30

RACF update

November 2002

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: fionah@xephon.com

North American office

Xephon
Post Office Box 350100
Westminster CO 80035-0100
USA
Telephone: (303) 410-9344

RACF Update on-line

Code from *RACF Update*, and complete issues in Acrobat PDF format, can be downloaded from http://www.xephon.com/racf; you will need to supply a word from the printed issue.

Subscriptions and back-issues

A year's subscription to *RACF Update* (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. The price includes postage. Individual issues, starting with the August 1999 issue, are available separately to subscribers for £48.50 ($72.75) each including postage.

Editor

Fiona Hewitt

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.

Contributions

When Xephon is given copyright, articles published in *RACF Update* are paid for at £170 ($260) per 1000 words and £100 ($160) per 100 lines of code for the first 200 lines of original material. The remaining code is paid for at the rate of £50 ($80) per 100 lines. In addition, there is a flat fee of £30 ($50) per article. To find out more about contributing an article, without any obligation, please contact us at any of the addresses above or download a copy of our *Notes for Contributors* from http://www.xephon.com/index/nfc

Printed in England.

Unobtrusive password validation

Applications that require you to validate who you are against the security product database (eg TSO, CICS) will typically issue a RACROUTE REQUEST=VERIFY,PASSCHK=YES,.... macro call against the userid and password in question. If the userid and password provided are valid and not in a status that would otherwise prevent system access (such as the password being expired, the userid being revoked, the system time being outside the WHEN day/time window for the userid, etc), the RACROUTE call will return a success condition and access to the application will be permitted. However, if the userid and password provided are invalid, this format of the RACROUTE macro call will cause the invalid password verification count to be incremented by 1. Enough of these invalid verification attempts will eventually cause the userid to be placed into REVOKE status, after which only intervention by the security product administrator can restore the active status of this particular user.

This action is commonly referred to as ‘intruder lockout’. The basic assumption is that a number of consecutive invalid password values could be an indication of someone attempting to gain access to a system with a userid that is not rightfully theirs. In practice, however, it’s most often indicative of a valid user simply forgetting his or her password value.

It is possible in RACF environments to issue a combination of RACROUTE macro calls to validate a userid and password combination that won’t trigger an increment to the invalid password count field if the userid/password are invalid. This is accomplished using a series of RACROUTE REQUEST=EXTRACT macro calls that extract information from the RACF database and perform encryption of the userid/password combination using several available techniques. The basic approach is to extract the current encrypted password from the RACF database. The userid/password combination is then encrypted using each of the available RACROUTE encryption algorithms (DES, HASH, INST, STDDES). A

comparison can then be done against the current encrypted password value and each of the test encrypted values. If there's a match in one of the tests, a valid userid/password combination has been specified.

This password verification technique would not be recommended for use in normal application log-on. It is, however, a viable authentication method for trusted applications that don't use an on-line log-on panel for performing userid/password validation. This technique can be deployed in scenarios where userid/password validation is required, but maintaining the integrity of the invalid log-on counter value is also an important consideration.

This article provides a #PWDVRFY macro and PWDVRFY API. The #PWDVRFY macro invokes the PWDVRFY API to determine whether the specified userid and password are correct. Different macro return codes are used, depending on the status of the password for the userid and whether or not the password is expired or the userid is revoked. These return codes are well documented in both the #PWDVRFY macro and the PWDVRFY source.

To invoke the PWDVRFY API you can use the #PWDVRFY macro as follows:

```
          #PWDVRFY USERID=USRID,PWD=PWDVAL,WORKAREA=WORKAREA
            .
            .
            .
USRID     DC    CL8'userid'
PWDVAL    DC    CL8'pwdval'
WORKAREA  DS    8F
```

The #PWDVRFY macro also supports register notation convention for specifying the parameter value addresses.

To successfully use the PWDVRFY API you will need to assemble the PWDVRFY program and then include its object code in the linkedit for the corresponding application similar to the following:

```
  OBJECT(yourapp)
  OBJECT(PWDVRFY)
```

```
  ENTRY yourapp
  SETCODE AC(1)
  NAME yourapp(R)
```

The RACROUTE calls made in the PWDVRFY API require the load module to minimally be APF authorized so you will require your requesting load module to be linked AC(1) and it must reside in an APF authorized library.

#PWDVRFY MACRO

```
MACRO
&LABEL   #PWDVRFY &USERID=,                                            X
               &PWD=,                                                  X
               &WORKAREA=
         MNOTE *,''
         MNOTE *,'##########   #PWDVRFY MACRO   ##########'
         MNOTE *,''
.*-------------------------------------------------------------------*
.*                                                                   *
.*       MACRO: #PWDVRFY                                             *
.*                                                                   *
.*    FUNCTION: THIS MACRO PROVIDES THE INTERFACE TO AN API THAT     *
.*              WILL VERIFY A PASSWORD VALUE FOR A USERID WITHOUT     *
.*              CAUSING A VIOLATION IF THE PASSWORD IS INVALID.      *
.*                                                                   *
.* CONVENTIONS: RØ       - NOT TO BE USED ON MACRO CALL              *
.*              R1       - NOT TO BE USED ON MACRO CALL              *
.*              R13      - NOT TO BE USED ON MACRO CALL              *
.*              R14      - NOT TO BE USED ON MACRO CALL              *
.*              R15      - NOT TO BE USED ON MACRO CALL              *
.*                                                                   *
.*       PARMS: USERID   - SPECIFIES THE ADDRESS OF THE USERID FOR   *
.*                         WHICH THE REQUEST IS BEING MADE.  THE     *
.*                         ADDRESS CAN BE IN LABEL OR REGISTER       *
.*                         NOTATION.  IF REGISTER NOTATION IS USED,  *
.*                         VALID REGISTERS ARE R2 - R12.             *
.*              PWD      - SPECIFIES THE ADDRESS OF THE REQUESTED    *
.*                         PASSWORD VALUE.  THE ADDRESS CAN BE       *
.*                         IN LABEL OR REGISTER NOTATION.  IF        *
.*                         REGISTER NOTATION IS USED, VALID REGISTERS*
.*                         ARE R2 - R12.                             *
.*              WORKAREA - SPECIFIES THE ADDRESS OF A 32 BYTE,       *
.*                         FULL WORD ALIGNED WORKAREA TO BE USED     *
.*                         BY THE #PWDVRFY MACRO.                    *
.*                         THE ADDRESS CAN BE IN LABEL OR REGISTER   *
.*                         NOTATION.  IF REGISTER NOTATION IS USED,  *
.*                         VALID REGISTERS ARE R2 - R12.             *
.*                                                                   *
```

```
.*      OUTPUT: R15      - RETURN CODE FROM #PWDVRFY MACRO            *
.*                         Ø - THE PASSWORD VALUE IS VALID            *
.*                         4 - THE USERID WAS NOT LOCATED             *
.*                         8 - THE PASSWORD VALUE WAS INVALID FOR     *
.*                             THE SPECIFIED USERID                   *
.*                        12 - THE PASSWORD VALUE WAS VALID, BUT      *
.*                             EXPIRED                                *
.*                        16 - THE PASSWORD VALUE WAS VALID, BUT      *
.*                             THE USERID IS IN REVOKED STATUS        *
.*                        2Ø - THE PASSWORD VALUE WAS VALID, BUT      *
.*                             EXPIRED AND THE USERID IS REVOKED      *
.*                        24 - UNKNOWN PASSWORD STATUS                *
.*                                                                    *
.*--------------------------------------------------------------------*
.*
         LCLB  &REG1
         LCLB  &REG2
         LCLB  &REG3
         AIF   ('&LABEL' EQ '').NOLABEL
&LABEL   EQU   *
.NOLABEL ANOP
.*
.*--------------------------------------------------------------------*
.*                                                                    *
.*   USERID CHECK - CHECK WHETHER REGISTER NOTATION IS BEING USED.    *
.*                - IF IT IS, TRY TO MAKE SURE RØ, R1, R13, R14,      *
.*                  R15 ARE NOT USED.                                 *
.*                                                                    *
.*--------------------------------------------------------------------*
.UIDCHK  ANOP
&REG1    SETB  Ø                        DEFAULT IS NOT REG NOTATION
         AIF   ('&USERID' EQ '').UIDERR1
         AIF   ('&USERID'(1,1) NE '(').PWDCHK
&REG1    SETB  1                        SET REG NOTATION ON
         AIF   ('&USERID'(2,2) EQ 'Ø)').UIDERR2
         AIF   ('&USERID'(2,3) EQ 'RØ)').UIDERR2
         AIF   ('&USERID'(2,3) EQ 'ØØ)').UIDERR2
         AIF   ('&USERID'(2,4) EQ 'RØØ)').UIDERR2
         AIF   ('&USERID'(2,2) EQ '1)').UIDERR2
         AIF   ('&USERID'(2,3) EQ 'R1)').UIDERR2
         AIF   ('&USERID'(2,3) EQ 'Ø1)').UIDERR2
         AIF   ('&USERID'(2,4) EQ 'RØ1)').UIDERR2
         AIF   ('&USERID'(2,2) EQ 'D)').UIDERR2
         AIF   ('&USERID'(2,3) EQ 'RD)').UIDERR2
         AIF   ('&USERID'(2,3) EQ '13)').UIDERR2
         AIF   ('&USERID'(2,4) EQ 'R13)').UIDERR2
         AIF   ('&USERID'(2,2) EQ 'E)').UIDERR2
         AIF   ('&USERID'(2,3) EQ 'RE)').UIDERR2
         AIF   ('&USERID'(2,3) EQ '14)').UIDERR2
         AIF   ('&USERID'(2,4) EQ 'R14)').UIDERR2
         AIF   ('&USERID'(2,2) EQ 'F)').UIDERR2
```

```
         AIF   ('&USERID'(2,3) EQ 'RF)').UIDERR2
         AIF   ('&USERID'(2,3) EQ '15)').UIDERR2
         AIF   ('&USERID'(2,4) EQ 'R15)').UIDERR2
         AGO   .PWDCHK
.UIDERR1 ANOP
         MNOTE 8,'#PWDVRFY REQUIRES A USERID'
         AGO   .END
.UIDERR2 ANOP
         MNOTE 8,'USERID REGISTER &USERID INVALID.  MUST BE R2 - R12'
         AGO   .END
.*
.*-------------------------------------------------------------------*
.*                                                                   *
.*   PWD CHECK - CHECK WHETHER REGISTER NOTATION IS BEING USED.       *
.*             - IF IT IS, TRY TO MAKE SURE RØ, R1, R13, R14,        *
.*               R15 ARE NOT USED.                                   *
.*                                                                   *
.*-------------------------------------------------------------------*
.*
.PWDCHK ANOP
&REG2    SETB  Ø                          DEFAULT IS NOT REG NOTATION
         AIF   ('&PWD' EQ '').PWERR1
         AIF   ('&PWD'(1,1) NE '(').WRKACHK
&REG2    SETB  1                          SET REG NOTATION ON
         AIF   ('&PWD'(2,2) EQ 'Ø)').PWERR2
         AIF   ('&PWD'(2,3) EQ 'RØ)').PWERR2
         AIF   ('&PWD'(2,3) EQ 'ØØ)').PWERR2
         AIF   ('&PWD'(2,4) EQ 'RØØ)').PWERR2
         AIF   ('&PWD'(2,2) EQ '1)').PWERR2
         AIF   ('&PWD'(2,3) EQ 'R1)').PWERR2
         AIF   ('&PWD'(2,3) EQ 'Ø1)').PWERR2
         AIF   ('&PWD'(2,4) EQ 'RØ1)').PWERR2
         AIF   ('&PWD'(2,2) EQ 'D)').PWERR2
         AIF   ('&PWD'(2,3) EQ 'RD)').PWERR2
         AIF   ('&PWD'(2,3) EQ '13)').PWERR2
         AIF   ('&PWD'(2,4) EQ 'R13)').PWERR2
         AIF   ('&PWD'(2,2) EQ 'E)').PWERR2
         AIF   ('&PWD'(2,3) EQ 'RE)').PWERR2
         AIF   ('&PWD'(2,3) EQ '14)').PWERR2
         AIF   ('&PWD'(2,4) EQ 'R14)').PWERR2
         AIF   ('&PWD'(2,2) EQ 'F)').PWERR2
         AIF   ('&PWD'(2,3) EQ 'RF)').PWERR2
         AIF   ('&PWD'(2,3) EQ '15)').PWERR2
         AIF   ('&PWD'(2,4) EQ 'R15)').PWERR2
         AGO   .WRKACHK
.PWERR1 ANOP
         MNOTE 8,'#PWDVRFY REQUIRES A PWD'
         AGO   .END
.PWERR2 ANOP
         MNOTE 8,'PWD REGISTER &PWD INVALID.  MUST BE R2 - R12'
         AGO   .END
```

```
.*
.*-------------------------------------------------------------------*
.*                                                                   *
.*   WORKAREA CHECK - CHECK WHETHER REGISTER NOTATION IS BEING USED.  *
.*                  - IF IT IS, TRY TO MAKE SURE RØ, R1, R13, R14,    *
.*                    R15 ARE NOT USED.                              *
.*                                                                   *
.*-------------------------------------------------------------------*
.*
.WRKACHK ANOP
&REG3    SETB  Ø                        DEFAULT IS NOT REG NOTATION
         AIF   ('&WORKAREA' EQ '').WKAERR1
         AIF   ('&WORKAREA'(1,1) NE '(').CALLAPI
&REG3    SETB  1                        SET REG NOTATION ON
         AIF   ('&WORKAREA'(2,2) EQ 'Ø)').WKAERR2
         AIF   ('&WORKAREA'(2,3) EQ 'RØ)').WKAERR2
         AIF   ('&WORKAREA'(2,3) EQ 'ØØ)').WKAERR2
         AIF   ('&WORKAREA'(2,4) EQ 'RØØ)').WKAERR2
         AIF   ('&WORKAREA'(2,2) EQ '1)').WKAERR2
         AIF   ('&WORKAREA'(2,3) EQ 'R1)').WKAERR2
         AIF   ('&WORKAREA'(2,3) EQ 'Ø1)').WKAERR2
         AIF   ('&WORKAREA'(2,4) EQ 'RØ1)').WKAERR2
         AIF   ('&WORKAREA'(2,2) EQ 'D)').WKAERR2
         AIF   ('&WORKAREA'(2,3) EQ 'RD)').WKAERR2
         AIF   ('&WORKAREA'(2,3) EQ '13)').WKAERR2
         AIF   ('&WORKAREA'(2,4) EQ 'R13)').WKAERR2
         AIF   ('&WORKAREA'(2,2) EQ 'E)').WKAERR2
         AIF   ('&WORKAREA'(2,3) EQ 'RE)').WKAERR2
         AIF   ('&WORKAREA'(2,3) EQ '14)').WKAERR2
         AIF   ('&WORKAREA'(2,4) EQ 'R14)').WKAERR2
         AIF   ('&WORKAREA'(2,2) EQ 'F)').WKAERR2
         AIF   ('&WORKAREA'(2,3) EQ 'RF)').WKAERR2
         AIF   ('&WORKAREA'(2,3) EQ '15)').WKAERR2
         AIF   ('&WORKAREA'(2,4) EQ 'R15)').WKAERR2
         AGO   .CALLAPI
.WKAERR1 ANOP
         MNOTE 8,'#PWDVRFY REQUIRES A WORKAREA'
         AGO   .END
.WKAERR2 ANOP
       MNOTE 8,'WORKAREA REGISTER &WORKAREA INVALID.  MUST BE R2 - R12'
         AGO   .END
.*
.*-------------------------------------------------------------------*
.*                                                                   *
.*   SET UP THE CALL TO THE PWDVRFY ROUTINE.                         *
.*                                                                   *
.*-------------------------------------------------------------------*
.*
.CALLAPI ANOP
         AIF   (&REG3 EQ 1).WAREG1
         LA    1,&WORKAREA              GET WORKAREA ADDRESS
```

```
         AGO   .CALLAPI2
.WAREG1  ANOP
         LR    1,&WORKAREA               GET WORKAREA ADDRESS
.CALLAPI2 ANOP
         XC    Ø(16,1),Ø(1)              CLEAR THE AREA
         AIF   (&REG3 EQ 1).WAREG2
         CALL  PWDVRFY,                                              X
               (&USERID,                                             X
               &PWD),VL,MF=(E,(1))
         AGO   .END
.WAREG2  ANOP
         CALL  PWDVRFY,                                              X
               (&USERID,                                             X
               &PWD),VL,MF=(E,(1))
         AGO   .END
.END     ANOP
         MEND
```

