19 RACF

February 2000

update

In this issue

RACF Update

Published by

Xephon
27-35 London Road
Newbury
Berkshire RG14 1JL
England
Telephone: 01635 38030
From USA: 01144 1635 38030
E-mail: trevore@xephon.com

North American office

Xephon/QNA
1301 West Highway 407, Suite 201-405
Lewisville, TX 75077-2150
USA
Telephone: 940 455 7050

Contributions

Articles published in *RACF Update* are paid for at the rate of £170 ($250) per 1000 words and £90 ($140) per 100 lines of code for original material. To find out more about contributing an article, without any obligation, please contact us at any of the addresses above and we will send you a copy of our *Notes for Contributors*.

RACF Update on-line

Code from *RACF Update* can be downloaded from our Web site at http://www.xephon.com/racfupdate.html; you will need the user-id shown on your address label.

Editor

Trevor Eddolls

Disclaimer

Readers are cautioned that, although the information in this journal is presented in good faith, neither Xephon nor the organizations or individuals that supplied information in this journal give any warranty or make any representations as to the accuracy of the material it contains. Neither Xephon nor the contributing organizations or individuals accept any liability of any kind howsoever arising out of the use of such material. Readers should satisfy themselves as to the correctness and relevance to their circumstances of all advice, information, code, JCL, and other contents of this journal before making any use of it.

Subscriptions and back-issues

A year's subscription to *RACF Update*, comprising four quarterly issues, costs £190.00 in the UK; $290.00 in the USA and Canada; £196.00 in Europe; £202.00 in Australasia and Japan; and £200.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the August 1995 issue, are available separately to subscribers for £50.50 ($77.50) each including postage.

Printed in England.

A RACF exit to return installation data

Having recently converted from CICS Version 2.1.2 to CICS Version 4.1.0 running alongside RACF Version 2.2, my installation encountered a problem that we had not anticipated. RACLISTed profiles are now stored in a dataspace, meaning that RACF installation data is therefore not so easily accessible. We had been using the RACF installation data field in a variety of our application programs for different reasons. For instance, on transaction profiles the installation data field had been used by us to retain an application description that could be subsequently displayed on an application menu screen or from a title area within the application display screen itself.

Many of our application programs were affected, and so we had the problem of either populating a database with application descriptions or returning the address of the installation data to an application program by using a RACF exit. I decided that a RACF exit invoked from specific programs would result in the fewest changes to our existing applications.

The RACF REQUEST=FASTAUTH exits allow the installation to make additional security checks. They are invoked for any FASTAUTH query, including the EXEC CICS QUERY SECURITY command. There are two RACROUTE REQUEST=FASTAUTH preprocessing exits, where ICHRFX01 is used for non-cross-memory calls and ICHRFX03 is used for cross-memory calls.

These exits are entered before the RACROUTE REQUEST=FASTAUTH service routine performs authorization checking. Similarly, there are two RACROUTE REQUEST=FASTAUTH postprocessing exits, where ICHRFX02 is used with local in-storage profiles and ICHRFX04 is called when FASTAUTH is running in cross-memory mode, or when FASTAUTH is processing in-storage profiles RACLISTed to a dataspace with GLOBAL=YES. ICHRFX04 receives control after authorization checking has been performed using the RACF dataspace profiles and therefore was selected as the most appropriate exit for our purpose.

In order to utilize any ESM exit, the SIT parameter ESMEXITS=INSTLN must be specified (the default is NOINSTLN).

ESMEXITS=INSTLN must be specified in the SIT only and not as an override; it ensures that CICS-related and installation-supplied data can be passed to the ESM using the INSTLN parameter of the RACROUTE macro.

The assembled exit routine must be located in the Link Pack Area and named ICHRFX04. If the RACROUTE REQUEST=FASTAUTH routine (ICHRCT00 or IGC0013) is placed in the Fixed Link Pack Area (FLPA), this exit should also be in the FLPA.

ICHRFX04 must be reentrant. It may have any RMODE but must have AMODE(31) or AMODE(ANY) because it is always invoked in AMODE(31). It is also invoked in AR (Access Register) mode. It is passed the parameter list RFXAP, located in the primary address space, which contains the ALET (Access List Entry Token) to the dataspace and the address of the profile used for authority checking, as well as the address of the exit parameter list RFXP.

The application program requiring installation data from a RACF profile stores the address of the area that will contain the information and its length in fields in the TCT user area, and issues an EXEC CICS QUERY SECURITY against the relevant profile. This is shown in the following application program extract:

```
*********************************************************
** Set up TCTUA for ESM exit to return Installation Data  **
*********************************************************
         USING TCTUAREA,TCTUAR
         EXEC  CICS ADDRESS TCTUA(TCTUAR)
         C     TCTUAR,=X'FF000000'         TCTUA not found
         BNE   GETCLASS
         MVC   0(8,TCTUAR),=C'NO TCTUA'
         B     RETURN
*****************************************************
**   Find transaction class                        **
*****************************************************
GETCLASS DS    0H
         LA    R14,L'DESC                  length of receiving area
         STC   R14,TLENGTH                 set up output area
         ST    R5,TESFAREA                 set up output area
         MVC   LOGM,DFHVALUE(NOLOG)
         EXEC  CICS QUERY SECURITY                                    X
            RESTYPE('TRANSATTACH')                                    X
            .
            RESID(APPL)                                               X
```

```
            READ(READSTAT)                                          X
            LOGMESSAGE(LOGM)                                        X
            RESP(CRES)
.  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
.  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
.  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
.  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
*************************************************
**       TCTUA DSECT                           **
*************************************************
TCTUAREA DSECT
TESFAREA DS   F             FOR USE BY SECURITY EXIT
TLENGTH  DS   X             LENGTH OF INSTALLATION DATA
```

ICHRFX04

Please note that the user of this exit should replace YOURPROG with the application program name that contains the code extract shown above.

```
TITLE 'ICHRFXØ4  -  RACF - FASTAUTH POST-PROCESSING EXIT'
*********************************************************************
***                                                               ***
***      MODULE - ICHRFXØ4                                        ***
***                                                               ***
***      RETURN CODES: Register 15                                ***
***        Ø - Exit routine processing is complete, normal        ***
***            FASTAUTH processing is to continue.                ***
***                                                               ***
***      FUNCTION                                                 ***
***       This exit allows the installation to access             ***
***       RACF Installation Data from a FASTAUTH call.            ***
***       CICS 4.1.Ø can optionally provide a parameter           ***
***       list to RACF exits via the ESMEXITS parameter.          ***
***       This must be set to ESMEXITS=INSTLN to ensure           ***
***       the exit has access to the parameter list including     ***
***       access to the TCTUA of the user terminal because        ***
***       the installation data is returned to the user           ***
***       program at an address specified in the TCTUA.           ***
***                                                               ***
***      REGISTERS                                                ***
***       RØ   =  Ø                                               ***
***       R1   =  RFXAPL parameter list                           ***
***       R2   =  RFXAPL parameter list                           ***
***       R3   =  TCTUA                                           ***
***       R4   =  RACF profile                                    ***
***       R5   =  length of installation data                     ***
***       R6   =  RFXAPL parameter list                           ***
```

```
***       R7   =  DFHXSUXP parameter list                        ***
***       R8   =  target address for installation data          ***
***       R9   =  length of target field                        ***
***       R1Ø  =  base                                          ***
***       R14  =  return address                                ***
***       R15  =  entry point                                   ***
***                                                             ***
***       AR4   = ALET of RACF address space                    ***
***                                                             ***
*****************************************************************
*****************************************************************
***                                                             ***
***      LOAD MODULE NAME          SYSTEM LIBRARY               ***
***      ________________          ______________               ***
***      ICHRFXØ4 (RENT,REFR)      SYS1.LPALIB                  ***
***      AMODE(31) RMODE(ANY)                                   ***
***                                                             ***
*****************************************************************
ICHRFXØ4 AMODE 31
ICHRFXØ4 RMODE ANY
         SPLEVEL  SET=1
         SYSSTATE ASCENV=AR      AR mode macro expansion
ICHRFXØ4 CSECT
************************************************
*  Linkage and addressability                  *
************************************************
         USING ICHRFXØ4,R15      temporary base register
         MODID BR=YES
         DROP  R15
         BAKR  R14, Ø            push environment onto linkage stack
         SAC   512               set AR mode
         LAE   R1Ø,Ø(R15,Ø)      load the base
         USING ICHRFXØ4,R1Ø      base register
         LAE   R6,Ø(R1,Ø)        save pointer to parameter list
         USING RFXAPL,R6         map the exit parm list
         L     PARMREG,ARFXEXPL
         LAE   PARMREG,Ø(PARMREG,Ø)   get address of ICHRFXØ2 parm
         USING RFXPL,PARMREG          map ICHRFXØ2 parm list
         L     R7,RFXANSTL
         LAE   R7,Ø(R7,Ø)        address installation parameter list
         USING DFHXSUXP,R7
         LTR   R7,R7             no installation parameter list?
         BZ    NOINST
         L     TCTUAR,UXPTCTUA   address TCTUA
         LAE   TCTUAR,Ø(TCTUAR,Ø)
         LTR   TCTUAR,TCTUAR
         BZ    NOTCTUA           no TCTUA for this terminal
         USING CISTCTUA,TCTUAR
************************************************
*  INSTDATA returned for QUERY SECURITY call     *
```

```
*************************************************
         L     R4,UXPPHASE
         LAE   R4,Ø(R4,Ø)
         TM    Ø(R4),USER_QUERY_CHECK
         BNO   RETURN
***********************************************
*  Confirm that we are in the right program     *
***********************************************
         L     R4,UXPPROG
         LAE   R4,Ø(R4,Ø)
         LTR   R4,R4             no program name
         BZ    RETURN
         CLC   =C'YOURPROG',Ø(R4)
         BE    GETINST
**  Other programs can be catered for here       **
         B     RETURN
***********************************************
*   Search for required INSTALLATION DATA         *
***********************************************
GETINST  DS    ØH
         L     R8,TESFAREA
         LAE   R8,Ø(R8,Ø)        address for INSTDATA
         LTR   R8,R8
         BZ    RETURN            no target address given
         XR    R9,R9
         IC    R9,TLENGTH        get length of target area
         LAM   R4,R4,ARFXALET    load AR4 with dataspace ALET
         L     R4,ARFXPROF       load GPR4 with profile address
         USING RACRPE,R4
         AH    R4,RPEINSOF       offset to installation data
         USING RPEINST,R4
         XR    R5,R5
         IC    R5,RPEINSTL       check length of data
         LTR   R5,R5             anything there?
         BZ    RETURN            no
         ICM   R5,8,=X'4Ø'       insert padding character
         LA    R4,RPEINSTD       address installation data
         MVCL  R8,R4
         B     RETURN
*
NOTCTUA  EQU   *
*
NOINST   EQU   *
*
RETURN   DS    ØH
         SR    R15,R15           load return code
         PR                      return to caller
*
* DATA AREAS
*
```

```
         DS    ØD
         LTORG
*
* REGISTER EQUATES
*
         DFHREGS
PARMREG  EQU   2
TCTUAR   EQU   3
*
*************************************************
*        Mapping MACROs                         *
*************************************************
         PRINT NOGEN
         PRINT   GEN
         CISTCTUA
         ICHRFXAP
         ICHRFXP
         DFHXSUXP
         COPY  ICHPISP
         PRINT NOGEN
         END   ,
***************************** Bottom of Data ****************************
```

John Hall
CICS Systems Programmer
Cooperative Insurance Society (UK)

Resource profiles utility

I keep the RACF commands I've used for different user profiles in different members of a partitioned dataset. This enables me to retrieve previously executed commands, and to see exactly what was done. All commands can be executed directly through the editor using my EXCMD edit macro (published in *MVS Update*, Issue 121, October 1996).

Sometimes, it can be useful to 'refresh' all the commands for a given dataset profile (or generic resource) and have all the commands to ADDsd (or RDEFine) or PERMit the profiles automatically generated.

The RACFREV macro allows you to ask for a profile in edit mode, and add new commands to the member, ready for execution or update.

For example, ‘RACFREV TSOUSR1.**’ in the command line will insert several lines, including one for ADDSD ‘TSOUSR1.**’ and one for each PERMIT ‘TSOUSR1.**’ ID(usergrp) ACCESS(access).

The syntax is:

```
RACFREV profile <class>
                <DATASET>
```

Unless specified otherwise, the class parameter defaults to ‘DATASET’. You also have to indicate the target line, ie where you want to insert the new lines. This can be either (B)efore or (A)fter a specific line (as with the COPY and MOVE edit commands). An error message will be shown if the target line is not specified.

After execution, there will be several new lines inserted in the edited member, containing all the RACF commands needed to reconstruct the profile.

