33

RACF

August 2003

In this issue

update

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.

MATRIX – cross-checking userids with groups

I never realized how much I needed this program until I wrote it! The first version was written for a colleague who had to cross-check a large number of userids in order to find out which groups, if any, they had in common. As he was telling me how much trouble he was going to have, I was already thinking how could it be done with a program. It was a lot of fun to write, it was a big help for my colleague, and, in its present form, it has proven itself a most valuable tool.

As I said, the original purpose of MATRIX was to cross-check several userids with all of their groups, but I've also been using it to – amongst other things – check whether all the userids of a functional group have the same profile, or which group is the better one to extend to a group of userids.

MATRIX lists all the groups associated with a userid, so, if you pass it groups, it will first get all the associated userids. If any of the received items turns out to be an invalid RACF item, a message will be displayed, and the process will be interrupted. Once all the userids to be treated are identified, they will be alphabetically sorted, and only then will the true MATRIX process begin. Every userid will be listed, and its associated groups will be recorded. After the last userid has been processed, the results will be displayed, with a line for each userid and a column for each group. An 'X' will be placed in each valid intersection, where a userid line crosses with a group column to which he is connected. The group names will be vertically placed at the top of the columns, and the userids will be placed at the left of the lines. By default, the RACF userid names will appear after the userids, but these can be omitted by specifying NAME(NO) at invocation time. Also by default, there will be a '|' as the separator between columns. This can be changed by specifying SEP(NO), also at invocation time.

If you invoke MATRIX under ISPF, the results will be put in a temporary work file, and shown by means of BROWSE. If you run

		CGRRAF1	CGDVT01	CERTIFY5	DVOWNER	FCQUSRS7X	IMOWNERX	MCQDB2SL	MDTROUB	MTUSERSS	PRODUCT	QUALITY
DUSR001	ALDOUS STEINBECK					X						
DUSR003	JOHN HUXLEY					X						
DUSR004	OSCAR ALAN CROWN			X					X			
FUSR004	EDGAR WILDE		X				X			X		
FUSR032	GINGER CHRISTIE			X				X			X	X
PUSR001	AGATHA ROGERS			X				X				
PUSR004	JAMES ANTHONY			X				X				
PUSR098	BLUE BEARD JR	X		X	X				X			
YBUSR002	STC DVT HELP USR							X				

Figure 1: Printed results

it on a BATCH JOB, the results will be shown on SYSTSPRT (by means of SAYs).

So, if you invoke MATRIX, under ISPF, with something like

```
* MATRIX group13 userid37
```

you might get something like the following:

```
                                |C|C|C|D|F|I|M|M|M|P|Q|
                                |G|G|E|V|C|M|C|D|T|R|U|
                                |R|D|R|O|Q|O|Q|T|U|O|A|
                                |R|V|T|W|U|W|D|R|S|D|L|
                                |A|T|I|N|S|N|B|O|E|U|I|
                                |F|Ø|F|E|R|E|2|U|R|C|T|
                                |1|1|Y|R|S|R|S|B|S|T|Y|
                                | | | |5| |7|X|L| |S| |
                                |-|-|-|-|-|-|-|-|-|-|-|
DUSRØØ1  ALDOUS STEINBECK       | | | | |X| | | | | | |
DUSRØØ3  JOHN HUXLEY            | | | | |X| | | | | | |
DUSRØØ4  OSCAR ALAN CROWN       | | |X| | | | |X| | | |
FUSRØØ4  EDGAR WILDE            | |X| | | |X| | |X| | |
FUSRØ32  GINGER CHRISTIE        | | |X| | | |X| | |X|X|
PUSRØØ1  AGATHA ROGERS          | | |X| | | |X| | | | |
PUSRØØ4  JAMES ANTHONY          | | |X| | | |X| | | | |
```

```
PUSRØ98  BLUE BEARD JR        |X| |X|X| | | |X| | | |
YBUSRØØ2 STC DVT HELP USR     | | | | | | |X| | | | |
```

Sometimes the results will be too complex to be managed by means of a 3270 DISPLAY, so I often capture the work file, transfer it to the PC as a .txt file, and use a spreadsheet to open, edit, and print it (see Figure 1).

MATRIX

```
/* rexx
                        *************************
                        *                       *
                        *  Joao Bentes de Jesus  *
                        *      MATRIX 1.2.Ø     *
                        *                       *
                        *************************
-                                                                    */
arg input_data
if  input_data="" then
    do
        say"You did not specify the group(s) and/or userid(s) to check"
    end
else
    do
        call verify_input
    end
return
/* - - - - - - - - - - - - - - */
verify_input:
px=wordpos("NAME(NO)",input_data) /* should racf names be displayed ? */
if  px>Ø then
    do        /* do not show racf names */
        name=Ø
        fill=9
        input_data=delword(input_data,px,1)
    end
else
    do        /* display racf names */
        name=1
        fill=3Ø
    end
px=wordpos("SEP(NO)",input_data) /* use '|' as a column separator ? */
if  px>Ø then
    do        /* no, use blanks as column separator */
        sep=" "
        fill=fill-1
        input_data=delword(input_data,px,1)
```

```
    end
else
    do        /* yes, use '|' as column separator */
        sep="|"
    end
groups=""
users =""
errors=""
x=outtrap("ON")
do  a=1 to words(input_data)
    item=word(input_data,a)
    "LU "item
    if  rc=0 then
        do
            users=users item
        end
    else
        do
            "LG "item
            if  rc=0 then
                do
                    groups=groups item
                end
            else
                do
                    errors=errors item
                end
        end
end
x=outtrap("OFF")
qx=words(errors)
if  qx>0 then
    do
        if  qx=1 then
            do
                say"The item "errors" was not found in RACF"
            end
        else
            do
                say"The items "errors" were not found in RACF"
            end
    end
else
    do
        call get_info
    end
return
/* - - - - - - - - - - - - */
get_info:
do  a=1 to words(groups)
```

```
    parse value users(word(groups,a)) with g_users
    users=users g_users
end
if  words(users)>Ø then
    do
        new_users=""
        do  a=1 to words(users)
            user=word(users,a)
            if  wordpos(user,new_users)=Ø then
                do
                    new_users=new_users user
                end
        end
        call get_matrix
    end
else
    do
        say"No valid users were selected for MATRIX processing"
    end
return
/* - - - - - - - - - - - - */
users: procedure
arg group .
x=outtrap(racf.,,"NOCONCAT")
"LG "group
x=outtrap("OFF")
if  rc=Ø then
    do
        users=""
        do  a=1 to racf.Ø
            if  wordpos("USER(S)=",racf.a)>Ø then
                do  a=a+1 by 3 to racf.Ø
                    users=users word(racf.a,1)
                end
        end
    end
drop racf.
return users
/* - - - - - - - - - - - - */
get_matrix:
users=sort_tab(new_users)
all_groups=""
qz=words(users)
do  a=1 to qz
    user.a=word(users,a)
    parse value groups(user.a) with "ß"nome.a"ß" groups.a
    qx.a=words(groups.a)
    do  g=1 to qx.a
        group=word(groups.a,g)
        if  wordpos(group,all_groups)=Ø then
```

```
            do
                all_groups=all_groups group
            end
    end
end
all_groups=sort_tab(all_groups)
qx=words(all_groups)
groups=""
do  a=1 to qx
    group.a=word(all_groups,a)
    groups=groups group.a
end
do  i=1 to 8
    o_line.i=left("",fill)
    do  a=1 to qx
        o_line.i=o_line.i substr(group.a,i,1)
    end
end
o_line.i=left("",fill) copies("- ",qx)
ln=0
do  a=1 to qz
    i=i+1
    if  name then
        do
            o_line.i=left(user.a,8) left(nome.a,20)
        end
    else
        do
            o_line.i=left(user.a,8)
        end
    do  z=1 to qx.a
        group=word(groups.a,z)
        p=wordpos(group,groups)
        interpret "o_line.i=overlay('X',o_line.i,"fill+2*p",2)"
    end
    ln=max(ln,length(o_line.i))
end
do  a=1 to i
    do  b=1+fill by 2 to ln
        o_line.a=overlay(sep,o_line.a,b,1)
    end
end
if  i>0 then
    do
        if  sysvar("SYSENV")="FORE" & sysvar("SYSISPF")="ACTIVE" then
            do
                call browse_output
            end
        else
            do  a=1 to i
```

```
                say o_line.a
            end
    end
else
    do
        say"Matrix processing returned no data"
    end
return
/* - - - - - - - - - - */
groups: procedure
arg user .
groups=""
x=outtrap(racf.,,"NOCONCAT")
"LU "user
x=outtrap("OFF")
parse value racf.1 with "NAME="nome"OWNER="
do  a=2 to racf.Ø
    if  pos("GROUP=",racf.a)>Ø then
        do
            parse value racf.a with "GROUP="group .
            if  wordpos(group,groups)=Ø then
                do
                    groups=groups group
                end
        end
end
drop racf.
return "ß"nome"ß" groups
/* - - - - - - - - - - */
browse_output:
dsn ="'"userid(),
   ||"."mvsvar("SYSNAME"),
   ||".D"date("J"),
   ||".T"space(translate(time(),,":"),Ø),
   ||".FWKØØØ'"
dd="O"time("S")
"alloc f("dd") da("dsn") recfm(v b) lrecl("ln+4") space(1Ø 5)",
    "tracks new"
if  rc=Ø then
    do
        "execio "i" diskw "dd" (finis stem o_line.)"
        address "ISPEXEC" "control errors return"
        address "ISPEXEC" "browse dataset("dsn")"
        "free f("dd") delete"
    end
else
    do
        say"Error "rc" no ALLOC de "dsn
    end
return
```

```
/* - - - - - - - - - - - - */
sort_tab:
arg tab_input
zx=words(tab_input)
do  a=1 to zx
    var.a=word(tab_input,a)
end
do  w1=1 to zx-1
    w2=w1+1
    if  var.w1>var.w2 then
        do
            var_wk=var.w1
            var.w1=var.w2
            var.w2=var_wk
            do  w4=w1 by -1 to 2
                w3=w4-1
                if  var.w3>var.w4 then
                    do
                        var_wk=var.w3
                        var.w3=var.w4
                        var.w4=var_wk
                    end
                else
                    do
                        leave w4
                    end
            end
        end
end
tab_output=""
do  a=1 to zx
    tab_output=tab_output var.a
end
return tab_output
/* - - - - - - - - - - - - */
```

Joao Bentes de Jesus
Systems Programmer (Portugal)

Dynamic RACF exits

Except for the IRREVX01 exit in RACF, which is supported by OS/390 dynamic exit services, all other RACF exits are expected to be static and must be present in the LPALST concatenation during system IPL in order to be recognized and activated in the RACF RCVT control block. Although this may be feasible for exits that have gone through testing and are ready for their production activation, this is more than just a minor inconvenience during exit development.

The RACFXITS program presented here provides a general-purpose RACF exit dynamic activation/deactivation tool. This utility supports 15 of the defined RACF exits which are traditionally required to be present in the LPALST during IPL before they can be used.

The RACFXITS utility can be used to load RACF exits dynamically at start-up through program PARM= reference. Alternatively, RACF exits can be dynamically activated and deactivated using the operator command interface. The nice feature of this approach is that an exit can be activated and deactivated many times during testing without requiring a system IPL to try out a new 'version' of the exit code. This can speed up exit development time significantly. Another advantage of the utility is that it enables you to activate or deactivate RACF exits on production systems dynamically using operator commands. This is useful if, for example, you want exits to be active only during certain time periods during the day – automatic commands can be used to enable or disable the appropriate exit at specified times.

A system started task or long-running batch job should be used for the RACFXITS utility. The basic JCL for running the program is shown below.

```
//RACFXITS EXEC PGM=RACFXITS
//STEPLIB  DD   DSN=auth.load.library,DISP=SHR
//XITLIB   DD   DSN=dynamic.racf.exit.library,DISP=SHR
```

The example JCL shown above will start the RACFXITS utility, but it won't activate any RACF exits. This is because no exit names have been provided via the PARM= specification. If RACF exits are to be dynamically activated when the RACFXITS utility starts, a PARM= keyword should be provided on the JCL EXEC statement. For example:

```
//RACFXITS EXEC PGM=RACFXITS,PARM='ICHPWXØ1,ICHRIXØ1,ICHRCXØ1'
```

The above example would cause the RACFXITS utility to attempt to activate three RACF exits at start-up – ICHPWX01, ICHRIX01, and ICHRCX01. If a corresponding load module for the specified exit isn't located in the XITLIB defined dataset, an error message will be issued to the console.

The RACFXITS utility supports four operator commands, as follows:

```
F racfxits,ADD=exitname
F racfxits,DELETE=exitname
F racfxits,DISPLAY=EXITSTATUS
F racfxits,SHUTDOWN
```

where 'racfxits' is the jobname of the started task or batch job being used for the RACFXITS utility, and 'exitname' is the common name for the RACF exit to be added or deleted dynamically (ICHPWX01, ICHDEX01, ICHRIX01, etc). See the XITMAP table in the program source for the list of supported exit names.

The ADD modify operand can be used to add exits which reside in the XITLIB DD dataset, but which weren't included in the PARM= exit list or have been deleted by a prior DELETE command. You cannot ADD an exit that is already dynamically activated – it must first be DELETEd and then it is eligible to be ADDed again. Exits can be refreshed in the XITLIB DD statement without restarting the utility, speeding up exit development.

The DELETE modify operand can be used to delete exits that were specified in the PARM= exit list or were added with the ADD modify command.

```
RACFXITS - RACF EXIT STATUS
Name      Address   RCVT offset  Load Status
--------  --------  -----------  -----------
ICHCCX00  00000000  00B8         N/A
ICHCNX00  00000000  00B4         N/A
ICHDEX01  00000000  01A0         N/A
ICHDEX02  00000000  02DC         N/A
ICHFRX01  00000000  00C8         N/A
ICHFRX02  00000000  01AC         N/A
ICHFRX03  00000000  014C         N/A
ICHFRX04  00000000  02D8         N/A
ICHPWX01  80E56000  00EC         STATIC
ICHRCX01  00000000  001C         N/A
ICHRCX02  00000000  00A4         N/A
ICHRDX01  00000000  0020         N/A
ICHRDX02  00000000  017C         N/A
ICHRLX01  00000000  00CC         N/A
ICHRLX02  00000000  00D0         N/A
ICHRIX01  86DED000  0018         DYNAMIC
ICHRIX02  86995000  00A0         DYNAMIC
END OF RACF EXIT STATUS DISPLAY
```

Figure 1: Console display for a DISPLAY=EXITSTATUS request

The DISPLAY=EXITSTATUS modify operand provides an operator display of the current exit status:

- STATIC – RACF loaded the current exit at IPL.
- DYNAMIC – the RACFXITS utility has loaded the current exit.
- N/A – no exit is currently active.

Figure 1 shows an example operator console display for a DISPLAY=EXITSTATUS request.

The RACFXITS utility can be terminated with either a modify operation as mentioned earlier (modify operand of SHUTDOWN) or with the STOP (P) operator command.

The RACFXITS utility is protected by an ESTAE which will restore the RACF exit environment to the same status as existed when the utility was started if a catastrophic utility failure occurs.