PWDVRFY SOURCE

```
*************************************************************************
*                                                                       *
*   MODULE: PWDVRFY                                                    *
*                                                                       *
* FUNCTION: USE DIRECT CALLS TO DATABASE EXTRACTION TOOLS AND           *
*           ENCRYPTION TOOLS TO ALLOW A USERID/PASSWORD TO BE           *
*           VERIFIED WITHOUT CAUSING A SECURITY PRODUCT PASSWORD        *
*           VIOLATION IF THE PASSWORD IS NOT VALID.                     *
*                                                                       *
*    INPUT: PARM1    - ADDRESS OF THE USERID                            *
*           PARM2    - ADDRESS OF THE VERIFY PASSWORD VALUE             *
*                                                                       *
*   OUTPUT: RETURN CODE DESIGNATION                                     *
*           R15 -   Ø - THE PASSWORD VALUE IS VALID                     *
*               -   4 - THE USERID WAS NOT LOCATED                      *
*               -   8 - THE PASSWORD VALUE WAS INVALID FOR              *
*               -       THE SPECIFIED USERID                            *
*               -  12 - THE PASSWORD VALUE WAS VALID, BUT               *
*               -       EXPIRED                                         *
*               -  16 - THE PASSWORD VALUE WAS VALID, BUT               *
*               -       THE USERID IS IN REVOKED STATUS                 *
*               -  2Ø - THE PASSWORD VALUE WAS VALID, BUT               *
*               -       EXPIRED AND THE USERID IS REVOKED               *
*               -  24 - UNKNOWN PASSWORD STATUS                         *
*                                                                       *
* REQ'MENT: PROGRAMS USING THIS API MUST MINIMALLY BE APF               *
*           AUTHORIZED AND RESIDE IN AN APF AUTHORIZED DATASET          *
*                                                                       *
*************************************************************************
*  START OF MODULE                                                      *
*************************************************************************
```

```
*
PWDVRFY  CSECT
PWDVRFY  AMODE 31
PWDVRFY  RMODE ANY
         BAKR  R14,Ø                    SAVE INCOMING REGISTERS
         LR    R12,R15                  COPY MODULE BASE REGISTER
         USING PWDVRFY,R12              SET ADDRESSABILITY
         B     BEGIN                    BRANCH PAST EYECATCHER
         DC    C'PWDVRFY  '
         DC    C'&SYSDATE &SYSTIME '
BEGIN    DS    ØH
*********************************************************************
*  REGISTERS AFTER ENTRY:                                           *
*                                                                   *
*    R12 - 1ST BASE REGISTER                                        *
*    R13 - SAVEAREA AND DYNAMIC STORAGE AREA                        *
*                                                                   *
*  REGISTER USAGE IN MODULE:                                        *
*                                                                   *
*    R5  - ADDRESS OF USERID                                        *
*    R6  - ADDRESS OF VERIFY PASSWORD VALUE                         *
*                                                                   *
*********************************************************************
         LR    R11,R1                   COPY INCOMING PARM REGISTER
         LR    R3,R13                   COPY INCOMING SAVEAREA
         STORAGE OBTAIN,LENGTH=WORKLEN,LOC=ANY
         LR    RØ,R1                    COPY THE STORAGE ADDRESS
         LR    R13,R1                   AGAIN
         LR    R14,R1                   AGAIN
         L     R1,=A(WORKLEN)           GET STORAGE LENGTH
         XR    R15,R15                  SET FILL BYTE
         MVCL  RØ,R14                   SANITIZE THE STORAGE
         USING WORKAREA,R13             TEMP STORAGE ADDRESSABILITY
         MVC   SAVEAREA+4(4),=C'F1SA'   SET STACK CONVENTION
*********************************************************************
         ST    R11,#PARMØ               SAVE PARM ADDRESS
         MVC   #PARM1(4),Ø(R11)         COPY PARM 1 ADDRESS
         MVC   #PARM2(4),4(R11)         COPY PARM 2 ADDRESS
         L     R5,#PARM1                GET USERID ADDRESS
         L     R6,#PARM2                GET PASSWORD ADDRESS
         N     R6,=X'7FFFFFFF'          CLEAR THE X'8Ø' BIT
*********************************************************************
         MVC   ROUTWRK1(ROUTLEN1),RACROUT1 MOVE IN RACROUTE MODEL
         XC    RACWORK(256),RACWORK     CLEAR RACROUTE
         XC    RACWORK+256(256),RACWORK+256  WORKAREA
         RACROUTE REQUEST=EXTRACT,                                  X
               TYPE=EXTRACT,                                        X
               ENTITY=(R5),                                         X
               RELEASE=1.9.2,                                       X
               FIELDS=FLDLIST1,                                     X
               SUBPOOL=1,                                           X
```

```
              WORKA=RACWORK,MF=(E,ROUTWRK1)
         ST   R15,RETCODE               SAVE THE RETURN CODE
         ST   R1,EXTADDR                SAVE EXTRACT AREA ADDRESS
*********************************************************************
         CLC  RETCODE(4),=F'Ø'          EXTRACT WAS OK?
         BNE  RETURNØ4                  NO - NO USERID SO FAILURE
         L    R1,EXTADDR                GET EXTRACT AREA ADDRESS
         USING EXTWKEA,R1               SET ADDRESSABILITY
         XR   R9,R9                     CLEAR R9
         ICM  R9,B'ØØ11',EXTWOFF        GET OFFSET OF FIELD DATA
         LA   R9,Ø(R9,R1)               POINT TO FIELD DATA
         ICM  R15,B'1111',Ø(R9)         GET FIELD LENGTH
         ST   R15,CURRPWDL              SAVE THE LENGTH
         C    R15,=F'8'                 EIGHT BYTES?
         BNE  RLSEEXTA                  NO - RELEASE EXTRACT AREA
         MVC  CURRPWD(8),4(R9)          COPY ENCRYPTED PASSWORD
         LA   R9,4(R15,R9)              POINT TO PASSDATE LENGTH
         MVC  EXPDATE(3),4(R9)          COPY PASSDATE
         LA   R9,4+3(,R9)               POINT TO FLAG4 LENGTH
         MVC  REVKFLG(1),4(R9)          COPY REVOKE FLAG
*********************************************************************
RLSEEXTA EQU  *
         XR   RØ,RØ                     CLEAR RØ
         ICM  RØ,B'Ø111',EXTWLN         GET LENGTH
         STORAGE RELEASE,LENGTH=(RØ),ADDR=(R1),SP=1
         DROP R1
*********************************************************************
         CLC  CURRPWDL(4),=F'8'         CORRECT LENGTH?
         BNE  RETURNØ4                  NO - SOMETHING IS GOOFY
         MVC  ENCRYPT1+1(8),Ø(R6)       COPY THE PASSWORD
         MVC  ENCRYPT2+1(8),Ø(R6)       COPY THE PASSWORD
         MVC  ENCRYPT3+1(8),Ø(R6)       COPY THE PASSWORD
         MVC  ENCRYPT4+1(8),Ø(R6)       COPY THE PASSWORD
         MVI  ENCRYPT1,X'Ø8'            SET STATIC LENGTH
         MVI  ENCRYPT2,X'Ø8'            SET STATIC LENGTH
         MVI  ENCRYPT3,X'Ø8'            SET STATIC LENGTH
         MVI  ENCRYPT4,X'Ø8'            SET STATIC LENGTH
*********************************************************************
         MVC  ROUTWRK2(ROUTLEN2),RACROUT2 MOVE IN RACROUTE MODEL
         XC   RACWORK(256),RACWORK      CLEAR RACROUTE
         XC   RACWORK+256(256),RACWORK+256  WORKAREA
         RACROUTE REQUEST=EXTRACT,                                     X
               TYPE=ENCRYPT,                                           X
               ENTITY=(R5),                                            X
               ENCRYPT=(ENCRYPT1,DES),                                 X
               RELEASE=1.9.2,                                          X
               WORKA=RACWORK,                                          X
               MF=(E,ROUTWRK2)
*********************************************************************
         MVC  ROUTWRK2(ROUTLEN2),RACROUT2 MOVE IN RACROUTE MODEL
         XC   RACWORK(256),RACWORK      CLEAR RACROUTE
```

```
         XC    RACWORK+256(256),RACWORK+256  WORKAREA
         RACROUTE REQUEST=EXTRACT,                                     X
               TYPE=ENCRYPT,                                           X
               ENTITY=(R5),                                            X
               ENCRYPT=(ENCRYPT2,HASH),                                X
               RELEASE=1.9.2,                                          X
               WORKA=RACWORK,                                          X
               MF=(E,ROUTWRK2)
*********************************************************************
         MVC   ROUTWRK2(ROUTLEN2),RACROUT2 MOVE IN RACROUTE MODEL
         XC    RACWORK(256),RACWORK      CLEAR RACROUTE
         XC    RACWORK+256(256),RACWORK+256  WORKAREA
         RACROUTE REQUEST=EXTRACT,                                     X
               TYPE=ENCRYPT,                                           X
               ENTITY=(R5),                                            X
               ENCRYPT=(ENCRYPT3,INST),                                X
               RELEASE=1.9.2,                                          X
               WORKA=RACWORK,                                          X
               MF=(E,ROUTWRK2)
*********************************************************************
         MVC   ROUTWRK2(ROUTLEN2),RACROUT2 MOVE IN RACROUTE MODEL
         XC    RACWORK(256),RACWORK      CLEAR RACROUTE
         XC    RACWORK+256(256),RACWORK+256  WORKAREA
         RACROUTE REQUEST=EXTRACT,                                     X
               TYPE=ENCRYPT,                                           X
               ENTITY=(R5),                                            X
               ENCRYPT=(ENCRYPT4,STDDES),                              X
               RELEASE=1.9.2,                                          X
               WORKA=RACWORK,                                          X
               MF=(E,ROUTWRK2)
*********************************************************************
         CLC   CURRPWD(8),ENCRYPT1+1     A PASSWORD MATCH?
         BE    CHKFLGS                   YES - RETURN A MATCH CONDITION
         CLC   CURRPWD(8),ENCRYPT2+1     A PASSWORD MATCH?
         BE    CHKFLGS                   YES - RETURN A MATCH CONDITION
         CLC   CURRPWD(8),ENCRYPT3+1     A PASSWORD MATCH?
         BE    CHKFLGS                   YES - RETURN A MATCH CONDITION
         CLC   CURRPWD(8),ENCRYPT4+1     A PASSWORD MATCH?
         BE    CHKFLGS                   YES - RETURN A MATCH CONDITION
         B     RETURNØ8                  RETURN NO MATCH CONDITION
*********************************************************************
CHKFLGS  EQU   *
         CLC   EXPDATE(3),=X'ØØØØØF'     PWD EXPIRED?
         BNE   CHKREVK                   NO - CHECK EXPIRED STATUS
         TM    REVKFLG,X'8Ø'             REVOKED?
         BO    RETURN2Ø                  YES - RETURN EXPIRED & REVOKED
         B     RETURN12                  RETURN EXPIRED
CHKREVK  EQU   *
         TM    REVKFLG,X'8Ø'             REVOKED?
         BO    RETURN16                  RETURN REVOKED
         B     RETURNØØ                  RETURN OK
```

```
*********************************************************************
RETURNØØ EQU   *
         LA    R15,Ø                     SET RETURN CODE
*********************************************************************
*                                                                   *
*   TERMINATION.                                                    *
*                                                                   *
*********************************************************************
RETURN   EQU   *
         LR    R5,R15                    SAVE THE RETURN CODE
         LR    R1,R13                    GET STORAGE ADDRESS
         STORAGE RELEASE,LENGTH=WORKLEN,ADDR=(R1)
         LR    R15,R5                    COPY RETURN CODE
         PR    ,                         POP THE STACK AND RETURN
*********************************************************************
RETURNØ4 DS    ØH
         LA    R15,4                     SET RETURN CODE
         B     RETURN                    RETURN
*********************************************************************
RETURNØ8 DS    ØH
         LA    R15,8                     SET RETURN CODE
         B     RETURN                    RETURN
*********************************************************************
RETURN12 DS    ØH
         LA    R15,12                    SET RETURN CODE
         B     RETURN                    RETURN
*********************************************************************
RETURN16 DS    ØH
         LA    R15,16                    SET RETURN CODE
         B     RETURN                    RETURN
*********************************************************************
RETURN2Ø DS    ØH
         LA    R15,2Ø                    SET RETURN CODE
         B     RETURN                    RETURN
*********************************************************************
RETURN24 DS    ØH
         LA    R15,24                    SET RETURN CODE
         B     RETURN                    RETURN
*********************************************************************
*                                                                   *
*   CONSTANTS                                                       *
*                                                                   *
*********************************************************************
FLDLIST1 DC    F'3'
         DC    C'PASSWORD'
         DC    C'PASSDATE'
         DC    C'FLAG4   '
*********************************************************************
RACROUT1 RACROUTE REQUEST=EXTRACT,                                  X
               TYPE=EXTRACT,                                        X
               CLASS='USER',                                        X
```

```
               RELEASE=1.9.2,                                          X
               MF=L
ROUTLEN1 EQU   *-RACROUT1
*********************************************************************
RACROUT2 RACROUTE REQUEST=EXTRACT,                                     X
               TYPE=ENCRYPT,                                           X
               ENCRYPT=(*-*,DES),                                      X
               RELEASE=1.9.2,                                          X
               MF=L
ROUTLEN2 EQU   *-RACROUT2
*********************************************************************
         LTORG
*********************************************************************
*                                                                     *
*   MODULE WORKAREA DEFINITION                                        *
*                                                                     *
*********************************************************************
WORKAREA DSECT
SAVEAREA DS    18F                     SAVEAREA
#PARMØ   DS    F                       INCOMING PARM ADDRESS
#PARM1   DS    F                       ADDRESS OF USERID
#PARM2   DS    F                       ADDRESS OF PASSWORD
RETCODE  DS    F                       RETURN CODE SAVE AREA
EXTADDR  DS    F                       EXTRACT AREA ADDRESS
CURRPWDL DS    F                       LENGTH OF ENCRYPTED PWD
CURRPWD  DS    CL8                     CURRENT VALID ENCRYPTED PWD
PWDL     DS    XL1
PWD      DS    CL8
ENCRYPT1 DS    CL9
ENCRYPT2 DS    CL9
ENCRYPT3 DS    CL9
ENCRYPT4 DS    CL9
EXPDATE  DS    CL3                     PASSWORD EXPIRE DATE
REVKFLG  DS    CL1                     REVOKE STATUS FLAG
ROUTWRK1 DS    ØD,CL(ROUTLEN1)
ROUTWRK2 DS    ØD,CL(ROUTLEN2)
RACWORK  DS    ØD,CL(512)              RACROUTE WORK AREA
WORKLEN  EQU   *-WORKAREA
         PRINT NOGEN
         IRRPRXTW
RØ       EQU   Ø
R1       EQU   1
R2       EQU   2
R3       EQU   3
R4       EQU   4
R5       EQU   5
R6       EQU   6
R7       EQU   7
R8       EQU   8
R9       EQU   9
R1Ø      EQU   1Ø
```

```
R11      EQU   11
R12      EQU   12
R13      EQU   13
R14      EQU   14
R15      EQU   15
         END
```