RACFREV

```
/*--REXX/RACFREV————————————————————————————————————————————————————*/
/* Reverse access from profiles (General resources + Dataset).        */
/* Insert lines in EDIT with commands to recreate profiles.           */
/* Syntax: RACFREV profile <class>                                    */
/*                         <DATASET>                                  */
/*————————————————————————————————————————————————————————————————*/

    Address ISPEXEC
    'ISREDIT MACRO (PROFILE CLASS) NOPROCESS'
    if Class = ''
       then Class = 'DATASET'
    if Profile = ''
       then Profile = UserId()'.**'

/*———————— Process destination line ——————————*/

    'ISREDIT PROCESS DEST'
    Select
      When Rc = 0
           Then 'ISREDIT (DESTLINE) = LINENUM .ZDEST'
      When Rc <= 8
           then Do
                zedsmsg = 'Enter "A"/"B" line cmd'
                zedlmsg = 'RACFREV requires an "A" or "B" line command'
                'SETMSG MSG(ISRZ001)'
                Exit 12
```

```
                End
      When rc = 20                                        /* Empty file */
           then DestLine = 0
      Otherwise exit 12
    End

/*------------ RACF command construction ------------*/

    if Class = 'DATASET'
       then do
            RacfCmd = 'LISTDSD DA(''''Profile'''') ALL GEN'
            end
       else do
            RacfCmd = 'RLIST' Class Profile 'ALL'
            end

/*-------------- Command execution ---------------*/
    address TSO
    x = outtrap('LINE.')
    address TSO RacfCmd
    RcCmd = Rc
    x = outtrap('OFF')

    if RcCmd <> 0
       then do
            do Ind = 1 to line.0
               say Line.Ind
            end
            say 'RACFREV: Rc='Rc'/'RacfCmd
            exit Rc
            end

/*------------ Retrieve command output ------------*/

    do Ind = 1 to Line.0
       parse var Line.Ind FirstW .
       First2W = Subword(Line.Ind,1,2)
/*-- Generic information (Class name, Profile name, and Generic) --*/
       if FirstW = 'INFORMATION'
          then do
               parse var Line.Ind . . ClassName ClassProf Gen .
               if Gen = '(G)'
                  then Generic = 'GENERIC'
                  else Generic = 'NOGENERIC'
               CmdRevDef  = 'ALTDSD '''ClassProf'''' Generic
               CmdRevPerm = 'PERMIT '''ClassProf'''' Generic
               end
/*------ Generic information (Class name, Profile name) ------*/
       if FirstW = 'CLASS'
          then do
```

```
                Ind = Ind + 2
                parse var Line.Ind ClassName ClassProf .
                CmdRevDef  = 'RALTER' ClassName ClassProf
                CmdRevPerm = 'PERMIT' ClassProf 'CLASS('ClassName')'
                end
/*------- Other information (Owner, UACC, Warning, ...) ------*/
       if FirstW = 'LEVEL'
          then do
                Ind = Ind + 2
                parse var Line.Ind . Owner Uacc Yacc Warning .
                if Warning = 'NO'
                   then Warning = 'NOWARNING'
                   else Warning = 'WARNING'
                end
/*-------------- Installation Data ---------------*/
       if FirstW = 'INSTALLATION'
          then do
                Ind = Ind + 2
                parse var Line.Ind InstData
                if InstData = 'NONE'
                   then InstData = ''
                end
       if Line.Ind = 'NO INSTALLATION DATA'
          then InstData = ''
/*---------------- Notify -----------------*/
       if FirstW = 'NOTIFY'
          then do
                Ind = Ind + 2
                parse var Line.Ind Notify .
                if Notify = 'NO'
                   then Notify = ''
                end
/*---------------- Access list ----------------*/
       if (FirstW = 'USER' & Class <> 'DATASET') |,
          Strip(Line.Ind) = 'ID     ACCESS'
          then do
                AccessList = ''
                do Ind = Ind+2 while FirstW <> ''
                   parse var Line.Ind FirstW Access .
                   AccessList = AccessList FirstW Access
                end
                end
    end

/*-------- Insertion of the generated commands --------*/

    call DATALINE '*** RACFREV: Class('Class') Profile('ClassProf')',
                  Date(E) Time()
    call DATALINE RacfCmd
    call DATALINE CmdRevDef 'OWNER('Owner') UACC('Uacc')'
```

```
    call DATALINE CmdRevDef 'NOTIFY('Notify')' Warning
    call DATALINE CmdRevDef 'DATA('''InstData''')'

    do Ind = 1 to words(AccessList) by 2
       Parse value subword(AccessList,Ind,2) with User Access .
       call DATALINE CmdRevPerm 'ID('User') ACCESS('Access')'
    end
exit

/*———————Insertion of a line in the edited member ———————*/

DATALINE:

    Line = ' 'Arg(1)
    'ISREDIT LINE_AFTER 'DestLine' = DATALINE (LINE)'
    DestLine = DestLine + 1
    return
```

Patrick Leroy (Portugal)

Maintaining RACF authorizations – part 2

This month we conclude the REXX procedure that generates a cross-reference between resource profiles and the users or groups that have access to them, allowing you to maintain most of your RACF authorizations interactively.

```
LADEN_CLASSTEXT:
ctxt.1 = 'TSO Account numbers'
ctyp.1 = 'TSO'
acctnum = 1
ctxt.2 = 'CICS Program Control Table'
ctyp.2 = 'CICS'
acicspct = 2
ctxt.3 = 'Controlling access to applications'
ctyp.3 = 'MVS'
appl = 3
ctxt.4 = 'Resourcegroup class for ACICSPCT'
ctyp.4 = 'CICS'
bcicspct = 4
ctxt.5 = 'CICS Protecting System programmer Commands'
ctyp.5 = 'CICS'
ccicscmd = 5
ctxt.6 = 'Protecting DASD Volumes'
```



```
ctyp.6 = 'MVS'
dasdvol = 6
ctxt.7 = 'Protecting Datasets'
ctyp.7 = 'MVS'
dataset = 7
ctxt.8 = 'CICS destination control table'
ctyp.8 = 'CICS'
dcicsdct = 8
ctxt.9 = 'Resourcegroup class for DCICSDCT'
ctyp.9 = 'CICS'
ecicsdct = 9
ctxt.1Ø= 'TSO Account numbers'
ctyp.1Ø= 'TSO'
acctnum = 1Ø
ctxt.11= 'Protecting miscellaneous functions (IDCAMS)'
ctyp.11= 'MVS'
facility = 11
ctxt.12= 'CICS File control table'
ctyp.12= 'CICS'
fcicsfct = 12
ctxt.13= 'Resourcegroup class for TCICSTRN'
ctyp.13= 'CICS'
gcicstrn = 13
ctxt.14= 'Resourcegroup class for DASDVOL'
ctyp.14= 'MVS'
gdasdvol = 14
ctxt.15= 'Global access checking table entry'
ctyp.15= 'MVS'
global = 15
ctxt.16= 'Memberclass for global class'
ctyp.16= 'MVS'
gmbr = 16
ctxt.17= 'Resourcegroup class for SDSF'
ctyp.17= 'MVS'
gsdsf = 17
ctxt.18= 'Resourcegroup class for FCICSFCT'
ctyp.18= 'CICS'
hcicsfct = 18
ctxt.19= 'CICS Journal control table'
ctyp.19= 'CICS'
jcicsjct = 19
ctxt.2Ø= 'Control access to jesspool datasets(SYSIN SYSOUT)'
ctyp.2Ø= 'MVS'
jesspool = 2Ø
ctxt.21= 'Resourcegroup class for JCICSJCT'
ctyp.21= 'CICS'
kcicsjct = 21
ctxt.22= 'CICS Processing program table'
ctyp.22= 'CICS'
mcicsppt = 22
ctxt.23= 'Resourcegroup class for MCICSPPT'
```

```
ctyp.23= 'CICS'
ncicsppt = 23
ctxt.24= 'Controlling who can issue operator commands'
ctyp.24= 'MVS'
opercmds = 24
ctxt.25= 'CICS Program specification blocks'
ctyp.25= 'CICS'
pcicspsb = 25
ctxt.26= 'Member class for PROGRAM class'
ctyp.26= 'MVS'
pmbr = 26
ctxt.27= 'Controlled programs'
ctyp.27= 'MVS'
program = 27
ctxt.28= 'Resourcegroup class for PCICSPSB'
ctyp.28= 'CICS'
qcicspsb = 28
ctxt.29= 'CICS Temporary storage table'
ctyp.29= 'CICS'
scicstst = 29
ctxt.3Ø= 'Control use of auth. commands in SDSF'
ctyp.3Ø= 'MVS'
sdsf = 3Ø
ctxt.31= 'Controls which users can act as surrogates'
ctyp.31= 'MVS'
surrogat = 31
ctxt.32= 'CICS Transactions'
ctyp.32= 'CICS'
tcicstrn = 32
ctxt.33= 'TSO User attributes such as OPER or MOUNT'
ctyp.33= 'TSO'
tsoauth = 33
ctxt.34= 'TSO Logon procedures'
ctyp.34= 'TSO'
tsoproc = 34
ctxt.35= 'Resourcegroup class for SCICSTST'
ctyp.35= 'CICS'
ucicstst = 35
ctxt.36= 'Resourcegroup class for CCICSCMD'
ctyp.36= 'CICS'
vcicscmd = 36
ctxt.37= 'Controlling the use of JES writers'
ctyp.37= 'MVS'
writer = 37
ctxt.38= 'User attributes for started Tasks'
ctyp.38= 'MVS'
started = 38
ctxt.39= 'Resourcegroup for Systemview'
ctyp.39= 'MVS'
sysmview = 39
return
```

```
PRF_NEWPROF:
 if special = 'N'
    then do
         kz_nprof = 2
         return
         end
 xx = outtrap(ld.)
 address tso "LG "nhlq""
 xx = outtrap(off)
 parse var ld.1 msgnr .
 /* ICH51003I NAME NOT FOUND IN RACF DATASET       */
 if msgnr = 'ICH51003I'
    then do
         drop ld.
         xx = outtrap(ld.)
         address tso "LU "nhlq""
         xx = outtrap(off)
         parse var ld.1 msgnr .
         /* ICH30001I UNABLE TO LOCATE USER ENTRY xxxxx     */
         if msgnr = 'ICH30001I' then kz_nprof = 1
                                else kz_nprof = 0
         end
    else kz_nprof = 0
 drop ld.
 xx = outtrap(ld.)
 address tso "LD DA('"nprof"') GEN"
 xx = outtrap(off)
 parse var ld.1 msgnr .
 /* ICH35003I NO RACF DESCRIPTION FOUND FOR nprof */
 if msgnr = 'ICH35003I' then kz_exist = 0
                        else kz_exist = 1
 drop ld.
return

REPLACE_VERIFICATION:
/************************************************/
  "CONTROL DISPLAY SAVE"
  "ADDPOP"
  "VGET (ZFKA) PROFILE"
  if zfka ¬= 'OFF'
     then do
          cmdstack = 'FKA OFF'
          'DISPLAY COMMAND(CMDSTACK)'
          cfka = '1'
          end
     else cfka = 0
  "DISPLAY PANEL(RACFXAC9)"
  retp9 = rc
  if cfka = 1
     then do
```

```
          cmdstack = 'FKA ON'
          'DISPLAY COMMAND(CMDSTACK)'
          cfka = '0'
          end
  "REMPOP"
  "CONTROL DISPLAY RESTORE"
/************************************************/
return

COPY_STDATA:
  if stgrp = '' then stgruppe = 'NOGROUP'
                else stgruppe = 'GROUP('stgrp')'
  cmd = "RALT "class" ("nprof") STDATA( USER("stusr")",
   stgruppe" PRIVILEGED("stpriv") TRUSTED("sttrst")",
   "TRACE("sttrace"))"
  address tso cmd
return

HELP_RACFXACC:
say "      RACF - access control cross-reference syntax            "
say "                                                              "
say "                                                              "
say "      Format : RACFXACC filterlist | ALL class( LIST HOLD     "
say "                                                              "
say "      filterlist  :  Profilename   could be generic           "
say "      class       :  RACF-class    default: DATASET           "
say "                                                              "
say "                     class = ? gives a selectionlist of all   "
say "                               active RACF-classes without    "
say "                          USER, GROUP, GLOBAL, GMBR           "
say "                          and TAPEVOL                         "
say "                                                              "
say "      LIST   :  optional. Output is directed to dataset       "
say "                userid.RACFXACC.DATAF and will be displayed   "
say "                with EDIT.                                    "
say "      HOLD   :  optional. Outputdataset won't be deleted after"
say "                display. HOLD could be used only with LIST    "
say "                option.                                       "
return
```