The RACFXITS program should be assembled and linked into an authorized load library using the following linkedit control cards:

```
INCLUDE OBJECT(RACFXITS)
ENTRY   RACFXITS
SETCODE AC(1)
NAME    RACFXITS(R)
```

Although there are some areas for improvement (for example, the ability to ADD an exit that is already dynamically added), the RACFXITS utility provides a very robust addition to your RACF environment, and it's very flexible in that it provides support for multiple exits. Try it in your environment to see if it can save you any time.

```
*********************************************************************
*                                                                   *
*   The RACFXITS utility is designed to run as a started task or    *
*   long running batch job that can be used to dynamically activate *
*   RACF exits that traditionally need to reside in the Link Pack   *
*   Area (LPA) at system IPL time in order to be active.            *
*                                                                   *
*   The PARM= value on the EXEC statement in the startup JCL for    *
*   RACFXITS is used to indicate which RACF exits should be         *
*   dynamically activated when the started task or batch job is     *
*   started.  Additionally, an operator command can be used to      *
*   activate exits not specifically indicated in the startup PARM.  *
*                                                                   *
*   This program should be linkedited into an authorized target load*
*   library using the following linkedit control cards:             *
*                                                                   *
*        INCLUDE OBJECT(RACFXITS)                                   *
*        ENTRY   RACFXITS                                           *
*        SETCODE AC(1)                                              *
*        NAME    RACFXITS(R)                                        *
*                                                                   *
*   The following JCL can be used to load the RACF new password exit*
*   (ICHPWXØ1) at startup:                                          *
*                                                                   *
*   //RACFXITS EXEC PGM=RACFXITS,PARM='ICHPWXØ1'                    *
*   //STEPLIB  DD   DSN=auth.load.library,DISP=SHR                  *
*   //XITLIB   DD   DSN=racf.exit.load.library,DISP=SHR             *
*                                                                   *
*   The RACFXITS utility supports four operator modify commands     *
*   as follows:                                                     *
*                                                                   *
*   F racfxits,ADD=exitname                                         *
```

```
*   F racfxits,DELETE=exitname                                          *
*   F racfxits,DISPLAY=EXITSTATUS                                       *
*   F racfxits,SHUTDOWN                                                 *
*                                                                       *
*   where 'racfxits' is the jobname of the started task or batch job    *
*   being used for the RACFXITS utility, and 'exitname' is the          *
*   common name for the RACF exit to be added or deleted dynamically    *
*   (eg. - ICHPWXØ1, ICHDEXØ1, ICHRIXØ1, etc.).  See the XITMAP         *
*   table below for the list of currently supported exits and the       *
*   names by which they are referenced by this program.                 *
*                                                                       *
*   The ADD modify operand can be used to add exits that reside in      *
*   the XITLIB DD dataset, but were not included in the PARM= exit      *
*   list or have been deleted by a prior DELETE command.  The           *
*   DELETE modify operand can be used to delete exits that were         *
*   specified in the PARM= exit list or were added with the ADD         *
*   modify command.                                                     *
*                                                                       *
*   The DISPLAY=EXITSTATUS provides an operator display of the          *
*   current exit status (STATIC - RACF loaded the current exit at       *
*   IPL; DYNAMIC - the RACFXITS utility has loaded the current exit;    *
*   N/A - no exit is currently active).                                 *
*                                                                       *
*   The SHUTDOWN operand will cause the RACFXITS utility to restore     *
*   the RACF exit environment to the state that existed at its          *
*   startup (this is equivalent to issuing a 'P racfxits' operator      *
*   command).                                                           *
*                                                                       *
*   The RACFXITS utility is protected by an ESTAE that will             *
*   effectively perform a SHUTDOWN operation in the event that a        *
*   failure occurs during normal operation.                             *
*                                                                       *
************************************************************************
RACFXITS CSECT
RACFXITS AMODE 31
RACFXITS RMODE 24
         STM   R14,R12,12(R13)      Save registers
         LR    R12,R15              Copy base register
         LA    R11,4Ø95(,R12)       Set second base ...
         LA    R11,1(,R11)             register address
         USING RACFXITS,R12,R11     Set addressability
         LR    R2,R1                Copy parm register
         LR    R3,R13               Copy savearea register
         STORAGE OBTAIN,LENGTH=WORKLEN,LOC=ANY
         LR    RØ,R1                Copy storage address
         LR    R14,R1               Again
         LR    R13,R1               And again
         L     R1,=A(WORKLEN)       Get length
         XR    R15,R15              Set fill byte
         MVCL  RØ,R14               Clear the storage
```

```
         USING WORKAREA,R13         Set addressability
         ST    R3,SAVEAREA+4        Save old savearea address
*-------------------------------------------------------------------*
         LR    R1,R2                Copy parm address
         L     R2,Ø(,R1)            Get address of parm data
         CLC   Ø(2,R2),=H'Ø'        Any parms?
         BE    RETURN               No - then we're done
         XR    R15,R15              Clear R15
         ICM   R15,B'ØØ11',Ø(R2)    Get parm length
         LA    R2,2(,R2)            Point to data
         LA    R14,Ø(R15,R2)        Point past data
         LA    R3,XITNMTBL          Get exit name table address
         LR    R5,R3                Copy R3
         LA    R4,XITNMTLN(,R3)     Get exit name table end address
         XR    R15,R15              Clear R15
XITNAMLP DS    ØH
         CR    R2,R14               End of parm data?
         BNL   XITNAMDN             Yes - done with parm data
         CR    R3,R4                End of table area?
         BNL   XITNAMDN             Yes - done with parm table
         C     R15,=F'8'            Max length?
         BE    NAMFLUSH             Yes - flush parm data
         CLI   Ø(R2),C','           Separator?
         BE    NEXTENT              Yes - set up for next table entry
         MVC   Ø(1,R3),Ø(R2)        Move next byte of exit name
         LA    R2,1(,R2)            Next source byte
         LA    R3,1(,R3)            Next target byte
         LA    R15,1(,R15)          Counter increment
         B     XITNAMLP             Check next byte
NAMFLUSH DS    ØH
         CR    R2,R14               End of parm data?
         BNL   XITNAMDN             Yes - done with parm data
         CLI   Ø(R2),C','           Separator?
         BE    NEXTENT              Yes - set up for next table entry
         LA    R2,1(,R2)            Next source byte
         B     NAMFLUSH             Keep flushing
NEXTENT  DS    ØH
         XR    R15,R15              Clear counter
         LA    R2,1(,R2)            Skip past separator
*-------------------------------------------------------------------*
*                                                                   *
*   CHECK TO SEE IF THIS IS A REPEAT ENTRY.                         *
*                                                                   *
*-------------------------------------------------------------------*
         LA    R1,XITNMTBL          Get table start address
XITNMLP1 DS    ØH
         CR    R1,R5                End of search?
         BNL   NEXTENT2             Yes - continue with next entry
         CLC   Ø(8,R1),Ø(R5)        A match?
         BE    XITNMMCH             Yes - don't add another entry
```

```
         LA    R1,ENTLEN(,R1)       Point to next table entry
         B     XITNMLP1             Go check it out
XITNMMCH DS    ØH
         MVC   XITWTOØ+8+16(8),Ø(R5) Move in exit name
XITWTOØ  WTO   'RACFXITS - EXIT XXXXXXXX ALREADY SPECIFIED',          X
               ROUTCDE=(1),DESC=(6)
         LR    R3,R5                Reuse last entry
         XC    Ø(8,R5),Ø(R5)        Clear the entry
         B     XITNAMLP             Check for more data
*-------------------------------------------------------------------*
NEXTENT2 DS    ØH
         LA    R3,ENTLEN(,R5)       POINT TO NEXT TABLE ENTRY
         LR    R5,R3                Copy the address
         B     XITNAMLP             Check for more data
XITNAMDN DS    ØH
*-------------------------------------------------------------------*
         OPEN  (XITLIB),MODE=31     Open exit load library
         LA    R5,REGSAVE           Get param address
         ESTAEX ESTAERTN,                                          x
               CT,                                                 x
               XCTL=NO,                                            x
               PARAM=(R5),                                         x
               PURGE=NONE,                                         x
               ASYNCH=YES,                                         x
               TERM=YES
         LA    R3,XITNMTBL          Get table address
         LA    R4,XITNMTLN(,R3)     Get exit name table end address
         MODESET MODE=SUP,KEY=ZERO
         L     R1,16                Get CVT address
         USING CVT,R1               Set addressability
         L     R6,CVTRAC            Get RCVT address
         USING RCVT,R6              Set addressability
         DROP  R1
XITLP1   DS    ØH
         CR    R3,R4                End of table?
         BNL   CMDWAIT              Yes - go wait for command request
*-------------------------------------------------------------------*
XITLD    DS    ØH
         LA    R7,XITMAP            Get exit map address
         LA    R9,XITMAPE(,R7)      Get exit map end address
XITMPLP1 DS    ØH
         CR    R7,R9                End of table?
         BNL   NOLD                 Yes - bail out
         CLC   Ø(8,R7),Ø(R3)        An exit name match?
         BE    LDXIT                Yes - process exit
         LA    R7,12(,R7)           Point to next entry
         B     XITMPLP1             Go check it out
LDXIT    DS    ØH
         L     R9,8(,R7)            Get RCVT offset of exit addr
         L     R8,Ø(R9,R6)          Get current exit addr
```

```
         ST    R8,8(,R3)            Save in table entry
         LA    R1,Ø(,R3)            Point to exit name
         BAL   R14,EXITLOAD         Go load the exit
         LTR   R15,R15              Success?
         BNZ   XITNEXT1             No - go process next exit
         L     R1,12(,R3)           Get new exit address
         ST    R1,Ø(R9,R6)          Copy exit address into RCVT
         B     XITNEXT1             Process next entry
NOLD     DS    ØH
         CLC   Ø(8,R3),=8X'ØØ'      Blank entry?
         BE    XITNEXT1             Yes - skip it
         MVC   XITWTO1+8+28(8),Ø(R3) Move in unknown name
XITWTO1  WTO   'RACFXITS - UNSUPPORTED EXIT XXXXXXXX SPECIFIED',       x
               ROUTCDE=(1),DESC=(6)
XITNEXT1 DS    ØH
         LA    R3,ENTLEN(,R3)       Point to next entry
         B     XITLP1               Check it out
*-------------------------------------------------------------------*
         DROP  R6
CMDWAIT  DS    ØH
         MODESET MODE=PROB,KEY=NZERO
         CLOSE (XITLIB),MODE=31     Close exit load mod dataset
         WTO   'RACFXITS - LOADED EXIT SUMMARY',ROUTCDE=(1),DESC=(6)
         LA    R3,XITNMTBL          Get exit name table address
         LA    R4,XITNMTLN(,R3)     Get exit name table end address
XITSUMLP DS    ØH
         CR    R3,R4                End of table?
         BNL   STCMDINT             Yes - set command interface
         CLC   Ø(8,R3),=8X'ØØ'      Null entry?
         BE    SUMBYPAS             Yes - bypass
         CLC   12(8,R3),=8X'ØØ'     An exit address?
         BE    SUMBYPAS             No - bypass
         MVC   DBL2(4),12(R3)       Copy exit address
         UNPK  DBL1(9),DBL2(5)      Unpack it
         NC    DBL1(8),=8X'ØF'      Turn off high nibbles
         TR    DBL1(8),=C'Ø123456789ABCDEF' Make it readable
         MVC   XSUMWTO1+19(8),DBL1  Copy exit addr into message
         MVC   XSUMWTO1+8(8),Ø(R3)  Copy exit name into message
XSUMWTO1 WTO   'XXXXXXXX:  xxxxxxxx',ROUTCDE=(1),DESC=(6)
SUMBYPAS DS    ØH
         LA    R3,ENTLEN(,R3)       Next entry
         B     XITSUMLP             Check it out
STCMDINT DS    ØH
         LA    R9,COMADDR           Answer address at R9
*                                   Get Command Scheduler comm list
         EXTRACT (R9),FIELDS=COMM,MF=(E,EXTRACT)
*                                   Using execute form of EXTRACT macro
         L     R9,COMADDR           Get address of the area
         USING COM,R9               R9 is base address of comm area
         ICM   R7,15,COMCIBPT       Get CIB address from comm  area
```

```
         BZ    SETCOUNT            No CIB, Task was not started
         USING CIB,R7
         QEDIT ORIGIN=COMCIBPT,BLOCK=(R7) Free current CIB
SETCOUNT DS    ØH
         QEDIT ORIGIN=COMCIBPT,CIBCTR=1 One oper cmd at a time
*-------------------------------------------------------------------*
         L     R6,COMECBPT         Get addr of comm ecb
         ST    R6,ECBS             Save in ECB list
         OI    ECBS,X'8Ø'          Set last ECB flag
         WTO   'RACFXITS - COMMAND INTERFACE ENABLED',                X
               ROUTCDE(1),DESC(6)
WAIT     DS    ØH
         WAIT  1,ECBLIST=ECBS      Wait for a stop or modify
*-------------------------------------------------------------------*
* A STOP or MODIFY command for the job has been issued.  Let's      *
* examine which command it is and perform more detailed processing  *
* for a MODIFY command.                                             *
*-------------------------------------------------------------------*
         ICM  R7,15,COMCIBPT       Get CIB address from COM area
         CLI   CIBVERB,CIBSTOP     Was it a STOP?
         BNE   CHKMODFY            No - check MODIFY
         QEDIT ORIGIN=COMCIBPT,BLOCK=(R7)
*                                  Free the CIB, clear the ECB
         B     DONE                We're all done
CHKMODFY DS    ØH
         CLI   CIBVERB,CIBMODFY    Was it a MODIFY?
         BE    MODIFY              Yes - let's check the data
         QEDIT ORIGIN=COMCIBPT,BLOCK=(R7)
*                                  Free the CIB, clear the ECB
         B     WAIT                And keep on waiting.
MODIFY   DS    ØH
         XR    R14,R14             Clear R14
         ICM   R14,B'ØØ11',CIBDATLN Get data length
         B     CHKCMDØ1            Go check valid commands
FREECIB  DS    ØH
         QEDIT ORIGIN=COMCIBPT,BLOCK=(R7)
*                                  Free the CIB, clear the ECB
         B     WAIT                And keep on waiting.
CHKCMDØ1 DS    ØH
         C     R14,=F'8'           Proper length for test?
         BNE   CHKCMDØ2            No - go check next command type
         CLC   CIBDATA(8),=C'SHUTDOWN' Shutdown request?
         BNE   CHKCMDØ2            No - go check next command type
         QEDIT ORIGIN=COMCIBPT,BLOCK=(R7)
*                                  Free the CIB, clear the ECB
         B     DONE                And we're all done
*-------------------------------------------------------------------*
CHKCMDØ2 DS    ØH
*-------------------------------------------------------------------*
         C     R14,=F'18'          Proper length for test?
```

```
         BNE   CHKCMDØ3             No - go check next command type
         CLC   CIBDATA(18),=C'DISPLAY=EXITSTATUS' Display status?
         BNE   CHKCMDØ3             No - go check next command type
         L     R1,16                Get CVT address
         USING CVT,R1               Set addressability
         L     R6,CVTRAC            Get RCVT address
         USING RCVT,R6              Set addressability
         DROP  R1
*-------------------------------------------------------------------*
         LA    R1,STATWTO+MODSTART  Point to first modification line
         LA    R15,EXITCNT          Get loop limit
         LA    R14,XITMAP           Get exit table address
STATLP1  DS    ØH
         MVC   DBL2(4),8(R14)       Copy exit offset
         UNPK  DBL1(9),DBL2(5)      Unpack it
         NC    DBL1(8),=8X'ØF'      Turn off high nibbles
         TR    DBL1(8),=C'Ø123456789ABCDEF' Make it readable
         MVC   OFFSTOFF(4,R1),DBL1+4 Move in RCVT exit offset
         L     R8,8(,R14)           Copy exit offset
         LA    R8,Ø(R8,R6)          Get exit addr in RCVT
         MVC   DBL2(4),Ø(R8)        Copy exit addr
         UNPK  DBL1(9),DBL2(5)      Unpack it
         NC    DBL1(8),=8X'ØF'      Turn off high nibbles
         TR    DBL1(8),=C'Ø123456789ABCDEF' Make it readable
         MVC   ADDROFF(8,R1),DBL1   Move in exit addr
         CLC   Ø(4,R8),=F'Ø'        An active exit?
         BE    NOEXIT               No - indicate no exit
         LA    R3,XITNMTBL          Get exit name table address
         LA    R4,XITNMTLN(,R3)     Get exit name table end address
STATLP2  DS    ØH
         CR    R3,R4                End of table?
         BNL   STATIND              Yes - indicate static
         CLC   Ø(8,R1),Ø(R3)        Exit name match?
         BNE   NEXTENT4             No - go check next entry
         CLC   12(4,R3),=F'Ø'       An active exit addr?
         BE    STATIND              No - indicate static
         MVC   LDSTOFF(7,R1),=C'DYNAMIC' Set dynamic indicator
         B     NXTSTATX             Check next exit status
NEXTENT4 DS    ØH
         LA    R3,ENTLEN(,R3)       Point to next entry
         B     STATLP2              Check it out
STATIND  DS    ØH
         MVC   LDSTOFF(7,R1),=C'STATIC ' Set static indicator
         B     NXTSTATX             Check next exit status
NOEXIT   DS    ØH
         MVC   LDSTOFF(7,R1),=C'N/A    ' Set N/A indicator
         B     NXTSTATX             Check next exit status
NXTSTATX DS    ØH
         LA    R1,MODINCR(,R1)      Point to next target area
         LA    R14,12(,R14)         Point to next exit info
```

```
         BCT   R15,STATLP1          Process next entry
         MODESET MODE=SUP,KEY=ZERO
         WTO   MF=(E,STATWTO)       Issue the WTO
         MODESET MODE=PROB,KEY=NZERO
         DROP  R6
*------------------------------------------------------------------*
         B     FREECIB              Go wait for more commands
*------------------------------------------------------------------*
CHKCMDØ3 DS    ØH
*------------------------------------------------------------------*
         C     R14,=F'7'            Proper length for test?
         BL    CHKCMDØ4             No - go check next cmd type
         CLC   CIBDATA(7),=C'DELETE=' Delete request?
         BNE   CHKCMDØ4             No - go check next command type
         C     R14,=F'15'           Proper length for test?
         BNE   CMDØ3NOX             No - issue message
         LA    R3,XITNMTBL          Get exit name table address
         LA    R4,XITNMTLN(,R3)     Get exit name table end address
         L     R1,16                Get CVT address
         USING CVT,R1               Set addressability
         L     R6,CVTRAC            Get RCVT address
         USING RCVT,R6              Set addressability
         DROP  R1
CMDØ3LP1 DS    ØH
         CR    R3,R4                End of table?
         BNL   CMDØ3NOX             Yes - requested exit nat matched
         CLC   Ø(8,R3),CIBDATA+7    Exit name match?
         BE    CMDØ3XON             Yes - check if dynamic load
         LA    R3,ENTLEN(,R3)       Point to next entry
         B     CMDØ3LP1             Check it out
CMDØ3NOX DS    ØH
         S     R14,=F'7'            Reduce length by 7
         C     R14,=F'8'            Still too large?
         BNH   CMDØ3ØØ5             No - use R14 length
         L     R14,=F'8'            Set to max of 8
CMDØ3ØØ5 DS    ØH
         BCTR  R14,Ø                Reduce by one for EX
         EX    R14,CMDØ3MV1         Move in specified name
CMDØ3WT1 WTO   'RACFXITS -          NOT AN ELIGIBLE EXIT',             X
               ROUTCDE=(1),DESC=(6)
         B     FREECIB              Go wait for more commands
CMDØ3XON DS    ØH
         CLC   12(4,R3),=F'Ø'       An active dynamic exit?
         BE    CMDØ3XOF             No - issue appropriate message
         LA    R14,XITMAP           Get exit map address
         LA    R15,XITMAPE(,R14)    Get exit map end address
CMDØ3LP2 DS    ØH
         CR    R14,R15              End of table?
         BNL   CMDØ3NOX             Yes - not possible, but bail out
         CLC   Ø(8,R3),Ø(R14)       An exit name match?
```

```
         BE    CMDØ3DLX             Yes - go delete exit
         LA    R14,12(,R14)         Point to next entry
         B     CMDØ3LP2             Check it out
CMDØ3DLX DS    ØH
         L     R8,8(,R14)           Save exit addr offset
         MODESET MODE=SUP,KEY=ZERO
         L     R1,8(,R3)            Get old exit address
         ST    R1,Ø(R8,R6)          Restore old address into RCVT
         LA    R1,Ø(,R3)            Point to exit name
         BAL   R14,EXITDEL          Go delete the exit
         MODESET MODE=PROB,KEY=NZERO
         XC    12(16,R3),12(R3)     Clear table information
         B     FREECIB              Go wait for more commands
CMDØ3XOF DS    ØH
         MVC   CMDØ3WT4+8+11(8),CIBDATA+7 MOVE IN REQUESTED NAME
CMDØ3WT4 WTO   'RACFXITS - xxxxxxxx NOT DYNAMICALLY LOADED',          X
               ROUTCDE=(1),DESC=(6)
         B     FREECIB              Go wait for more commands
*-------------------------------------------------------------------*
*   Executed instruction                                            *
*-------------------------------------------------------------------*
CMDØ3MV1 MVC   CMDØ3WT1+8+11(*-*),CIBDATA+7 Move in specified name
*-------------------------------------------------------------------*
CHKCMDØ4 DS    ØH
*-------------------------------------------------------------------*
         C     R14,=F'4'            Proper length for test?
         BL    CHKCMDØ5             No - go check next cmd type
         CLC   CIBDATA(4),=C'ADD='  Add request?
         BNE   CHKCMDØ5             No - go check next command type
         C     R14,=F'12'           Proper length for test?
         BNE   CMDØ4NOX             No - issue message
         LA    R3,XITMAP            Get exit map address
         LA    R4,XITMAPE(,R3)      Get exit map end address
         L     R1,16                Get CVT address
         USING CVT,R1               Set addressability
         L     R6,CVTRAC            Get RCVT address
         USING RCVT,R6              Set addressability
         DROP  R1
CMDØ4LP1 DS    ØH
         CR    R3,R4                End of table?
         BNL   CMDØ4NOX             Yes - requested exit not matched
         CLC   Ø(8,R3),CIBDATA+4    Exit name match?
         BE    CMDØ4XON             Yes - check if already active