RACF in focus – DSMON reports

'RACF in focus' is a regular column focusing on specific aspects of RACF. In this issue, we look at DSMON and its various reports, review the valuable information they contain, and consider how to maximize their usefulness.

WHAT IS DSMON?

DSMON (Data Security Monitor) is a program that comes with RACF. The reports it produces can be used to find any high-level weaknesses in the way you're using RACF, and are often used by auditors. RACF security specialists therefore need to know how to run and interpret them, so as to identify and resolve potential security exposures ahead of an IT audit.

You should run DSMON reports regularly to make sure no potential security exposures are creeping in, and also to familiarize yourself with the contents of these reports. Note that DSMON should be used in conjunction with other audit methods; it should not be the only means of measuring the health of security at your installation.

HOW TO PRODUCE DSMON REPORTS

Naturally, DSMON reports contain sensitive information – to run them you need the RACF AUDITOR attribute.

DSMON produces a set of ten reports. You have the choice of

running all the reports, or just a subset of them. For more information on DSMON, see the *z/OS Security Server RACF Auditor's Guide*.

The following JCL will produce all ten reports:

```
//STEP1         EXEC  PGM=ICHDSMØØ
//SYSPRINT  DD     SYSOUT=A
//SYSUT2      DD     SYSOUT=A
//SYSIN        DD     *
    FUNCTION ALL
/*
```

VALUE OF DSMON REPORTS

System report

As the name implies, the system report provides information related to the operating system environment – CPU-ID, model, RACF version and release, whether RACF is active in the system, etc.

Value of this report

There are times when you need to know system-specific information such as the RACF release level you're running, and this is a good place to find it. In the case of multiple RACF databases, and after a RACF release upgrade, you should make sure you're running the right version of RACF.

Group tree report

The group tree report shows how your RACF group structure looks. It's not an easy report to read, because the group tree typically runs to several pages, and there are no graphics.

Value of this report

You can use this report to get the overall view of your group structure – is it flat? Is it hierarchical? Does it make sense? Can it be improved upon? If you're giving group-level special privileges (SPECIAL, OPERATIONS), this report will give you an idea of what this means in terms of security administration.

Also, you may be surprised to find groups (or sub-groups) that are no longer valid, in which case it's time to do some clean-up!

Program properties table report

The Program Properties Table (PPT) is an MVS table that contains, mostly, operating system programs capable of bypassing RACF security. The table is maintained by the z/OS system programmer.

Value of this report

Because the programs contained in this report can bypass RACF security, you need to ensure that each and every entry in this list is a valid one. To do this, you'll need to work with your z/OS system programmer. Normally, the entries in this report should not change. Find out if new programs are showing up (you can do this by running DSMON periodically and comparing two successive reports). If you find new programs, you should get a justification for their addition by talking to the system programmer.

RACF authorized caller table report

The authorized caller table report shows programs that can make RACF calls in an authorized state, and thus change critical RACF control blocks (for example, the ACEE).

Value of this report

Normally there should be no entries in this table. If there is one, you should find out who put it there (your z/OS system programmer should be able to help you on this one).

RACF class descriptor table report

The class descriptor table report shows all the resource classes defined at your installation, and, for each resource class, provides information such as whether the class is active, the default universal access for the class, whether there is auditing turned on for the class, etc.

Value of this report

Use this report to verify that resource classes that should be active are indeed active. Quite often, someone with special powers may deactivate a resource class for some testing, and forget to reactivate it. If the class isn't one of the important ones, this mistake may go unnoticed. You can also use this report to review your audit options for each class.

Also, review the default universal access (UACC) for each class. This will apply when you don't specify a default in any profile you create for this class.

RACF exits report

The RACF exits report shows RACF exits that are active at your installation. It also shows the module length (size) of each exit.

Value of this report

RACF exits can bypass security checking. All the security you've defined in RACF profiles means nothing if an exit is programmed to bypass some aspect of security checking. And this can happen without your knowledge if you're not aware of exits implemented at your installation. RACF exits are coded in Assembler language, and may therefore be difficult for most security administrators to understand.

If this report shows any active RACF exits, make sure you know what they do. You should also compare the size of the exits between two runs of this report, to ensure that the module length hasn't changed without your knowledge. A change in the module length generally signifies a change in the processing done by the exit.

Work with your z/OS system programmer to find out what each exit does.

RACF global access table report

The global access table report shows classes that are activated for global access checking – a typical example is the dataset

class, but there may be others. The report also shows, for each active class, the profiles eligible for global access checking, together with the level of access allowed. Entries in this table are checked before a check is done in the RACF database. If access is allowed in the global access table, the database check is bypassed.

Value of this report

Global access checking is designed for performance reasons. In cases where you want to grant universal access to a profile, and the profile is heavily used, it may be advisable to put this profile and the desired universal access in the global access table. But remember to put the corresponding universal access in the actual profile too. If you do not, and the GLOBAL class somehow becomes deactivated, there will be a lot of access failures.

Another way to use this report is to compare the universal accesses in it with the ones specified in the corresponding profiles, and ensure that they're the same.

RACF started procedures table report

The started procedures table report shows all the started procedures at your installation (all profiles in the STARTED class). It shows special attributes assigned to each started procedure (privileged or trusted).

Value of this report

Since trusted and privileged attributes allow a started procedure to access all information without specific checking or authorization, you should ensure that started procedures with either of these attributes really deserve these special powers. Also, if you notice that new entries have shown up in the report, you may want to verify with the system programmers or the operations department that these entries are legitimate.

You should also check whether the named started procedures actually exist in procedure libraries (proclibs). If not, consider removing the entry from RACF. Check, too, whether these

proclibs are adequately protected. Are there auditable processes in place to update members in these proclibs? If not, you may have security exposures. It's best to work with your systems and operations people to identify inappropriate entries in this report.

Selected user attribute report

The selected user attribute report shows user IDs with the SPECIAL, OPERATIONS, and AUDITOR attributes. The report tells you whether the attribute is system-wide, or at the group level. It also shows user IDs that have been revoked.

Value of this report

Use this report to monitor the use of special attributes. Generally, user IDs with the SPECIAL attribute should be kept to a minimum. Userids with the OPERATIONS attribute should be reviewed, as this may not be desirable (see 'RACF in focus – OPERATIONS attribute', *RACF Update* 29, August 2002, pp 13-15).

You can also use this report to conduct a review and validation of these special attributes. Typically, this involves having the supervisors or managers of the user IDs confirm and approve the special powers.

You can also use the report to enforce the 'segregation of duties' principle. This states that no one person should have multiple special privileges. For example, if a person has both SPECIAL and OPERATIONS attribute, there is a potential security exposure.

Lastly, the report can be used to clean up user IDs that have been revoked.

Selected datasets report

The selected datasets report lists several important datasets (APF-authorized datasets, linklist libraries, the RACF databases, system catalogs, etc). For each, it tells you whether there is RACF protection, and what the universal access (UACC) is.

Value of this report

APF-authorized libraries contain programs that can bypass RACF security, and should therefore be tightly controlled. Use this report to ensure that all the datasets listed have RACF protection (RACF PROTECTED should be YES in the report). If not, you may want to create a new profile to protect the dataset. Also, go through the report and make sure that universal access (UACC) for each dataset is NONE or READ; if you see a UACC of UPDATE or ALTER, you may have a security exposure. You should also go through the access lists of all the corresponding profiles periodically, to ensure that the accesses permitted are what you would expect.

If an APF-authorized library in this report is flagged as 'not found' (NF), there may be a security exposure. Resolve the issue with your system programmers.

Work with your system programmers to make sure all APF-authorized and linklist libraries are still valid. Software products may have been removed or retired, or libraries may belong to older releases of software products, making them obsolete.

Dinesh Dattani (dddattani@rogers.com)
Security Consultant (Canada)

How to survive a RACF audit

One of the responsibilities of the RACF administrator is to ensure that adequate security controls are in place, and that policies and standards are being adhered to. Eventually, the RACF administrator will be faced with an internal or external audit of security controls, typically within the scope of an audit that covers MVS security or multiple platforms. Although audits vary in both scope and depth, the areas described below are common to most, and following the steps outlined in this article will help you survive your RACF audit.

AUDIT PREPARATION

Several steps can be taken to prepare for the audit and mitigate any subsequent findings, including the following:

- Request in writing that the audit team provide a scope of the upcoming audit. The scope will detail the areas of concentration. Review the scope statement and examine the areas relating to RACF or security. We have frequently encountered scope statements that were either unclear or vague in their intent. When that occurs, promptly notify the audit team of the issues and ask them for clarification. This will prevent any misunderstanding of the audit scope.
- Run a DSMON to obtain a snapshot of the environment. Review the listed RACF groups and determine which groups are obsolete (those that have no userids connected or own no datasets). Review those userids with the OPERATIONS and SPECIAL privilege and determine whether the privilege level is appropriate for the function of the userid. Review the Class Desriptor Table information and become familiar with the active RACF classes. *(Editor's note: for more details on DSMON, see 'RACF in focus – DSMON reports' on pp 15-21 of this issue.)*
- Run a report to identify any obsolete groups or userids

remaining in access lists. Remove obsolete entries from all access lists.

- Review Global RACF (SETROPTS) options and be prepared to discuss these with the auditors.
- Review any profiles in WARN Mode and determine what is needed to change these profiles to FAIL mode.

WHAT THE AUDITORS WILL EXAMINE

The auditors will examine the following areas.

Global RACF (SETROPTS) options

The audit team will review all of the global RACF (SETROPTS) options currently in effect. Some of the areas they will concentrate on are as follows:

- *Protect-all status.* If protect-all fail is in effect, all datasets will be protected, regardless of whether a RACF profile exists or not. If protect-all warn is in effect, only those datasets covered by a RACF profile will be protected. Auditors will include the lack of protect-all fail mode as a finding in their report. Note that if non-standard dataset high-level-qualifiers are used (ie one-character high-level qualifier), ensure that the naming convention conversion table is accurately translating the non-standard qualifier to one that is recognized by RACF. At a company where I used to work, the naming convention table was not translating correctly and the RACF profiles were not recognized, resulting in a warning each time the file was accessed. The high-level-qualifiers had to be altered to those that were recognized by RACF.
- *Inactive userids are being automatically revoked after x number of days.* Auditors will recommend that inactive userids be revoked after a period of 30-90 days. If the number of days specified is much higher than 90, there's a high chance that this will be reported as an audit finding. One id of special interest to the auditors is the vendor-

supplied default userid, IBMUSER. Ensure that this userid is not being used and has been revoked.