RACFXAC1

```
/* REXX */
/* Subroutine RACFXAC1 */
trace o
address ispexec
  "VGET (nuacc,nasys,narz,natso,weiter,cntw,class,prof)"
  call aufbau_weitertab
  "ADDPOP"
  "TBDISPL RACFXAC0 PANEL(RACFXAC2)"
```

```
retw = rc
do while retw < 8
   do while ztdsels > Ø
   if class = 'DATASET' then dlm = "'"
                        else dlm = ""
   if class = 'PROGRAM' then generic = ""
                        else generic = "GENERIC"
      select
        when sel = 'D'
          then do
               "TBDELETE RACFXACØ"
               cmd = "PE "dlm||prof||dlm" DELETE CLASS("class")",
                     "ID("stdu")" generic
                  address tso cmd
               updkz = 1
               cntw = cntw - 1
              end
        when sel = 'A'
          then do
              "CONTROL DISPLAY SAVE"
              "ADDPOP"
              "VGET (ZFKA) PROFILE"
              if zfka ¬= 'OFF'
                 then do
                      cmdstack = 'FKA OFF'
                      'DISPLAY COMMAND(CMDSTACK)'
                      cfka = '1'
                      end
                      else cfka = Ø
              "DISPLAY PANEL(RACFXAC5)"
              retp = rc
              if cfka = 1
                 then do
                      cmdstack = 'FKA ON'
                      'DISPLAY COMMAND(CMDSTACK)'
                      cfka = 'Ø'
                      end
              if retp = Ø
                 then do
                      "TBPUT RACFXACØ"
                      cmd = "PE "dlm||prof""dlm" ACC("stda")",
                       "CLASS("class") ID("stdu")" generic
                          address tso cmd
                      updkz = 1
                      end
              "REMPOP"
              "CONTROL DISPLAY RESTORE"
              end
        when sel = 'L'
          then do
              "CONTROL DISPLAY SAVE"
```

```
      call aufbau_usertab
      "ADDPOP"
      "TBDISPL RACFXAC4 PANEL(RACFXAC7)"
      "TBEND RACFXAC4"
      "REMPOP"
      "CONTROL DISPLAY RESTORE"
      end
when sel = 'I'
  then do
      "CONTROL DISPLAY SAVE"
      "ADDPOP"
      "VGET (ZFKA) PROFILE"
      if zfka ¬= 'OFF'
         then do
              cmdstack = 'FKA OFF'
              'DISPLAY COMMAND(CMDSTACK)'
              cfka = '1'
              end
              else cfka = Ø
      "DISPLAY PANEL(RACFXAC5)"
      retp = rc
      if cfka = 1
         then do
              cmdstack = 'FKA ON'
              'DISPLAY COMMAND(CMDSTACK)'
              cfka = 'Ø'
              end
      if retp = Ø
         then do
              "TBTOP RACFXACØ"
              "TBSCAN RACFXACØ ARGLIST(STDU)"
              retsc = rc
              if retsc = 8
                 then do
                     "TBADD RACFXACØ ORDER"
                     cmd = "PE "dlm||prof||dlm" ACC("stda")",
                            "CLASS("class")",
                            "ID("stdu")" generic
                         address tso cmd
                      updkz = 1
                      cntw = cntw + 1
                      end
                 else do
                      smsg = 'User already defined'
                      lmsg = 'Access for User',
                              stdu 'already defined'
                      "SETMSG MSG(RACFØØØA) MSGLOC(CLASS)"
                      end
              end
      "REMPOP"
      "CONTROL DISPLAY RESTORE"
```

```
                end
            otherwise nop
          end
          if ztdsels > 1 then "TBDISPL RACFXACØ"
                         else ztdsels = Ø
       end
      "TBDISPL RACFXACØ PANEL(RACFXAC2)"
      retw = rc
      end
  call auslesen_weitertab
  "TBEND RACFXACØ"
  "VPUT (nuacc,nasys,narz,natso,weiter,cntw)"
  exit

AUFBAU_WEITERTAB:
  "TBCREATE RACFXACØ NAMES(STDU STDA) WRITE"
  "TBSORT RACFXACØ FIELDS(STDU)"
     select
      when nuacc = 'A' then nuacc = 'ALTER'
      when nuacc = 'C' then nuacc = 'CONTROL'
      when nuacc = 'U' then nuacc = 'UPDATE'
      when nuacc = 'R' then nuacc = 'READ'
      when nuacc = 'E' then nuacc = 'EXECUTE'
      when nuacc = 'N' then nuacc = 'NONE'
      otherwise nuacc = ' '
      end
     select
      when nasys = 'A' then nasys = 'ALTER'
      when nasys = 'C' then nasys = 'CONTROL'
      when nasys = 'U' then nasys = 'UPDATE'
      when nasys = 'R' then nasys = 'READ'
      when nasys = 'E' then nasys = 'EXECUTE'
      when nasys = 'N' then nasys = 'NONE'
      otherwise nasys = ' '
      end
     select
      when narz = 'A' then narz = 'ALTER'
      when narz = 'C' then narz = 'CONTROL'
      when narz = 'U' then narz = 'UPDATE'
      when narz = 'R' then narz = 'READ'
      when narz = 'E' then narz = 'EXECUTE'
      when narz = 'N' then narz = 'NONE'
      otherwise narz = ' '
      end
     select
      when natso = 'A' then natso = 'ALTER'
      when natso = 'C' then natso = 'CONTROL'
      when natso = 'U' then natso = 'UPDATE'
      when natso = 'R' then natso = 'READ'
      when natso = 'E' then natso = 'EXECUTE'
      when natso = 'N' then natso = 'NONE'
```

```
      otherwise natso = ' '
      end
  wrkw = weiter
  do ii = 1 to cntw
     parse var wrkw w1 wrkw
     parse var w1 stdu '-' stda
     stda = strip(stda,'L','-')
     select
      when stda = 'A' then stda = 'ALTER'
      when stda = 'C' then stda = 'CONTROL'
      when stda = 'U' then stda = 'UPDATE'
      when stda = 'R' then stda = 'READ'
      when stda = 'E' then stda = 'EXECUTE'
      when stda = 'N' then stda = 'NONE'
      otherwise stda = ' '
      end
     "TBADD RACFXAC0 ORDER"
     end
  "TBTOP RACFXAC0 "
return

AUSLESEN_WEITERTAB:
  if updkz = 1
     then do
          "TBTOP RACFXAC0"
          weiter = ''
          do ii = 1 to cntw
             stdu = ' '
             "TBSKIP RACFXAC0"
             "TBGET RACFXAC0"
             weiter = weiter||left(stdu,9,'-')||left(stda,1)' '
             end
          end
return

AUFBAU_USERTAB:
  "TBCREATE RACFXAC4 NAMES(USERID NAME) WRITE"
  "TBSORT RACFXAC4 FIELDS(USERID)"
  address tso
  xx = outtrap(lg.)
  'LG' stdu
  retg = rc
  xx = outtrap(off)
  if retg = 0
     then do
          grp = 'Y'
          do i = 1 to lg.0
             zeile = strip(lg.i)
             if left(zeile,8) = 'USER(S)='
                then leave
             end
```

```
          do j = i+1 by 3 to lg.Ø
             parse var lg.j userid .
             xx = outtrap(lu.)
             'LU' userid
             xx = outtrap(off)
             parse var lu.1 . 'NAME=' name 'OWNER=' .
             address ispexec 'TBADD RACFXAC4 ORDER'
             end
          end
     else do
          grp = 'N'
          xx = outtrap(lu.)
          'LU' stdu
          xx = outtrap(off)
          parse var lu.1 . 'NAME=' name 'OWNER=' .
          userid = stdu
          address ispexec 'TBADD RACFXAC4 ORDER'
          end
  address ispexec
  'TBTOP RACFXAC4'
return
```

RACFXAC2

```
/* REXX */
/* subroutine RACFXAC2 */
address ispexec
  "VGET (prof)"
  call aufbau_dsntab
  "ADDPOP"
  "TBDISPL RACFXAC1 PANEL(RACFXAC6)"
  retw = rc
  do while retw < 8
     do while ztdsels > Ø
        select
          when datatype(sysvol) = 'NUM' then nop
          when sel = 'B'
            then do
                 if sysdsorg = 'VS'
                    then address tso "BV '"strip(protdsn,'T')"'"
                    else "BROWSE DATASET('"strip(protdsn,'T')"') "
                 end
          when sel = 'E'
            then do
                 if sysdsorg = 'VS'
                    then address tso "BV '"strip(protdsn,'T')"'"
                    else "EDIT DATASET('"strip(protdsn,'T')"') "
                 end
          otherwise nop
        end
```

```
        if ztdsels > 1 then "TBDISPL RACFXAC1"
                       else ztdsels = Ø
       end
      "TBDISPL RACFXAC1 PANEL(RACFXAC6)"
      retw = rc
      end
  "TBEND RACFXAC1"
  exit

AUFBAU_DSNTAB:
  "TBCREATE RACFXAC1 NAMES(PROTDSN SYSVOL SYSDSORG) WRITE"
  "TBSORT RACFXAC1 FIELDS(PROTDSN)"
  xx = outtrap(li.,99999)
  address tso "LD DA('"prof"') DSNS NORACF"
  xx = outtrap(off)
  do i = 5 to li.Ø
     parse var li.i protdsn .
     protdsn = strip(protdsn,'B')
     xx =  listdsi("'"protdsn"'  NORECALL")
     if xx = 16
        then do
             sysvol = '??????'
             sysdsorg = ' '
             if sysreason = 9 then do
                                  sysvol = 'MIGRAT'
                                  sysdsorg = ' '
                                  end
             if sysreason = 22 then do
                                  sysvol = 'n/Mnt '
                                  sysdsorg = ' '
                                  end
             end
        else do
             sysvol  = sysvolume
             end
     "TBADD RACFXAC1 ORDER"
     end
  "TBTOP RACFXAC1 "
return
```

RACFXAC3

```
/*  REXX   */
/*  RACF Cross-Reference – Output is displayed as edit-member */
trace o
    arg fstrg class '(' opt
    if index(opt,'HOLD') > Ø then hold = 'Y'
    if class = '' then class = 'DATASET'
    select
     when fstrg = '' | fstrg = '?'   then do
```

```
         say ' RACF - Access control cross-reference          '
         say '    '
         say '    '
         say 'Format : RACFXACC filterlist | ALL class( HOLD'
         say '    '
         say 'filterlist  :  Profilename   possibly generic   '
         say 'class       :  RACF-Class    default: DATASET   '
         say '    '
         say 'Option HOLD holds the Outputdataset for later use'
        exit
        end
   when fstrg = 'ALL' then cmd = 'SR CLASS('class') NOMASK'
   when length(fstrg) = 1 then cmd = 'SR CLASS('class') MASK('fstrg')'
   otherwise cmd = 'SR CLASS('class') FILTER('fstrg')'
   end
  x = outtrap(cmd.,999999)
address tso
      cmd
  x = outtrap(off)
  x = outtrap(xx.)
  x = outtrap(li.,999999)
  parse var cmd.1 msgnr .
  if msgnr = 'ICH31Ø05I' then do
     say "No profiles with search criteria '"fstrg"' found"
     exit
     end
  if cmd.1 = 'NO ENTRIES MEET SEARCH CRITERIA' then do
     say "No profiles with search criteria '"fstrg"' found"
     exit
     end
  say cmd.Ø' profiles meet search criteria 'fstrg
  do i = 1 to cmd.Ø
     parse var cmd.i prof .
     if class = 'DATASET'
        then "LD DA('"prof"') GEN ALL"
        else "RL "class" ("prof") ALL"
     end
  x = outtrap(off)
  x = outtrap(xx.)
  temp  =  userid()".RACFXACC.DATAF"
  "FREE   DS('"temp"')"
  "DELETE ('"temp"')"
  "ALLOC  F(TEMPN) DS('"temp"') NEW CATALOG SPACE(1,4) TRACKS",
      "LRECL(255) REUSE"
  lrc  = rc
  if   lrc >Ø  then  exit
  t = '|'
  text = 'Access-Level :'
  aus.1 = text '  (A)lter (C)ontrol (U)pdate (R)ead (E)xecute (N)one'
  text = '==================='
  aus.2 = text'=========================================================='
```

```
    aus.3 = 'P r o f i l e -          C l a s s :        '||t
    aus.3 = aus.3'U- 't'*** STD-Acclist ***'
    aus.4 = '    n a m e              'left(class,19)||t
    aus.4 = aus.4'Acc't'SYS't'RZ 't'TSO't'More    '
    text = copies('-',44)'+-+-+-+-+---->>-'copies('-',20)
    aus.0 = 4
       if class = 'DATASET' then call dataset
                            else call genresource
    call ausgabe
    "EXECIO * DISKW TEMPN (STEM AUS.  FINIS "
  address ispexec
              "EDIT DATASET('"TEMP"')  "
  address tso
    "FREE   F(TEMPN) "
    if hold ¬= 'Y' then "DELETE ('"temp"')"
    x = outtrap(off)
 exit

AUSGABE:
  aus.0 = aus.0 + 1
  j = aus.0
  aus.j = text
return

DATASET:
    do i = 1 to li.0
       select
       when pos('INFORMATION FOR DATASET',li.i) > 0
          then do
               call ausgabe
               parse var li.i . . . prof .
               text = left(prof,44)||t
               end
       when pos('UNIVERSAL ACCESS',li.i) > 0
          then do
               i = i + 2
               parse var li.i . . uacc .
               text = text||' 'left(uacc,1)' 't
               end
       when pos('ID     ACCESS',li.i) > 0
          then do
               sacctab = '   't'   't'   't
               i = i + 2
               parse var li.i grp sacc .
               select
                when grp = 'SYS' then do
                  htxt = left(sacc,1)
                  sacctab = overlay(htxt,sacctab,2)
                  end
                when grp = 'RZ' then do
                  htxt = left(sacc,1)
```

```
                  sacctab = overlay(htxt,sacctab,6)
                  end
                when grp = 'TSO' then do
                  htxt = left(sacc,1)
                  sacctab = overlay(htxt,sacctab,1Ø)
                  end
                when grp = 'NO' & sacc = 'ENTRIES' then nop
                otherwise do
                  sacctab = sacctab||left(grp,9,'-')||left(sacc,1)' '
                  end
                end
               i = i + 1
               do while li.i ¬= ' '
                  parse var li.i grp sacc .
                  select
                   when grp = 'SYS' then do
                     htxt = left(sacc,1)
                     sacctab = overlay(htxt,sacctab,2)
                     end
                   when grp = 'RZ' then do
                     htxt = left(sacc,1)
                     sacctab = overlay(htxt,sacctab,6)
                     end
                   when grp = 'TSO' then do
                     htxt = left(sacc,1)
                     sacctab = overlay(htxt,sacctab,1Ø)
                     end
                   otherwise do
                    sacctab = sacctab||left(grp,9,'-')||left(sacc,1)' '
                     end
                   end
                  i = i + 1
                  end
                  text = text||sacctab
               end
       otherwise nop
       end
    end
return

GENRESOURCE:
    do i = 1 to li.Ø
       select
       when pos(class,li.i) = 1
          then do
               call ausgabe
               parse var li.i . prof .
               text = left(prof,44)||t
               end
       when pos('UNIVERSAL ACCESS',li.i) > Ø
          then do
```

```
        i = i + 2
        parse var li.i . . uacc .
        text = text||' 'left(uacc,1)' 't
        end
when pos('USER      ACCESS',li.i) > 0
   then do
        sacctab = '   't'   't'   't
        i = i + 2
        parse var li.i grp sacc .
        select
         when grp = 'SYS' then do
           htxt = left(sacc,1)
           sacctab = overlay(htxt,sacctab,2)
           end
         when grp = 'RZ' then do
           htxt = left(sacc,1)
           sacctab = overlay(htxt,sacctab,6)
           end
         when grp = 'TSO' then do
           htxt = left(sacc,1)
           sacctab = overlay(htxt,sacctab,10)
           end
         when grp = 'NO' & sacc = 'ENTRIES' then nop
         otherwise do
           sacctab = sacctab||left(grp,9,'-')||left(sacc,1)' '
           end
         end
        i = i + 1
        do while li.i ¬= ' '
           parse var li.i grp sacc .
           select
            when grp = 'SYS' then do
              htxt = left(sacc,1)
              sacctab = overlay(htxt,sacctab,2)
              end
            when grp = 'RZ' then do
              htxt = left(sacc,1)
              sacctab = overlay(htxt,sacctab,6)
              end
            when grp = 'TSO' then do
              htxt = left(sacc,1)
              sacctab = overlay(htxt,sacctab,10)
              end
            otherwise do
             sacctab = sacctab||left(grp,9,'-')||left(sacc,1)' '
              end
            end
           i = i + 1
           end
           text = text||sacctab
        end
```

```
       otherwise nop
       end
    end
return
```