         LA    R3,12(,R3)           Point to next entry
         B     CMDØ4LP1             Check it out
CMDØ4NOX DS    ØH
         S     R14,=F'4'            Reduce length by 4
         C     R14,=F'8'            Still too large?
         BNH   CMDØ4ØØ5             No - use R14 length
         L     R14,=F'8'            Set to max of 8
```

```
CMDØ4ØØ5 DS    ØH
         BCTR  R14,Ø                Reduce by one for EX
         EX    R14,CMDØ4MV1         Move in specified name
CMDØ4WT1 WTO   'RACFXITS -          NOT AN ELIGIBLE EXIT',            X
               ROUTCDE=(1),DESC=(6)
         B     FREECIB              Go wait for more commands
CMDØ4XON DS    ØH
         LA    R14,XITNMTBL         Get exit name table address
         LA    R15,XITNMTLN(,R14)   Get exit name table end address
CMDØ4LP2 DS    ØH
         CR    R14,R15              End of table?
         BNL   CMDØ4AD1             Yes - add is ok
         CLC   Ø(8,R3),Ø(R14)       An exit name match?
         BE    CMDØ4CHA             Yes - check if exit is active
         LA    R14,ENTLEN(,R14)     Point to next entry
         B     CMDØ4LP2             Check it out
CMDØ4CHA DS    ØH
         CLC   12(4,R14),=F'Ø'      Already dynamic exit?
         BE    CMDØ4AD2             No - add is OK
         MVC   CMDØ4WT2+8+11(8),CIBDATA+4 MOVE IN REQUESTED NAME
CMDØ4WT2 WTO   'RACFXITS - xxxxxxxx ALREADY DYNAMICALLY LOADED',      X
               ROUTCDE=(1),DESC=(6)
         B     FREECIB              Go wait for more commands
CMDØ4AD1 DS    ØH
         LR    R14,R3               Save XITMAP entry addr
         LA    R3,XITNMTBL          Get exit name table address
         LA    R4,XITNMTLN(,R3)     Get exit name table end address
CMDØ4LP3 DS    ØH
         CR    R3,R4                End of table?
         BNL   CMDØ4ER1             Yes - issue error
         CLC   Ø(8,R3),=8X'ØØ'      A blank entry?
         BE    CMDØ4AD3             Yes - go add entry
         LA    R3,ENTLEN(,R3)       Point to next entry
         B     CMDØ4LP3             Check it out
CMDØ4AD2 DS    ØH
         LR    R14,R3               Save XITMAP entry addr
         LA    R3,XITNMTBL          Get exit name table address
         LA    R4,XITNMTLN(,R3)     Get exit name table end address
CMDØ4LP4 DS    ØH
         CR    R3,R4                End of table?
         BNL   CMDØ4ER1             Yes - issue error
         CLC   Ø(8,R3),CIBDATA+4    The correct entry?
         BE    CMDØ4AD3             Yes - go add entry
         LA    R3,ENTLEN(,R3)       Point to next entry
         B     CMDØ4LP4             Check it out
CMDØ4ER1 DS    ØH
         MVC   CMDØ4WT3+8+11(8),CIBDATA+4 Move in requested name
CMDØ4WT3 WTO   'RACFXITS - xxxxxxxx NO AVAILABLE TABLE ENTRY',        X
               ROUTCDE=(1),DESC=(6)
         B     FREECIB              Go wait for more commands
```

```
CMDØ4AD3 DS    ØH
         MVC   Ø(8,R3),CIBDATA+4    Move in exit name
         L     R4,8(,R14)           Get RCVT offset of exit addr
         L     R8,Ø(R4,R6)          Get current exit addr
         ST    R8,8(,R3)            Save in table entry
         OPEN  (XITLIB),MODE=31     Open exit load library
         MODESET MODE=SUP,KEY=ZERO
         LA    R1,Ø(,R3)            Get exit name addr
         BAL   R14,EXITLOAD         Go load the exit
         LTR   R15,R15              Success?
         BNZ   CMDØ4END             No - we're done
         L     R1,12(,R3)           Get new exit addr
         ST    R1,Ø(R4,R6)          Copy exit address into RCVT
CMDØ4END DS    ØH
         MODESET MODE=PROB,KEY=NZERO
         CLOSE (XITLIB),MODE=31     Close exit dataset
         B     FREECIB              Go wait for more commands
*------------------------------------------------------------------*
*   Executed instruction                                           *
*------------------------------------------------------------------*
CMDØ4MV1 MVC   CMDØ4WT1+8+11(*-*),CIBDATA+4 Move in specified name
*------------------------------------------------------------------*
CHKCMDØ5 DS    ØH
*------------------------------------------------------------------*
         WTO   'RACFXITS - SPECIFIED MODIFY OPERATION NOT SUPPORTED',  x
               ROUTCDE=(1),DESC=(6)
         B     FREECIB              Release the CIB and issue msg
         DROP  R6
*------------------------------------------------------------------*
DONE     DS    ØH                   Prepare to return
*------------------------------------------------------------------*
         LA    R4,XITNMTBL          Get table address
         LA    R3,XITNMTLN(,R4)     Get exit name table end address
         S     R3,=A(ENTLEN)        Point to last entry
         MODESET MODE=SUP,KEY=ZERO
         L     R1,16                Get CVT address
         USING CVT,R1               Set addressability
         L     R6,CVTRAC            Get RCVT address
         USING RCVT,R6              Set addressability
         DROP  R1
XITLP2   DS    ØH
         CR    R3,R4                Before start of table?
         BL    ALLDONE              Yes - we're all done
*------------------------------------------------------------------*
XITDL1   DS    ØH
         LA    R7,XITMAP            Get exit map address
         LA    R9,XITMAPE(,R7)      Get exit map end address
XITLOC2  DS    ØH
         CR    R7,R9                End of table?
         BNL   XITNEXT2             Yes - bail out
```

```
         CLC   Ø(8,R7),Ø(R3)        An exit name match?
         BE    DLXIT1              Yes - process exit
         LA    R7,12(,R7)          Point to next entry
         B     XITLOC2             Go check it out
DLXIT1   DS    ØH
         L     R9,8(,R7)           Get RCVT offset of exit addr
         L     R1,8(,R3)           Get old exit address
         ST    R1,Ø(R9,R6)         Restore old address into RCVT
         LA    R1,Ø(,R3)           Point to exit name
         BAL   R14,EXITDEL         Go delete the exit
XITNEXT2 DS    ØH
         S     R3,=A(ENTLEN)       Point to prev entry
         B     XITLP2              Check it out
*-------------------------------------------------------------------*
         DROP  R6
ALLDONE  DS    ØH
         ESTAEX Ø
         MODESET MODE=PROB,KEY=NZERO
*-------------------------------------------------------------------*
RETURN   DS    ØH
         L     R3,SAVEAREA+4       Save old savearea address
         LR    R1,R13              Copy storage address
         STORAGE RELEASE,LENGTH=WORKLEN,ADDR=(R1)
         LR    R13,R3              Copy old savearea address
         LM    R14,R12,12(R13)     Restore the regs
         XR    R15,R15             RC=Ø
         BR    R14                 Return
*-------------------------------------------------------------------*
EXITLOAD DS    ØH
         STM   RØ,R15,REGSAVE      Save the registers
         MVC   NAMESAVE(8),Ø(R1)   Save the exit name
         CLC   12(4,R3),=F'Ø'      Already an exit loaded?
         BE    XLDCONT1            No - keep going
         MVC   XLDWTO1+8+16(8),Ø(R3) MOVE IN EXIT NAME
XLDWTO1  WTO   'RACFXITS - EXIT XXXXXXXX ALREADY LOADED',             x
               ROUTCDE=(1),DESC=(6)
         B     XLDFAIL             Return
XLDCONT1 DS    ØH
         LA    R7,DYLPAM1
         USING LPMEA,R7
         XC    DYLPAM1(LPMEA_LEN),DYLPAM1
         MVC   LPMEANAME,NAMESAVE
         CSVDYLPA REQUEST=ADD,                                       X
               MODINFOTYPE=MEMBERLIST,                               X
               MODINFO=DYLPAM1,                                      X
               NUMMOD=1,                                             X
               DCB=XITLIB,                                           X
               REQUESTOR==CL16'RACFXITS',                            X
               ERRORDATA=DYLPAE1
         LTR   R15,R15             OK?
```

```
         BZ    XLOADOK              Yes - exit module loaded fine
         MVC   XLDWTO2+19(8),NAMESAVE Copy the exit name into WTO
         ST    R15,DBL2             Save the return code
         UNPK  DBL1(9),DBL2(5)      Unpack it
         NC    DBL1(8),=8X'ØF'      Turn off high order nibbles
         TR    DBL1(8),=C'Ø123456789ABCDEF' Make it readable
         MVC   XLDWTO2+46(8),DBL1   Move into WTO
         ST    RØ,DBL2              Save the reason code
         UNPK  DBL1(9),DBL2(5)      Uunpack it
         NC    DBL1(8),=8X'ØF'      Turn off high order nibbles
         TR    DBL1(8),=C'Ø123456789ABCDEF' Make it readable
         MVC   XLDWTO2+61(8),DBL1   Move into WTO
         MVC   DBL2(4),DYLPAE1      Copy first part of error data
         UNPK  DBL1(9),DBL2(5)      Unpack it
         NC    DBL1(8),=8X'ØF'      Turn off high order nibbles
         TR    DBL1(8),=C'Ø123456789ABCDEF' Make it readable
         MVC   XLDWTO2+8Ø(8),DBL1   Move into WTO
         MVC   DBL2(4),DYLPAE1+4    Copy second part of error data
         UNPK  DBL1(9),DBL2(5)      Unpack it
         NC    DBL1(8),=8X'ØF'      Turn off high order nibbles
         TR    DBL1(8),=C'Ø123456789ABCDEF' Make it readable
         MVC   XLDWTO2+89(8),DBL1   Move into WTO
XLDWTO2  WTO   'RACFXITS - XXXXXXXX LOAD FAILED - RC: xxxxxxxx  RSN: xxX
               xxxxxx  ERRDATA: xxxxxxxx xxxxxxxx',                  X
               ROUTCDE=(1),DESC=(6)
XLDFAIL  DS    ØH
         LM    R2,R14,REGSAVE+8     Reload registers
         LA    R15,4                Set failure return code
         BR    R14                  Return
XLOADOK  DS    ØH
         MVC   XLDWTO3+19(8),NAMESAVE Copy the exit name into WTO
XLDWTO3  WTO   'RACFXITS - XXXXXXXX SUCCESSFULLY LOADED',             X
               ROUTCDE=(1),DESC=(6)
         L     R1,LPMEAENTRYPOINTADDR
         ST    R1,12(,R3)           Save exit address
         MVC   2Ø(8,R3),LPMEADELETETOKEN Save token value for delete
         LM    R2,R14,REGSAVE+8     Reload registers
         LA    R15,Ø                Set success return code
         BR    R14                  Return
         DROP  R7
*-------------------------------------------------------------------*
EXITDEL  DS    ØH
         STM   RØ,R15,REGSAVE       Save the registers
         MVC   NAMESAVE(8),Ø(R1)    Copy exit name
         CLC   2Ø(8,R3),=8X'ØØ'     A valid token?
         BE    XDELRET1             No - bypass the delete
         LA    R7,DYLPAM2
         USING LPMED,R7
         XC    DYLPAM2(LPMED_LEN),DYLPAM2
         MVC   LPMEDNAME,NAMESAVE
```

```
         MVC   LPMEDDELETETOKEN(8),2Ø(R3)
         CSVDYLPA REQUEST=DELETE,                                      X
               MODINFO=DYLPAM2,                                        X
               NUMMOD=1
         LTR   R15,R15              OK?
         BZ    XDELOK               Yes - exit module was delete fine
         MVC   XDELWTO+19(8),NAMESAVE Copy the exit name into WTO
XDELWTO  WTO   'RACFXITS - XXXXXXXX DELETE FAILED',                    X
               ROUTCDE=(1),DESC=(6)
XDELRET1 DS    ØH
         LM    R2,R14,REGSAVE+8     Reload registers
         LA    R15,4                Set failure return code
         BR    R14                  Return
XDELOK   DS    ØH
         MVC   XDELWTO2+19(8),NAMESAVE Copy the exit name into WTO
XDELWTO2 WTO   'RACFXITS - XXXXXXXX DELETED',                          X
               ROUTCDE=(1),DESC=(6)
         DROP  R7
         LM    R2,R14,REGSAVE+8     Reload registers
         LA    R15,Ø                Set success return code
         BR    R14                  Return
*-------------------------------------------------------------------*
XITLIB   DCB   MACRF=R,DDNAME=XITLIB,DSORG=PO
*-------------------------------------------------------------------*
XITMAP   DS    ØD
*                Exitname    RCVT offset
XITØ1    DC    C'ICHCCXØØ',A(RCVTNCDX-RCVT)  Specific CMD exit (DELETE)
XITENTLN EQU   *-XITØ1
XITØ2    DC    C'ICHCNXØØ',A(RCVTNCX-RCVT)   Specific CMD exit
XITØ3    DC    C'ICHDEXØ1',A(RCVTDESX-RCVT)  PWD AUTH exit DEXØ1
XITØ4    DC    C'ICHDEX11',A(RCVTDX11-RCVT)  PWD AUTH exit DEX11
XITØ5    DC    C'ICHFRXØ1',A(RCVTFRXP-RCVT)  REQUEST=FASTAUTH pre exit
XITØ6    DC    C'ICHFRXØ2',A(RCVTFRX2-RCVT)  REQUEST=FASTAUTH post exit
XITØ7    DC    C'ICHFRXØ3',A(RCVTFRX3-RCVT)  REQUEST=FASTAUTH pre exit
XITØ8    DC    C'ICHFRXØ4',A(RCVTFRX4-RCVT)  REQUEST=FASTAUTH post exit
XITØ9    DC    C'ICHPWXØ1',A(RCVTPWDX-RCVT)  New password exit
XIT1Ø    DC    C'ICHRCXØ1',A(RCVTRCX-RCVT)   REQUEST=AUTH pre exit
XIT11    DC    C'ICHRCXØ2',A(RCVTRCXP-RCVT)  REQUEST=AUTH post exit
XIT12    DC    C'ICHRDXØ1',A(RCVTRDX-RCVT)   REQUEST=DEFINE pre exit
XIT13    DC    C'ICHRDXØ2',A(RCVTRDXP-RCVT)  REQUEST=DEFINE post exit
XIT14    DC    C'ICHRIXØ1',A(RCVTRIX-RCVT)   VERIFY pre exit
XIT15    DC    C'ICHRIXØ2',A(RCVTRIXP-RCVT)  VERIFY post exit
XIT16    DC    C'ICHRLXØ1',A(RCVTRLX-RCVT)   REQUEST=LIST pre exit
XIT17    DC    C'ICHRLXØ2',A(RCVTRLXP-RCVT)  REQUEST=LIST post exit
XITMAPE  EQU   *-XITMAP
EXITCNT  EQU   (XITMAPE/XITENTLN)
*-------------------------------------------------------------------*
STATWTO  WTO   ('RACFXITS - RACF EXIT STATUS                ',D),     X
               ('Name      Address   RCVT offset  Load Status',D),     X
               ('--------  --------  -----------  -----------',D),     X
```

```
                ('ICHCCX00  xxxxxxxx  xxxx         xxx         ',D),     X
                ('ICHCNX00  xxxxxxxx  xxxx         xxx         ',D),     X
                ('ICHDEX01  xxxxxxxx  xxxx         xxx         ',D),     X
                ('ICHDEX02  xxxxxxxx  xxxx         xxx         ',D),     X
                ('ICHFRX01  xxxxxxxx  xxxx         xxx         ',D),     X
                ('ICHFRX02  xxxxxxxx  xxxx         xxx         ',D),     X
                ('ICHFRX03  xxxxxxxx  xxxx         xxx         ',D),     X
                ('ICHFRX04  xxxxxxxx  xxxx         xxx         ',D),     X
                ('ICHPWX01  xxxxxxxx  xxxx         xxx         ',D),     X
                ('ICHRCX01  xxxxxxxx  xxxx         xxx         ',D),     X
                ('ICHRCX02  xxxxxxxx  xxxx         xxx         ',D),     X
                ('ICHRDX01  xxxxxxxx  xxxx         xxx         ',D),     X
                ('ICHRDX02  xxxxxxxx  xxxx         xxx         ',D),     X
                ('ICHRIX01  xxxxxxxx  xxxx         xxx         ',D),     X
                ('ICHRIX02  xxxxxxxx  xxxx         xxx         ',D),     X
                ('ICHRLX01  xxxxxxxx  xxxx         xxx         ',D),     X
                ('ICHRLX02  xxxxxxxx  xxxx         xxx         ',D),     X
                ('END OF RACF EXIT STATUS DISPLAY                ',DE),    X
                ROUTCDE=(1),DESC=(6),MF=L
MODSTART EQU   156
MODINCR  EQU   48
MODEND   EQU   (MODSTART+(EXITCNT*MODINCR))
ADDROFF  EQU   10
OFFSTOFF EQU   20
LDSTOFF  EQU   33
*-------------------------------------------------------------------*
         LTORG
*-------------------------------------------------------------------*
WORKAREA DSECT
SAVEAREA DS    18F
REGSAVE  DS    18F
*-------------------------------------------------------------------*
*   Each exit entry had the following format:
*XITNAME  DS    CL8                One of the supported exit names
*XITOLDA  DS    F                  Addr of old exit
*XITNEWA  DS    F                  Addr of new exit
*XITRSRV1 DS    F                  Reserved - was new exit length
*XITTOKEN DS    XL8                CSVDYLPA LOAD add value
XITNMTBL DS    0D,CL(30*ENTLEN)   30 entries - 28 bytes each
XITNMTLN EQU   *-XITNMTBL
ENTLEN   EQU   28
*-------------------------------------------------------------------*
ECBS     DS    5F                 ECB list for WAIT
COMADDR  DS    F                  ADDR(COMAREA) from EXTRACT
EXTRACT  EXTRACT MF=L             EXTRACT parameter list
DBL1     DS    2D
DBL2     DS    2D
NAMESAVE DS    CL8
DYLPAM1  DS    0D,CL(LPMEA_LEN)
DYLPAE1  DS    XL8
```

```
DYLPAM2  DS    ØD,CL(LPMED_LEN)
WORKLEN  EQU   *-WORKAREA
         CVT   DSECT=YES
         ICHPRCVT ,
COM      DSECT
         IEZCOM   ,                COM area
CIB      DSECT
         IEZCIB   ,                CIB
         CSVLPRET
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
ESTAERTN CSECT
ESTAERTN AMODE 31
ESTAERTN RMODE 24
*-------------------------------------------------------------------*
*                                                                   *
*   Register contents on entry if an SDWA is available:             *
*                                                                   *
*     RØ  - Ø, 4, 8, or 16                                          *
*     R1  - address of the SDWA                                     *
*     R2  - address of the parm list specified on the ESTAE PARAM=  *
*     R3-R12 - no relevant info                                     *
*     R13 - savearea                                                *
*     R14 - return address                                          *
*     R15 - address of the ESTAE routine                            *
*                                                                   *
*   Register contents on entry if no SDWA is available:             *
*                                                                   *
*     RØ  - 12                                                      *
*     R1  - completion code                                         *
*     R2  - address of the parm list specified on the ESTAE PARAM=  *
*     R3-R13 - no relevant info                                     *
*     R14 - return address                                          *
*     R15 - address of the ESTAE routine                            *
*                                                                   *
*-------------------------------------------------------------------*
```

```
         USING ESTAERTN,R15
         C     RØ,=F'12'            SDWA present?
         BE    NOSDWA1              No - process as such
         STM   R14,R12,12(R13)      Save environment
         B     SETUP                Continue
NOSDWA1  DS    ØH
         STM   R14,R12,12(R2)       R2 points to save area parm
         LR    R13,R2               Point to save area
SETUP    DS    ØH
         DROP  R15
         LR    R11,R15              Set up ...
         USING ESTAERTN,R11            new addressability
         LR    R3,RØ                Save SDWA flag
         LR    R4,R1                Save SDWA address
         ST    R13,ERRSAVE+4        Save old save area address
         LA    R13,ERRSAVE          Get new save area address
         XR    R12,R12              Clear R12
         C     R3,=F'12'            SDWA?
         BE    NOSDWA2              No - bypass SDWA processing
         LR    R12,R4               Set up addressability ...
         USING SDWA,R12                to the SDWA
         L     R2,SDWAPARM          Get parameter area address
         L     R2,Ø(,R2)            Get parameter area address
NOSDWA2  DS    ØH
         LTR   R12,R12              SDWA?
         BZ    NOSDWA3              No - bypadd SDWA
         UNPK  ERRDBL1(7),SDWAABCC+1(4) Unpack abend code
         B     ERROR1
NOSDWA3  DS    ØH
         ST    R4,ERRDBL2           Save abend code
         UNPK  ERRDBL1(7),ERRDBL2+1(4) Unpack abend code
ERROR1   DS    ØH
         NC    ERRDBL1(6),=6X'ØF'   Make abend ...
         TR    ERRDBL1(6),=C'Ø123456789ABCDEF' code readable
         MVC   ERWTOWRK(ERWTOLEN),ERRWTO Move in WTO
         MVC   ERWTOWRK+22(3),ERRDBL1 Move in abend code
         CLC   ERRDBL1(3),=3C'Ø'    System code zero?
         BNE   ISSUEERR             No - issue error
         MVC   ERWTOWRK+16(5),=C'USER ' Move in new text
         MVC   ERWTOWRK+22(3),ERRDBL1+3 Move in user code
ISSUEERR DS    ØH
         WTO   MF=(E,ERWTOWRK)      Issue WTO
         LA    R4,72(,R2)           Get address of exit table
         LA    R3,XITNMTLN(,R4)     Get exit name table end address
         S     R3,=A(ENTLEN)        Point to last entry
         MODESET MODE=SUP,KEY=ZERO
         L     R1,16                Get CVT address
         USING CVT,R1               Set addressability
         L     R6,CVTRAC            Get RCVT address
         USING RCVT,R6              Set addressability
```

```
         DROP  R1
XITLP3   DS    ØH
         CR    R3,R4               Past start of table?
         BL    ALLDONE2            Yes - we're all done
*-------------------------------------------------------------------*
XITDL2   DS    ØH
         L     R7,=A(XITMAP)       Get exit map address
         LA    R9,XITMAPE(,R7)     Get exit map end address
XITLOC3  DS    ØH
         CR    R7,R9               End of table?
         BNL   XITNEXT3            Yes - bail out
         CLC   Ø(8,R7),Ø(R3)       An exit name match?
         BE    DLXIT2              Yes - process exit
         LA    R7,12(,R7)          Point to next entry
         B     XITLOC3             Go check it out
DLXIT2   DS    ØH
         L     R9,8(,R7)           Get RCVT offset of exit addr
         L     R1,8(,R3)           Get old exit address
         ST    R1,Ø(R9,R6)         Restore old address into RCVT
         LA    R1,Ø(,R3)           Point to exit name
         BAL   R14,EXITDEL2        Go delete the exit
         B     XITNEXT3            Process next entry
XITNEXT3 DS    ØH
         S     R3,=A(ENTLEN)       Point to prev entry
         B     XITLP3              Check it out
*-------------------------------------------------------------------*
         DROP  R6
ALLDONE2 DS    ØH
         MODESET MODE=PROB,KEY=NZERO
*-------------------------------------------------------------------*
         L     R13,ERRSAVE+4       Get old save area address
         LTR   R12,R12             SDWA?
         BZ    END                 No - end
         SETRP WKAREA=(R12),REGS=(14),DUMP=YES,RC=Ø
END      DS    ØH
         LM    RØ,R12,2Ø(R13)      Restore environment
         XR    R15,R15             Clear R15
         L     R14,12(,R13)        Point to RTM
         BR    R14                 Return
ERRRTN2  DS    ØH
         L     R13,ERRSAVE+4       Restore savearea address
         LM    R14,R12,12(R13)     Restore environment
         LA    R15,4               Set return code
         BR    R14                 Return
*-------------------------------------------------------------------*
EXITDEL2 DS    ØH
         STM   RØ,R15,REGSAVE      Save the registers
         MVC   NAMESAV2(8),Ø(R1)   Copy exit name
         CLC   2Ø(8,R3),=8X'ØØ'    A valid token?
         BE    XDELRET2            No - bypass the delete
```

```
         LA    R7,DYLPAM3
         USING LPMED,R7
         XC    DYLPAM3(LPMED_LEN),DYLPAM3
         MVC   LPMEDNAME,NAMESAV2
         MVC   LPMEDDELETETOKEN(8),2Ø(R3)
         CSVDYLPA REQUEST=DELETE,                                      X
               MODINFO=DYLPAM3,                                        X
               NUMMOD=1
         LTR   R15,R15             OK?
         BZ    XDELOK2             YES - exit module delete was fine
         MVC   XDELWTO3+19(8),NAMESAV2 Copy the exit name into WTO
XDELWTO3 WTO   'RACFXITS - XXXXXXXX DELETE FAILED',                    X
               ROUTCDE=(1),DESC=(6)
XDELRET2 DS    ØH
         LM    R2,R14,REGSAVE+8    Reload registers
         LA    R15,4               Set failure return code
         BR    R14                 Return
XDELOK2  DS    ØH
         MVC   XDELWTO4+19(8),NAMESAV2 Copy the exit name into WTO
XDELWTO4 WTO   'RACFXITS - XXXXXXXX DELETED',                          X
               ROUTCDE=(1),DESC=(6)
         DROP  R7
         LM    R2,R14,REGSAVE+8    Reload registers
         LA    R15,Ø               Set success return code
         BR    R14                 Return
*-------------------------------------------------------------------*
ERRWTO   WTO   'RACFXITS  - ABEND XXX DETECTED.  ERROR RECOVERY IN PROGX
               RESS.',ROUTCDE=(1),DESC=(1),MF=L
ERWTOLEN EQU   *-ERRWTO
ERRSAVE  DS    18F
ERR2SAVE DS    18F
ERRDBL1  DS    2D
ERRDBL2  DS    2D
NAMESAV2 DS    CL8
DYLPAM3  DS    ØD,CL(LPMED_LEN)
         DS    F
ERWTOWRK DS    CL(ERWTOLEN)
         LTORG
         IHASDWA
         END
```

