20

RACF update

May 2000

In this issue

RACF Update

Published by

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

North American office

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

Contributions

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

RACF Update on-line

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

Editor

Fiona Hewitt

Subscriptions and back-issues

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

Dynamic access

The programs presented here are designed to help systems programmers who may occasionally need the RACF SPECIAL or OPERATIONS attribute added, in order to perform certain functions. They dynamically give the caller the SPECIAL,OPERATIONS attribute for the life of the session only. Note that although the access is granted by the programs rather than the RACF administrator, the administrator should maintain the programs, by controlling who can access them. Note also that all commands issued to the sysprog invoking the program will be reported in SMF for later auditing.

THE PROCESS

The first program is a user SVC program designed to place the caller in an Authorized state and check which user(s) are allowed to use the program. The second program is a command processor which the user runs to be granted the access given by the SVC program.

DETAIL

You should place the IGC0023C program in SYS1.PARMLIB (IEALPAxx) and code as follows:

```
INCLUDE LIBRARY(dsn)
  MODULES(IGC0023C)
```

(Dsn – the location of IGC0023C module; the dsn must be APF authorized.)

The RACACC program must be placed in an APF authorized dataset. This dataset should be concatenated in your log-on proc via //STEPLIB DD.

INVOKING THE PROGRAM

Once the programs have been assembled and linked, an IPL is required for the SVC module. To invoke, issue:

```
TSO RACACC
```

IGC0023C SOURCE

```
IGCØØ23C TITLE 'AUTHORIZATION SVC'
*-----------------------------------------------------------------------
* FUNCTION -
*
*  THIS USER SVC WILL PUT THE USER IN AN AUTHORIZED STATE.
* PLACE IN IEALPAXX MEMBER OF SYS1.PARMLIB
*
* PROCESS: ONLY SELECTED USERS CAN INVOKE THIS PROGRAM.
*
*  LINKED AS IGCØØ23C RENT,REFR
*             THIS MODULE TO BE MLPA
*
*  IF RØ CONTAINS Ø AUTH IS GRANTED, IF NOT AUTH IS RESET
*
*-----------------------------------------------------------------------
         EJECT
IGCØØ233 CSECT
         LR    12,6                LOAD ENTRY POINT ADDR
         USING IGCØØ233,12         ADDRESSABILITY
         L     2,X'6C'(7)          POINT AT ASXB
         CLC   X'CØ'(3,2),USER1
         BE    OK
         CLC   X'CØ'(4,2),USER2
         BE    OK
         CLC   X'CØ'(4,2),USER3
         BE    OK
         B     RETURN                  NOT AUTH USER
*
OK       L     2,18Ø(4)            POINT R2 TO JSCB
         BCT   Ø,AUTHOFF           IF,AFTER BCT, RØ NOT = Ø ,TURN OFF
AUTHON   EQU   *
         OI    236(2),X'Ø1'        TURN ON AUTHORIZATION BIT
         B     RETURN              EXIT IN AUTHORIZED STATE
AUTHOFF  EQU   *
         NI    236(2),X'FE'        TURN OFF AUTHORIZATION BIT
RETURN   EQU   *
         BR    14                  RETURN TO SPFCOPY
*list of users allowed to run the program

USER1    DC    C'O6Ø'              ALLOW THIS USER
USER3    DC    C'O56'              ALLOW THIS USER
USER2    DC    C'CICS'             ALLOW THIS USER
         END   IGCØØ233
```