ABFRAGE

```
/* REXX Abfrage */
trace o
   parse arg auswahl , affrage
   if auswahl = '' then auswahl = '(Y/N)'
   verlist = strip(auswahl,'L','(')
   verlist = strip(verlist,'T',')')
   verlist = translate(verlist,',','/')
   afflen = length(affrage)
   if afflen > 68 then do
      afind = 69
      do forever
         if substr(affrage,afind,1) = ' ' then leave
         afind = afind - 1
         if afind < 2 then leave
      end
      affrage2 = substr(affrage,afind+1)
      affrage  = left(affrage,afind-1)
   end
   address ispexec
   pcol  = 1Ø
   panid = 'ABFRAGE'
   call disppan
   if drc ¬< 8 then afantw = '*'
   affrage = ''
return afantw

disppan:
   ptit = '  'ptit'  '
   ptit = center(ptit,68,'-')
   nok = 1
   do while nok = 1
   "VGET (ZFKA) PROFILE"
   if zfka ¬= 'OFF' then do
      cmdstack = 'FKA OFF'
      'DISPLAY COMMAND(CMDSTACK)'
      cfka = '1'
   end
   else cfka = Ø
   "ADDPOP ROW(6) COLUMN("pcol")"
   "DISPLAY PANEL("panid")"
   drc = rc
   "REMPOP"
   if cfka = 1 then do
```

```
      cmdstack = 'FKA ON'
      'DISPLAY COMMAND(CMDSTACK)'
      cfka = 'Ø'
   end
return
```

MESSAGEP

```
/* REXX MESSAGEP shows a message while procedure is still processing  */
trace o
parse arg lmsg
address ISPEXEC
msgtxt = 1
lmsg.1 = ''
lmsg.2 = ''
lmsg.3 = ''
lmsg.4 = ''
if length(lmsg) > 2ØØ then,
   lmsg.1 = 'MESSAGEP: message longer than 2ØØ characters'
else do forever
   if length(lmsg) > 5Ø then do
      if msgtxt = 4 then do
         lmsg.4 = 'MESSAGEP: part of message longer than 5Ø characters'
         leave
      end
      else do
         msgind  = 51
         do forever
            if substr(lmsg,msgind,1) = ' ' then leave
            msgind = msgind - 1
            if msgind = 2 then leave
         end
         lmsg.msgtxt = left(lmsg,msgind-1)
         lmsg        = substr(lmsg,msgind+1)
         msgtxt      = msgtxt + 1
      end
   end
   else do
      lmsg.msgtxt = lmsg
      leave
   end
end
if lmsg.3 = '' then do
   lmsg.3 = lmsg.2
   lmsg.2 = lmsg.1
   lmsg.1 = ''
end
lmsg1 = lmsg.1
lmsg2 = lmsg.2
lmsg3 = lmsg.3
lmsg4 = lmsg.4
```

```
'ADDPOP ROW(2) COLUMN(15)'
"VGET (ZFKA) PROFILE"
altfka = zfka
cmdstk = 'FKA OFF'
'DISPLAY PANEL(MESSAGEP) COMMAND(CMDSTK)'
'CONTROL DISPLAY LOCK' ; 'DISPLAY PANEL(MESSAGEP)'
if altfka ¬= 'OFF' then do
   cmdstk = 'FKA ON'
   'DISPLAY PANEL(MESSAGEP) COMMAND(CMDSTK)'
   end
'REMPOP'
exit
```

RACFXAC0 PANEL

```
)PANEL KEYLIST(RACFXACC,RACF)
)Attr Default(%/_)
/* % type(text  ) intens(high)              Defaults displayed for */
/* / type(text  ) intens(low )                   information only  */
/* _ type( input) intens(high) caps(on ) just(left )               */
   ! type( input) intens(high) caps(on ) just(left ) pad('_')
   ¬ type(output) intens(low ) caps(off) just(asis ) pad(' ')
   # type(output) intens(high) caps(off) just(asis ) pad(' ')
)Body  Expand(//) Smsg(smsgx) Lmsg(lmsgx)
%-/-/- R A C F   C r o s s   R e f e r e n c e -/-/-
%Command ===>_zcmd                                / /%Scroll ===>_amt /
¬smsgx¬lmsgx%
/RACF-Class   : ¬class
/Access-Level :  (A)lter (C)ontrol (U)pdate (R)ead (E)xecute (N)one
/Action-Codes :  (C)opy  (D)elete  (E)xpand (S)how datasets
/
¬================================================================================
%P r o f i l e -                                  /U- %*** STD Access ***
%    n a m e                                      /Acc%SYS/RZ%TSO/Others
%──────────────────────────────────────
/
)Model clear(sel)
!z¬prof                                           ¬z   #z/ ¬z/#z/   ¬z   /
)Init
  .ZVARS = '(sel uacc asys arz atso cntw)'
  &amt = csr
)Reinit
)Proc
                                  /* Process )BODY fields here      */
  If (&ztdsels ¬= 0000)           /* If user selected some rows ... */
                                  /* ... process )MODEL fields here */
  if (&zcmd = 'SHIFT RI')
     exit
  if (&zcmd = 'SHIFT LE')
     exit
```

```
  VER (&SEL,list,' ',E,D,C,S)
)End
```

RACFXAC1 PANEL

```
)PANEL KEYLIST(RACFXACC,RACF)
)Attr Default(%/_)
/* % type(text  ) intens(high)             Defaults displayed for */
/* / type(text  ) intens(low )                  information only  */
/* _ type( input) intens(high) caps(on ) just(left )              */
   ! type( input) intens(high) caps(on ) just(left ) pad('_')
   ¬ type(output) intens(low ) caps(off) just(asis ) pad(' ')
   # type(output) intens(high) caps(off) just(asis ) pad(' ')
)Body  Expand(//) Smsg(smsgx) Lmsg(lmsgx)
%-/-/- R A C F   C r o s s   R e f e r e n c e -/-/-
%Command ===>_zcmd                              / /%Scroll ===>_amt /
¬smsgx¬lmsgx%
/RACF-Class   : ¬class
/Access-Level :  (A)lter (C)ontrol (U)pdate (R)ead (E)xecute (N)one
/Action-Codes :  (C)opy  (D)elete  (E)xpand (S)how datasets
/
¬=============================================================================
%P r o f i l e -              Other  STD Access (without SYS RZ TSO)
%    n a m e                  Cnt.  ─── User or groups  ───
%─────────────────────────────────────
/
)Model clear(sel)
!z¬prof                     ¬z¬z  ¬z
¬z
)Init
  .ZVARS = '(sel fkz cntw weiter fkzw)'
  &amt = csr
)Reinit
)Proc
                                  /* Process )BODY fields here      */
  If (&ztdsels ¬= 0000)           /* If user selected some rows ... */
                                  /* ... process )MODEL fields here */
  if (&zcmd = 'SHIFT RI')
     exit
  if (&zcmd = 'SHIFT LE')
     exit
  VER (&SEL,list,' ',E,D,C)
)End
```

RACFXAC2 PANEL

```
)PANEL KEYLIST(RACFXACC,RACF)
)Attr Default(%+?)
```

```
/* % type(text  ) intens(high)             Defaults displayed for */
/* + type(text  ) intens(low )                  information only  */
   _ type( input) intens(high) caps(on ) just(left ) pad('_')
   ¬ type( input) intens(low ) caps(on ) just(left ) pad(' ')
   ! type( input) intens(high) caps(on ) just(left ) pad(' ')
   # type(output) intens(high) caps(off) just(asis ) pad(' ')
   \ type(output) intens(low ) caps(off) just(asis ) pad(' ')
)Body  Expand(//) window(59,21) smsg(smsgx) lmsg(lmsgx)
#smsgx#lmsgx%
%Command ===>!zcmd    / /%Scroll==>!amt +
%
+RACF-Class   : #class   +
+Profile      : #prof
+
+Universal Access    SYS-User   RZ-User   TSO-User
+  ¬nuacc  +        ¬nasys  +  ¬narz   + ¬natso  +
+
%Other entries in standard access list:
\action                                                 +
+
_z%User - group       access-level
)Model clear(sel)
_z+  ¬stdu    +        ¬stda   +
)Init
  .Help = tutorpan               /* insert name of tutorial panel  */
  .ZVARS = '(sel1 sel)'
  &amt = CSR
  &sel1 = &z
  &action = 'Action-Codes : (I)nsert'
  if (&cntw > Ø)
     .ATTR(sel1) = 'TYPE(OUTPUT) PAD('' '')'
     &action = 'Action-Codes : (A)lter (D)elete (I)nsert (L)ist-Group'
)Reinit
)Proc
 VER (&sel1,LIST,' ',I)
 VER (&sel,LIST,' ',D,I,A,L)
 VER (&nuacc,NB,LIST,' ',ALTER,CONTROL,UPDATE,READ,EXECUTE,NONE)
 VER (&nasys,LIST,' ',ALTER,CONTROL,UPDATE,READ,EXECUTE,NONE)
 VER (&narz,LIST,' ',ALTER,CONTROL,UPDATE,READ,EXECUTE,NONE)
 VER (&natso,LIST,' ',ALTER,CONTROL,UPDATE,READ,EXECUTE,NONE)
 if (&sel1 ¬= &z)
    &sel = &sel1
    &ztdsels = 1
)End
```

RACFXAC3 PANEL

```
)PANEL KEYLIST(RACFXAC5,RACF)
)ATTR DEFAULT(%+_)
```

```
      /*  % TYPE(TEXT) INTENS(HIGH)      defaults displayed for      */
      /*  + TYPE(TEXT) INTENS(LOW)       information only            */
      /*  _ TYPE(INPUT) INTENS(HIGH) CAPS(ON) JUST(LEFT)             */
  $ TYPE(INPUT) INTENS(LOW) PAD(_)    /* input field padded with '_' */
  ! TYPE(INPUT) INTENS(LOW) PAD(' ')  /* input field padded with ' ' */
  # TYPE(OUTPUT) INTENS(LOW) PAD(' ')
)BODY expand(//) smsg(smsgx) lmsg(lmsgx) window(66,6)
%/-/  Copy Profile to  /-/
+Command ===> _zcmd
#smsgx#lmsgx+
+ model  profile: #aprof                                    +
+ target profile: !nprof                                    +
)INIT
  .HELP = TUTORPAN          /* Insert name of tutorial panel    */
  &nprof = &z
)PROC
    VER (&nprof,NB)
)END
```

RACFXAC4 PANEL

```
)PANEL KEYLIST(RACFXACC,RACF)
)ATTR DEFAULT(%+_)
      /*  % TYPE(TEXT) INTENS(HIGH)      defaults displayed for      */
      /*  + TYPE(TEXT) INTENS(LOW)       information only            */
      /*  _ TYPE(INPUT) INTENS(HIGH) CAPS(ON) JUST(LEFT)             */
  $ TYPE(INPUT) INTENS(LOW) PAD(_)    /* input field padded with '_' */
  ! TYPE(INPUT) INTENS(LOW) PAD(' ')  /* input field padded with ' ' */
  # TYPE(OUTPUT) INTENS(LOW) PAD(' ')
)BODY expand(//) smsg(smsgx) lmsg(lmsgx) window(55,6)
%/-/  Delete Verification  /-/
#smsgx#lmsgx+
+Profile:#prof                                    +
+
+Should profile really deleted              J/N ?$z+
+
)INIT
  .HELP = TUTORPAN          /* Insert name of tutorial panel    */
  .ZVARS = '(delok)'
  &delok = &z
)PROC
    VER (&DELOK,NB,LIST,J,N)
)END
```

RACFXAC5 PANEL

```
)PANEL KEYLIST(RACFXACC,RACF)
)ATTR DEFAULT(%+_)
```

```
      /*  % TYPE(TEXT) INTENS(HIGH)      defaults displayed for      */
      /*  + TYPE(TEXT) INTENS(LOW)       information only            */
  _ TYPE(INPUT) INTENS(HIGH) CAPS(ON) JUST(LEFT)
  $ TYPE(INPUT) INTENS(LOW) PAD(_)    /* input field padded with '_' */
  ! TYPE(INPUT) INTENS(LOW) PAD(' ')  /* input field padded with ' ' */
  # TYPE(OUTPUT) INTENS(LOW) PAD(' ')
  \ TYPE(OUTPUT) INTENS(HIGH) PAD(' ')
)BODY expand(//) smsg(smsgx) lmsg(lmsgx) window(40,6)
%/-/ \fkt   %Standart Access Entry  /-/
+Command ===> _zcmd
#smsgx#lmsgx+
+ User / group  : _stdu    +
+ Access-Level  : _stda   +
)INIT
  .HELP = TUTORPAN           /* Insert name of tutorial panel    */
  &zcmd = &z
  if (&SEL = I)
     .ATTR(stdu) = 'TYPE(INPUT)'
     &stdu = &z
     &stda = &z
     &fkt  = 'Insert'
  if (&SEL = A)
     .ATTR(stdu) = 'TYPE(OUTPUT)'
     &stda = &z
     &fkt  = 'Update'
)PROC
    VER (&stdu,NB)
    VER (&stda,NB,LIST,ALTER,CONTROL,UPDATE,READ,EXECUTE,NONE)
)END
```

