered with multiple key parts. In Version 3 of the TKE customization of the TKE.INI file must be done so the key part values are not displayed as they are entered.

The MULTIKEY application makes use of the fact that a double-length key (16-bytes, 32 characters) can emulate the cryptographic functions of a single-length key (8-bytes, 16 characters). At the TKE, for instance, a desired DATA type key can be entered in one or more parts under multiple custody, such that the 8 key bytes are replicated and concatenated to themselves thereby giving a double-length key that cryptographically produces the same output as its single-length counterpart. MULTIKEY also makes use of the fact that under a controlled process a CCA key type can be modified.

To use the MULTIKEY application program you will run a batch job specifying input data. This documentation describes all the tasks involved in order to LOAD key values for use as DATA keys via TKE. This tasks are

- entering the key parts for the data key into the TKE
- having MULTIKEY application required keys in the CKDS
- running the MULTIKEY application to transform the keys entered from the TKE to DATA keytype

Enter the DATA key parts on the TKE

As an example, assume that a DATA key (to be used for CVC/CVV generation) is to be entered with multiple key parts.
Key part one consists of the 8-byte value 0123456789ABCDEF.
Key part two consists of the 8-byte value ABCDEF0123456789.

At the TKE, the OPERATIONAL KEY LOAD function is selected, and a keytype of EXPORTER is selected. This is required because the MULTIKEY application expects a key of this type and the TKE does not allow you to specify DATA as a keytype.

FIRST KEY PART would be:

0123456789ABCDEF0123456789ABCDEF

Notice that the key part value has been duplicated to create 32 digits representing a double-length key value.

FINAL KEY PART would be:

ABCDEF0123456789ABCDEF0123456789

Bring the key parts into the CKDS

Use of ICSF panels is required to bring the key parts into the CKDS from the secure host crypto module storage. At the ICSF ISPF panel, option OPKEY is used to move the two key parts from the crypto co-processor to the CKDS (Cryptographic Key Data Set) under encryption of the Master Key. The CKDS name must be entered, as well as a name by which the resulting key can be referenced. You would not indicate on the panel for the last key part that the key is a NOCV key (one to be used without control vector).

Assume the name or key label of that the target key (the single-length EXPORTER) is specified as "EXPORTER.CVCKEYA". The actual name is not significant, but the key value must be eventually made available to the MULTIKEY application and it will use the key label to access the value.

How to Use MULTIKEY to "convert" the key to a DATA key

The MULTIKEY application makes some assumptions and has specific requirements in order to have the DATA key parts entered via the TKE.

Assumptions:

- The MULTIKEY required keys are controlled by the administrator and cannot be removed or monitored to prevent misuse.

Requirements:

- 2 specific keytypes in the CKDS for use by MULTIKEY:
  - EXPORTER
  - IMPORTER
- Desired clear data key values have been entered on the TKE as EXPORTER key types.
- Only single length Data keys will be processed by the MULTIKEY application.

MULTIKEY Required Keys

The MULTIKEY application requires for its use, two additional keys. These keys are used only by the application and consist of one key that is an EXPORTER key type, the other is an IMPORTER key

type. In order to prevent misuse of these key types, and to preserve the integrity of the user key to be entered that will be converted to a DATA key, special preparation of these application keys is required.

The two keys share the same clear key values with a slight addition for one of the keys. The EXPORTER is "prepared" such that an additional key part is used to create its content.

Assume that the desired Key Encrypting Key values are:
Key Part 1:
63C3 0331 1581 4B07 57B9 6C52 994D 4DC1
Key Part 2:
0123 4567 89AB CDEF FEDC BA98 7654 3210

The application required IMPORTER can be entered into the system (via the TKE for instance) using the two above key parts, one being FIRST or initial, the other being LAST or final.

The application required EXPORTER must be entered as THREE key parts. The above key parts would be entered as FIRST and MIDDLE. An additional key part having the value shown below must be entered as a FINAL key part. This required third key part value is

0041 7D00 0341 0000 0041 7D00 0341 0000

The purpose of this extra step is to provide a key that cryptographically, when used to export another EXPORTER key (and only an EXPORTER key), will effectively remove the Control Vector from the LEFT half of the other EXPORTER, thereby making the left half of the other EXPORTER into a DATA key.

This function only works for EXPORTER keys that are SINGLE-length. Double-length EXPORTERS are not at risk. It is also assumed that the application required EXPORTER/IMPORTER are under the control of a Key Administrator, and can be removed from the system as required so that misuse of these cannot occur.

Running the MULTIKEY Program

The MULTIKEY JCL input are a set of control cards with data starting in column 1. An asterisk in column 1 denotes a comment card. The control cards required are shown in the examples below and highlighted in **bold blue** text.