RACF in focus – migrating DB2 security to RACF

'RACF in focus' is a regular column focusing on specific aspects of RACF. Here, we consider how best to migrate DB2 security to RACF, a task currently being undertaken by many installations.

BACKGROUND ON DB2 INTERNAL SECURITY

DB2's internal security involves protecting the following:

- DB2 objects, such as plans, views, tables, table spaces, databases, etc.
- DB2 special privileges, such as SYSADM, SYSCTRL, DBADM, DBCTRL, DBMAINT, etc.

DB2 has its own ways of controlling its internal security. Typically, a person knowledgeable in DB2 issues the SQL 'grants' and 'revokes', which are similar to RACF's 'permit' and 'permit delete' commands. DB2 keeps its security information in its own tables.

This means that DB2 security remains within the DB2 universe, and separate from RACF. (Note, however, that external objects such as DB2 logs, catalogs, and databases have always been protected by RACF.)

However, while RACF security has been busily advancing with the times, providing better facilities to administer, monitor, report, and audit, DB2 security remains pretty much where it was when it was first introduced. Add-on vendor tools help to address DB2's shortcomings, but even these have their limitations, providing a 'front-end' to simplify security administration, but doing little to address the core issues.

RACF NOW PROVIDES DB2 SECURITY

The pitfalls of DB2 security did not go unnoticed. With Version 5 of DB2 and OS/390 Version 2 release 4, IBM announced the capability to bring DB2's internal security within the RACF fold.

Since most installations now have these levels of software or better, migrating to RACF is only a matter of having the will to do the conversion, and overcoming political hurdles such as where DB2 security administration function should reside. When you migrate DB2 security to RACF, you have RACF profiles controlling access to DB2 objects and privileges.

WHY MIGRATE?

So why migrate? Well, a better question might be, why resist? Remember the time when CICS had its own internal security? Nowadays, it's rare to find a shop where CICS security resides within itself. The same drama is unfolding in the DB2 world.

RACF-controlled DB2 security provides several benefits:

- RACF provides 'wild-carding', or masking of profiles, so one profile can protect a number of DB2 objects. This reduces the security maintenance and administration effort. In DB2, there is one grant per DB2 object, and masking is not allowed. Can you imagine the administrative nightmare we would have if we had to protect each dataset separately? Such is the case with DB2's internal security.
- In RACF, you can provide DB2 access at the group level instead of at the individual level. This has obvious benefits.
- The drawbacks of DB2 security, such as 'cascading revokes', don't apply in RACF. Again, this reduces the security administration effort.
- Using RACF, you can pre-define a profile for a DB2 object (or objects), even when the DB2 object (or objects) does not exist. In DB2, an object has to exist before its security can be specified. What's more, removing (dropping) a DB2 object doesn't remove the RACF profiles protecting it. With DB2's internal security, if a DB2 object is removed (dropped), its security is dropped too.
- The trend is towards bringing all IBM mainframe security under the RACF umbrella. Security administration,

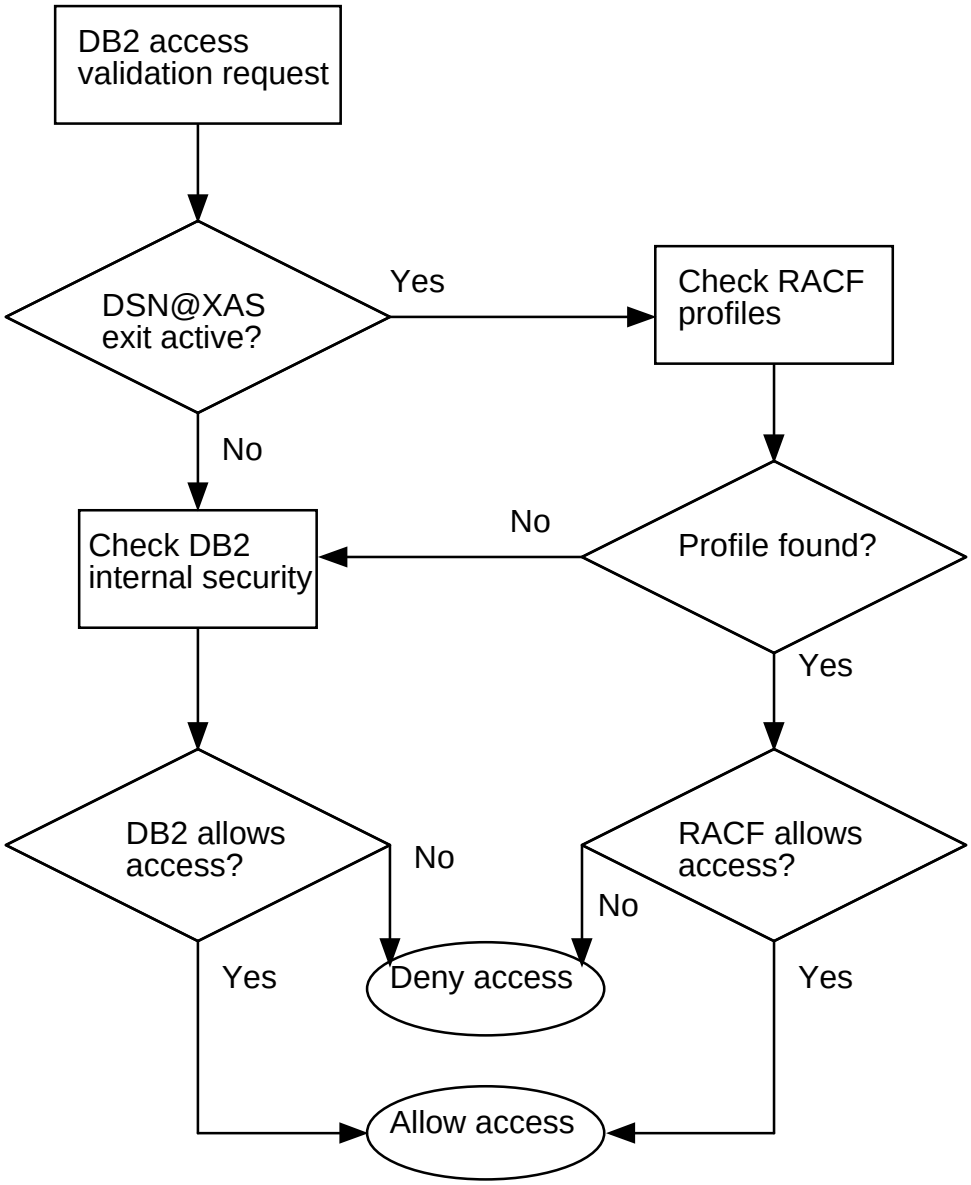

Figure 1: How the exit works

monitoring, and reporting become easy when there's a central repository for all security. There is also a standard protection mechanism (RACF profiles) across multiple applications (RACF profiles for DB2 versus DB2 grants and revokes; RACF profiles for CICS versus CICS internal security).

The best thing about moving DB2 security to RACF is the relative

ease with which it can be done: DB2 security can be phased in gradually. You don't need to convert all of the DB2 security to RACF at once, although that's possible. You can pick a portion of DB2 security – such as plans – and move them under RACF profiles. When you're happy with the results, you can pick the next portion to migrate, and so on. During this transition phase, some security will reside in RACF, and some in DB2.

The transition period could be a week or a year – the choice is yours. Of course, the shorter the transition period, the less the confusion about who (RACF or DB2) protects what (DB2 object), and who is responsible for security (DB2 folk or RACF folk).

TRAINING AND EDUCATION

Introducing DB2 security to RACF will mean that the RACF administrators will have to learn new terms, which, in the case of DB2, may seem like a daunting task. But it's no different from learning about, say, CICS terms such as transactions and FCTs – the RACF administrators don't have to know what a transaction or an FCT is, merely how to protect it. The same goes for DB2. Over time, the RACF administrators will learn the DB2 terms, and a brief introduction from the DB2 team may help pave the way.

MIGRATING TO RACF

Needless to say, migrating DB2 security to RACF should be a joint project between the RACF and the DB2 folks. You need to decide what to migrate first, how much to migrate, and so on. The biggest change at the end of the migration effort will be that security administration for DB2 will be done in RACF instead of in DB2.

IBM provides a utility that converts DB2's internal security (the grants) into RACF commands. You can use this utility to convert the existing DB2 security into RACF profiles.

RACF security checking for DB2 is activated by implementing

the DB2 Authorization Exit DSNX@XAS, which intercepts DB2 authorization requests and calls RACF for access validation. If RACF cannot validate the request (for example, because the DB2 class for the object is not activated), DB2 reverts to its own internal security. This makes it possible to have a phased approach, whereby you migrate security for one class of DB2 objects at a time, learn from the experience, and then migrate the security for the next class of DB2 objects. IBM provides a sample DSNX@XAS exit. This exit doesn't have to be modified unless you want to change the defaults. The flowchart in Figure 1 shows how the exit works.

Migrating to RACF security involves activating RACF resource classes for DB2, and defining appropriate profiles. So let's look at each DB2 object, and consider how its security validation can be migrated to RACF. But first, there are some common features.

Some common features among all RACF profiles for DB2

The first qualifier of the profile name is the DB2 subsystem name, such as DB2P; the last qualifier denotes the DB2 privilege or administrative authority. The middle qualifiers are different, according to the type of DB2 object. In the discussion below, we use DB2P as the subsystem name, but it can be any subsystem name at your installation. In fact, you can even have more than one subsystem, in which case you'll have sets of profiles for each subsystem.

In all cases, an access level of READ in the RACF profile means that the userid (or group) has access to the DB2 object covered by the profile.

Each DB2 resource class in RACF has a corresponding grouping class, with the exception of the DSNADM class. Thus, the grouping class for MDSNPN is GDSNPN, and so on. These grouping classes will not be discussed here, as the idea behind them is similar to other grouping classes, such as CICS grouping classes.

What follows can be used to phase in RACF security – that is, to

migrate DB2 plans first, then DB2 table spaces, and so on But if you prefer a different order, that's OK, too – the end result will be the same.

Defining RACF profiles for DB2 plans

Activate the RACF resource class MDSNPN.

The RACF profiles for DB2 plans have two formats:

- DB2P.plan_name.EXECUTE, for execute access to the plan.
- DB2P.plan_name.BIND, for bind access to the plan.

Example

DB2P.plan*.EXECUTE protects the execute privilege for all plans beginning with 'plan' in DB2P.

Defining RACF profiles for DB2 table spaces

Activate the RACF resource class MDSNTS.

The RACF profiles for DB2 table spaces have the following format:

- DB2P.database.table_space.USE

Example

DB2P.DB123.TS45*.USE protects the USE privilege for all table spaces starting with 'TS45' in database DB123 in DB2P.

Defining RACF profiles for DB2 storage groups

Activate the RACF resource class MDSNSG.

The RACF profiles for DB2 storage groups have the following format:

- DB2P.storage_group.USE

Example

DB2P.STGRP987.USE protects the USE privilege for storage group STGRP987 in DB2P.

Defining RACF profiles for DB2 databases

Activate the RACF resource class MDSNDB.

The RACF profiles for DB2 databases have the following format:

- DB2P.database_name.PRIVILEGE

where privilege can be any of the dozen or so database privileges, such as STOPDB, LOAD, DROP, REORG, REPAIR, etc.

Example

DB2P.SALESDB.REORG protects the REORG privilege for database SALESDB in DB2P.

Defining RACF profiles for DB2 buffer pools

Activate the RACF resource class MDSNBP.

The RACF profiles for DB2 buffer pools have the following format:

- DB2P.buffer_pool.USE

Example

DB2P.BUFP*.USE protects the USE privilege for all buffer pools beginning with BUFP in DB2P.

Defining RACF profiles for DB2 packages

Activate the RACF resource class MDSNPK.

The RACF profiles for DB2 packages have the following formats:

- DB2P.collection_id.package.BIND
- DB2P.collection_id.package.COPY

- DB2P.collection_id.package.EXECUTE

Example

DB2P.COLLECT9.*.EXECUTE specifies the EXECUTE privilege for any package in collection COLLECT9 in DB2P.

Defining RACF profiles for DB2 user-defined functions

Activate the RACF resource class MDSNUF.

The RACF profiles for DB2 user-defined functions have the following formats:

- DB2P.schema_name.function_name.DISPLAY
- DB2P.schema_name.function_name.EXECUTE

Example

DB2P.*.*.DISPLAY protects the DISPLAY privilege for any user-defined function in DB2P.

Defining RACF profiles for DB2 collections

Activate the RACF resource class MDSNCL.

The RACF profiles for DB2 table spaces have the following format:

- DB2P.collection_name.CREATEIN

Example

DB2P.COL*.CREATEIN protects the CREATEIN privilege for all collections beginning with COL.

Defining RACF profiles for DB2 tables

Activate the RACF resource class MDSNTB.

The RACF profiles for DB2 tables have the following format:

- DB2P.table_owner.table_name.PRIVILEGE

where PRIVILEGE is one of: ALTER, DELETE, INDEX, INSERT, SELECT, REFERENCES, UPDATE, or TRIGGER.

Example

DB2P.OWN1.TABL123.* protects all the privileges for TABL123 whose owner is OWN1 in DB2P.

Defining RACF profiles for DB2 schemas

Activate the RACF resource class MDSNSC.

The RACF profiles for DB2 schemas have the following formats:

- DB2P.schema.CREATIN for the CREATIN privilege
- DB2P.schema.object.PRIVILEGE, where privilege is either ALTERIN or DROPIN.

Example

DB2P.SCH001.*.DROPIN protects the DROPIN privilege for all objects in schema SCH001 in DB2P.

Defining RACF profiles for DB2 stored procedures

Activate the RACF resource class MDSNSP.

The RACF profiles for DB2 stored procedures have the following format:

- DB2P.schema.procedure.PRIVILEGE

where PRIVILEGE is either DISPLAY or EXECUTE.

Example

DB2P.SC001.PR*.DISPLAY protects the DISPLAY privilege for all procedures starting with PR in schema SC001 in DB2P.

Defining RACF profiles for DB2 user-defined distinct types

Activate the RACF resource class MDSNUT.

The RACF profiles for DB2 user-defined distinct types have the following format:

- DB2P.schema.type_name.USAGE

Example

DB2P.SCHPAY.*.USAGE protects the USAGE privilege for all type names in schema SCHPAY in DB2P.

Defining RACF profiles for DB2 system privileges

Activate the RACF resource class MDSNSM.

The RACF profiles for DB2 system privileges have the following formats:

- DB2P.package_owner.BINDAGENT
- DB2P.PRIVILEGE

where PRIVILEGE is one of ARCHIVE, DISPLAY, STOPALL, TRACE, etc (there are many).

Example

DB2P.* protects all system privileges in DB2P.

Defining RACF profiles for DB2 administrative authorities

Activate the RACF resource class DSNADM. (Note the slight variation in name from other DB2 classes. This class is the only one that doesn't have a corresponding grouping class.)

For system authorities, the RACF profiles for DB2 administrative authorities have the following format:

- DB2P.PRIVILEGE

where PRIVILEGE is one of SYSADM, SYSCTRL, SYSOPR.

Example

DB2P.SYSCTRL protects the SYSCTRL privilege in DB2P.

For database authorities, the RACF profiles for DB2 administrative authorities have the following format:

- DB2P.database.PRIVILEGE

where PRIVILEGE is one of DBADM, DBCTRL, or DBMAINT.