RACFXAC6 PANEL

```
)PANEL KEYLIST(RACFXACC,RACF)
)Attr Default(%/_)
/* % type(text  ) intens(high)             Defaults displayed for */
/* / type(text  ) intens(low )                  information only  */
/* _ type( input) intens(high) caps(on ) just(left )               */
   ! type( input) intens(high) caps(on ) just(left ) pad('_')
   ¬ type(output) intens(low ) caps(off) just(asis ) pad(' ')
   # type(output) intens(high) caps(off) just(asis ) pad(' ')
)Body  Expand(//) Smsg(smsgx) Lmsg(lmsgx)
%-/-/- R A C F   C r o s s   R e f e r e n c e -/-/-
%Command ===>_zcmd                              / /%Scroll ===>_amt /
¬smsgx¬lmsgx%
/Profile      : ¬prof
/
/ Following datasets are protected by the Profile above :
/
/Action-Codes :  (E)dit  (B)rowse
/
```

```
¬=============================================================================
%   D a t a s e t n a m e                          Volume    DS-Org
%———————————————————————————————————————
/
)Model clear(sel)
!z ¬protdsn                                       ¬z    +   ¬z   +
)Init
  .ZVARS = '(sel,sysvol,sysdsorg)'
  &amt = csr
)Reinit
)Proc
                                  /* Process )BODY fields here      */
  VER (&SEL,list,' ',E,B)
)End
```

RACFXAC7 PANEL

```
)PANEL KEYLIST(RACFXACC,RACF)
)Attr Default(%+?)
/* % type(text  ) intens(high)             Defaults displayed for */
/* + type(text  ) intens(low )                  information only  */
   _ type( input) intens(high) caps(on ) just(left ) pad('_')
   ¬ type( input) intens(low ) caps(on ) just(left ) pad(' ')
   ! type( input) intens(high) caps(on ) just(left ) pad(' ')
   # type(output) intens(high) caps(off) just(asis ) pad(' ')
   \ type(output) intens(low ) caps(off) just(asis ) pad(' ')
)Body  Expand(//) window(59,21) smsg(smsgx) lmsg(lmsgx)
#smsgx#lmsgx%
%Command ===>!zcmd    / /%Scroll==>!amt +
%
+RACF-Class   : #class   +
+Profile      : #prof
+
\text1
\text2
+
+Userid    Name
)Model
#userid  +#name                                        +
)Init
  .Help = tutorpan               /* insert name of tutorial panel  */
  &amt = CSR
  if (&grp = Y)
     &text1 = 'Following users in group &STDU have'
     &text2 = '&STDA.-access to the profile above:'
  if (&grp ¬= Y)
     &text1 = 'Following user has &STDA.-access to the profile:'
     &text2 = &Z
)Reinit
```

```
)Proc
)End
```

RACFXAC8 PANEL

```
)PANEL KEYLIST(RACFXACC,RACF)
)Attr Default(%+?)
/* % type(text  ) intens(high)              Defaults displayed for */
/* + type(text  ) intens(low )                   information only  */
   _ type( input) intens(high) caps(on ) just(left ) pad('_')
   ¬ type( input) intens(low ) caps(on ) just(left ) pad(' ')
   ! type( input) intens(high) caps(on ) just(left ) pad(' ')
   # type(output) intens(high) caps(off) just(asis ) pad(' ')
   \ type(output) intens(low ) caps(off) just(asis ) pad(' ')
)Body  Expand(//) smsg(smsgx) lmsg(lmsgx)
#smsgx#lmsgx%
%Cmd ===>!zcmd    / /%Scroll==>!amt +
%
+/*/ List of all active RACF-classes /*/
+
+Please mark class of your choice with 'S'
+
+   classname typ   description
)Model clear(sel)
_z+#aclass   #ctyp #ctxt
)Init
  .Help = tutorpan               /* insert name of tutorial panel  */
  &amt = CSR
  .ZVARS = '(sel)'
)Reinit
)Proc
                                 /* Process )BODY fields here      */
  If (&ztdsels ¬= 0000)          /* If user selected some rows ... */
                                 /* ... process )MODEL fields here */
  VER (&SEL,LIST,S)
)End
```

RACFXAC9 PANEL

```
)PANEL KEYLIST(RACFXACC,RACF)
)ATTR DEFAULT(%+_)
      /*  % TYPE(TEXT) INTENS(HIGH)      defaults displayed for      */
      /*  + TYPE(TEXT) INTENS(LOW)       information only            */
      /*  _ TYPE(INPUT) INTENS(HIGH) CAPS(ON) JUST(LEFT)             */
  $ TYPE(INPUT) INTENS(LOW) PAD(_)    /* input field padded with '_' */
  ! TYPE(INPUT) INTENS(LOW) PAD(' ')  /* input field padded with ' ' */
  # TYPE(OUTPUT) INTENS(LOW) PAD(' ')
)BODY expand(//) smsg(smsgx) lmsg(lmsgx) window(60,8)
```

```
%/-/  Replace Verification  /-/
#smsgx#lmsgx+
+Profile:#nprof                                          +
+
+Profile already exists    ||
+Should profile be replaced     J/N ?$z+
+
)INIT
  .HELP = TUTORPAN           /* Insert name of tutorial panel    */
  .ZVARS = '(repok)'
  &repok = &z
)PROC
    VER (&REPOK,NB,LIST,J,N)
)END
```

RACFXACA PANEL

```
)PANEL KEYLIST(RACFXACC,RACF)
)Attr Default(%/_)
/* % type(text  ) intens(high)             Defaults displayed for */
/* / type(text  ) intens(low )                  information only  */
/* _ type( input) intens(high) caps(on ) just(left )               */
   ! type( input) intens(high) caps(on ) just(left ) pad('_')
   ¬ type(output) intens(low ) caps(off) just(asis ) pad(' ')
   # type(output) intens(high) caps(off) just(asis ) pad(' ')
)Body  Expand(//) Smsg(smsgx) Lmsg(lmsgx)
%-/-/- R A C F   C r o s s   R e f e r e n c e -/-/-
%Command ===>_zcmd                                / /%Scroll ===>_amt /
¬smsgx¬lmsgx%
/RACF-Class   : ¬class
/
/Action-Codes :  (C)opy  (D)elete  (E)dit  (A)dd
/
¬================================================================================
%P r o f i l e -              *** STDATA Information ***
%    n a m e       /User    %Group   /Trusted %Privileged /Trace
%————————————————————————————————————————
/
)Model clear(sel)
!z¬prof           ¬z       #z       /  ¬z  /       #z  /  ¬z  /
)Init
  .ZVARS = '(sel stusr stgrp sttrst stpriv sttrace)'
  &amt = csr
)Reinit
)Proc
                                  /* Process )BODY fields here      */
  If (&ztdsels ¬= 0000)           /* If user selected some rows ... */
                                  /* ... process )MODEL fields here */
  if (&zcmd = 'SHIFT RI')
```

```
     exit
  if (&zcmd = 'SHIFT LE')
     exit
  VER (&SEL,list,' ',E,D,C,A)
)End
```

RACFXACB PANEL

```
)PANEL KEYLIST(RACFXACC,RACF)
)ATTR DEFAULT(%+_)
      /*  % TYPE(TEXT) INTENS(HIGH)      defaults displayed for      */
      /*  + TYPE(TEXT) INTENS(LOW)       information only            */
  _ TYPE(INPUT) INTENS(HIGH) CAPS(ON) JUST(LEFT)
  $ TYPE(INPUT) INTENS(LOW) PAD(_)    /* input field padded with '_' */
  ! TYPE(INPUT) INTENS(LOW) PAD(' ')  /* input field padded with ' ' */
  # TYPE(OUTPUT) INTENS(LOW) PAD(' ')
  \ TYPE(OUTPUT) INTENS(HIGH) PAD(' ')
)BODY expand(//) smsg(smsgx) lmsg(lmsgx) window(40,11)
%/-/   \fkt   %STDATA Information /-/
+Command ===> _zcmd
#smsgx#lmsgx+
+ Profile       : \nprof           +
+
+ Userid        : _nstusr  +
+ Group         : _nstgrp  +
+ Trusted       : _z  +
+ Privileged    : _z  +
+ Trace         : _z  +
)INIT
  .ZVARS = '(nsttrst nstpriv nsttrace)'
  .HELP = TUTORPAN           /* Insert name of tutorial panel    */
  &fkt = 'Alter'
  if (&SEL = A)
     .ATTR(nprof) = 'TYPE(INPUT)'
     &fkt  = 'Insert'
  &zcmd = &z
)PROC
    VER (&nprof,NB)
    VER (&nstusr,NB)
    VER (&nsttrst,NB,LIST,NO,YES)
    VER (&nstpriv,NB,LIST,NO,YES)
    VER (&nsttrace,NB,LIST,NO,YES)
)END
```

ABFRAGE PANEL

```
)ATTR DEFAULT(%#_)
       #   TYPE(TEXT) INTENS(LOW) SKIP(ON)
       $   TYPE(TEXT) INTENS(NON) SKIP(ON)
```

```
       @   TYPE(OUTPUT) INTENS(LOW) CAPS(OFF) SKIP(ON) COLOR(YELLOW)
       _   TYPE(INPUT) INTENS(HIGH) CAPS(ON) JUST(LEFT)
)body expand(//) window(70,4)
@affrage
@affrage2
%
%Antwort : _Z#    @auswahl
)INIT
  .ZVARS  = '(AFANTW)'
  &AFANTW = &Z
  .CURSOR = afantw
  IF ( &auswahl = &z )
     &auswahl = '(Y/N)'
     &verlist = 'Y,N'
)REINIT
)PROC
 VER (&afantw,NONBLANK,LISTV,&verlist)
)END
```

MESSAGEP PANEL

```
)ATTR DEFAULT(%#_)
       @   TYPE(OUTPUT) INTENS(HIGH) SKIP(ON)  CAPS(OFF)
)BODY window(52,4)
@lmsg1                                                  %
@lmsg2                                                  %
@lmsg3                                                  %
@lmsg4                                                  %
)INIT
)REINIT
)PROC
)END
```

RACF00 MESSAGES

```
RACF000A '&SMSG' .WINDOW=NORESP .TYPE=ACTION
'&LMSG'
RACF000C '&SMSG' .WINDOW=RESP .TYPE=CRITICAL
'&LMSG'
RACF000N '&SMSG' .WINDOW=NORESP .TYPE=NOTIFY
'&LMSG'
RACF000W '&SMSG' .WINDOW=NORESP .TYPE=WARNING
'&LMSG'
```

Roman Hawlitschek
Systems Programmer (Germany)

Managing DFHSM, DFHRMM, and RACF

Under OS/390 we use DFHSM, DFHRMM, and RACF to manage security. Because we have a library, all dumps, back-ups, and recycles are done during the night.

To make sure that there are no problems with DFHHSM, DFHRMM, and RACF, I have written two programs in COBOL to ensure that everything is all right:

- DIVDHSM highlights discrepancies between DFHSM and DFHRMM.
- DIVRACF highlights discrepancies between DFHSM and RACF.