Let us assume that the Key Administrators have set up the application required key types EXPORTER/IMPORTER in the ICSF CKDS and labeled these as "MULTIKEY.EXPORTER" and "MULTIKEY.IMPORTER". Additionally, key values for four desired DATA keys have been entered from the TKE as EXPORTER keys . These four keys are referred to as "SOURCE" by the application. Assume they are labeled:

"EXPORTER.CVCKEYA", "EXPORTER.CVCKEYB",
"KEK.TO.DATA.1", "KEK.TO.DATA.2"

MULTIKEY JCL Invocation

The invocation JCL for the MULTIKEY application would look like the following:

```
//   JOB
//CONVERT EXEC PGM=MULTIKEY,REGION=100K
//SYSPRINT DD SYSOUT=A
//SYSIN        DD *
*all control cards start in column 1.
*an asterisk denotes a comment card
IMPORTER=MULTIKEY.IMPORTER
EXPORTER=MULTIKEY.EXPORTER
SOURCE=EXPORTER.CVCKEYA
TARGET=EXPORTER.CVCKEYA
*The above implies that the target will replace the source.
*The source is first exported to an application work area.
*Target can be any valid key label or name.
*If a CKDS record already exists for the target key name,
*it will first be deleted, then added and updated with the new value.
*TARGET also triggers execution of the application.
*TARGET= should therefore follow SOURCE= .   At least one
*instance of IMPORTER=  and EXPORTER=  and one instance of
*SOURCE= and TARGET=  must be specified.
*Target key label need not be the same as source key label, but by using the same
*key label values, it eliminates the need to later remove the source key entries from the CKDS.
*From a practical and auditing standpoint, though, it may be better
*to specify the TARGET as the actual desired name, then also specify SOURCE equal to *TARGET.
SOURCE=EXPORTER.CVCKEYB
TARGET=CVCKEYB
SOURCE=KEK.TO.DATA.1
TARGET=DEBIT.CVVKEYA
SOURCE=KEK.TO.DATA.2
TARGET=DEBIT.CVCKEYB
*Note that IMPORTER and EXPORTER stay in effect unless another
*IMPORTER= or EXPORTER= card is encountered.
/*
//
```

Using Encrypted Key Values with MULTIKEY

The MULTIKEY application can also be used to enter the cryptograms of a desired DATA type key. For instance, an encrypted key value is presented pre-encrypted by a transport key (KEK). That transport key is made available via a secure method by the Key Custodians as an IMPORTER key type. The MULTIKEY application can now be run with reference to this IMPORTER and the encrypted DATA key value.

Assume that the IMPORTER has been stored as label "CVC.IMPORTER" and that "C874 E3B2 9DC5 6FAA" is the value of the CVC key encrypted with this IMPORTER.
The control cards for MULTIKEY in this case would be:

```
IMPORTER=CVC.IMPORTER
ENCKEY=C874E3B29DC56FAA
TARGET=CVCKEYA.TEST
```

These cards may be merged also into the above JCL example meaning that not only can EXPORTER to DATA conversion be done, but also encrypted DATA key entry can be done all in the same JCL invocation:

```
//   JOB
//CONVERT EXEC PGM=MULTIKEY,REGION=100K
//SYSPRINT DD SYSOUT=A
//SYSIN       DD *
IMPORTER=MULTIKEY.IMPORTER
EXPORTER=MULTIKEY.EXPORTER
SOURCE=EXPORTER.CVCKEYA
TARGET=EXPORTER.CVCKEYA
SOURCE=EXPORTER.CVCKEYB
TARGET=CVCKEYB
SOURCE=KEK.TO.DATA.1
TARGET=DEBIT.CVVKEYA
SOURCE=KEK.TO.DATA.2
TARGET=DEBIT.CVCKEYB
IMPORTER=CVC.IMPORTER
ENCKEY=C874E3B29DC56FAA
TARGET=CVCKEYA.TEST
/*
//
```

Using MULTIKEY to Generate or Verify CVV/CVC

The MULTIKEY application can be used to generate or verify CVV/CVC values. The control cards to perform this function are listed below. Some of these control cards represent multiple options only one of which would be specified. Following the list is an example of how the input values would be specified for each type of control card. A comment card follows each with the meaning of the preceeding card.

PAN13 | PAN16=
EXPIRY=
SC=
KEYA=
KEYB=
[CVV | CVC=
CVVGEN | CVCGEN]

PAN13=1234567890123 or PAN16=1234567890123456
*either PAN13 or PAN16 would normally be used, not both
EXPIRY=0205
*this is the 4 digit expiry date
SC=101
*this is the 3 digit service code (000 for CVV2/CVC2)
KEYA=LABEL.OF.CVVCVC.KEYA
*label of the CVVKEYA or CVCKEYA
KEYB=LABEL.OF.CVVCVC.KEYB
*label of the CVVKEYB or CVCKEYB
CVV= or CVC=
*1 to 5 digit existing CVV/CVC to be verified
CVVGEN
CVCGEN
*request to generate a CVV/CVC using the already presumed entered
*PAN, EXPIRY, SC, KEYA and KEYB values
*5 digits of CVV/CVC will be generated. The digits (left to right, 1
*through 5) represent the values that would have been generated for
*CVV-1, CVV-2, CVV-3, CVV-4 and CVV-5