Example

DB2P.SALES*.DBMAINT protects the DBMAINT privilege for any database that starts with SALES in DB2P.

Some additional considerations

The following additional points also need to be considered:

- IBM's DB2 to RACF conversion utility will not group specific profiles into more generic ones. To derive the full benefit of your conversion, you may have to do some clean-up work after each DB2 class is migrated to RACF. This clean-up work can also be done before executing the commands generated by the conversion utility. For example, you may end up with a large number of the following profiles, each having similar access control requirements:
  - DB2P.PLANR123.EXECUTE
  - DB2P.PLANR124.EXECUTE
  - DB2P.PLANR88.EXECUTE
  - . . .
  - . . .
  - DB2P.PLANR999.EXECUTE

  If the access requirements are the same for these profiles, you can write a single profile to replace them all:

  DB2P.PLANR*.EXECUTE
- Since DB2's internal security is always invoked for cases where no matching RACF profile is found, it's important, when you finally want to 'take charge' of all DB2 security for

a particular class (remember, you can do this on a class by class basis), that you create in that class a 'back-stop' profile of the type DB2P.** which will always match the 'left-over' profiles in the class, and thus never pass on control to DB2's internal security.

- RACF security request forms will need to be enhanced. You probably have an on-line request system for the user community for requesting additions, deletions, and changes to RACF security. These will need to be updated to allow for the DB2 security objects mentioned above.

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

Making it easy to switch IDs in TSO

Have you ever hit PF3 in the ISPF Editor and had RACF refuse to save your changes? And then had to CANCEL, get all the way out of ISPF, log on to a different TSO ID, find the dataset, and make all the changes again before you can hit PF3? The chances are that you didn't even remember all the changes you made the first time.

The TSO command USEID solves this – and many other problems – by allowing you to switch IDs in the middle of virtually anything in ISPF, TSO, or any application that runs on ISPF. Simply type TSO USEID in ISPF, or just USEID at a TSO READY prompt or in ISPF Option 6, followed by the user ID you want to switch to, and you'll be prompted for the ID's password and taken back to where you left off.

Specifying USEID without an ID returns you to your log-on ID. About the only things to remind you that you logged on with your original ID are the JOBNAME and OWNER fields that you'll see

if you look at your TSO session with SDSF's DA (Display Active) command.

If you submit a batch job, the SDSF-displayed OWNER field and the value of &SYSUID in the job's JCL will be your assumed ID. It's just as if you had originally logged on to TSO with the assumed ID and submitted the batch job. The only thing you lose is NOTIFY=&SYSUID on the JOB card of the batch job; if you code it, you won't receive a message when the job completes because, technically, you're still logged on in TSO with your original rather than your assumed ID.

IMPLEMENTATION EXAMPLE

USEID is used in one organization to try to dissuade technical staff from always logging on with their 'super power' RACF user IDs. In fact, the RACF TSO segment of all such IDs has been deleted – users must log on with their normal, slightly less powerful ID, and then use USEID to switch to super powers only when they need them.

It's a nice theory and, so far, it has satisfied the auditors. But it's easy to switch to the super power ID and forget to switch back, which can be a problem, given that most systems programmers stay logged on all day. However, that still doesn't take away from the fact that USEID is an extremely handy command.

HISTORY

This code was written over 20 years ago by Trevor Howard when he first joined the Alberta Government in Canada, and he recently described it as "the most useful 100 lines of code I ever wrote".