- *RVARY function.* Make sure that the RVARY passwords for the switch and status function are changed from the vendor-supplied default. The audit team will also review protection over issuing the RVARY switch and status functions.
- *Password controls.* The review will examine existing password controls:
  - What is the password change interval at your site? The standard is usually 30-45 days.
  - What kind of password history is being maintained? A password history between three and six is a customary standard.
  - How many invalid password attempts will result in a revoked userid? Most auditors will recommend that a policy of three invalid passwords before revocation be set as the standard.
  - What are the password syntax rules (minimum and maximum characters, composition of characters)? A standard password length is between six and eight characters, composed of alpha/numeric characters. At least one character should be a required numeric.

Policies and documentation

The focus in the area of policies and documentation will be on the existence of an overall Information Security policy. Does such a policy exist? If so, does it address RACF global standards (ie password formats, password and userid naming conventions, and information classifications)? Is it being adhered to? The auditors will also seek documentation on Service Level Agreements and RACF administrative procedures. If there are no procedures in place before the audit, begin drafting procedures immediately. The auditors will view this as a work-in-progress and any findings will be mitigated.

Audit trails and reviews

Auditors will look at three areas in the section on audit trails and reviews.

- Are RACF audit trails being generated? There should be a mechanism to generate audit trails that include the following:
  - invalid passwords/sign-ons report
  - resource violation report
  - dataset violation report
  - activities of privileged userids (ie OPERATIONS, SPECIAL)
  - Userid modification report
  - RACF warnings report (accesses against resources in WARN mode)
- Are audit trails being reviewed? Audit trails should be reviewed on a daily basis. Any items that require subsequent investigation should be documented and kept for a designated period of time (one to two years).
- How are incidents being evaluated and reported? What criteria are used for selecting incidents for subsequent review? What are the next steps if there is evidence of misuse? There should be incident reporting and escalation procedures in place for the resolution of security incidents.

Set-up and removal of userids

Some of the questions to expect from the audit team about the set-up and removal of userids are as follows. Are there procedures for the set-up and removal of userids? Who authorizes the requests? How do userids get removed from the system? How are the administrators notified of terminations? How many userids have OPERATIONS and SPECIAL attribute? What userids have the Auditor privilege? How many Group Specials are there, and what functions do they perform?

One of the first things auditors will look for is the existence of generic userids. Generic ids provide no accountability and should not be used for updating any resources. I've experienced several business units requesting generics for temporary staff, due to a large and rapid turnover. To provide some accountability, I assigned them to the hiring manager, or requested the names of the people using the ids on a daily basis. Access was limited to inquiry whenever possible.

Auditors will also review termination lists to ensure that userids were deactivated on a timely basis. It's a good idea to obtain a listing of cumulative terminations from Human Resources before the audit, and check that their userids have been deactivated.

The team will want to know how accesses are adjusted when a user transfers to another department. This requires some coordination with the two departments (departing and incoming) and notification from at least one of them.

Logical security – access to resources

Part of the review of logical security will include examining the process of granting access to resources. How is this requested and authorized? Is approval required from the manager or resource owner? Are there processes and procedures in place? How are the CICS system transactions protected (ie CEMT, CEDA, etc)? Are production CICS resources separated from test and development resources? Who has update access to production datasets? Who has update access to system datasets (ie SYS1.Parmlib, Proclib)? How are APF authorized libraries protected? How is security interfaced with third-party products (eg CA7, DB2, Endevor, etc)? Who has access to the Bypass Label Processing function? Have the RACF database files been adequately protected?

Before the audit, review all rule profiles for system and production files and transactions. Ensure that all system transactions, files, and libraries are adequately protected. Remove any obsolete or inappropriate entries. Remove any RACF groups from access lists that no longer exist or have no userids attached.

Once the audit is over, it's usual to meet with the auditors to review the audit findings and resolve any outstanding issues. I've usually found that many audit findings can be negotiated with the audit team. The schedule for correcting audit findings is discussed at this point. Some items can be corrected on the spot and reflected as such on the audit report.

Request a draft of the audit report for review, before it's issued to management – this is your last chance to catch any errors and resolve any inconsistencies.

Bruce Josephs and Joel Mandelkorn
(USA)

RACF restructuring: implementation

The fourth and final article in our series on RACF restructuring concentrates on implementation. For part one of this series, see RACF Update *27, February 2002, pp 8-22; for part two, see* RACF Update *28, May 2002, pp 35-50; for part three, see* RACF Update *29, August 2002, pp 20-36.*

WHAT WE'RE DOING TODAY

In the final episode of this seemingly endless drama, we'll look at the actual preparation, transfer, and implementation of your tested RACF database into the 'live' environment. You'll see how to broadcast your intentions to all of the people that the changes will affect. You'll find out some neat tips on how to get your user community to accept new user ID structures. We'll look at the actual preparations required for the transfer of your new database from the RAC1 test LPAR to its new home. You'll see how to do the actual transfer itself, and what steps are vital to ensure that the database 'catches' properly in the new environment. You'll learn why the first 48 hours after the transfer are so critical, and what you need to do to address this. Also, you'll get helpful hints on how to handle the inevitable crises that will arise. We'll discuss how to handle the final test of any database transfer – your user community – and how to provide them with adequate support to iron out any final errors, bugs, or boo-boos. Finally, we'll discuss what steps you should take after the mayhem has subsided and everything is running tickety-boo.

INFORMING YOUR USERS

If you've been following the standard structure of the Systems Development Life Cycle (SDLC), then you know that any change to the system must undergo an approval process. That should also include the scheduling of the proposed transfer of the new and improved RACF database. Generally, the following

people should sign off (and I mean a physical signature here, because verbal approval can be easily denied later):

- Technical support
- Computer operations management
- Production control
- Programming department management (for development and test LPARs)
- Business department management (for production LPARs)
- Quality control
- Test management
- Information technology division management
- Internal audit.

Naturally, you want to make sure that you include the date of the proposed transfer. Here's a tip: wait for a three-day weekend, if possible. Don't schedule this around the Christmas-New Year time frame, though – that's the last time anyone wants to do real work. Oh, and forget about using Thanksgiving to extend a four-day weekend. Your spouses and/or loved ones will disown you.

Make sure that everybody is comfortable with the date you've selected. If they're not, and they can give you good cause for their apprehension, be flexible and change the date. You're going to need a whole lot of support when you do the changeover.

But these aren't the only people who need to know what's going on: once a date is set, you're going to have to tell the users. And make sure you give them plenty of warning – at least two weeks is a good rule of thumb. Use e-mail messages, fliers, bulletin boards, your company's intranet, newsletters, skywriters, etc, but make sure that you've exhausted every means at your disposal to ensure that people are aware. (Though this won't of course stop some people claiming that they knew nothing about it. …)

About three days before you do the transfer, send out a 'final warning' message to the users. Let them know that they should save all of their PDS, database, and flat files. Make sure they know that they should also change any JCL to reflect their new user ID. For TSO users, remind them that all of their files using their user ID as the High Level Qualifier (HLQ) will be renamed to their new user ID, and to make the appropriate changes in any jobs that use those files. This can take some time, which is why you should give them three days. Hopefully, most of this will have been done when you sent out your original alert 11 days before this warning.

One more thing: remind all your users that this changeover is confidential, ie not to be discussed outside the organization. You don't want to let the world know about this kind of thing, because somebody might want to use the confusion that inevitably follows a major system changeover to their advantage and your company's detriment.

PASSING ALONG NEW USER IDS WITHOUT INJURING YOURSELF

If your user ID structure was a mess before you decided to restructure RACF (and the odds are that it was), you'll need to generate new user IDs for the staff. You've done the coding for it, you've transferred the permissions, etc, but now you've got to inform your staff of their new IDs. And this must be done in a secure manner.

This is one instance where MS Word comes in *very* handy. You can create a draft letter, and then use the mail merge facility to create the individual letters. If you took my advice a few issues ago and created a database, your job will be a lot easier. If you didn't, well, see what happens when you don't listen?

The memo should tell the user their old and new user IDs, as well as their initial password. Make sure they understand that it is a 'one-time-only' password, and that they'll have to change it at their first log-on. This is also a good place to remind them of the password structure, how often it needs to be changed, basic security rules, etc.

Oh, one point here. Try not to give everybody the same password. There are plenty of random generators out there that will give you eight-character strings of numbers and letters. They may be harder to code in your JCL, but they're a lot more secure.

You should also make sure that these forms are distributed in a secure manner. That means that you'll be stuffing envelopes and putting mailing labels on them. Put the mailing label on the *back* of the envelope, over the flap, to ensure that it hasn't been opened or tampered with. Alternatively, you can put the mailing label on the front, and use a stick-on seal of some sort over the flap on the back. A company logo is best – a smiley face shouldn't even be considered.

You'll need to make sure that you include a 'receipt' that must be signed by the user and returned to you. The receipt can double as a 'terms and conditions' document so that the user understands his/her rights and responsibilities regarding system security. Another bit of insurance you should use is a delivery log – the mail delivery staff will need to obtain the signature of the user when they receive the envelope, and shouldn't leave until they've collected the signed receipt as well. And no-one should sign for or accept the envelope for a co-worker. This is hand-delivery in its truest sense.

Appendix A shows a draft memo for the new user ID, and Appendix B shows a draft of the 'terms and conditions'.

PREPARATIONS FOR DATABASE TRANSFER

You've tested your new database out on the RAC1 test LPAR, so you should just be able to plug the new one into your recipient LPAR and everything will be both hunky and dory, right? Well, perhaps this would be true somewhere else – say, for instance, on another planet. For us mere Earthers, though, it's a different story.

Remember, you're trying to do an update to what is, ostensibly, a moving target – your old RACF database will be continuing to evolve while you're coding and testing the new one. So, before

you start patting yourself on the back, you'll need to do the tedious and boring task of making sure that all the updates implemented on the old database are incorporated into the new one. This will save you many headaches and heartaches as your new database goes 'live'.

There are two ways to do this. The easy way is to duplicate any changes made on the old database at the same time on the new one. This involves dual coding, I know, but it cuts down on the problems you'd have in the second option. That option – the hard way – is to save up all of the RACF change documents and do them all in one fell swoop. This concentrates all the work into one small period of time, instead of spreading it around. It also reduces the ability to test the updates in the new database, which means that you can end up with security problems in the real world. The easy way is the better choice here (and you don't hear that very often, do you?).

You've also got to get some major code changes prepared for your TSO user files. If you're changing the user ID structure (and you'll almost certainly have to do this), you'll need to get production control or technical support to create some jobs that will change the HLQ on those files from the old user ID to the new user ID. I'm not going to get into how to code this for you. Your technical staff have a wide variety of tools they can choose from to handle this odious chore.

Don't forget to bring *all* of the documentation generated during the testing of this LPAR's new database. You may need to trace a problem back to its source, and this is the best way to do it. Be sure you have the documentation in both hard and soft copy. Make sure you've brought the RACF changes you've made on the target LPAR as well, in case you run into any access conflicts or violations that you can't immediately identify. A soft copy of the RACF manuals would be a wise precaution. And make sure you've got a dedicated PC available for your use. Some of the work you'll be doing will be on the system console, but you should have a PC available for your other work – keeping a log of changes, saving JCL code to diskettes, playing solitaire, etc.

Okay, now we've covered the technical issues you need to address when preparing for the database transfer. But what do *you* need to do to prepare yourself for the task ahead? Remember, you'll be virtually living at your computer centre for several days. Yes, working and eating and sleeping (though not too much of that last one). This isn't one of those things you can do from an office, or just wait for a phone call for. No matter how hard you've prepared, no matter how thoroughly you've tested, there is one universal truth: Things Will Go Wrong.

There are several things you need to bring to the computer centre to ensure a minimum level of comfort:

- Two changes of clothing and three of underwear.
- Extra socks.
- Comfortable shoes and/or slippers.
- A sweater.
- A good-quality sleeping bag and pillow.
- Long underwear and a woolly hat that can cover your ears
  - Trust me on this one.
- Aspirin, Tylenol, or Paracetamol.
- Toothbrush, toothpaste, and extra deodorant.
  - Everyone will thank you for these three items.
- One packet of cigarettes (minimum).
  - Note: for you non-smokers, this is a good chance to start!
  - Note: remember, you can only smoke outside the computer centre.
- Anti-depressants.
  - Avoid tri-cyclics if at all possible.
  - Selective Serotonin Re-Uptake Inhibitors (SSRI) are your best bet here.

  - Start taking these at least four to six weeks before the transfer, in order to get the full effect.
- Valium or any mild form of benzodiazepam (low dosages only).
- USD $1,000 or equivalent local currency.
- Discount coupons for local fast food restaurants that deliver.
- Your favourite teddy bear.
  - At times in the transfer process, this will be your only friend.

DOING THE DIRTY DEED ...

The date has arrived. Everybody is ready and waiting. Doughnuts have been served. You're now going to get things started. So what on earth do you do?

First, make sure that everything, and I do mean *everything*, has been backed up. System files, programs, data files, the whole nine yards. Also, you'll want to make a separate back-up of your old RACF database to a tape and to a non-SYSRES pack. You should make sure that all the data file back-ups from the previous day are available as well (you'll see why later).

Now, copy the new RACF database to a non-SYSRES pack (contiguous tracks) and prep an IRRUT400 job to copy that file into SYSRES, with a different name to your old database. You'll need to do this twice (primary and back-up, of course, with the back-up going to the SYSBAK pack). You're now ready to do the switch with the RVARY commands. Make sure you've got the passwords for this in your hand.

Once the switch is completed, have the operators perform a complete IPL. Don't bother with a hot start or a warm start – get down and dirty and do a cold start of the system. You're going to have to ensure that the system can start from a complete power-off status anyway, so you may as well try it first.

Now, monitor the operations log very carefully as the system comes up. The early stages (pre-RACF implementation) shouldn't cause you any problems, but things will get interesting when RACF activates. Make sure your operators tell you if anything on the log looks out of the ordinary, or if there are problems in the start-up sequence. They've seen these IPLs a few dozen times by now, and will have a feel for when things are acting oddly. Rely on that experience. And note any specific error messages or Started Task problems that crop up. If they're RACF-related, you'll need to do some fast corrections (making sure you enter any of the problems into your error log, of course), and restart the task.