DIVDHSM

```
//DIVDHSM  JOB SYS,'psy1',CLASS=Y,MSGCLASS=J,REGION=4M,
//         NOTIFY=psy1,MSGLEVEL=(1,1),RESTART=*
//JOBLIB   DD  DSN=RDVLYO.BATCH.LOADLIB,DISP=SHR            LOAD-MODULE
//*
//*********************************************************************
//**                                                                 **
//**       check HSM tapes owned by RMM                              **
//**       chech HSM tapes owned by RACF                             **
//**                                                                 **
//*********************************************************************
//** SYS:    MVS520, DFHSM 1.2, DFRMM 1.2, AND  RACF 2.2             **
//*********************************************************************
//*
//IDCAMS   EXEC  PGM=IDCAMS
//*=======================
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 DELETE EXPL69.HSM.PRIK7       NONVSAM PURGE
 DELETE EXPL69.HSM.ALTK7       NONVSAM PURGE
 DELETE EXPL69.HSM.ABRK7       NONVSAM PURGE
 DELETE EXPL69.HSM.ALLK7       NONVSAM PURGE
 DELETE EXPL69.RMM.HSMK7       NONVSAM PURGE
 DELETE EXPL69.DIV.ERRK7       NONVSAM PURGE
 DELETE EXPL69.RACF.TAPEVOL.OK NONVSAM PURGE
 IF MAXCC = 8 THEN SET MAXCC = 0
/*
//*************************************************************
//**   CREATION OF A FILE OF PRIMARY HSM VOLUMES              **
```

```
//************************************************************
//**  RESULT CMD TSO HSEND LIST TTOC OUTDATASET() (M,B)     **
//**  RESULT CMD TSO HSEND LIST DVOL OUTDATASET() (D)       **
//**  RESULT CMD TSO HSEND LIST AGGR OUTDATASET() (ABARS)   **
//************************************************************
//**               SEE JOB EDGHSKP                          **
//************************************************************
//SORTHSMP EXEC  PGM=ICEMAN
//*=========================
//SORTLIB  DD DSN=SYS1.SORTLIB,DISP=SHR
//*
//SYSOUT   DD SYSOUT=*
//*
//SORTIN   DD DISP=SHR,DSN=EXPL69.LST3.HSM  VOLUME HSM MIGR AND BACKUP
//         DD DISP=SHR,DSN=EXPL69.LST5.HSM  VOLUME HSM DUMP
//SORTOUT  DD DSN=EXPL69.HSM.PRIK7,DISP=(NEW,CATLG,DELETE),
//            SPACE=(TRK,(5,5),RLSE),UNIT=SYSDA,RECFM=FBA,LRECL=121
/*
//SORTDIAG DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 SORT     FIELDS=(2,6,CH,A)
 OPTION   DYNALLOC
 INCLUDE  COND=(2,1,CH,EQ,C'4',OR,2,1,CH,EQ,C'L')
/*
//************************************************************
//** CREATION OF A FILE OF HSM ALTERNATIVES (MIG)          **
//************************************************************
//SORTHSMS EXEC  PGM=ICEMAN
//*=========================
//SORTLIB  DD DSN=SYS1.SORTLIB,DISP=SHR
//*
//SYSOUT   DD SYSOUT=*
//*
//SORTIN   DD DISP=SHR,DSN=EXPL69.LST3.HSM  VOLUME HSM MIGR AND BACKUP
//SORTOUT  DD DSN=EXPL69.HSM.ALTK7,DISP=(NEW,CATLG,DELETE),
//            SPACE=(TRK,(5,5),RLSE),UNIT=SYSDA,RECFM=FBA,LRECL=121
/*
//SORTDIAG DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 SORT     FIELDS=(94,6,CH,A)
 OPTION   DYNALLOC
 INCLUDE  COND=(94,1,CH,EQ,C'4',OR,94,1,CH,EQ,C'L')
 OUTREC   FIELDS=(1X,94,6,13X,C'ALT',98X)
/*
//************************************************************
//** CREATION OF A FILE OF HSM ABARS VOLUMES               **
//************************************************************
//SORTHSMA EXEC  PGM=ICEMAN
//*=========================
```

```
//SORTLIB  DD DSN=SYS1.SORTLIB,DISP=SHR
//*
//SYSOUT   DD SYSOUT=*
//*
//SORTIN   DD DISP=SHR,DSN=EXPL69.LST7.HSM  VOLUME ABARS
//SORTOUT  DD DSN=EXPL69.HSM.ABRK7,DISP=(NEW,CATLG,DELETE),
//            SPACE=(TRK,(5,5),RLSE),UNIT=SYSDA,RECFM=FBA,LRECL=121
/*
//SORTDIAG DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 SORT     FIELDS=(1Ø,6,CH,A)
 OPTION   DYNALLOC
 INCLUDE  COND=(3,5,CH,EQ,C'VOLS=')
 OUTREC   FIELDS=(1X,1Ø,6,13X,C'ABR',98X)
/*
//*********************************************************************
//**  MERGE FILES OF HSM PRIMARY + ALTERNATE + ABARS VOLUMES        **
//*********************************************************************
//MERGEHSM EXEC  PGM=ICEMAN
//*=========================
//SORTLIB  DD DSN=SYS1.SORTLIB,DISP=SHR
//*
//SYSOUT   DD SYSOUT=*
//*
//SORTIN1  DD DSN=EXPL69.HSM.PRIK7,DISP=OLD
//SORTIN2  DD DSN=EXPL69.HSM.ALTK7,DISP=OLD
//SORTIN3  DD DSN=EXPL69.HSM.ABRK7,DISP=OLD
//SORTOUT  DD DSN=EXPL69.HSM.ALLK7,DISP=(NEW,CATLG,DELETE),
//            SPACE=(TRK,(5,5),RLSE),UNIT=SYSDA,RECFM=FBA,LRECL=121
/*
//SORTDIAG DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 MERGE    FIELDS=(2,6,CH,A)
 OPTION   DYNALLOC
 SUM      FIELDS=NONE
/*
//*************************************************************
//** CREATION OF A FILE OF RMM VOLUMES WITH FILES          **
//*************************************************************
//** FROM RESULT OF PROGAM EDGHSKP (MAJ RMM)               **
//*************************************************************
//** SEE JOB EDGHSKP                                       **
//*************************************************************
//SORTRMM  EXEC  PGM=ICEMAN
//*=========================
//SORTLIB  DD DSN=SYS1.SORTLIB,DISP=SHR
//*
//SYSOUT   DD SYSOUT=*
//*
```

```
//SORTIN   DD DISP=SHR,DSN=SMS.DFRMM.REPORT
//SORTOUT  DD DSN=EXPL69.RMM.HSMK7,DISP=(NEW,CATLG,DELETE),
//            SPACE=(TRK,(2Ø,1),RLSE),UNIT=SYSDA,RECFM=VBA,LRECL=125
/*
//SORTDIAG DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 SORT     FIELDS=(72,6,CH,A)
 OPTION   DYNALLOC
 INCLUDE  COND=((72,1,CH,EQ,C'4',OR,72,1,CH,EQ,C'L'),AND,
               (16,3,CH,EQ,C'HSM'))
 SUM      FIELDS=NONE
/*
//*************************************************************
//** COMPARISON BETWEEN THE HSM AND RMM VOLUMES              **
//*************************************************************
//DIVDHSM  EXEC  PGM=DIVDHSM,TIME=2
//*==========================
//SYSABOUT DD SYSOUT=*
//SYSDBOUT DD SYSOUT=*
//SYSOUT   DD SYSOUT=B
//VOLHSM   DD DISP=SHR,DSN=EXPL69.HSM.ALLK7
//VOLRMM   DD DISP=SHR,DSN=EXPL69.RMM.HSMK7
//COMAND   DD DSN=EXPL69.DIV.ERRK7,DISP=(NEW,CATLG,DELETE),
//            SPACE=(TRK,(1,1),RLSE),RECFM=FB,LRECL=8
//BALANCE    DD SYSOUT=B
//*
//*************************************************************
//** CREATION OF A FILE OF RACF VOLUMES                      **
//*************************************************************
//** RESULT CMD TSO RLIST TAPEVOL HSMHSM                     **
//*************************************************************
//** SEE JOB EDGHSKP                                         **
//*************************************************************
//SORTRACF EXEC  PGM=ICEMAN
//*=========================
//SORTLIB  DD DSN=SYS1.SORTLIB,DISP=SHR
//*
//SYSOUT   DD SYSOUT=*
//*
//SORTIN   DD DSN=EXPL69.RACF.TAPEVOL,DISP=SHR  VOLUMES TAPEVOL HSMHSM
//SORTOUT  DD DSN=EXPL69.RACF.TAPEVOL.OK,DISP=(NEW,CATLG,DELETE),
//            SPACE=(TRK,(5,5),RLSE),UNIT=SYSDA,RECFM=VBA,LRECL=137
//SORTDIAG DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 SORT     FIELDS=COPY
 OPTION   DYNALLOC
 INCLUDE  COND=(6,2,CH,EQ,C'4Ø',OR,6,2,CH,EQ,C'LN')
/*
//*************************************************************
```

```
//** COMPARISON BETWEEN HSM AND RACF VOLUMES               **
//*************************************************************
//DIVDRACF EXEC  PGM=DIVDRACF,TIME=1
//*==========================
//SYSABOUT DD SYSOUT=*
//SYSDBOUT DD SYSOUT=*
//SYSOUT   DD SYSOUT=B,OUTLIM=100000
//VOLHSM   DD DISP=SHR,DSN=EXPL69.HSM.ALLK7
//VOLRACF  DD DISP=SHR,DSN=EXPL69.RACF.TAPEVOL.OK
//BALANCE    DD SYSOUT=B
//*
//SOUMET   EXEC PGM=IKJEFT01,COND=EVEN
//*------------
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD *
 SUBMIT 'EXPL69.EXPL.CNTL(RMMSCRAT)'
 SUBMIT 'EXPL69.EXPL.CNTL(RMMSCRA8)'
 SUBMIT 'EXPL69.EXPL.CNTL(RMMGASTO)'
/*
//
//
//
//
//
```