Because it was written so long ago, it uses the original RACF macros, which are no longer supported, but still work. One of IBM Canada's top technical people recently reviewed the code and commented that no RACF short-cuts were taken, which doubtless explains why it has never caused any RACF problems or errors over the many years of its use at several local z/OS sites.

```
USEID    TITLE  '- TSO COMMAND PROCESSOR'
*************************************************************************
*                                                                       *
*               USEID                                                   *
*   PURPOSE:                                                            *
*     ALLOW A TSO USER TO ASSUME THE RACF IDENTITY OF ANOTHER ID        *
*                                                                       *
*   AUTHOR/ORIGIN:                                                      *
*     T.HOWARD, ALTA PWSS                                               *
*                                                                       *
*   CALLING SEQUENCE (ASSEMBLER):                                       *
*                                                                       *
*           CALL USEID,(CPPL)                                           *
*                                                                       *
*   INVOCATION:                                                         *
*                                                                       *
*           USEID ID</PASSWORD>                                         *
*                                                                       *
*           THE PASSWORD IS OPTIONAL BUT IF OMITTED IT WILL BE          *
*           PROMPTED FOR, UNLESS ID SPECIFIES THE SAME ID AS            *
*           ORIGINALLY LOGGED ON, IN WHICH CASE THE COMMAND             *
*           WILL BE ALLOWED TO PROCEED. IF ID SPECIFIES A USER ID       *
*           WHICH IS NOT IN THE CURRENT ID'S GROUP, READ ACCESS         *
*           IS REQUIRED TO APPL USEIDG.                                 *
*   INPUT:                                                              *
*           CPPL IS THE COMMAND PROCCESSOR PARAMETER LIST AS            *
*           DEFINED IN TSO:GUIDE TO WRITING A TMP.                      *
*                                                                       *
*           BY A PASSWORD, WITH THE TWO SEPARATED BY A SEMICOLON.       *
*           IF THE PASSWORD IS NOT SUPPLIED, IT WILL BE PROMPTED        *
*           FOR.                                                        *
*           NOTE: BOTH THE ID AND PASSWORD ARE OPTIONAL - IF NO         *
*           PASSWORD IS SUPPLIED AND THE ID IS THE SAME AS THE          *
*           ID RUNNING THE PROGRAM, THE ACEE WILL BE REFRESHED          *
*           WITH THE USER'S OWN PROFILE. IF NO ID IS GIVEN, THE         *
*           ID IS ASSUMED TO BE THAT OF THE CALLER OF THE PROGRAM.      *
*                                                                       *
*   OUTPUT:                                                             *
*                                                                       *
*           PRINTS MESSAGE AT TERMINAL OR OPERATOR CONSOLE              *
*                                                                       *
*   FUNCTION:                                                           *
*                                                                       *
*     THIS PROGRAM WILL, UPON VERIFICATION OF THE ID AND PASSWORD,      *
*     CAUSE THE ACEE OF THE INVOKING JOB TO BE REPLACED WITH ONE        *
*     REPRESENTING THE SPECIFIED USERID. THIS EFFECTIVELY MAKES         *
*     THE CALLING ID HAVE THE SAME ACCESS AS THE SPECIFIED ID.          *
*                                                                       *
```

```
*   RESTRICTIONS: THE NEW ID MUST HAVE AUTHORITY TO THE CALLER'S      *
*     CURRENT GROUP AND TERMINAL.                                     *
*                                                                     *
*   INSTALLATION CONSIDERATIONS:                                      *
*                                                                     *
*   ENVIRONMENT:                                                      *
*      MUST BE APF AUTHORIZED                                         *
*                                                                     *
*   VERSION: 1.Ø                                                      *
*                                                                     *
*   CHANGES: Ø1/13/88 - REMOVED IN-HOUSE MACRO DEPENDENCIES           *
*            Ø6/21/9Ø - ADDED RELEASE PARAMETER TO ALL RACF MACROS    *
*            27/Ø1/92 - CHANGED RELEASE TO 1.9 ON RACF MACROS         *
*            17/Ø4/93 - CHANGED RELEASE BACK TO 1.8.1                 *
*                     - ADDED INIT CODE TO BLANK NEW USERID FIELD     *
*            3Ø/Ø7/96 - CHANGED RELEASE BACK TO 1.8.1                 *
*            JUNE3/Ø3 - FIX >8 CHAR. ID FAILURE AND TIDY UP SOURCE    *
*                       (JON PEARKINS)                                *
*                                                                     *
***********************************************************************
         PRINT ON,NOGEN
         YREGS
         IHAASCB
         IHAACEE
         IKJPSCB
         IKJCPPL
         IKJIOPL
         IKJPPL
         IKJPGPB
         PRINT ON,GEN
USEID    CSECT
         SAVE  (14,12),,MOD_Ø1/13/88_DIS_99/99/99_ASM_&SYSDATE
         USING USEID,R12               SET UP BASE ADDRESSABILITY
         LR    R12,R15                 LOAD BASE REG WITH ENTRY POINT
***********************************************************************
*                                                                     *
*   GETMAIN - GET STORAGE FOR A WORKAREA AND CHAIN THE SAVE AREAS     *
*   TOGETHER.                                                         *
*                                                                     *
***********************************************************************
         LR    R2,R1
         L     RØ,GETPARM
         GETMAIN R,LV=(Ø)
         LR    R6,R1
         USING WORKAREA,R6
         LR    R1,R13
         LA    R13,SAVEAREA
         ST    R13,8(R1)
         ST    R1,4(R13)
         XC    USERAREA(CLEARL),USERAREA
```

```
         MVI   ID,C' '                     BLANK NEW USERID FIELD
         MVC   ID+1(L'ID-1),ID
         LR    R1,R2
*********************************************************************
*                                                                   *
*   INITIALIZATION - INIT VARIOUS PARM. BLOCKS, DETERMINE USERID AND  *
*   DEFAULT GROUP.                                                  *
*                                                                   *
*********************************************************************
         LR    R1Ø,R1
         USING CPPL,R1Ø
         ST    R1Ø,CPPLPTR
*
*  INITIALIZE IOPL
*
         LA    R15,MYIOPL
         USING IOPL,R15
         MVC   IOPLUPT(4),CPPLUPT
         MVC   IOPLECT(4),CPPLECT
         LA    RØ,PUTLECB
         ST    RØ,IOPLECB
*
*  INITIALIZE PUTLINE PARAMETER BLOCK
*
         LA    RØ,MYPTPB
         ST    RØ,IOPLIOPB
*
*  SET UP PPL FOR IKJPARS
*
         LA    R15,MYPPL
         USING PPL,R15
         MVC   PPLUPT(4),CPPLUPT
         MVC   PPLECT(4),CPPLECT
         LA    RØ,PARSEECB
         ST    RØ,PPLECB
         XC    PARSEECB,PARSEECB
         L     RØ,=A(MYPCL)
         ST    RØ,PPLPCL
         LA    RØ,MYANS
         ST    RØ,PPLANS
         MVC   PPLCBUF(4),CPPLCBUF
         L     R1,CPPLCBUF
         ST    R1,CBUFPTR
         MVC   CBUFHDR(4),Ø(R1)
         LA    RØ,MYUWA
         ST    RØ,PPLUWA
         DROP  R15
*
*  DETERMINE THE USERID OF THE CALLER
*
```

```
GETPSCB  DS    ØH
         MVC   EXTRACTA(EXTRACTL),EXTRACTM
         EXTRACT PSCBADDR,FIELDS=PSB,MF=(E,EXTRACTA)
         CLC   PSCBADDR,ZEROS
         BE    ENVERR                        NOT TSO - TOO BAD
         MVC   USERID,BLANKS                 SET TO BLANKS
         L     R2,PSCBADDR
         USING PSCB,R2
         IC    R3,PSCBUSRL                   GET LENG. OF USERID
         STC   R3,USERIDL                    SAVE IT.
         BCTR  R3,Ø                          PREPARE FOR EX
         B     SKIPUSER
USERMVC  MVC   USERID(*-*),PSCBUSER
SKIPUSER EX    R3,USERMVC                    EX ABOVE MVC
         DROP  R2
*
*    GET CURRENT GROUP AND TERMINAL PARMS
*
         L     R1,16
         L     R1,Ø(R1)                       POINT TO 4 WORD AREA
         L     R1,8(R1)                       GET ASCB
         L     R1,1Ø8(R1)                     GET ASXB
         L     R1,2ØØ(R1)                     GET ACEE
         LR    R1Ø,R1
         USING ACEE,R1Ø
         MVC   CURGRPL(L'ACEEGRP),ACEEGRP     GET CURRENT GROUP
         MVC   CURIDL(L'ACEEUSER),ACEEUSER    GET CURRENT ID
         MVC   CURTERM(L'ACEETRID),ACEETRID
         LA    R1,CURTERM
         ST    R1,CTERMAD
         DROP  R1Ø
*
*   GET CURRENT USER'S DEFAULT GROUP
*
         MVC   REXTRACT(REXMDLL),REXMDL
         RACXTRT  TYPE=EXTRACT,MF=(E,REXTRACT),ENTITY=USERID,SUBPOOL=Ø,*
               RELEASE=1.8.1
         LTR   R15,R15
         BNE   EXTRERR
EXTR1OK  DS    ØH
         USING RACXAREA,R1
         MVC   DFLTGRP(L'RACXDGRP),RACXDGRP
         LA    R14,L'DFLTGRP
         LA    R1,DFLTGRP
         BAL   R11,GETLEN
         STC   R15,DFLTGRPL
         DROP  R1
*********************************************************************
*                                                                   *
*    MAINLINE - PARSE THE COMMAND, AND DETERMINE IF A PASSWORD IS   *
```

```
*   NEEDED. IF SO, OBTAIN IT.                                          *
*                                                                      *
************************************************************************
*
*  PARSE THE COMMAND
*
PARSECMD DS    ØH
         LA    R1,MYPPL
         LINK  EP=IKJPARS,SF=(E,PARMLIST)
         LTR   R15,R15
         BNZ   PARSERR
PARSEOK  DS    ØH
         CLC   CBUFHDR,ZEROS
         BNE   NOSAVE
         L     R1Ø,CBUFPTR
         MVC   CBUFHDR(4),Ø(R1Ø)   SAVE ORIGINAL CBUF HEADER
NOSAVE   EQU   *
         L     R3,MYANS
         USING IKJPARMD,R3
         LA    R1Ø,UID
         TM    UID+6,X'8Ø'         WAS A USERID SPECIFIED?
         BZ    NOID                NO - USE THE CURRENT USERID.
         LH    R1,UID+4
         LTR   R1,R1               CHECK LENGTH
         BNP   NOID                ASSUME NOID  IF NOT > Ø
*  CHECK THAT SPECIFIED ID DOES NOT EXCEED LENGTH OF INTERNAL FIELD
*  USED TO STORE USER ID (CURRENTLY 8, THE RACF MAXIMUM)
*        LA    RØ,L'ID             MAXIMUM LENGTH OF A USER ID
*        CR    R1,RØ               SEE IF USER ID SPECIFIED EXCEEDS LEN
*        BH    MAXIDLEN            IF LEN EXCEEDS, ERROR MESSAGE & EXIT
         STC   R1,IDL              PUT IN LENGTH
         L     R14,UID             GET ADDR OF TEXT
         ST    R14,IDPTR           STORE ADDR OF USERID IN CBUF
         BCTR  R1,Ø                PREP FOR EX
         B     SKIP1               SKIP OVER MVC
IDMVC    MVC   ID(*-*),Ø(R14)      MOVE SPECIFIED USERID
SKIP1    EX    R1,IDMVC            EXECUTE THE ABOVE MVC
PWDCHK   DS    ØH
         IC    R2,IDL              IS ID SAME AS LOGON ID?
         EX    R2,IDCLC
         BE    NPWDRACI            YES - PWD NOT REQUIRED.
         EX    R2,IDCLC3           IS ID THE CHARACTERS 'DEFAULT'?
         BE    NOID                YES - USE THE CURRENT USERID
*
* PASSWORD IS REQUIRED
*
         TM    UID+14,X'8Ø'        WAS A PASSWORD SPECIFIED?
         BZ    REPARSE             NO  - REPARSE COMMAND.
         LH    R1,UID+12
         LTR   R1,R1               CHECK LENGTH
```

```
         BZ    REPARSE             REPARSE IF Ø LENGTH PWD.
         STC   R1,PWDL             PUT IN LENGTH
         L     R14,UID+8           GET ADDR OF TEXT
         BCTR  R1,Ø                PREP FOR EX
         B     SKIP2               SKIP OVER MVC
PWDMVC   MVC   PWD(*-*),Ø(R14)     MOVE SPECIFIED PASSWORD
SKIP2    EX    R1,PWDMVC           EXECUTE THE ABOVE MVC
         B     PWDRACI
*
* ADD A '/' TO THE END OF THE COMMAND AND REPARSE. THIS WILL CAUSE
* PARSE TO PROMPT FOR A PASSWORD.
*
REPARSE  DS  ØH
         XC    MYANS,MYANS          RESET ANSWER AREA.
         L     R1,IDPTR             GET ADDR OF USEID IN CBUF
         SR    R4,R4
         IC    R4,IDL
         LA    R1,Ø(R1,R4)          POINT TO END OF USERID
         MVC   Ø(2,R1),=C'/ '       ADD PASSWORD PROMPT IND.
         L     R9,CBUFPTR
         MVC   Ø(4,R9),CBUFHDR      RESTORE ORIGINAL CBUF HEADER
         LH    R8,Ø(R9)             GET CMD BUFFER LEN
         LA    R7,Ø(R8,R9)          POINT TO END OF CMD BUFFER
         CR    R7,R1                IS END PAST USERID?
         BH    PARSECMD             YES - DON'T INCREMENT LEN.
         LA    R8,1(R8)             BUMP CBUF LEN.
         STH   R8,Ø(R9)             AND SAVE IT.
         B     PARSECMD             RE-PARSE CMD.
*
*********************************************************************
*                                                                   *
*   ID AND PASSWORD HAVE BEEN OBTAINED AND ARE IN ID AND PWD        *
*   IF THE USERID IS NOT THE CURRENT ID, OBTAIN THE DEFAULT         *
*   GROUP OF THE DESIRED ID.                                        *
*   ISSUE RACINITS TO DELETE THE CURRENT USER PROFILE AND           *
*     INSTALL THE NEW ONE.                                          *
*                                                                   *
*********************************************************************
PWDRACI  DS    ØH
DORACD   DS    ØH
         MVC   NEWGRPL(L'ACEEGRP),CURGRP       PRIME NEWGRP
*
*  GET SPECIFIED USER'S DEFAULT GROUP
*
         IC    R2,IDL
         EX    R2,IDCLC                  IF ID SAME AS USERID, DON'T
         BE    GROUPOK                   BOTHER TO CHECK GROUP.
         MVC   REXTRACT(REXMDLL),REXMDL
         RACXTRT  TYPE=EXTRACT,MF=(E,REXTRACT),ENTITY=ID,SUBPOOL=Ø,    *
               RELEASE=1.8.1
```

```
         B     REXJMP(R15)
REXJMP   B     EXTROK
         B     EXTRERR
         B     USRNDEF1
         B     EXTRERR
         B     EXTRERR
         B     EXTRERR
         B     EXTRERR
EXTROK   DS    ØH
         LR    R1Ø,R1
         USING RACXAREA,R1Ø
         MVC   NEWGRP(L'RACXDGRP),RACXDGRP
         LA    R14,8
         LA    R1,NEWGRP
         BAL   R11,GETLEN
         STC   R15,NEWGRPL
         DROP  R1Ø
*  IF THIS IS DIFFERENT FROM THE CURRENT GROUP, NEED SPECIAL AUTH
         CLC   NEWGRP(8),CURGRP
         BE    GROUPOK
*        CALL  AUTHCHEK,(USEIDG),VL,MF=(E,PARMLIST)
*        LTR   R15,R15
*        BNZ   AUTHERR
*
GROUPOK  DS    ØH
         MVC   RACINIT(RACIML),RACIM
         RACINIT MF=(E,RACINIT),ENVIR=DELETE,RELEASE=1.8.1
         LTR   R15,R15
         BNZ   RACDERR
DORACI   DS    ØH
         MVC   RACINIT(RACIML),RACIM
         RACINIT MF=(E,RACINIT),ENVIR=CREATE,USERID=IDL,PASSWRD=PWDL,  +
               GROUP=NEWGRPL,PASSCHK=YES,RELEASE=1.8.1
         B     RACIJMP(R15)
RACIJMP  DS    ØH
         B     EXIT
         B     USERNDEF    /* SHOULD NOT HAPPEN  */
         B     PWDNAUTH
         B     PWDEXPR
         B     PWDINVLD
         B     GRPNDEF
         B     INSTFAIL
         B     USRREVKD
         B     RACFNACT
         B     GRPACCRV
         B     NOOID
         B     OIDINVL
         B     TRMNAUTH
         B     APLNAUTH
USERNDEF DS    ØH
```

```
         LA    RØ,L'MSGØØ
         LA    R1,MSGØØ
         BAL   R11,PUTLINE
         B     RESTRID
PWDNAUTH DS    ØH
         LA    RØ,L'MSGØ1
         LA    R1,MSGØ1
         BAL   R11,PUTLINE
         B     RESTRID
PWDEXPR  DS    ØH
         LA    RØ,L'MSGØ2
         LA    R1,MSGØ2
         BAL   R11,PUTLINE
         B     RESTRID
PWDINVLD DS    ØH
         LA    RØ,L'MSGØ3
         LA    R1,MSGØ3
         BAL   R11,PUTLINE
         B     RESTRID
GRPNDEF  DS    ØH
         LA    RØ,L'MSGØ4
         LA    R1,MSGØ4
         BAL   R11,PUTLINE
         B     RESTRID
INSTFAIL DS    ØH
         LA    RØ,L'MSGØ5
         LA    R1,MSGØ5
         BAL   R11,PUTLINE
         B     RESTRID
USRREVKD DS    ØH
         LA    RØ,L'MSGØ6
         LA    R1,MSGØ6
         BAL   R11,PUTLINE
         B     RESTRID
RACFNACT DS    ØH
         LA    RØ,L'MSGØ7
         LA    R1,MSGØ7
         BAL   R11,PUTLINE
         B     RESTRID
GRPACCRV DS    ØH
         LA    RØ,L'MSGØ8
         LA    R1,MSGØ8
         BAL   R11,PUTLINE
         B     RESTRID
NOOID    DS    ØH
         LA    RØ,L'MSGØ9
         LA    R1,MSGØ9
         BAL   R11,PUTLINE
         B     RESTRID
OIDINVL  DS    ØH
```

```
         LA    RØ,L'MSG1Ø
         LA    R1,MSG1Ø
         BAL   R11,PUTLINE
         B     RESTRID
TRMNAUTH DS    ØH
         LA    RØ,L'MSG11
         LA    R1,MSG11
         BAL   R11,PUTLINE
         B     RESTRID
APLNAUTH DS    ØH
         LA    RØ,L'MSG12
         LA    R1,MSG12
         BAL   R11,PUTLINE
         B     RESTRID
*********************************************************************
*                                                                   *
*   RESTORE ORIGINAL USER PROFILE.                                  *