Once you've got a clean IPL, and have corrected any problems with the Started Tasks, have the operators run another IPL – this time a warm start – and ensure that you don't have any repetition of problems from the cold start. You should have a clean start this time.

At this point, you'll want your production control or technical support staff to do the HLQ renames from the old user IDs to the new. This is one of those very *very* rare occasions where I feel that the OPERATIONS profile on a RACF user ID is warranted. Make sure, however, that once the renaming has been completed, you remove OPERATIONS from that ID.

Now, begin doing testing on a selection of normal processes for that particular LPAR – CICS regions, batch jobs, compiles, TSO usage, whatever is generally performed on that system. These tests should be assisted by someone from QC, technical support, and a selection of business areas.

THE FIRST 48 HOURS

Now the fun part. Remember when we saved all of those files earlier? You'll now set the system back a full day and repeat your normal processing for that day, with the new database in place. The output can be compared with the previous day's output if you wish, but remember, you're testing the RACF

portion of the system. If your QC guys want to go over hash totals and balance sheets, let them have their fun. But your main goal is to see if you run into any RACF error messages while going through a normal process cycle for that LPAR. Keep a weather eye on the console log (you can do this from TSO, by the way, so you can scroll back on the SDSF log and do searches) and note *every error*! You'll spend the next several hours picking through each error, and working out the correction(s) required to ensure that a) system security isn't compromised and b) the message doesn't return.

If everything turns out okay, you're ready for the next phase. Restore the data files again, and get your QC testers to do the same tests that they did in the RAC1 LPAR. Check all of the CICS transactions, have the programmers do compiles, do queries on SDSF, everything. You'll be looking for any errors or problems you missed in the RAC1 tests, and fixing them post haste.

Now, I can hear you saying, "That's all well and good for development or test LPARs, but what about production? What if you're a 24/7 operation that can't afford to interrupt regular processing?" These are very good questions, and here's a very good answer – one that will keep your internal and external auditors happy as well. Switch your production to a separate LPAR or site. You should already have business continuity/ contingency planning/disaster recovery (pick whichever name you like best) plans in place to handle such things. If you don't, well shame on you! But you can do this exercise as a rather nice by-product of your RACF restructuring, and you'll get an extra feather in your cap. Just make sure that any transaction processing done over the three-day weekend can be reprocessed on your primary production LPAR and incorporated in time for when your users get back to their desks.

WHAT TO DO IF IT ALL GOES WRONG ...

Let's face the facts here. Things can go really wrong, really fast, in any major project. Sometimes faster than you can normally

cope with. The questions is, what do you do about it? And when do you say, “Okay, we can fix this, let’s keep going” or “We can’t fix this, we’ll have to take this back to test”?

Back-out strategies are fundamental to any good SDLC project, and this one is no different. You have to be able to bring your system back to a ‘last working copy’ status in a short amount of time, should the need arise. To do this, you’ll need to know a few things:

- How long does it take to restore all systems and files to the original point?
- What determines a ‘do or die’ scenario?
- Who will make the final decision?

The first point is vital in deciding when you’ve got to make your final decision. If your systems must be available at 08:00 on Monday and it takes 10 hours to restore everything, then your point of no return must be 22:00 the previous Sunday. If you’re busy trying to fix major system failures at 21:59, and there seems no end in sight, it’s time to call it quits and get things back into order.

The second point is more nebulous. ‘Do or die’ can be problematic, and depends on the scale and scope of the problem itself. If you’re having intermittent problems with a minor system that doesn’t have critical impact, you can limp along with the error for a day or two. If it’s your 20-squillion transaction per second money-maker that has a maximum downtime measured in femtoseconds, and finding a solution requires looking through the Hubble, you’ve reached a level of criticality that can make ‘do or die’ a real issue.

That’s when you have to gather the various people together in a room and hash out the problems. This must be done in a fast and concise manner, so no doughnuts. Get opinions, get options, and then get a decision. Who makes that decision? The project manager – namely, you (or your boss) – is the final arbiter. This is the person who ‘carries the can’, as it were. The contents of the can, of course, are dependent on whether or not

things work out well. If they do, the can is full of gold. If they don't, well, that ain't gold in the can, that's for sure. ...

THE REAL TEST!

So far you've been playing with the system – testing it to see if it fits your criteria, making sure the technical support and operations and QC staff can work with it. But now you've got the ultimate test: users.

If your office is open for business at 08:00, expect the telephone to start ringing off the hook some time around 08:00:01. Oh, of course you'll get a few early birds in the office, but they're simple and well interspersed. The flood of calls will seem like a 50 metre tsunami. And nine times out of ten, it'll be dumb calls:

- "What happened to my old user ID?"
- "Why won't the system accept my old password any more?"
- "Why won't the system accept my new password now?"
- "I can log on, but I can't find any of my old files. Where are they?"
- "Why won't my JCL work?"
- "What is the nature of the universe?"
- "I'm getting a weird error message, but I don't remember what it said. What does that mean?"
- "Can you reset my password? I forgot the one I just created."
- "I don't like this new user ID. Can I go back to the old one?"

You should make sure to route all of these calls to your office at the data centre, and not your regular office. There are several good reasons for this. First, you won't end up with all of the normal office distractions, and can concentrate solely on fixing any and all problems that come your way. Second, a data centre is (or should be) a secure facility with very limited access, so users will have a harder time finding you in order to beat you up.

Third, the data centre is going to end up with teething problems on that first day anyway, so it's better to be on-hand to help out when the occasional crisis arises. You'll find this third one a welcome distraction from irate phone calls.

Users, as a breed, have very low tolerance for change. But they also have even shorter attention spans. The initial problems do blow over after the first day or so. Oh, you'll get some schmoe who was on holiday while you were doing all your work, and he or she will be caught a bit unaware, but that's much more sporadic, and much easier to handle.

As the day wears on, and turns into night, the phone calls will abate, and the nightly batch processing will begin. This is the first 'real' night of processing, and you'll need to be nearby to fight the occasional security brush fire. But if you've done your testing properly, you might get a few hours of sleep here and there. Cherish that sleep well, and take it whenever you can get it.

On the morning of the fifth day, after you've worked 96 hours straight, go home. The rest of your security staff should be able to handle the higher-than-normal user requests for the next couple of days. Be sure to hand over *all* of your notes, files, etc. Then roll up your sleeping bag, grab your dirty laundry, and, clutching your beloved teddy bear to your chest, have someone drive you home. Sleep for 24 hours or so, and you'll feel a whole lot better.

USER SUPPORT – YES, YOU HAVE TO DO THIS...

Oh, how secure our mainframe would be if we didn't have one little thing – users! They're always getting into things they shouldn't, or trying things they ought not to, or going where they mustn't. But, in the end, their mistakes are our continued employment opportunities, so I guess we do owe them some support.

User support is an administrative function. And you have to do this whether your RACF database is in pristine condition, or looks like a scrap metal sculpture done by a psychotic artist on

really bad acid. But for the purposes of this article, the way you support your users can either keep your RACF database looking and operating well, or let it devolve into what you just spent months of effort trying to fix.

So, let's start with some rules:

1 Only Security creates/updates/deletes RACF profiles. No Group-Special profiles allowed.

2 All RACF updates are done with formatted JCL.

3 All RACF updates must be supported by requests, signed by the owner of the resource.

4 All requests are matched against activity reports the next day.

5 All profiles identify primary and secondary owners.

6 Unused profiles (especially user IDs) are regularly checked and purged.

7 Anyone in your Security department who violates Rule 1 through 6 will be shot.

Once you've got a clean RACF database, it's relatively easy to maintain, provided you follow those simple rules above. You have to be quite strict about this though (see Rule 7), in order to keep the database functional, structured, informative, and clean.

WHAT TO DO AFTER YOU'RE DONE

You've spent months working on this project. You've created RACF databases for three separate LPARs, and have tested and implemented each one. You've nurtured this project like a parent, and have seen serious problems turn into major successes. You're tired, exhausted, and feel like you've aged about a decade. What should you do?

Well, since Disneyland is too energetic, have a 'wrap' party with the rest of the implementation team. A nice restaurant, a

chance to kick back and relax, and let off a bit of steam. Have a good meal, a few laughs, and swear to each other that you'll never do a project like this ever again in the history of the universe.

The next morning, get on a plane to Barbados. Rent a hut for three to four weeks. No telephone. No television. No beeper. No radio. No newspapers. But lots and lots of rum.

Just listen to the sound of the waves and the shore, throw your watch away, and forget you even heard of RACF. After all you've just gone through, you've earned the right.

And above all, DON'T TELL ANYBODY WHERE YOU WENT!!!

APPENDIX A: DRAFT MEMO FOR THE NEW USER ID

YOUR NEW USER ID

Name:	Doc Farmer
Department:	Information Security
Old user ID:	Farmer01
New user ID:	A234567
New password:	XG8H2V3R

DO NOT SHARE YOUR USER ID WITH ANYONE
DO NOT SHARE YOUR PASSWORD WITH ANYONE
KEEP THIS INFORMATION CONFIDENTIAL
DESTROY THIS FORM SECURELY AFTER YOUR FIRST SIGN-ON
DO NOT USE UNTIL AFTER 14 WOMBAT 2002

APPENDIX B: DRAFT OF THE 'TERMS AND CONDITIONS'

PASSWORD RESPONSIBILITIES AND DUTIES

Please note that you must change your password after the initial password has been entered. The new password must be entered twice.

Here are some important things to remember about choosing, using, and maintaining passwords:

- Passwords must be changed every 30 days.
- The mainframe(s) will notify you for five days before your password expires.
- If you enter an incorrect password three times in a row, your user ID is automatically revoked.
- If your ID has been revoked, you must contact IS security to be reinstated.
- User IDs will automatically revoke access after 45 days of non-use.
- Passwords must be six to eight characters in length.
- You cannot use the same password month after month – the system keeps track of the last 24 passwords you've used.
- Choose a password that is easy to remember, but hard for others to guess.
- Don't write your passwords down or store them in a computer file.
- Don't share passwords with anyone (including IS security!).
- It's a good idea not to use the following as passwords:
  - Your name
  - Your department name
  - Your employee ID number
  - Names of months, days, years, etc.
  - Family names, birthdates, phone numbers, address numbers/names.
  - Easy to guess characters (ABCDEF, QWERTY, ASDFJKL, AAAAAAAA, ABABABAB, etc.).
- Remember, YOU are responsible for all actions taken on your user ID.

ACKNOWLEDGEMENT OF RESPONSIBILITIES

Your user ID will be activated and the associated secret password established for you to access the mainframe(s). You will be requested to change your system password when you first log in.

Under no circumstances should you share your user ID and password with any other person. If you do this you are not complying with *Company X* policy and *Company X* may take disciplinary action against you.

By signing this document, you acknowledge your responsibilities regarding the safeguarding of data within *Company X*.

NOTE THAT IF OUR OFFICE DOES NOT RECEIVE THIS ACKNOWLEDGMENT WITHIN 10 DAYS OF MAILING, YOUR USER ID WILL BE SUSPENDED AND SYSTEM ACCESS PRIVILEGES REVOKED.

USER SIGNATURE

USER NAME: _____ DATE: _____

USER SIGNATURE: _____

Doc Farmer (DocFarmer@qatar.net.qa)
Manager and Senior IS Security Analyst (Middle East)

Determining the RACF access level for a DSN

Users and programs often need to know what access they have to a specific dataset from the RACF user ID they're running under. Unfortunately, as many a frustrated user will attest, looking in the obvious place for such functionality results in a big disappointment.

ISPF AND ISMF

RACF's ISPF interface allows you to search and display RACF profiles via options 1.8 and 1.9. But nowhere can you specify a dataset.

A less obvious approach is to use the dataset list (DSLIST) options of ISPF (Option 3.4) or ISMF (Option 1). If you specify the DataSet Name (DSN) or High Level Qualifier (HLQ) for the dataset, you'll receive the following message from RACF if you have NONE access to the dataset:

```
ICH408I USER(JONPE   ) GROUP(DEMOUSER) NAME(JONPE              )
  USERCAT.SYSTEM CL(DATASET ) VOL(UCAT10)
  INSUFFICIENT ACCESS AUTHORITY
  FROM USERCAT.SYSTEM (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
```

ISPF 3.4 will then actually list the HLQ or DSN specified, with Catalog Error in the upper right corner of the panel. At first, the RACF ICH408I message appears to tell you what you want to know, but it's actually misleading as ISPF has only tried to read the catalogue, not the dataset.

Instead, use ISPF Browse or View (Option 1). Specify the DSN in single quotes ('SYS6.PROCLIB') in the DSN field, and hit Enter; you'll receive the following error messages and then see Authorization Failed in the upper right corner of the same ISPF Option 1 panel:

```
ICH408I USER(JONPE   ) GROUP(DEMOUSER) NAME(JONPE              )
  SYS6.PROCLIB CL(DATASET ) VOL(ISPW11)
  INSUFFICIENT ACCESS AUTHORITY
  FROM SYS6.** (G)
```

```
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
IEC150I 913-38,IFG0194E,JONPE,ISPFPROC,ISP21009,0311,ISPW11,SYS6.PROCLIB
```

This is useful. As well as confirming that your access to the dataset is NONE on the fifth line (ACCESS ALLOWED), the fourth line, beginning FROM, indicates that the RACF profile controlling this dataset is named SYS6.**, with the parenthetical G denoting a Generic profile type. It may also be useful to know that your user ID, JONPE, is in a RACF Group named DEMOUSER.

If you want to know whether you have UPDATE access, use ISPF Edit (Option 2), again specifying the DSN in single quotes ('SYS1.PROCLIB') in the DSN field, and hitting Enter. If, as in this case, the dataset is a PDS, select any member. Type SAVE in the Editor's command field, and hit Enter; there's no need to make any changes. If you don't have at least UPDATE access, you'll see the following messages:

```
ICH408I USER(JONPE   ) GROUP(DEMOUSER) NAME(JONPE               )
  SYS1.PROCLIB CL(DATASET ) VOL(OS39M1)
  INSUFFICIENT ACCESS AUTHORITY
  FROM SYS1.*.** (G)
  ACCESS INTENT(UPDATE )  ACCESS ALLOWED(READ   )
IEC150I 913-38,IFG0194E,JONPE,ISPFPROC,ISP21011,0308,OS39M1,SYS1.PROCLIB
```

Once you've hit Enter to clear these messages, you'll see the same Editor panel, with Authorization Failed in the upper right corner. Type the CANCEL command to exit.