DIVDRACF

```
PROCESS APOST,XREF,VBREF
PROCESS NOADV,NOAWO,DATA(31),DYNAM,NOFSRT,RENT
PROCESS NUMPROC(NOPFD),SSRANGE,TRUNC(STD),ZWB
      ************************
       IDENTIFICATION DIVISION.
      ************************
      *
       PROGRAM-ID. DIVDRACF.
      *
      *****************************************************************
      ***                                                           ***
      *** search for tape discrepancies under DFSMS/HSM & RACF      ***
      ***                                                           ***
      ***********************
       ENVIRONMENT DIVISION.
      ***********************
      *
       CONFIGURATION SECTION.
      *======================
       SOURCE-COMPUTER. IBM-390.
       OBJECT-COMPUTER. IBM-390.
      *
       INPUT-OUTPUT SECTION.
```

```
*=====================
 FILE-CONTROL.
*
      SELECT FVOLRACF ASSIGN TO VOLRACF
        FILE STATUS W-FS-FVOLRACF.
*
      SELECT FVOLHSM  ASSIGN TO VOLHSM
        FILE STATUS W-FS-FVOLHSM.
*
      SELECT FBALANCE   ASSIGN TO BALANCE
        FILE STATUS W-FS-FBALANCE.
*
 EJECT
*
***************
 DATA DIVISION.
***************
*
 FILE SECTION.
*==============
*
** QSAM ************************************************************
*
 FD  FVOLHSM
     LABEL RECORD IS STANDARD
     RECORDING MODE F
     RECORD CONTAINS 121 CHARACTERS
     BLOCK CONTAINS Ø RECORDS.
 Ø1  VOLHSM.
     Ø5                                  PIC X(Ø1).
     Ø5 VOLHSM-VOLUME                    PIC X(Ø6).
     Ø5                                  PIC X(12).
     Ø5 VOLHSM-TYPE                      PIC X(Ø5).
        88 HSM-ML2      VALUE ' ML2 '.
        88 HSM-ALT      VALUE ' ALT '.
        88 HSM-ABR      VALUE ' ABR '.
        88 HSM-DUMP     VALUE '     '.
     Ø5                                  PIC X(97).
*
** QSAM ************************************************************
*
 FD  FVOLRACF
     LABEL RECORD IS STANDARD
     RECORDING MODE V
     RECORD CONTAINS 1 TO 133 CHARACTERS
     BLOCK CONTAINS Ø RECORDS.
 Ø1  VOLRACF-1                           PIC X(Ø1).
*
 Ø1  VOLRACF.
     Ø5 VOLRACF-CODE-SAUT                PIC X(ØØ1).
```

```
     05 VOLRACF-VOL1                    PIC X(006).
     05                                 PIC X(002).
     05 VOLRACF-VOL2                    PIC X(006).
     05                                 PIC X(002).
     05 VOLRACF-VOL3                    PIC X(006).
     05                                 PIC X(002).
     05 VOLRACF-VOL4                    PIC X(006).
     05                                 PIC X(102).
*
*
** QSAM ************************************************************
*
 FD  FBALANCE
     LABEL RECORD IS STANDARD
     RECORDING MODE F
     RECORD CONTAINS 133 CHARACTERS
     BLOCK CONTAINS 0 RECORDS.
 01  ZIMPB                                      PIC X(133).
*
*****************************************************************
*
 EJECT
*
 WORKING-STORAGE SECTION.
*========================
*
********* BALANCE REGISTERS  *********************************
*
 01  W-LUS-FVOLHSM         PIC S9(05) PACKED-DECIMAL VALUE +0.
 01  W-LUS-FVOLHSM-MVOL    PIC S9(05) PACKED-DECIMAL VALUE +0.
 01  W-LUS-FVOLHSM-BVOL    PIC S9(05) PACKED-DECIMAL VALUE +0.
 01  W-LUS-FVOLHSM-DVOL    PIC S9(05) PACKED-DECIMAL VALUE +0.
 01  W-LUS-FVOLHSM-AVOL    PIC S9(05) PACKED-DECIMAL VALUE +0.
 01  W-LUS-FVOLHSM-RVOL    PIC S9(05) PACKED-DECIMAL VALUE +0.
 01  W-LUS-FVOLRACF        PIC S9(05) PACKED-DECIMAL VALUE +0.
 01  W-CPT-DIFF            PIC S9(05) PACKED-DECIMAL VALUE +0.
 01  W-CPT-ERREUR1         PIC S9(05) PACKED-DECIMAL VALUE +0.
 01  W-CPT-ERREUR2         PIC S9(05) PACKED-DECIMAL VALUE +0.
*
********* TABLE OF ALTERNATIVE VOLUMES ***********************
*
 01  W-TYPE                PIC X(07) VALUE SPACES.
*
********* TABLE OF CLASS VOLUMES TAPEVOL HSMHSM OF RACF **TTT01990
*
 01  W-TABLE.
     02 W-TAB        OCCURS 5000 TIMES INDEXED BY W-IDX.
        03 W-VOLUME  PIC X(06).
*
********* PROCESSING DECISION *********************************
*
```

```
 Ø1  W-DECISION-PROCESSING       PIC X(Ø1) VALUE 'Ø'.
     88         PROCESSING                 VALUE 'Ø'.
     88     END-PROCESSING                 VALUE '1'.
     88 ABANDON-PROCESSING                 VALUE '9'.
*
********* FILE STATUS CODE AND VSAM CODE **********************
*
 Ø1  .
     Ø2  W-FS-FBALANCE            PIC 9(Ø2) VALUE  Ø.
 Ø1  .
     Ø2  W-FS-FVOLHSM             PIC 9(Ø2) VALUE  Ø.
 Ø1  .
     Ø2  W-FS-FVOLRACF            PIC 9(Ø2) VALUE  Ø.
*
********* ERROR ZONES ON FILE **********************************
*
 Ø1  W-ABANDON-TITRE1             PIC X(1ØØ) VALUE
      '  ==> CAUSE OF ABANDONMENT OF PROGRAM DIVDRACF : '.
 Ø1  W-ABANDON-TITRE2             PIC X(1ØØ) VALUE
      '  ABANDON DIVDRACF : ERROR E/S IN A FILE'.
 Ø1  W-ABANDON-FILE-STATUS        PIC X(1ØØ) VALUE ' '.
 Ø1  W-ABANDON-VSAM-CODE.
     Ø2  PIC X(Ø2Ø) VALUE ' VSAM RETURN CODE : '.
     Ø2  W-VRS                    PIC XX VALUE ' '.
     Ø2  W-VSAM-RETURN-CODE       REDEFINES W-VRS PIC 9(ØØ2).
     Ø2  PIC X(Ø23) VALUE ' VSAM COMPONENT CODE : '.
     Ø2  W-VFC                    PIC X VALUE ' '.
     Ø2  W-VSAM-FONCTION-CODE     REDEFINES W-VFC PIC 9(ØØ1).
     Ø2  PIC X(Ø2Ø) VALUE ' VSAM REASON CODE : '.
     Ø2  W-VSC                    PIC XXX VALUE ' '.
     Ø2  W-VSAM-REASON-CODE       REDEFINES W-VSC  PIC 9(ØØ3).
 Ø1  W-ABANDON-BALANCE              PIC X(1ØØ) VALUE ' '.
*
********* DATE MACHINE *****************************************
*
 Ø1  W-DAT.
     Ø2 AA                    PIC 9(Ø2) VALUE Ø.
     Ø2 MM                    PIC 9(Ø2) VALUE Ø.
     Ø2 JJ                    PIC 9(Ø2) VALUE Ø.
*
********* TIME MACHINE *****************************************
*
 Ø1  W-TIM.
     Ø2 HE                    PIC 9(Ø2) VALUE Ø.
     Ø2 MI                    PIC 9(Ø2) VALUE Ø.
     Ø2 SE                    PIC 9(Ø2) VALUE Ø.
     Ø2 MS                    PIC 9(Ø2) VALUE Ø.
*
********* WHEN-COMPILED ****************************************
*
 Ø1  W-WHEN-COMPILED          PIC X(16).
```

```
*
********* TEMPORARY BALANCE ************************************
*
 Ø1  B-TIT1.
     Ø2              PIC X(Ø7) VALUE ' '.
     Ø2              PIC X(23) VALUE 'CENTRE DE REDEVANCE DE '.
     Ø2              PIC X(1Ø) VALUE 'LYON'.
     Ø2              PIC X(1Ø) VALUE ' '.
     Ø2              PIC X(33) VALUE 'TEMPORARY BALANCE'.
     Ø2              PIC X(28) VALUE ' '.
     Ø2              PIC X(Ø7) VALUE 'ETAT : '.
     Ø2              PIC X(Ø8) VALUE 'DIVDRACF'.
*
 Ø1  B-TIT2.
     Ø2              PIC X(Ø4Ø) VALUE '       U P  DIVDRACF'.
     Ø2   PIC X(Ø55) VALUE
     ' COMPARISON BETWEEN DFSMS / H S M  &  R A C F      '.
     Ø2              PIC X(Ø16) VALUE ' '.
     Ø2              PIC X(ØØ7) VALUE 'DATE : '.
     Ø2 JJ           PIC X(ØØ2) VALUE ' '.
     Ø2              PIC X(ØØ1) VALUE ' '.
     Ø2 MM           PIC X(ØØ2) VALUE ' '.
     Ø2              PIC X(ØØ1) VALUE ' '.
     Ø2 AA           PIC X(ØØ2) VALUE ' '.
     Ø2              PIC X(ØØ1) VALUE ' '.
*
 Ø1  B-DET1.
     Ø2               PIC X(Ø9) VALUE SPACES.
     Ø2               PIC X(4Ø) VALUE
     '- NB OF VOLUMES HSM OBTAINED BY HSM'.
     Ø2 B-LUS-FVOLHSM PIC ZZZBZZZBZZ9.
*
 Ø1  B-DET1Ø.
     Ø2               PIC X(Ø9) VALUE SPACES.
     Ø2               PIC X(4Ø) VALUE
     '                     DONT MIGRVOL'.
     Ø2 B-LUS-FVOLHSM-MVOL PIC ZZZBZZZBZZ9.
*
 Ø1  B-DET11.
     Ø2               PIC X(Ø9) VALUE SPACES.
     Ø2               PIC X(4Ø) VALUE
     '                     DONT  ALTVOL'.
     Ø2 B-LUS-FVOLHSM-AVOL PIC ZZZBZZZBZZ9.
*
 Ø1  B-DET12.
     Ø2               PIC X(Ø9) VALUE SPACES.
     Ø2               PIC X(4Ø) VALUE
     '                     DONT BACKVOL'.
     Ø2 B-LUS-FVOLHSM-BVOL PIC ZZZBZZZBZZ9.
*
 Ø1  B-DET13.
```

```
     Ø2               PIC X(Ø9) VALUE SPACES.
     Ø2               PIC X(4Ø) VALUE
     '                     DONT DUMPVOL'.
     Ø2 B-LUS-FVOLHSM-DVOL PIC ZZZBZZZBZZ9.
*
 Ø1  B-DET14.
     Ø2               PIC X(Ø9) VALUE SPACES.
     Ø2               PIC X(4Ø) VALUE
     '                     DONT ABARS  '.
     Ø2 B-LUS-FVOLHSM-RVOL PIC ZZZBZZZBZZ9.
*
 Ø1  B-DET2.
     Ø2               PIC X(Ø9) VALUE SPACES.
     Ø2               PIC X(4Ø) VALUE
     '- NB OF HSM VOLUMES HELD BY RACF'.
     Ø2 B-LUS-FVOLRACF PIC ZZZBZZZBZZ9.
*
 Ø1  B-DET3.
     Ø2               PIC X(Ø9) VALUE SPACES.
     Ø2               PIC X(4Ø) VALUE
     '- RECORDED DIFFERENCES'.
     Ø2 B-CPT-DIFF    PIC ZZZBZZZBZZ9.
*
 Ø1  B-DET4.
     Ø2               PIC X(Ø9) VALUE SPACES.
     Ø2               PIC X(4Ø) VALUE
     ' * VOLUME HSM ABSENT IN RACF'.
     Ø2 B-CPT-ERREUR1  PIC ZZZBZZZBZZ9.
     Ø2               PIC X(2Ø) VALUE '  TO BE CORRECTED'.
*
 Ø1  B-DET5.
     Ø2               PIC X(Ø9) VALUE SPACES.
     Ø2               PIC X(4Ø) VALUE
     ' * VOLUME RACF ABSENT IN HSM'.
     Ø2 B-CPT-ERREUR2  PIC ZZZBZZZBZZ9.
     Ø2               PIC X(2Ø) VALUE '  TO BE CORRECTED'.
*
 Ø1  B-END1.
     Ø2               PIC X(55) VALUE SPACES.
     Ø2               PIC X(5Ø) VALUE
     'NORMAL END OF DIVDRACF  '.
*
 Ø1  B-END2.
     Ø2               PIC X(45) VALUE SPACES.
     Ø2               PIC X(73) VALUE
     'ABNORMAL END OF DIVDRACF'.
*
********************
 PROCEDURE DIVISION.
********************
*
```

```
*****************************************************************
*                GENERAL PROGRAM STRUCTURE                      *
*****************************************************************
*
 MODULE-DIVDRACF SECTION.
*
     PERFORM MODULE-START.
*
     PERFORM MODULE-PROCESSING.
*
     PERFORM MODULE-END.
*
     PERFORM END-PROG.
*
 END-MODULE-DIVDRACF.
     EXIT.
*
 EJECT
*
*****************************************************************
*                START OF PROCESSING                            *
*****************************************************************
*
 MODULE-START SECTION.
*
     PERFORM INIT-PROGRAM.
*
     PERFORM PRINTER-OPEN.
*
     PERFORM OPEN-FILES.
*
 END-MODULE-START.
     EXIT.
*
 EJECT
*
*****************************************************************
*     VARIOUS INITIALIZATIONS                                   *
*****************************************************************
*
 INIT-PROGRAM SECTION.
*
     PERFORM
       MOVE WHEN-COMPILED            TO W-WHEN-COMPILED
       DISPLAY 'PROGRAM DIVDRACF, '
               'COMPILATION THE ' W-WHEN-COMPILED (4:3)
                                  W-WHEN-COMPILED (1:3)
                                  W-WHEN-COMPILED (7:2)
                           ' A ' W-WHEN-COMPILED (9:8)
       ACCEPT             W-TIM    FROM TIME
```

```
       DISPLAY 'START A ' HE ':' MI ':' SE NO ADVANCING
       DISPLAY '        ' HE ' ' MI ' ' SE
       DISPLAY ' '
       ACCEPT             W-DAT    FROM DATE
       MOVE CORRESPONDING W-DAT      TO B-TIT2
     END-PERFORM.
*
     PERFORM
       INITIALIZE W-TABLE
     END-PERFORM.
*
 END-INIT-PROGRAM.
     EXIT.
*
*****************************************************************
*     OPEN PRINTER FILES                                        *
*****************************************************************
*
 OPEN-PRINTER SECTION.
*
     PERFORM
       OPEN OUTPUT FBALANCE
       IF W-FS-FBALANCE NOT = Ø AND NOT = 97 THEN
         PERFORM
           STRING 'ERROR OPENING FBALANCE FS : ' W-FS-FBALANCE
                  DELIMITED BY SIZE INTO W-ABANDON-FILE-STATUS
           SET ABANDON-PROCESSING TO TRUE
           PERFORM MODULE-END
         END-PERFORM
       END-IF
     END-PERFORM.
*
 END-OPEN-PRINTER.
     EXIT.
*
*****************************************************************
*     OPEN DISK FILES                                           *
*****************************************************************
*
 OPEN-FILES SECTION.
*
     PERFORM
       OPEN INPUT   FVOLHSM
       IF W-FS-FVOLHSM NOT = Ø AND NOT = 97 THEN
         PERFORM
           STRING 'ERROR OPENING FVOLHSM FS : ' W-FS-FVOLHSM
                  DELIMITED BY SIZE INTO W-ABANDON-FILE-STATUS
           SET ABANDON-PROCESSING TO TRUE
           PERFORM MODULE-END
         END-PERFORM
```

```
       END-IF
     END-PERFORM
*
     PERFORM
       OPEN INPUT   FVOLRACF
       IF W-FS-FVOLRACF NOT = Ø AND NOT = 97 THEN
         PERFORM
           STRING 'ERROR OPENING FVOLRACF FS : ' W-FS-FVOLRACF
                  DELIMITED BY SIZE INTO W-ABANDON-FILE-STATUS
           SET ABANDON-PROCESSING TO TRUE
           PERFORM MODULE-END
         END-PERFORM
       END-IF
     END-PERFORM.
*
 END-OPEN-FIC.
     EXIT.
*
 EJECT.
*
*****************************************************************
*     PROCESSING                                                *
*****************************************************************
*
 MODULE-PROCESSING SECTION.
*
     PERFORM CHARGING-TABLE.
*
     PERFORM READ-FVOLHSM.
*
     PERFORM WITH TEST BEFORE   UNTIL   END-PROCESSING

        PERFORM RESEARCH-IN-TABLE

        PERFORM READ-FVOLHSM

     END-PERFORM.
*
     PERFORM EXAMINE-TABLE.
*
 END-MODULE-PROCESSING.
     EXIT.
*
 EJECT.
*
*****************************************************************
* READ AND INFORMATION OF THE TABLE OF VOLUMES TAPEVOL RACF      *
*****************************************************************
*
 CHARGING-TABLE           SECTION.
```

```
*
     PERFORM READ-FVOLRACF.
*
     SET  W-IDX                TO 1.
*
     PERFORM WITH TEST BEFORE UNTIL VOLRACF-VOL1 = HIGH-VALUES
                                 OR W-IDX > 4999

        PERFORM INFORMATION-TABLE

        PERFORM READ-FVOLRACF

     END-PERFORM.
*
     MOVE HIGH-VALUES          TO W-VOLUME(W-IDX).
*
 END-CHARGING-TABLE.
     EXIT.
*
 EJECT.
*
*****************************************************************
*     READ VOLRACF                                              *
*****************************************************************
*
 READ-FVOLRACF SECTION.
*
     READ FVOLRACF
       AT END
         MOVE HIGH-VALUES   TO VOLRACF-VOL1
       NOT AT END
         IF W-FS-FVOLRACF > Ø THEN
             PERFORM
               STRING ' ERROR READ FVOLRACF  FS : '
                 W-FS-FVOLRACF
                 DELIMITED BY SIZE INTO W-ABANDON-FILE-STATUS
               SET ABANDON-PROCESSING TO TRUE
               PERFORM MODULE-END
             END-PERFORM
         END-IF
     END-READ.
*
 END-READ-FVOLRACF.
     EXIT.
*
*****************************************************************
*                                                               *
*****************************************************************
*
 INFORMATION-TABLE SECTION.
*
```

```
     IF VOLRACF-VOL1 NOT = SPACES THEN
        MOVE VOLRACF-VOL1         TO W-VOLUME (W-IDX)
        SET  W-IDX                UP BY 1
        ADD  1                    TO W-LUS-FVOLRACF
     END-IF.
     IF VOLRACF-VOL2 NOT = SPACES THEN
        MOVE VOLRACF-VOL2         TO W-VOLUME (W-IDX)
        SET  W-IDX                UP BY 1
        ADD  1                    TO W-LUS-FVOLRACF
     END-IF.
     IF VOLRACF-VOL3 NOT = SPACES THEN
        MOVE VOLRACF-VOL3         TO W-VOLUME (W-IDX)
        SET  W-IDX                UP BY 1
        ADD  1                    TO W-LUS-FVOLRACF
     END-IF.
     IF VOLRACF-VOL4 NOT = SPACES THEN
        MOVE VOLRACF-VOL4         TO W-VOLUME (W-IDX)
        SET  W-IDX                UP BY 1
        ADD  1                    TO W-LUS-FVOLRACF
     END-IF.
*
 END-INFORMATION -TABLE.
     EXIT.
*
*****************************************************************
*     READ OF VOLHSM                                            *
*****************************************************************
*
 READ-FVOLHSM SECTION.
*
     READ FVOLHSM
       AT END
         MOVE HIGH-VALUES   TO VOLHSM-VOLUME
         SET END-PROCESSING TO TRUE
       NOT AT END
         IF W-FS-FVOLHSM > Ø THEN
             PERFORM
               STRING ' ERROR READ FVOLHSM  FS : '
                 W-FS-FVOLHSM
                 DELIMITED BY SIZE  INTO W-ABANDON-FILE-STATUS
               SET ABANDON-PROCESSING TO TRUE
               PERFORM MODULE-END
             END-PERFORM
           ELSE
             ADD 1                    TO W-LUS-FVOLHSM
             EVALUATE TRUE
               WHEN HSM-ALT
                 ADD 1                TO W-LUS-FVOLHSM-AVOL
               WHEN HSM-ML2
                 ADD 1                TO W-LUS-FVOLHSM-MVOL
               WHEN HSM-ABR
```

```
                ADD 1                TO W-LUS-FVOLHSM-RVOL
              WHEN HSM-DUMP
                ADD 1                TO W-LUS-FVOLHSM-DVOL
              WHEN OTHER
                ADD 1                TO W-LUS-FVOLHSM-BVOL
            END-EVALUATE
         END-IF
     END-READ.
*
 END-READ-FVOLHSM.
     EXIT.
*
*****************************************************************
* ENQUIRY OF HSM VOLUME IN RACF TABLE                           *
*****************************************************************
*
 ENQUIRY-IN-TABLE SECTION.
*
     SET W-IDX TO 1.
*
     SEARCH W-TAB VARYING W-IDX
            AT END
                 PERFORM PROCESSING-VOLUME-HSM
            WHEN VOLHSM-VOLUME = W-VOLUME(W-IDX)
                 MOVE ALL 'Ø'    TO W-VOLUME(W-IDX)
     END-SEARCH.
*
 END-ENQUIRY-IN-TABLE.
     EXIT.
*
*
*****************************************************************
* VOLUME KNOWN BY HSM BUT NOT PROTECTED BY RACF                 *
*****************************************************************
*
 PROCESSING-VOLUME-HSM SECTION.
*
     ADD 1              TO W-CPT-DIFF
     ADD 1              TO W-CPT-ERREUR1.
     PERFORM LISTAGE-VOLUME-HSM.
*
 END-PROCESSING-VOLUME-HSM.
     EXIT.
*
*****************************************************************
* LISTING OF ABBERANT VOLUME IN HSM                             *
*****************************************************************
*
 LISTING-VOLUME-HSM SECTION.
*
     EVALUATE TRUE
```

```
       WHEN HSM-ALT
         MOVE 'ALTERNA'        TO W-TYPE
       WHEN HSM-ABR
         MOVE 'ABARS  '        TO W-TYPE
       WHEN HSM-ML2
         MOVE 'MIGRVOL'        TO W-TYPE
       WHEN HSM-DUMP
         MOVE 'DUMPVOL'        TO W-TYPE
       WHEN OTHER
         MOVE 'BACKVOL'        TO W-TYPE
     END-EVALUATE.
     DISPLAY ' '.
     DISPLAY 'VOLUME HSM ' VOLHSM-VOLUME ' (' W-TYPE ') N EST '
     'PAS PROTEGE RACF'.
*
 END-LISTING-VOLUME-HSM.
     EXIT.
 EJECT.
*
*****************************************************************
* EXAMINE THE TABLE : ENQUIRY OF NON A Ø VOLUMES                *
*****************************************************************
*
 EXAMINE-TABLE SECTION.
*
     PERFORM WITH TEST AFTER  VARYING W-IDX FROM 1 BY 1
            UNTIL   W-VOLUME(W-IDX) = HIGH-VALUES
                 OR W-VOLUME(W-IDX) = SPACES
                 OR W-IDX           > 4999

        IF     W-VOLUME(W-IDX) NOT = ALL 'Ø'
           AND W-VOLUME(W-IDX) NOT = HIGH-VALUES
           AND W-VOLUME(W-IDX) NOT = SPACES      THEN
               PERFORM PROCESSING-VOLUME-RACF
        END-IF

     END-PERFORM.
*
 END-EXAMINE-TABLE.
     EXIT.
*
*****************************************************************
* VOLUME KNOWN BY RACF AS HSM VOLUME BUT NOT IN THE CDS HSM      *
*****************************************************************
*
 PROCESSING-VOLUME-RACF SECTION.
*
     ADD 1              TO W-CPT-DIFF
     ADD 1              TO W-CPT-ERREUR2
     PERFORM LISTAGE-VOLUME-RACF.
*
```

```
 END-PROCESSING-VOLUME-RACF.
     EXIT.
*
*****************************************************************
* LISTING OF THE ABBERANT VOLUME IN RACF                        *
*****************************************************************
*
 LISTING-VOLUME-RACF        SECTION.
*
     DISPLAY ' '.
     DISPLAY 'VOLUME HSM ' W-VOLUME(W-IDX)' IS KNOWN BY RACF'
     ' BUT NOT IN HSM'.
*
 END-LISTING-VOLUME-RACF.
     EXIT.
*
 EJECT
*
*****************************************************************
*     NORMAL OR ABNORMAL END OF PROGRAM                         *
*****************************************************************
*
 MODULE-END SECTION.
*
     PERFORM INFORMATION-BALANCE
*
     PERFORM CREATION-BALANCE
*
     PERFORM CLOSURE-FILES
*
     PERFORM RETURN-CODE
*
     IF NOT END-PROCESSING THEN
       PERFORM END-PROG
     END-IF.
*
 END-MODULE-END.
     EXIT.
*
 EJECT
*
*****************************************************************
*     BALANCE ZONES INFORMATION                                 *
*****************************************************************
*
 INFORMATION-BALANCE SECTION.
*
     MOVE W-LUS-FVOLHSM          TO B-LUS-FVOLHSM
     MOVE W-LUS-FVOLHSM-AVOL     TO B-LUS-FVOLHSM-AVOL
     MOVE W-LUS-FVOLHSM-BVOL     TO B-LUS-FVOLHSM-BVOL
     MOVE W-LUS-FVOLHSM-DVOL     TO B-LUS-FVOLHSM-DVOL
```

```
     MOVE W-LUS-FVOLHSM-MVOL     TO B-LUS-FVOLHSM-MVOL
     MOVE W-LUS-FVOLHSM-RVOL     TO B-LUS-FVOLHSM-RVOL
     MOVE W-LUS-FVOLRACF         TO B-LUS-FVOLRACF
     MOVE W-CPT-DIFF             TO B-CPT-DIFF
     MOVE W-CPT-ERROR1          TO B-CPT-ERROR1
     MOVE W-CPT-ERROR2          TO B-CPT-ERROR2.
*
 END-INF-BALANCE.
     EXIT.
*
*****************************************************************
*     WRITING OF BALANCE                                        *
*****************************************************************
*
 CREATION-BALANCE      SECTION.
*
     WRITE ZIMPB  FROM B-TIT1  AFTER PAGE.
     WRITE ZIMPB  FROM B-TIT2  AFTER 3.
     WRITE ZIMPB  FROM B-DET1  AFTER 5.
     WRITE ZIMPB  FROM B-DET1Ø AFTER 2.
     WRITE ZIMPB  FROM B-DET11 AFTER 1.
     WRITE ZIMPB  FROM B-DET12 AFTER 1.
     WRITE ZIMPB  FROM B-DET13 AFTER 1.
     WRITE ZIMPB  FROM B-DET14 AFTER 1.
     WRITE ZIMPB  FROM B-DET2  AFTER 3.
     WRITE ZIMPB  FROM B-DET3  AFTER 3.
     WRITE ZIMPB  FROM B-DET4  AFTER 2.
     WRITE ZIMPB  FROM B-DET5  AFTER 1.
     PERFORM
       IF       W-LUS-FVOLHSM NOT =    Ø
          AND   W-LUS-FVOLRACF NOT =   Ø
          AND   W-LUS-FVOLRACF
              + W-CPT-ERREUR1
              - W-CPT-ERREUR2     = W-LUS-FVOLHSM
         THEN
           WRITE   ZIMPB FROM  B-END1 AFTER 4
         ELSE
           PERFORM
             WRITE ZIMPB FROM  B-END2 AFTER 4
             WRITE ZIMPB FROM  B-END2 AFTER Ø
             MOVE ' INCOHERENCE DETECTED,SEE TEMPORARY BALANCE
-                ' ' TO W-ABANDON-BALANCE
             SET ABANDON-PROCESSING TO TRUE
           END-PERFORM
       END-IF
     END-PERFORM.
*
 END-CREATION-BALANCE.
     EXIT.
*
```

```
*****************************************************************
*     CLOSE THE FILES                                           *
*****************************************************************
*
 CLOSE-FILES SECTION.
*
      PERFORM
        CLOSE FVOLHSM
        IF W-FS-FVOLHSM > Ø THEN
           DISPLAY '** ERROR CLOSING FVOLHSM FS : '
                   W-FS-FVOLHSM
        END-IF
      END-PERFORM
      PERFORM
        CLOSE FVOLRACF
        IF W-FS-FVOLRACF > Ø THEN
           DISPLAY '** ERROR CLOSING FVOLRACF FS : '
                   W-FS-FVOLRACF
        END-IF
      END-PERFORM
      PERFORM
        CLOSE FBALANCE
        IF W-FS-FBALANCE  > Ø THEN
           DISPLAY '** ERROR CLOSING FBALANCE  FS : '
                   W-FS-FBALANCE
        END-IF
      END-PERFORM.
*
 END-CLOSE-FILES.
      EXIT.
*
*****************************************************************
*     USE RETURN CODE ACCORDING TO PROCESSING DECISION          *
*****************************************************************
*
 RETURN-CODE  SECTION.
*
*    EVALUATION OF ZONE W-DECISION-PROCESSING
*
      EVALUATE TRUE
        WHEN     END-PROCESSING
          MOVE +ØØ     TO RETURN-CODE
        WHEN ABANDON-PROCESSING
          PERFORM
            PERFORM DISPLAY-ABANDON
            MOVE +15   TO RETURN-CODE
          END-PERFORM
        WHEN OTHER
          PERFORM
            DISPLAY '  ERROR CODE DECISION PROGRAM'
            MOVE +99   TO RETURN-CODE
```

```
         END-PERFORM
      END-EVALUATE.
*
 END-RETURN-CODE.
      EXIT.
*
*****************************************************************
* ABANDON PROCESSING : DISPLAY FILE STATUS                      *
*****************************************************************
*
 DISPLAY-ABANDON  SECTION.
*
      DISPLAY '  '.
      DISPLAY W-ABANDON-TITRE1.
      DISPLAY '  '.
      IF W-ABANDON-FILE-STATUS NOT = SPACES THEN
        DISPLAY '  ' W-ABANDON-TITRE2
        DISPLAY '  ' W-ABANDON-FILE-STATUS
      END-IF.
      IF W-VRS                 NOT = SPACES THEN
        DISPLAY '  ' W-ABANDON-VSAM-CODE
      END-IF.
      DISPLAY '  '.
      IF W-ABANDON-BALANCE       NOT = SPACES THEN
        DISPLAY '  ' W-ABANDON-BALANCE
      END-IF.
      DISPLAY '  '.
*
 END-DISPLAY-ABANDON.
      EXIT.
*
*****************************************************************
*     END OF PROGRAM                                            *
*****************************************************************
*
 END-PROG SECTION.
*
      PERFORM
        ACCEPT W-TIM FROM TIME
        DISPLAY ' '
        DISPLAY 'END   A ' HE ':' MI ':' SE NO ADVANCING
        DISPLAY '        ' HE ' ' MI ' ' SE
      END-PERFORM.
*
      STOP RUN.
*
 END-END-PROG.
      EXIT.
```