*                                                                   *
*********************************************************************
NPWDRACI DS    ØH
NOID     DS    ØH
         IC    R2,USERIDL
         EX    R2,IDCLC2                ID SAME AS CURRENT ACEE
         BNE   DOIT                     NO, DO RACINIT
         LA    RØ,L'MSG13
         LA    R1,MSG13
         BAL   R11,PUTLINE
         LA    RØ,L'MSG24
         LA    R1,MSG24
         BAL   R11,PUTLINE
         B     EXIT
DOIT     MVC   RACINIT(RACIML),RACIM
         RACINIT MF=(E,RACINIT),ENVIR=DELETE,RELEASE=1.8.1
         LTR   R15,R15
         BNZ   RACDERR
RESTRID  DS    ØH
*        WTO   'RESRID'
         LA    RØ,L'MSG14
         LA    R1,MSG14
         BAL   R11,PUTLINE
         MVC   RACINIT(RACIML),RACIM
         RACINIT MF=(E,RACINIT),ENVIR=CREATE,USERID=USERIDL,          +
               GROUP=DFLTGRPL,PASSCHK=NO,RELEASE=1.8.1
         LTR   R15,R15
         BNZ   FATALERR
         B     EXIT
*********************************************************************
*                                                                   *
*   ALL DONE                                                        *
*                                                                   *
```

```
*************************************************************************
EXIT     DS    ØH
*        WTO   'IKJRLSA'
         IKJRLSA  MYANS
*        WTO   'FREEMAIN'
         L     R13,4(R13)
         L     RØ,GETPARM
         FREEMAIN R,LV=(Ø),A=(R6)
         RETURN (14,12),RC=Ø
*************************************************************************
*                                                                       *
*   HANDLE ERROR CONDITIONS                                             *
*                                                                       *
*************************************************************************
USRNDEF1 DS    ØH
         LA    RØ,L'MSG16
         LA    R1,MSG16
         BAL   R11,PUTLINE
         LA    RØ,L'MSG24
         LA    R1,MSG24
         BAL   R11,PUTLINE
         B     EXIT
AUTHERR  DS    ØH
         LA    RØ,L'MSG17
         LA    R1,MSG17
         BAL   R11,PUTLINE
         B     EXIT
ENVERR   DS    ØH
         LA    RØ,L'MSG18
         LA    R1,MSG18
         BAL   R11,PUTLINE
         ABEND 999
         B     EXIT
EXTRERR  DS    ØH
         LA    RØ,L'MSG19
         LA    R1,MSG19
         BAL   R11,PUTLINE
         LA    RØ,L'MSG23
         LA    R1,MSG23
         BAL   R11,PUTLINE
         B     EXIT
RACDERR  DS    ØH
         LA    RØ,L'MSG2Ø
         LA    R1,MSG2Ø
         BAL   R11,PUTLINE
         LA    RØ,L'MSG23
         LA    R1,MSG23
         BAL   R11,PUTLINE
         B     EXIT
FATALERR DS    ØH
```

```
         LA    RØ,L'MSG21
         LA    R1,MSG21
         BAL   R11,PUTLINE
         LA    RØ,L'MSG23
         LA    R1,MSG23
         BAL   R11,PUTLINE
         B     EXIT
PARSERR  DS    ØH
         LA    RØ,L'MSG15
         LA    R1,MSG15
         BAL   R11,PUTLINE
         B     EXIT
MAXIDLEN DS    ØH
         LA    RØ,L'MSG25
         LA    R1,MSG25
         BAL   R11,PUTLINE
         B     EXIT
*************************************************************************
*                                                                       *
*   SUBROUTINES                                                         *
*                                                                       *
*************************************************************************
*
*  GET LENGTH OF A STRING
*
*  R1  POINTS TO STRING
*  R14 CONTAINS MAX LENGTH
*  R15 IS SET TO LENGTH OF STRING ON RETURN
*
GETLEN   DS    ØH
         SR    R15,R15
GLOOP    DS    ØH
         CLI   Ø(R1),X'ØØ'
         BER   R11
         CLI   Ø(R1),C' '
         BER   R11
         LA    R15,1(R15)
         LA    R1,1(R1)
         BCT   R14,GLOOP
         BR    R11
*************************************************************
*                                                           *
*        PUTLINE ROUTINE                                    *
*   RØ =  MSG LENGTH                                        *
*   R1 -> MSG                                               *
*                                                           *
*************************************************************
         SPACE
PUTLINE  STM   R14,R1,PUTLINS
         XC    MYOLD(8),MYOLD
```

```
         XC    MYSEG1(4),MYSEG1
         MVC   MYPTPB(12),PTPBMDL
         LA    R14,1               NO. OF MESSAGE SEGMENTS
         ST    R14,MYOLD
         LA    R14,MYSEG1          POINT TO 1ST SEGMENT
         ST    R14,MYOLD+4
         LR    R14,RØ              LENGTH IN RØ
         LA    R14,4(,R14)         ADD 4
         STH   R14,MYSEG1
         LR    R14,RØ
         BCTR  R14,Ø
         LA    R15,MYSEG1+4
         B     PSKIP
PUTLMVC  MVC   Ø(*-*,R15),Ø(R1)      MOVE MESSAGE IN
PSKIP    EX    R14,PUTLMVC
         SPACE
         PUTLINE PARM=MYPTPB,OUTPUT=(MYSEG1,,,DATA),MF=(E,MYIOPL)
         SPACE
         LM    R14,R1,PUTLINS
         BR    R11
         SPACE
*********************************************************************
*                                                                   *
*   CONSTANTS                                                       *
*                                                                   *
*********************************************************************
         DS    ØF
GETPARM  DC    AL1(WORKSP),AL3(WORKLEN)
BLANKS   DC    CL8' '
ZEROS    DC    XL8'ØØ'
USEIDG   DC    C'USEIDG'              APPL NAME FOR CROSS-GROUP USEID
MAXID    DC    F'8'
MINID    DC    F'Ø'
MAXPWD   DC    F'8'
MINPWD   DC    F'Ø'
MAXLEN   DC    F'17'
IDCLC    CLC   IDL(*-*),USERIDL         SPECIFIED ID SAME AS USERID?
IDCLC2   CLC   CURIDL(*-*),USERIDL      USERID SAME AS ACEE ID?
IDCLC3   CLC   IDL(*-*),DEFAULTL        USERID IS 'DEFAULT'?
RACIM    RACINIT RELEASE=1.8.1,MF=L
RACIML   EQU   *-RACIM
REXMDL   RACXTRT TYPE=EXTRACT,RELEASE=1.8.1,MF=L
REXMDLL  EQU   *-REXMDL
EXTRACTM EXTRACT ,FIELDS=PSB,MF=L
EXTRACTL EQU   *-EXTRACTM
PTPBMDL  PUTLINE OUTPUT=(1,TERM,SINGLE,DATA),                          X
               TERMPUT=(EDIT,WAIT,NOHOLD,NOBREAK),MF=L
PGPBMDL  PUTGET MF=L
PGPBL    EQU   *-PGPBMDL
DEFAULTL DC    AL1(7)
```

```
DEFAULT  DC    CL8'DEFAULT'
*
*   MESSAGES
*
MSGØØ    DC    C'SPECIFIED USER ID DOES NOT EXIST'
MSGØ1    DC    C'PASSWORD IS NOT AUTHORIZED'
MSGØ2    DC    C'PASSWORD HAS EXPIRED'
MSGØ3    DC    C'INVALID PASSWORD'
MSGØ4    DC    C'USER ID IS NOT DEFINED TO CURRENT GROUP'
MSGØ5    DC    C'RACINIT FAILED BY INSTALLATION EXIT'
MSGØ6    DC    C'SPECIFIED USER ID HAS BEEN REVOKED'
MSGØ7    DC    C'RACF IS NOT ACTIVE'
MSGØ8    DC    C'GROUP ACCESS HAS BEEN REVOKED'
MSGØ9    DC    C'OPERATOR ID CARD IS REQUIRED'
MSG1Ø    DC    C'INVALID OPERATOR ID CARD'
MSG11    DC    C'SPECIFIED USER ID NOT AUTHORIZED TO TERMINAL'
MSG12    DC    C'SPECIFIED USER ID NOT AUTHORIZED TO APPLICATION'
MSG13    DC    C'LOGON USER ID PROFILE ALREADY IN EFFECT'
MSG14    DC    C'REVERTING TO LOGON USER ID PROFILE'
MSG15    DC    C'UNABLE TO PARSE COMMAND'
MSG16    DC    C'SPECIFIED USER ID IS NOT DEFINED'
MSG17    DC    C'SPECIFIED USER ID IS NOT IN YOUR GROUP'
MSG18    DC    C'USEID MUST BE RUN UNDER TSO'
MSG19    DC    C'ERROR ISSUING RACXTRT MACRO'
MSG2Ø    DC    C'ERROR ISSUING RACINIT DELETE'
MSG21    DC    C'ERROR RESTORING LOGON USER ID PROFILE - PLEASE RE-LOGO+
               N'
MSG23    DC    C'PLEASE CONTACT HELP DESK'
MSG24    DC    C'NO ACTION TAKEN'
MSG25    DC    C'SPECIFIED USER ID EXCEEDS MAXIMUM LENGTH ALLOWED'
         LTORG
*************************************************************
*                                                           *
*        PARSE PCL                                          *
*                                                           *
*************************************************************
         SPACE
         PRINT ON,GEN
         CNOP Ø,4
MYPCL    IKJPARM
UID      IKJPOSIT USERID,DEFAULT='DEFAULT',                          +
               HELP='USER ID OR USER ID/PASSWORD TO CHANGE PROFILES, OR+
                BLANK TO REVERT TO YOUR LOGON PROFILE'
         IKJENDP
*************************************************************************
*                                                                       *
*   DSECTS                                                              *
*                                                                       *
*************************************************************************
*
```

```
*        DYNAMIC SAVE/WORK AREA
*
WORKAREA DSECT ,                    DYNAMIC WORK AREA
SAVEAREA DS    18F
USERAREA DS    ØF
USERIDL  DS    AL1
USERID   DS    CL8
IDL      DS    AL1
ID       DS    CL8
PWDL     DS    AL1
PWD      DS    CL8
CURIDL   DS    AL1
CURID    DS    CL8
CURGRPL  DS    AL1
CURGRP   DS    CL8
DFLTGRPL DS    AL1
DFLTGRP  DS    CL8
NEWGRPL  DS    AL1
NEWGRP   DS    CL8
CURTERM  DS    CL8
         DS    ØH
CBUFHDR  DS    CL4
CBUFPTR  DS    CL4
CLEARL   EQU   *-USERAREA
         DS    ØF
CTERMAD  DS    AL4
PSCBADDR DS    AL4
CPPLPTR  DS    AL4
IDPTR    DS    AL4
RACINIT  RACINIT RELEASE=1.8.1,MF=L
REXTRACT RACXTRT TYPE=EXTRACT,RELEASE=1.8.1,MF=L
EXTRACTA DS    CL(EXTRACTL)
PUTLECB  DS    AL4
PGECB    DS    AL4
PARSEECB DS    AL4
MYPPL    DS    7F
MYANS    DS    F
MYUWA    DS    F
MYECB    DS    F                    USED BY PUTLINE ROUTINE
MYIOPL   DS    4F                   USED BY PUTLINE ROUTINE
MYPTPB   DS    3F                   USED BY PUTLINE ROUTINE
MYOLD    DS    3F                   USED BY PUTLINE ROUTINE
MYPGPB   DS    CL(PGPBL)
MYSEG1   DS    2H,CL1ØØ             USED BY PUTLINE ROUTINE
PUTLINS  DS    4F                   USED BY PUTLINE ROUTINE
PARMLIST DS    2ØF
WORKLEN  EQU   *-WORKAREA
WORKSP   EQU   ØØ
         SPACE 2
*
```

```
*  FOLLOWING DSECT MAPS AREA RETURNED BY RACXTRT
*
RACXAREA DSECT
RACXSUBP DS    AL1
RACXLEN  DS    XL3
RACXPWDO DS    AL2
RACXRSVD DS    CL18
RACXDUID DS    CL8
RACXDGRP DS    CL8
         END
AUTHCHEK
         TITLE  'AUTHCHEK'
*************************************************************************
*                                                                       *
*                 AUTHCHEK                                              *
*   PURPOSE:                                                            *
*                                                                       *
*     DO RACF AUTHORIZATION CHECKING FOR APPL RESOURCES                 *
*                                                                       *
*   AUTHOR/ORIGIN:                                                      *
*     T.HOWARD,  ALBERTA PUBLIC WORKS, SUPPLY AND SERVICES              *
*                                                                       *
*   CALLING SEQUENCE (ASSEMBLER):                                       *
*                                                                       *
*           CALL AUTHCHEK,(APPLNAME)                                    *
*                                                                       *
*   INPUT:                                                              *
*           APPLNAME - CHAR(8)    NAME OF APPL RESOURCE, LEFT           *
*                                 JUSTIFIED, BLANK PADDED               *
*                                                                       *
*   OUTPUT:                                                             *
*                                                                       *
*           R15 =    Ø            ACCESS GRANTED                       *
*                     4            ACCESS REFUSED                       *
*                     8            INVALID INPUT PARAMETERS             *
*                    12            APPL RESOURCE NOT DEFINED            *
*                                                                       *
*   FUNCTION:                                                           *
*                                                                       *
*     JUST DOES A RACHECK                                               *
*                                                                       *
*   INSTALLATION CONSIDERATIONS:                                        *
*                                                                       *
*   ENVIROMENT:                                                         *
*      CALLER MUST BE AMODE 24 TO CALL THIS ROUTINE                     *
*                                                                       *
*   VERSION: 1.Ø                                                        *
*                                                                       *
*   CHANGES: Ø1/14/88 - REMOVE DEPENDENCIES ON IN-HOUSE MACROS          *
*            Ø6/21/9Ø - ADD RELEASE PARAMETER ON RACF MACROS            *
```

```
*            27/Ø1/92 - CHANGED RELEASE TO 1.9 ON RACF MACROS         *
*            17/Ø4/93 - CHANGED RELEASE BACK TO 1.8.1                 *
*            3Ø/Ø7/96 - CHANGED RELEASE TO 1.8.1                      *
*                                                                     *
***********************************************************************
         YREGS
AUTHCHEK CSECT
         SAVE  (14,12),,MOD_Ø1/13/88_DIS_99/99/99_ASM_&SYSDATE
         USING AUTHCHEK,R12            SET UP BASE ADDRESSABILITY
         LR    R12,R15                 LOAD BASE REG WITH ENTRY POINT
***********************************************************************
*                                                                     *
*   GETMAIN - GET STORAGE FOR A WORKAREA AND CHAIN THE SAVE AREAS     *
*   TOGETHER.                                                         *
*                                                                     *
***********************************************************************
         LR    R2,R1
         L     RØ,GETPARM
         GETMAIN R,LV=(Ø)
         LR    R6,R1
         USING WORKAREA,R6
         LR    R1,R13
         LA    R13,SAVEAREA
         ST    R13,8(R1)
         ST    R1,4(R13)
         LR    R1,R2
***********************************************************************
*                                                                     *
*   GETPARM - GET ANY PARAMETER WHICH WAS PASSED TO THE PROGRAM       *
*                                                                     *
***********************************************************************
         LTR   R1,R1
         BZ    EXITRC8
         L     R1Ø,Ø(R1)          CLEAR R4 FOR PARM LENGTH
         MVC   RACFPARM(RACFPRML),RACFBGN
***********************************************************************
*                                                                     *
*   MAINLINE                                                          *
*                                                                     *
***********************************************************************
         RACHECK ENTITY=((R1Ø)),RELEASE=1.8.1,MF=(E,RACFPARM)
         B     JMPTABL(R15)
JMPTABL  B     EXITRCØ                    USER IS AUTHORIZED
         B     EXITRC12                   RESOURCE IS NOT DEFINED
         B     EXITRC4                    USER IS NOT AUTHORIZED
         B     EXITRC8                    SNO
***********************************************************************
*                                                                     *
*   ALL DONE                                                          *
*                                                                     *
```

```
*********************************************************************
EXITRC12 DS    ØH
         LA    R15,12
         B     EXIT
EXITRC8  DS    ØH
         LA    R15,8
         B     EXIT
EXITRC4  DS    ØH
         LA    R15,4
         B     EXIT
EXITRCØ  DS    ØH
         LA    R15,Ø
         B     EXIT
EXIT     DS    ØH
         LR    R2,R15
         L     R13,4(R13)
         L     RØ,GETPARM
         FREEMAIN R,LV=(Ø),A=(R6)
         LR    R15,R2
         RETURN (14,12),RC=(15)
*********************************************************************
*                                                                   *
*   CONSTANTS                                                       *
*                                                                   *
*********************************************************************
         EJECT
CLASS    DC    X'Ø7',CL7'APPL   '
RACFBGN  EQU   *
         RACHECK CLASS=CLASS,ATTR=READ,RELEASE=1.8.1,MF=L
RACFEND  EQU   *
RACFPRML EQU   RACFEND-RACFBGN
GETPARM  DC    AL1(WORKSP),AL3(WORKLEN)
         LTORG
*********************************************************************
*                                                                   *
*   DSECTS                                                          *
*                                                                   *
*********************************************************************
*
*        DYNAMIC SAVE/WORK AREA
*
WORKAREA DSECT                     DYNAMIC WORK AREA
SAVEAREA DS    18F
RACFPARM DS    CL(RACFPRML)
WORKLEN  EQU   *-WORKAREA
WORKSP   EQU   ØØ
         END
```