PROGRAMMING

As you might guess, you could do something similar within a program by trying to open the dataset, or a member if it's a PDS. Perform an Open for Input or Open for Update, and then see if it fails.

SYSDSN is available in both REXX and CLIST. It's a TSO/E external function in REXX. Given a DSN, it returns a message: PROTECTED DATASET if the user ID has NONE access; OK otherwise. But note the odd behaviour in this test routine:

```
/* REXX */
say "SYS1.PROCLIB" "|"SYSDSN("'SYS1.PROCLIB'")"|"
```

```
say "SYS6.PROCLIB" "|"SYSDSN("'SYS6.PROCLIB'")"|"
say "SYS1.PROCLIB(TSO)" "|"SYSDSN("'SYS1.PROCLIB(TSO)'")"|"
say "SYS1.PROCLIB(A)" "|"SYSDSN("'SYS1.PROCLIB(A)'")"|"
say "SYS6.PROCLIB(A)" "|"SYSDSN("'SYS6.PROCLIB(A)'")"|"
The output is:
SYS1.PROCLIB |OK|
SYS6.PROCLIB |OK|
SYS1.PROCLIB(TSO) |OK|
SYS1.PROCLIB(A) |MEMBER NOT FOUND|
ICH408I USER(JONPE   ) GROUP(DEMOUSER) NAME(JONPE               )
  SYS6.PROCLIB CL(DATASET ) VOL(ISPW11)
  INSUFFICIENT ACCESS AUTHORITY
  FROM SYS6.** (G)
  ACCESS INTENT(READ   )  ACCESS ALLOWED(NONE   )
SYS6.PROCLIB(A) |PROTECTED DATASET|
```

The lesson is that, to test a PDS with SYSDSN, specify any member name, even if the member doesn't exist. Without a member name, you'll get OK for a PDS to which you have NONE access.

&SYSDSN is a built-in CLIST function. The following code is the equivalent of the REXX code above, and outputs identical results when you run it:

```
WRITE SYS1.PROCLIB |&SYSDSN('SYS1.PROCLIB')|
WRITE SYS6.PROCLIB |&SYSDSN('SYS6.PROCLIB')|
WRITE SYS1.PROCLIB(TSO) |&SYSDSN('SYS1.PROCLIB(TSO)')|
WRITE SYS1.PROCLIB(A) |&SYSDSN('SYS1.PROCLIB(A)')|
WRITE SYS6.PROCLIB(A) |&SYSDSN('SYS6.PROCLIB(A)')|
```

A RACF SOLUTION

At the beginning of this discussion, we seemingly gave up on RACF. But a detailed look at the *z/OS SecureWay Security Server RACF General User's Guide* reveals that RACF offers functionality beyond that provided by its ISPF panels. RACF has TSO commands, most notably LISTDSD, which tells you what access level the current ID has to the specified dataset.

Details of LISTDSD and the output it generates can be found in Section 6.4 ('Finding out how a dataset is protected') of the manual, originally published for z/OS Version 1.1.0, but still current for Version 1.4.0. The Order number is SA22-7685-00, but it can also be viewed on-line at:

http://publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/BOOKS/ICHZA100

Add a /6.4 to the end of that URL to get directly to LISTDSD.

LISTDSD doesn't have a lot of options – there are just two ways to code it:

```
LISTDSD DATASET('dataset-name') ALL GENERIC
LISTDSD DATASET('dataset-name') ALL
```

The first form lists any generic profile that protects the dataset; the second, a discrete profile. In theory, you should check for both, discrete first. In practice, most sites use only generic profiles, allowing you to ignore discrete altogether. However, it's important to know for sure. If both are being used, performing a LISTDSD GENERIC on a dataset with a discrete profile will show you only the generic profile that would protect it if the discrete profile didn't exist!

LISTDSD OUTPUT

If the dataset is protected by a generic profile and the current user ID has access other than NONE,

```
LISTDSD DATASET('SYS1.PROCLIB') ALL GENERIC
```

will output:

```
INFORMATION FOR DATASET SYS1.*.** (G)
LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ---------------    -------   -----
 ØØ    P39Ø            READ          NO      NO

AUDITING
--------
FAILURES(READ)

NOTIFY
--------
NO USER TO BE NOTIFIED

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    READ        SYS1            NON-VSAM

NO INSTALLATION DATA
```

```
CREATION DATE  LAST REFERENCE DATE  LAST CHANGE DATE
(DAY) (YEAR)       (DAY) (YEAR)     (DAY) (YEAR)
------------  ------------------  ---------------
164    95      NOT APPLICABLE FOR GENERIC PROFILE

ALTER COUNT  CONTROL COUNT  UPDATE COUNT  READ COUNT
-----------  ------------  ------------  ----------
NOT APPLICABLE FOR GENERIC PROFILE
```

But, if you forget the GENERIC parameter on the LISTDSD command, you may be puzzled by the output in the following example:

```
LISTDSD DATASET('SYS1.PROCLIB') ALL
ICH35003I NO RACF DESCRIPTION FOUND FOR SYS1.PROCLIB
```

The dataset is protected by a generic profile, not a discrete profile, and the command looks only for discrete profiles.

Back to the output from the first command. The information you're looking for is found under the heading YOUR ACCESS. The user ID has READ access to the SYS1.PROCLIB dataset.

Although beyond the scope of this article, the first line is useful because it names the generic profile that protects the dataset. The name SYS1.*.** indicates that the same generic profile protects all datasets with a high level of SYS1. Well, maybe. Other generic profiles, or even discrete profiles, may have been defined to protect one or more SYS1 datasets. Lower-level profiles override higher level ones. For example, a lower-level generic profile might protect all datasets beginning SYS1.BACKUP.

Finally, if the user ID has access of NONE to the specified dataset, as in the following example, a helpful message is generated:

```
LISTDSD DATASET('SYS6.PROCLIB') ALL GENERIC
ICH35002I NOT AUTHORIZED TO LIST SYS6.**
```

This is helpful because it also lists the name of the generic profile that protects the dataset in question, SYS6.**, just the information you'll need if you want your Security Administrator to correct the situation.

CAPTURING THE OUTPUT FOR ANALYSIS

If you're a REXX programmer, especially if you've dealt with DFSMShsm in REXX, your first question will be: "Can I capture the output from LISTDSD?" The following REXX program was written to answer that question using two of the LISTDSD GENERIC commands shown above.

```
/* REXX */
call capture "LISTDSD DATASET('SYS1.PROCLIB') ALL GENERIC"
call saytrapout "listcat"
call capture "LISTDSD DATASET('SYS6.PROCLIB') ALL GENERIC"
call saytrapout "listcat"
exit

capture:
arg command
/* Just in case OUTTRAP does not set listcat.Ø to number of
   entries in listcat.  */
listcat.Ø = Ø
junk = outtrap("listcat.", "*", "noconcat")
command
junk = outtrap("off")
return

saytrapout:
/*  used to print trapped output:  call saytrapout "listcat"  */
arg trapvar
recs = value(trapvar || ".Ø")
say trapvar || ".n has" recs "records:"
do i=1 to recs
   say value(trapvar || "." || i)
   end
if recs = Ø then say value(trapvar || ".1") ,
   "record one when Ø lines"
return
```

Here's the output it generates, indicating that both standard output and messages can be captured with the REXX outtrap function:

```
LISTCAT.n has 28 records:

INFORMATION FOR DATASET SYS1.*.** (G)

LEVEL  OWNER    UNIVERSAL ACCESS   WARNING   ERASE
-----  -------- ---------------    -------   -----
 ØØ    P39Ø            READ          NO      NO
```

```
AUDITING
--------
FAILURES(READ)

NOTIFY
--------
NO USER TO BE NOTIFIED

YOUR ACCESS  CREATION GROUP  DATASET TYPE
-----------  --------------  ------------
    READ        SYS1           NON-VSAM

NO INSTALLATION DATA

CREATION DATE  LAST REFERENCE DATE  LAST CHANGE DATE
(DAY) (YEAR)        (DAY) (YEAR)      (DAY) (YEAR)
------------  ------------------  ---------------
 164    95      NOT APPLICABLE FOR GENERIC PROFILE

ALTER COUNT  CONTROL COUNT  UPDATE COUNT  READ COUNT

-----------  ------------  ------------  ----------
NOT APPLICABLE FOR GENERIC PROFILE
LISTCAT.n has 1 records:
ICH35002I NOT AUTHORIZED TO LIST SYS6.**
```

USER USAGE

A user asking the question “what access do I have to this dataset?” can use the LISTDSD TSO command in a number of environments. On an ISPF Option or Command line, type:

```
TSO LISTDSD DATASET('dataset-name') ALL GENERIC
```

Unfortunately, the command disappears from the command line after being executed, making it tedious for checking more than one dataset. And although ISPF allows you to precede a command with an ampersand (‘&’) to have it remain on the command line after being executed, this unfortunately only applies to Edit commands within the ISPF Editor.

Option 6 of ISPF allows you to enter most TSO commands directly, without the TSO preceding it. And it has some other advantages too. Although it doesn’t leave the command on the command line after execution, it does add it to a Retrieve list of the 10 most recent commands, which is displayed in the bottom

half of the panel. You can also enter the LISTDSD command in TSO itself, after a READY prompt.

That solves one problem, but there is also another. Whenever more than a screenful of output is generated in TSO, including within ISPF, you'll see three asterisks ('***') at the bottom of the screen. Hit Enter to see the next screenful of output. But there's no way to scroll back.

The ability to capture the output in a manner that can be scrolled, or even retained for future reference, makes batch a worthwhile option. Running TSO in batch effectively gives you three options:

- Print the LISTDSD output on paper
- Output to a dataset or PDS member
- Route the output to a SYSOUT JESx hold class.

The second option allows you to scroll the output with the ISPF editor. The last option allows you to scroll the output with SDSF or other JES*x* tools, but is a temporary solution since JES*x*-held output is generally purged after a specified number of days. Here's the JCL you would use:

```
//TSO      EXEC PGM=IKJEFT01
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD *
LISTDSD DATASET('SYS1.PROCLIB') ALL GENERIC
//
```

IKJEFT01 is TSO/E, which can be run in batch. For the second option, a change in the SYSTSPRT DD statement is required. For example:

```
//SYSTSPRT DD DSN=&SYSUID..FB132.DATA(LISTDSD),DISP=SHR
```

THE PROBLEM WITH LISTDSD

One peculiar habit of LISTDSD is definitely worth noting: it doesn't display an error message if the dataset doesn't exist! Conceivably, this could be useful before a dataset is created, to check what level of access one or more user IDs will have. The

downside of this behaviour is that it doesn't provide the usual check for mistyped DSNs that users (and programmers) expect.

In REXX and CLIST programs, the solution is to perform a SYSDSN on the DSN first, to be sure a dataset exists with that DSN. For anyone using LISTDSD directly, it probably makes more sense (to avoid DSN typos) to use it within the dataset list (DSLIST) of ISPF 3.4 or ISMF 1. Beside and over the DSN displayed, you would type:

```
LISTDSD DATASET(/) ALL GENERIC
```

The slash indicates the current dataset. In ISPF 3.4, begin typing in the command field to the left of the DSN. You can keep typing right over the top of the DSN itself. In ISMF 1, the command (LISTDSD) must be typed in the LINE OPERATOR field to the left of the DSN. Type the parameters, beginning with DATASET, in the DSN field, right over the top of the displayed DSN.