NEON Systems is shipping Version 4.5 of its Halo SSO, giving NT users a single sign-on tool for accessing System/390-based applications without the need for custom coding or installing software on PCs.

Halo SSO consists of distinct NT and MVS components. The NT bit listens for account changes originated on the OS/390-MVS security system and sends changes to NT domains. The MVS part interfaces with OS/390 security packages (eg RACF), listens for account change requests from remote NT servers, then synchronizes the mainframe and NT accounts.

There's a GUI and a Windows API that can be called by C/C++, Java, and Visual BASIC applications for security administration.

For further details contact:
NEON Systems, 14141 Southwest Freeway, Suite 6200, Sugar Land, TX 77478, USA.
Tel: (281) 491 4200.
NEON Systems, Third Floor, 26-30 London Road, Twickenham, TW1 3RW, UK.
Tel: (0181) 607 9911.
URL: http://www.neonsys.com.

* * *

BMC has announced availability of its INCONTROL for Security Management, a security management framework for single-point enterprise administration of third-party systems. The products include CONTROL-SA, CONTROL-SA/Work Flow, and CONTROL-SA/PassPort.

CONTROL-SA is the centralized security administration tool, allowing security administrators to manage security products running on a range of platforms in the enterprise.

It supports mainframe security packages, eg RACF, plus native security of NOS and mid-range operating systems, including NT, Unix, AS/400, NetWare, OS/2 LAN Server, VAX/VMS, and Tandem. Also supported is applications security for SAP R/3, Oracle, Sybase, Lotus Notes, Microsoft Exchange, SeOS, Proxima SSO, TrustBroker, and PassGo.

CONTROL-SA/WorkFlow, meanwhile, with its Web-based workflow engine, automates authorization requests and approvals, enabling security administrators to grant individuals access rights to computer systems.

The CONTROL-SA/PassPort management tool is said to simplify password changes, allowing end users to initiate a password change request using a Web browser. The password is changed and synchronized on all the systems the user needs.

Separately, the company announced CONTROL-SA/Links to automate the security administration process and to allow security managers to monitor security tasks and implement security policies from a central point.

The product gives security managers an option to create event definitions and automated reactions.

For further details contact:
BMC Software, 2101 Citywest Blvd, Houston, TX 77042, USA, USA.
Tel: (713) 918 8800.
BMC Software, Compass House, 207-215 London Road, Camberley, Surrey, GU15 3EY, UK.
Tel: (01276) 24622.
URL: http://www.bmc.com.