NOTES

Two important areas of USEID are commented out because I haven't had a chance to fully test them. They are as follows:

- The group checking section, including the call to AUTHCHEK, was removed years ago at the site's request. The group checking code is commented out just before the GROUPOK label. Obviously, if you leave group checking commented out, you won't need to assemble or include AUTHCHEK in the load module.
- Code to address a new bug that surfaced recently and was tracked down to a missing maximum length check on the user-supplied ID. The ID length checking code is commented out after the NOSAVE label.

ASSEMBLY

You could just separately assemble USEID and AUTHCHEK, and linkedit them together into a load module and store it in SYS1.CMDLIB(USEID). The only other step you would need to complete would be to add USEID to SYS1.PARMLIB(IKJTSO00), as is required for all APF-authorized TSO commands:

```
AUTHCMD NAMES(               /*   AUTHORIZED COMMANDS    */ +
   LISTB    LISTBC           /*                          */ +
     {rest of APF-authorized TSO commands}
   USEID                     /* CHANGE RACF USERID       */ +
                )
```

Alternatively, you can fool SMP/E into thinking that USEID is an update to a null module – in the example below, SYS1.AUSERLNK(USEID) doesn't exist – and store the load module in a separate library, SYS1.USERLNK. Here is the JCL to do that, assuming an SMP/E procedure named SMPZOS12 and a target zone of MVST100:

```
//USEIDMOD JOB NOCPØØØ,'SYSMOD MODØØ3',MSGCLASS=H,CLASS=A,
// MSGLEVEL=(1,1),REGION=7M,TIME=(1)
/*JOBPARM LINES=999,CARDS=999999
//*
//* PURPOSE: USEID COMMAND
//* CONTACT: TECHNICAL SEVICES
```

```
//* CREATED: 26 JUL 96
//* EXPIRES: WHEN COMMAND NO LONGER REQUIRED/SUPPORTED
//* REMARKS.
//*
//* CHANGE ACTIVITY
//* DDMMMYY INIT CHANGE DESCRIPTION - MOST RECENT CHANGE FIRST !!!!!
//* ------- ---- ------------------------------------------------
//* Ø3JUNØ3 JONP FIX >8 ID PROBLEM AND TIDY UP SOURCE
//*
//** SOURCE ->  ************** 'TSGO.ZOS12.USERMODS.CNTL(MODØØ3  )'
//* ASSEMBLE
//*
//ASMUSEID EXEC PGM=ASMA9Ø,PARM='DECK,NOOBJECT'
//SYSLIB   DD   DISP=SHR,DSN=SYS1.MACLIB
//         DD   DISP=SHR,DSN=SYS1.MODGEN
//SYSUT1   DD   UNIT=VIO,SPACE=(CYL,(1,1)),DSN=&SYSUT1
//SYSPRINT DD   SYSOUT=*
//SYSPUNCH DD   DSN=&SMPPTFI1,UNIT=VIO,DCB=BLKSIZE=312Ø,
//  SPACE=(CYL,(1,1)),DISP=(MOD,PASS)
//SYSIN    DD   *
  {insert USEID source code here}
/*
//***
//AUTHCHEK EXEC PGM=ASMA9Ø,PARM='DECK,NOOBJECT'
//SYSLIB   DD   DSN=SYS1.MACLIB,DISP=SHR
//         DD   DSN=SYS1.MODGEN,DISP=SHR
//SYSUT1   DD   UNIT=VIO,SPACE=(CYL,(1,1)),DSN=&SYSUT1
//SYSPRINT DD   SYSOUT=*
//SYSPUNCH DD   DSN=&SMPPTFI2,UNIT=VIO,DCB=BLKSIZE=312Ø,
//  SPACE=(CYL,(1,1)),DISP=(MOD,PASS)
//SYSIN    DD   *
  {insert AUTHCHEK source code here}
/*
//***
//*
//* INSTALL WITH SMP/E                                                ***
//*
//TMODØØ3 EXEC SMPZOS12,COND=(Ø,NE)
//SMPPTFIN DD   DATA,DLM='##'
++USERMOD (TMODØØ3).
++VER (ZØ38) FMID(TMVSØØØ) .
++JCLIN .
//JOB      JOB  1,'TMODØØ3 USERMOD'
//STEP1    EXEC PGM=IEWL,PARM='NCAL,LET,LIST,XREF,AC=1'
//AUSERLNK DD   DSN=SYS1.AUSERLNK,DISP=SHR
//SYSLMOD  DD   DSN=SYS1.USERLNK,DISP=SHR
//SYSLIN   DD   *
  INCLUDE AUSERLNK(USEID)
  NAME    USEID(R)
/*
```

```
++MOD (USEID) LMOD(USEID) DISTLIB(AUSERLNK) .
##
//         DD   DSN=&SMPPTFI1,DISP=(OLD,DELETE,DELETE)
//         DD   DSN=&SMPPTFI2,DISP=(OLD,DELETE,DELETE)
//SYSIN    DD   *
  SET BDY(GLOBAL).
    REJECT SELECT(TMOD003) BYPASS(APPCHK).
    RESETRC.
    RECEIVE SELECT(TMOD003).
  SET BDY(MVST100).
    APPLY SELECT(TMOD003) RETRY(NO) REDO.
/*
//
```

In summary, I use USEID on a daily basis and I would recommend it to anyone with access to more than one RACF user ID.

Jon E Pearkins (Canada)

Consul/zAlert monitors z/OS, RACF, and Unix System Services, detecting unwanted log-ons and log-on attempts, and identifying internal security policy violations, dangerous configuration changes, and primary system resources at risk.

It immediately notifies administrators via mobile devices, e-mail, or the Consul/eAudit security event management console and takes actions and countermeasures such as revoking a user ID or shutting down an application. Over time, incidents are correlated to show particular patterns.

zAlert is part of zSecure Pro Suite and is integrated with tools such as Consul's Enterprise Audit (CeA), Tivoli, NetView and HP OpenView.

URL: http://www.consul.com/index.php3?cid=568

* * *

Candle's new PathWAI Secure for WebSphere MQ protects MQ data with full PKI support, Online Certificate Status Protocol (OCSP) support, message-embedded digital certificates, and auditability that meets US federal standards governing proof of unaltered data during transmission and archival. It runs on z/OS, Windows NT/2000/XP, Sun Solaris, AIX, HP-UX, and OS/400.

URL: http://www.candle.com/www1/cnd/portal/CNDportal_Channel_Master/0,2938,2683_2885,00.html

* * *

IBM has made a number of announcements, including the following:

- Immediate availability of the Enterprise Identity Mapping (EIM) infrastructure in z/OS 1.4 via APAR OW57137. By using the LDAP database as a central repository of user mapping information, user IDs are maintained across multiple platforms.

- z/VM 4.4 has secured the TCP/IP stack by logging all TCP/IP administrative commands that attempt to alter the active IP or Control Program (CP) configuration.

- The new eserver z990 has three cryptographic options: the PCI Cryptographic Accelerator (PCICA) feature, the new PCIX Cryptographic Coprocessor (PCIXCC) feature, and the CP Assist for Cryptographic Function (CPACF). PCIXCC replaces the z900's PCI Cryptographic Coprocessor (PCICC) and the CMOS Cryptographic Coprocessor Facility. CPACF's Message Security Assist Architecture provides DES and TDES data encryption/decryption and SHA-1 hashing.

- DB2 OLAP Server 8.1 allows only the RACF user ID that started the Integration Server to shut it down, allows sharing of a single group metadata catalogue by assigning users to a RACF group, and allows users to change their RACF password through the Application Manager or spreadsheet interface.

URL: http://www.ibmlink.ibm.com

* * *