REXX

Putting all this together into a callable REXX routine, you get:

```
/* REXX */
say dsnaccess("SYS1.PROCLIB")
say dsnaccess("SYS6.PROCLIB")
say dsnaccess("JONPE.ACTIVE.CNTL")
say dsnaccess("JONPE.DISCRETE.DATA")
say dsnaccess("JONPE.DOESNOT.EXIST")
exit

dsnaccess:
/* Given a DSN, return the RACF access level (a word:  NONE,
   READ, UPDATE, ALTER, CONTROL, EXECUTE) from the current
   user ID to the given dataset.                              */
arg dsn
/* Constants:  */
true = 1
false = Ø
/* A complete list of all possible RACF accesses to a dataset.
   NONE is omitted because LISTDSD never returns it, giving an
   ICH35ØØ2I error instead.                                   */
acc.Ø = 5    /* Number of entries in the stemmed variable list. */
acc.1 = "READ"
acc.2 = "UPDATE"
```

```
acc.3 = "ALTER"
acc.4 = "CONTROL"
acc.5 = "EXECUTE"
/*  Initialize here to make the ABORT routine work properly.  */
capture.Ø = Ø
/*  Be sure that RACF is installed and available  */
if sysvar("sysracf") <> "AVAILABLE" then do
   say "Fatal error:  RACF is" sysvar("sysracf")
   call abort
   end
/*  The following check for existence of the given DSN
    can be removed if LISTDSD's behaviour with non-existent
    DSNs is understood and desired:  LISTDSD does a "What
    If" scenario, telling you what access you would have to
    the dataset, if it existed.
    The code that follows this check has been written with
    otherwise redundant checks so that the following code
    can be removed.  Code to be removed is marked ****A****  */
/* ****A**** */
/*  Check the DSN to be sure it is valid (no typos) and that
    the dataset exists.                                      */
sysdsnsays = sysdsn("'"dsn"'")
if sysdsnsays = "PROTECTED DATASET" then do
   /* RACF allows no access at all to the dataset */
   return "NONE"
   end
/* Other than PROTECTED DATASET, OK is the only other possible
   response from SYSDSN that does not indicate an error.       */
if sysdsnsays <> "OK" then do
   say "Fatal error:  SYSDSN reports an error on" dsn
   say "              "sysdsnsays
   call abort
   end
/* ****A**** */
/*  Check for a Discrete RACF profile protecting the dataset,
    and capture the output generated by the LISTDSD TSO
    command.                                                 */
call capture "LISTDSD DATASET('"dsn"') ALL"
/*  LISTDSD "always" displays something, so check for no output  */
if capture.Ø = Ø then do
   say "Fatal error:  LISTDSD (discrete) generated no output for" dsn
   call abort

   end
/*  First word of the first line of LISTDSD's display output:
    the RACF message number, except for the normal lengthy
    display that begins INFORMATION FOR DATASET               */
word1 = word(capture.1, 1)
/*  ICH35Ø03I NO RACF DESCRIPTION FOUND FOR dsn  */
if word1 = "ICH35Ø03I" then do
   /* No discrete profile exists, so try generic */
```

```
   call capture "LISTDSD DATASET('"dsn"') ALL GENERIC"
   if capture.Ø = Ø then do
      say "Fatal error:  LISTDSD GENERIC generated no output for" dsn
      call abort
      end
   /*  Message Number: first word of first line of LISTDSD
       output.                                              */
   word1 = word(capture.1, 1)
   /*  ICH35ØØ3I NO RACF DESCRIPTION FOUND FOR dsn  */
   if word1 = "ICH35ØØ3I" then do
      say "Error:  Neither discrete nor generic profile exists" ,
          "in RACF for dataset" dsn
      /*  This could happen if RACF is not set up to protect
          all datasets by default.  If you wish to handle that
          situation as "OK", then replace this call abort with:
             return "ALTER"
      */
      call abort
      end
   end
/*  ICH35ØØ2I NOT AUTHORIZED TO LIST generic profile name  */
if word1 = "ICH35ØØ2I" then do
   racfaccess = "NONE"
   end
else do
   /*  We have processed all expected errors from LISTDSD.
       Normal output begins INFORMATION FOR DATASET,
       so anything else is trouble.  It may just be
       a change to the output in a future version of RACF,
       requiring a review of the code below, especially the
       YOUR ACCESS part.                                    */
   if word1 <> "INFORMATION" then do
      /*  RACF errors are ICHnnnnna  */
      if left(word1,3) = "ICH" then do
         say "Fatal error:  unexpected error from LISTDSD:"
         end
      else do
         say "Fatal error:  unexpected output from LISTDSD:"
         end
      call abort
      end
   /*  Look for YOUR ACCESS at the beginning of a line of
       output from LISTDSD.                                 */
   do i = 1 to capture.Ø ,
      until subword(capture.i,1,2) = "YOUR ACCESS"
      end
   /*  At the end of this DO loop, i will be set to the line
       in capture.n where YOUR ACCESS is found beginning the
       line.  If YOUR ACCESS is not found, i will be set to
       capture.Ø plus one.                                  */
   /*  Be sure that YOUR ACCESS was found.  */
```

```
   if i > capture.Ø then do
      say "Fatal error: LISTDSD output for" dsn
      say "does not contain 'YOUR ACCESS' heading"
      call abort
      end
   /*  Look at the next line, where the hyphens should be
       underlining YOUR ACCESS                              */
   i = i + 1
   /*  Be sure the line was output by LISTDSD and that it
       contains hyphens.                                    */
   if i > capture.Ø | ,
      left(word(capture.i,1),2) <> "--" then do
      say "Fatal error: LISTDSD output for" dsn
      say "does not contain hyphens to underline YOUR ACCESS"
      call abort
      end
   /*  The next line is where you expect to find the RACF access
       "word" describing your access to the dataset    */
   i = i + 1
   /*  Be sure that there actually is a next line  */
   if i > capture.Ø then do
      say "Fatal error: LISTDSD output for" dsn
      say "ends with underscoring for YOUR ACCESS heading," ,
          "without specifying RACF access to dataset."
      call abort
      end
   /*  The RACF access should be the first word of the line  */
   racfaccess = word(capture.i,1)
   found = false
   /*  Check through the table (REXX stemmed variable acc.n)
       to be sure that the word is a valid RACF access keyword */
   do a = 1 to acc.Ø
      if acc.a = racfaccess then found = true
      end
   if ¬ found then do
      say "Fatal error: Invalid RACF access value in LISTDSD",
          "output for" dsn
      say "Access value found was:" racfaccess
      call abort
      end
   end
return racfaccess

abort:
/*  Fatal errors come here to have any output dumped
    before exiting the entire REXX code               */
if capture.Ø > Ø then do
   /*  Dump the entire displayed output to simplify
       diagnosis.                                    */
   say "LISTDSD output follows:"
   do i = 1 to capture.Ø
```

```
      say capture.i
      end
   end
exit 16  /* exit completely, not just this subroutine */

capture:
/* Capture the output from a command in the stemmed variable
   capture.   */
arg command
/* Just in case OUTTRAP does not set capture.Ø to number of
   entries in capture.  */
capture.Ø = Ø
junk = outtrap("capture.", "*", "noconcat")
command
junk = outtrap("off")
return
When it was run, it displays:
READ
NONE
ALTER
ALTER
Fatal error:  SYSDSN reports an error on JONPE.DOESNOT.EXIST
              DATASET NOT FOUND
```

CLIST

The equivalent CLIST is shown below:

```
SYSCALL DSNACCESS 'SYS1.PROCLIB'
WRITE &LASTCC
SYSCALL DSNACCESS 'SYS6.PROCLIB'
WRITE &LASTCC
SYSCALL DSNACCESS 'JONPE.ACTIVE.CNTL'
WRITE &LASTCC
SYSCALL DSNACCESS 'JONPE.DISCRETE.DATA'
WRITE &LASTCC
SYSCALL DSNACCESS 'JONPE.DOESNOT.EXIST'
WRITE &LASTCC
EXIT

DSNACCESS:  PROC 1 DSN
/* Given a DSN, return the RACF access level (a word:  NONE,
/* READ, UPDATE, ALTER, CONTROL, EXECUTE) from the current
/* user ID to the given dataset.                              */
/*  Constants:     */
SET &SYSOMAX = 1ØØ
/*  Be sure that RACF is installed and available  */
IF &SYSRACF ¬= AVAILABLE THEN DO
   WRITE Fatal error:  RACF is &SYSRACF
   EXIT CODE(16)
   END
```

```
/*  The following check for existence of the given DSN
/*  can be removed if LISTDSD's behaviour with non-existent
/*  DSNs is understood and desired:  LISTDSD does a "What
/*  If" scenario, telling you what access you would have to
/*  the dataset, if it existed.
/*  The code that follows this check has been written with
/*  otherwise redundant checks so that the following code
/*  can be removed.  Code to be removed is marked ****A****  */
/* ****A**** */
/*  Check the DSN to be sure it is valid (no typos) and that
/*  the dataset exists.                                      */
SET &SYSDSNSAYS = &SYSDSN(&DSN)
IF &SYSDSNSAYS = PROTECTED DATASET THEN DO
   /* RACF allows no access at all to the dataset */
   RETURN CODE(NONE)
   END
/* Other than PROTECTED DATASET, OK is the only other possible
/* response from SYSDSN that does not indicate an error.       */
IF &SYSDSNSAYS ¬= OK THEN DO
   WRITE Fatal error:  SYSDSN reports an error on &DSN
   WRITE &SYSDSNSAYS
   EXIT CODE(16)
   END
/* ****A**** */
/*  Check for a Discrete RACF profile protecting the dataset,
/*  and capture the output generated by the LISTDSD TSO
/*  command.                                                 */
SET &SYSOUTTRAP = &SYSOMAX
LISTDSD DATASET(&DSN) ALL
SET &SYSOUTTRAP = Ø
/*  ICH35ØØ3I NO RACF DESCRIPTION FOUND FOR dsn  */
IF &SUBSTR(1:9,&SYSOUTLINE1) = ICH35ØØ3I THEN DO
   /* No discrete profile exists, so try generic */
   SET &SYSOUTTRAP = &SYSOMAX
   LISTDSD DATASET(&DSN) ALL GENERIC
   SET &SYSOUTTRAP = Ø
   /*  ICH35ØØ3I NO RACF DESCRIPTION FOUND FOR dsn  */
   IF &SUBSTR(1:9,&SYSOUTLINE1) = ICH35ØØ3I THEN DO
      WRITE Error:  Neither discrete nor generic profile exists
      WRITE in RACF for dataset &DSN
      /*  This could happen if RACF is not set up to protect
      /*  all datasets by default.  If you wish to handle that
      /*  situation as "OK", then replace this EXIT CODE(16) with:
      /*     RETURN CODE(ALTER)
      */
      EXIT CODE(16)
      END
   END
/*  ICH35ØØ2I NOT AUTHORIZED TO LIST generic profile name  */
IF &SUBSTR(1:9,&SYSOUTLINE1) = ICH35ØØ2I THEN DO
   SET &ACCESS = NONE
```

```
   END
ELSE DO
   /*  Look for YOUR ACCESS at the beginning of a line of
   /*  output from LISTDSD.                              */
   DO I = 1 TO &SYSOMAX +
      UNTIL &SUBSTR(1:11,&LINE) = YOUR ACCESS
      SET &LINEVAR = &STR(&&SYSOUTLINE&I)
      SET &LINE = &STR(&LINEVAR                    )
      /*  Note that UNTIL is not executed until this point */
      END
   /*  At the end of this DO loop, I will be set to the line
   /*  in &SYSOUTLINE where YOUR ACCESS is found beginning the
   /*  line.  If YOUR ACCESS is not found, I will be set to
   /*  &SYSOMAX plus one.                                 */
   /*  2 lines to where you expect to find the RACF access
   /*  "word" describing your access to the dataset.   */
   SET &I = &I + 2
   SET &LINE = &STR(&&SYSOUTLINE&I)
   /*  The RACF access should be the first word of the line  */
   DO &ST = 1 TO 133 WHILE &SUBSTR(&ST,&LINE)=
      END
   DO &END = &ST TO 133 UNTIL &SUBSTR(&END,&LINE)=
      END
   SET &ACCESS = &SUBSTR(&ST:&END,&LINE)
   END
RETURN CODE(&ACCESS)
END
The output looks like this:
READ
NONE
ALTER
ALTER
FATAL ERROR:  SYSDSN REPORTS AN ERROR ON 'JONPE.DOESNOT.EXIST'
DATASET NOT FOUND
```

CLIST TO REXX

You may note that the CLIST above lacks the detailed error checking found in the REXX code. It's only included here in case you want to include this code in an existing CLIST; it usually makes more sense to call a REXX EXEC from a CLIST as shown below:

```
EXEC 'JONPE.ACTIVE.EXEC(CLDSNACS)' 'SYS1.PROCLIB'
WRITE &LASTCC
EXEC 'JONPE.ACTIVE.EXEC(CLDSNACS)' 'SYS6.PROCLIB'
WRITE &LASTCC
EXEC 'JONPE.ACTIVE.EXEC(CLDSNACS)' 'JONPE.ACTIVE.CNTL'
WRITE &LASTCC
```

```
EXEC 'JONPE.ACTIVE.EXEC(CLDSNACS)' 'JONPE.DISCRETE.DATA'
WRITE &LASTCC
EXEC 'JONPE.ACTIVE.EXEC(CLDSNACS)' 'JONPE.DOESNOT.EXIST'
WRITE &LASTCC
```

The start of the original REXX code would then have to change slightly:

```
/* REXX */
/* Callable version of dsnaccess for CLIST, which requires
   a numeric return code                                    */
arg dsn
/* A complete list of all possible RACF accesses to a dataset.
   NONE is omitted because LISTDSD never returns it, giving an
   ICH35ØØ2I error instead.                                 */
acc.Ø = 6    /* Number of entries in the stemmed variable list. */
acc.1 = "READ"
acc.2 = "UPDATE"
acc.3 = "ALTER"
acc.4 = "CONTROL"
acc.5 = "EXECUTE"
acc.6 = "NONE"
/* Call dsnaccess */
word = dsnaccess(dsn)
/* Translate word into a number */
do i = 1 to acc.Ø until acc.i = word
   end
return i

dsnaccess:
/* Given a DSN, return the RACF access level (a word:  NONE,
   READ, UPDATE, ALTER, CONTROL, EXECUTE) from the current
   user ID to the given dataset.                            */
arg dsn
/* Constants:  */
true = 1
false = Ø
/*  Initialize here to make the ABORT routine work properly.  */
capture.Ø = Ø
  .
  .
  .
```

The major difference is that a number, rather than a word, is returned. This is only because of CLIST's requirement that return codes be numeric. You could, of course, translate the number back into a word in the CLIST. The output from the CLIST is as follows:

```
1
6
```

```
3
3
Fatal error:  SYSDSN reports an error on JONPE.DOESNOT.EXIST
              DATASET NOT FOUND
16
```

ADDENDUM

It's worth mentioning a non-IBM solution suggested to me five years ago at my local telco: VRA has since been renamed Vanguard Administrator, and is the company's flagship product (see http://www.go2vanguard.com).

Although primarily an on-line tool, VRA also runs in batch. Its Dataset Authority Report, Program VRADSNA, accepts input records with user ID and DSN on DD name SYSIN, and outputs, on DD name REPORT, the RACF access level that the user ID has to the specified dataset: None, Read, Update, Alter, Control, or Execute. Typical JCL to run it looks like this:

```
//*  TELLS WHAT ACCESS A GIVEN USER HAS TO A GIVEN DATA SET //
*****************************************************************
//*                                                              *
//*      THE RACF ADMINISTRATOR                                  *
//*                                                              *
//*      MEMBER: VRADSNAJ - THE DATASET AUTHORITY REPORT         *
//*                                                              *
//*      FORMAT OF INPUT CARDS IS AS FOLLOWS:                    *
//*      COL 1-8 - USERID                                        *
//*      COL 9     BLANK                                         *
//*      COL 1Ø    DATASET NAME                                  *
//****************************************************************
//VRADSNAJ PROC RSIZE=6M
//STEPØ1  EXEC PGM=VRADSNA,REGION=&RSIZE
//VIPOPTS   DD DISP=SHR,DSN=SYS3.VRA.VANOPTS
//REPORT   DD  SYSOUT=*
//SYSUDUMP DD  SYSOUT=*
//        PEND
//GO   EXEC  VRADSNAJ
//SYSIN    DD  *      PLACE INPUT HERE
#JRP     LPWHPØØ1.PRODØ.COPY
#6ØT     LPLSMEDM.ARCHIVE.TEST
#JRR     LPLSMEDM.ARCHIVE.TEST
#JRP     LPLSMEDM.ARCHIVE.TEST
//
```

Jon E Pearkins
(Canada)

RACF – your questions answered

Welcome to our second instalment! I hope you found the information in last quarter's issue useful. Unfortunately, however, I didn't receive a single reply from any of our readers – no questions, no answers, no corrections of mistakes, no suggestions. So it looks like I'll have to make it up as I go along this issue. …

But please, folks, send me your questions or comments. *RACF Update* wants to make this column as valuable to you as possible, but in order to do that, we need to know what you think. Oh, and by the bye, I've got a new e-mail address – DocFarmer@qatar.net.qa – so please use this instead of the old Doc.Farmer@sbm.net.sa one.

PASSWORD STRUCTURE

Q: How can I get around RACF's password structuring limitations?

A: To say that RACF's password structuring is a bit limited is a bit like saying the oceans are slightly damp: the SETROPTS profile allows you only eight different variations on a particular theme. If you want to require a password to include a numeral, you've got to set up the profile as follows:

```
XXXXXXXN
XXXXXXNX
XXXXXNXX
XXXXNXXX
XXXNXXXX
XXNXXXXX
XNXXXXXX
NXXXXXXX
```

But if you want to have two numbers in the profile, in non-contiguous positions, you're in for a bit of a problem. You've got only eight slots for profiles, but you'd need 21 to fulfil this requirement. This means that you're limited in how you can define or specify your password structures to fill the needs of your security policies. Also, since RACF doesn't recognize

upper/lower case differences, you've cut your alphanumeric password (no special or national character) combinations down from 218,340,105,584,896 (628) to only 2,821,109,907,456 (368), a factor of 77 times! Which is a bit of a pain!

How do you get around all of these limitations? Well, quite frankly, you don't – until IBM does a serious upgrade to RACF, we're stuck with this situation. The best suggestion I can make is to lobby IBM to make these changes through SHARE, RACF user groups, and *RACF Update*. I'd like to ask all of you to send me a 'laundry list' of the things you'd like to see 'cleaned up' in RACF – not just the passwords, but everything. I'll compile the list and present it in a future article, as well as sending a copy to the Armonk Giant itself.

AUDIT – HOW MUCH IS TOO MUCH?

Q: RACF allows you to audit the work of users, groups, etc. But when does that audit function become less of a tracking system and more of a resource nightmare?

A: This partly depends on the type of organization you're working for. If it's a manufacturing and distribution centre for construction materials, you may need far less tracking capabilities than, say, a manufacturing and distribution centre for weapons systems. Also, it depends on your Orange Book level. If you're a B1, you need far more auditability than a C1 or C2.

It also depends on what you're going to use the tracking for. If you only want to monitor updates to critical production data files, or critical program modules, tracking becomes a far simpler process than auditing the entire system. It's also far less costly in system overhead and DASD storage requirements. You can blow an entire 3390 pack in a week (or less), if you're auditing everything in a medium- to large-scale environment.

There are a few simple ways to reduce your audit overhead. First, if your internal or external auditors ask you to increase the RACF Audit facility, ask them for specific reasons and justifications – "because we want it" or "because we said so" is

not a valid justification. Second, make sure that the auditors receive the audit trail results – preferably on a daily basis, and preferably via forklift. Also make sure they understand that they are responsible for the investigation of any finding they make in those audit trails. You can always clarify that understanding by charging them (at an hourly rate) for any assistance you must provide.

But what should you audit at a minimum? Here are some suggestions:

- SYS1 files
- Key 0 programs
- Technical support users
- Information security users
- Critical production programs
- Critical production data files
- Anyone with one of the following capabilities:
  - Users with SPECIAL
  - Users with OPERATIONS
  - Users/Groups with Supervisor-State UIDs/GIDs (0000000000).

Why not pass this along to your auditors? When they've finished reading it (and regained consciousness!), have them pass along their comments!

UID/GID STRUCTURE AND SETTINGS – THE NIGHTMARE BEGINS...

Q: How can I keep the UID and GID structures straight?

A: This is something I really get annoyed with. More often than not, keeping the UID and GID structures straight can turn into a logistical nightmare – making sure there are no duplicate IDs, limiting the dreaded supervisor key (0000000000), ensuring

that you're not giving conflicting permissions between the UID and GID, etc. But how do you get around this headache?

Here are a few tips:

- Use a unique ID for the UIDs – employee number, social security number, etc – with leading zeros.
- Wherever possible, use the cost centre number of the department or area for the GID.
- For system-related GIDs, use a unique prefix (700-998) depending on whether it's for CICS groups, started tasks, etc.
- Record all of these numbers in an Excel spreadsheet, and sort it every once in a while to check for duplicates.
- For users or groups with the supervisor key, make sure you turn the AUDIT function on (another good reason to limit the number of people with this access).
- Use Pentland Utilities RACF87 and RACF88 to check your GID and UID assignments every so often. Match them to that Excel spreadsheet I mentioned above.

As always, proper management from the get-go is the key to successfully managing these little Unix interlopers into our beloved RACF realm. ...

REQUEST FOR SETROPTS INFORMATION

I'm still hoping some of you will send in your SETROPTS settings information for that article I'm planning – your help would be greatly appreciated.

CONTACT ME!

Please feel free to make suggestions or pose questions of your own – that's what this column is here for. Please send comments and suggestions on this article to DocFarmer@qatar.net.qa.

Doc Farmer
Security and Senior IS Security Analyst (Middle East)

Information point – reviews

This issue, we look at the Knowledge Bases and Forums of major software companies to see what help they offer with their products from a RACF perspective. Like Web sites themselves, there's no consistency here.

SAS

SAS stands for Statistical Analysis System. It is both software and the company that developed it. In the mid-1970s, it surprised everyone by how quickly it dethroned reigning stats pack SPSS, and today, 98 of the Fortune 100 companies are its customers.

To search everything at once, go to:

http://www.sas.com/service/techsup/search

Leave the default 'Tech Support Area INCLUDING SAS Notes' and enter either a Simple or Advanced search. A simple search of RACF yields 93 results, sorted by relevance. 72% relevance is assigned to the first item, entitled 'Experimental SECPROFILE SAS System option along with a new SVC routine, allows use of the RACF Secured Sign-on function', which provides information on a hotfix for SAS 8.2 running in z/OS and OS/390.

The fifth and tenth items are the first not taken from SAS Notes. Both provide sample SAS/C code:

- RACF – MVS RACF parameter list mappings for the TRYRACF sample in prefix.SAMPLE.C
- TRYRACF – An example of issuing a MVS RACF call from a C program.

CA

Even though Computer Associates sells at least two competitive products to RACF, it still provides support for using its other products with RACF. The fastest way to find information is from

the Quick Search field in the upper left hand corner of the Technical Support home page at http://support.ca.com

Type RACF and hit Find, and you'll get 272 results, sorted by relevance. Number one scores 56% and is barely a month old: 'Refreshing RACF Without Cycling Multi-User'.

The second item is entitled 'SOLVE-IBM PTF UW31883 Resolves RACF exclusive lock problem on OS/390' and has a relevance of 55%. However, it's available only to CA customers, prompting you for a StarTCC ID and password when you click the link.

PHOENIX

Although Phoenix may not be well known, its (E)JES product is certainly popular in JES3 installations as an alternative to IBM's JES2-only SDSF.

Although the company offers neither manuals nor a knowledge base on its Web site, it does have a discussion group where you can ask questions and get answers. And it's a public forum, which means you don't have to be a customer: go to http://www.phoenixsoftware.com/discus/board.html

The left sidebar offers several ways to view what's there, and Tree View seems to be the fastest way to get around.

The only thing missing is a search facility. As an alternative, you might think of using the Advanced Search of Google, with its Domain field where you can specify phoenixsoftware.com, retaining the default Only option, and typing RACF in the 'Find results with all the words' field. Unfortunately, however, this doesn't work, presumably because the forum pages are not in Google's enormous database.

MICROSOFT

For most topics, the best Microsoft support information is found by clicking on Support in the menu bar in the upper right corner of the home page, and then selecting Knowledge Base from the drop-down menu. That takes you to the Advanced Search at:

http://support.microsoft.com/default.aspx?scid=fh;[ln];kbhowto

But all you'll get for RACF is two hits. Instead, on the home page, type RACF into the Search field in the upper left corner and hit the Go button. 51 results are displayed, many for older (versions of) products, with a few not in English. For more detailed searches, click the Advanced Search link just below the Search field on the home page, or go directly there:

http://search.microsoft.com/advanced_search.asp

IBM

IBM has a lot of other mainframe-related software besides RACF, much of which must work with RACF. The Technical Support Search is the place to start: http://www.ibm.com/support/search/index.html

As well as a keyword search in the centre of the page, the Other Searches to the right includes an APAR Search and an Advanced Search – both worth exploring. Searching APARs for RACF returns 1,020 items.

Just typing RACF as a keyword search gives you 845 items, but you can narrow it down in two ways. Before hitting the Submit button, you can select a Product Category from a drop-down list. Selecting Database & Data Management, for example, narrows it down to 69 items, mostly involving IMS or DB2. Alternatively, after you hit Submit, you can click on a Limit Your Search link to narrow it down, based on the same categories as the drop-down list. Then, Limiting Your Search links keep appearing, allowing you to further narrow the search as the following hierarchy shows:

```
Database & Data Management
      Databases
            IMS
                  IMS Connect
```

Note, however, that you may not be able to read everything that you've found: small key icons are located to the left of items that are locked, and items marked with this symbol are available only to customers who have purchased an IBM Passport

Advantage Software Maintenance Agreement.

LOTUS

Lotus Support seems to be in a state of transition confusion, as it moves to use IBM's on-line support systems. Click on Support in the upper right corner of the Lotus home page and you're taken to:

http://www.ibm.com/software/lotus/support

There, you'll find a search field labelled 'Keyword Search against Lotus content'. Type in RACF and you'll get the same 845 items you would on the IBM site, nearly all having nothing to do with Lotus products.

CUSTOMERS ONLY

Other software companies provide on-line information for their customers only. Here are a few:

- BMC – http://www.bmc.com/support.html
- Candle – http://www.candle.com/www1/cnd/portal/CNDportal_Article_Master/0,2245,2683_3009_44517,00.html
- CompuWare – http://frontline.compuware.com
- Oracle – http://www.oracle.com/support/metalink
- Serena – http://support.serena.com
- Tantia – http://www.tantiatech.com/support/scripts/kbsearch.cfm
- Tivoli – http://www.tivoli.com/support/knowledgebase

Of course, most software companies provide little or no on-line support knowledge base or forum. Many provide on-line forms or e-mail links for support. Or at least a telephone number to call.

Jon E Pearkins
(Canada)

Contributing to *RACF Update*

In addition to *RACF Update*, the Xephon family of *Update* publications now includes *CICS Update*, *MVS Update*, *TCP/SNA Update*, *VSAM Update*, *DB2 Update*, *AIX Update*, *and MQ Update*. Although the articles published are of a very high standard, the vast majority are not written by professional writers, and we rely heavily on our readers themselves taking the time and trouble to share their experiences with others. Many have discovered that writing an article is not the daunting task that it might appear to be at first glance.

They have found that the effort needed to pass on valuable information to others is more than offset by our generous terms and conditions and the recognition they gain from their fellow professionals. Often, just a few hundred words are sufficient to describe a problem and the steps taken to solve it.

If you have ever experienced any difficulties with RACF, or made an interesting discovery, you could receive a cash payment, a free subscription to any of our *Updates*, or a credit against any of Xephon's wide range of products and services, simply by telling us all about it.

More information about contributing an article to a Xephon Update, and an explanation of the terms and conditions under which we publish articles, can be found at http://www.xephon.com/nfc. Alternatively, please write to the editor, Fiona Hewitt, at any of the addresses shown on page 2, or e-mail her at fionah@xephon.com

IBM is adding Multilevel Security to z/OS, creating a Trusted Computing Base – MAC/DAC support using labelled resources – primarily in TCP/IP, Security Server, and Unix System Services. The March 2003 release of z/OS and z/OS.e Version 1.5 will extend the labelled security protection of z/OS to include TCP/IP and Unix System Services, and provide enhancements to Security Server, JES2, SDSF and others.

With RACF Name Hiding active, users will only see DSNs to which they have at least Read access. DFSMShsm Command Authorization Control will provide RACF Facility Class support for all storage administrator and end-user commands. Project eLiza's Enterprise Identity Mapping (EIM) will use the LDAP database as a central repository of user mapping information; it will provide a C/C++ interface to equate a RACF user ID with an ID on iSeries or other platform.

For further information, contact your local IBM representative, or visit the Web site at http://www.ibmlink.ibm.com/usalets&parms=H_202-190

* * *

Following its recent acquisition of Entact Information Security, ASG now has three integrated security products:

- ASG-Admin for Security Server is a user interface for real-time management of RACF profiles.
- ASG-Audit for Security Server is a Windows-based GUI alternative to the RACF Report Writer for reporting on RACF profile and SMF audit information.
- ASG-Entact ID offers multi-platform security through four modules. ASG-Entact ID Manager handles user access administration. ASG-Entact ID Request supports access request workflow. Self-service password resets are provided by ASG-Entact ID Reset. And ASG-Entact ID UniPass enables one-step password synchronization.

URL: http://www.asg.com/products/secmgmt.asp

* * *

Jacada Terminal Emulator provides Web-to-host 3270, iSeries, and VAX/VMS support for Windows, Mac, Unix, and Linux clients. Secure connections are offered via an SSL add-in for the client-side applet. An SSL Security Server is available if there isn't one already behind the Web server's firewall.

URL: http://www.jacada.com/Products/Jacada_Terminal_Emulator

* * *

CA eTrust Security Command Centre provides a single point of control console/portal to administer eTrust Access Management, eTrust Identity Management, and eTrust Threat Management. A Software Development Kit (SDK) can be used to interface to other security products.

URL: http://www3.ca.com/Solutions/SubSolution.asp?ID=4350