RACACC SOURCE

```
RACACC   TITLE 'ACEE'
         REPRO
    NAME RACACC(R)
```

```
         REGS
RACACC   CSECT
         SAVE  (14,12),,RACACC_&SYSDATE._&SYSTIME
         LR    R12,R15                  SET ADDRESSABILITY
         USING RACACC,R12
         LA    R6,SAVEAREA              POINT AT MY SA
         ST    R6,8(R13)                STORE IN CALLERS SA
         ST    R13,SAVEAREA+4           STORE CALLERS IN MY SA
         LR    R13,R6                   LOAD MY SA ADDR
*
         LA     Ø,1                     REQUEST AUTH
         SVC    233
         MODESET KEY=ZERO
         L      5,X'224'                 POINTER TO ASCB
         L      5,X'6C'(5)               POINTER TO ASXB
         L      5,X'C8'(5)               POINTER TO ACEE
         NI     X'26'(5),X'ØØ'           SPEC ATTR
         OI     X'26'(5),X'B1'           OPER ATTR
*        OI     X'27'(5),X'8Ø'           ALTER ACCESS
         NI     X'27'(5),X'ØØ'           TURN OFF NO ACCESS
         OI     X'27'(5),X'8Ø'           ALTER ACCESS
         MODESET KEY=NZERO
         LA    RØ,Ø                     AUTH OFF
         SVC   233                      CALL THIS SVC FOR AUTH
         B     RETURN
*
         OPEN  (SYSPRINT,(OUTPUT))
PRINT    PUT   SYSPRINT,PRINTLNE
         MVI   CC,X'4Ø'                 CLEAR
         MVC   PRLINE,CC                PRINT
         MVI   CC,X'Ø9'                 LINE
         BR    R1Ø
*
RETURN   DS    ØH
         CLOSE (SYSPRINT)
         L     13,SAVEAREA+4
         LH    R15,RCODE                LOAD RETURN CODE
         RETURN (14,12),RC=(15)
SAVEAREA DS    18F
RCODE    DC    H'Ø'
PRINTLNE DS    ØCL133
CC       DS    CL1
PRLINE   DS    CL132
SYSPRINT DCB   DSORG=PS,BLKSIZE=133,DDNAME=SYSPRINT,                   &
               MACRF=PM,RECFM=FM,LRECL=133
         LTORG
         TITLE 'ACEE'
         IHAACEE
         END   RACACC
```

Salah Balboul
Senior Systems Programmer (USA)

Authentication using the RACF PassTicket

The RACF PassTicket is an alternative to the RACF password that enables remote clients on workstations to get authenticated to the host system. This means that the user can gain access to the host without sending the RACF password in clear across a network which, with the advent of the Internet, is now ‘full of villains’ (in IBM speak). Note that, while the PassTicket is excellent for telnet and traditional 3270 applications, it faces stiff competition from the digital certificate targeted for Web applications and now supported by recent versions of OS/390.

The RACF PassTicket doesn’t replace the regular RACF password, which remains usable. Rather, it’s a cryptographically-generated, short-lifespan password substitute. It’s more secure than passwords because it’s valid for a period of plus or minus 10 minutes (as measured on the mainframe’s GMT clock) and it can’t be reused. So, even if some ‘villain’ did manage to capture it by eavesdropping or hacking the network or the routers, it would be useless to him.

The PassTicket is always an alphanumeric, eight-character string – say, for example, 5PX9A4UZ. At first, RACF can’t tell that a PassTicket is being presented rather than a regular password. But when it authenticates a password field and determines that it’s not the password for the userid, RACF performs a second authentication step to determine whether the password field is a valid PassTicket. This is why you may see two RACF messages in the SYSLOG if the PassTicket is invalid.

USAGES

The RACF PassTicket has a number of uses, for instance:

- It means that you can connect to your site from the outside world (ftp, telnet) without the security hazard of transmitting the password in clear.
- It enables you to avoid coding passwords in clear in batch jobs or input data (FTP commands in a batch job).

- With CICS or IMS, you may now submit a job under a user's authority without prompting the user for his password.
- You can submit jobs via NJE to other nodes; it is then recommended that your MVS or OS/390 systems should run with MVS GMT time = real GMT time.
- You can 'lend' a userid + a PassTicket value to somebody, for test or maintenance, for a short period of time. Note that once that user logs off, he cannot reconnect.

The PassTicket is not recommended for Web applications. This is because, when you access protected Web pages, the password is transmitted in the HTTP header at each interaction. Because the PassTicket is not reusable, a different PassTicket would be generated each time, which would be extremely inconvenient.

PASSTICKET GENERATION PROCESS

Unlike the standard password, the PassTicket applies to only one application. It must be generated locally. The algorithm that generates the PassTicket is a function of :

- The userid of the client.
- The application id (CICS applid, IMS id, etc).
- A secured sign-on application key, known to both sides (RACF and the local generator).
- A time and date stamp.

In order to generate a PassTicket, you need to define a shared key or secret between the systems on which the client and the security server are running. This key depends on the application and optionally (and hopefully) on the user. The application-id is the same as the profile you would put in the RACF APPL class for protecting the access to the application. For TSO, it is TSO + SMF id.

Applications that don't specify an APPL parameter get a default APPL, for PassTicket purposes only, of 'MVS' followed by the system's SMF id. This works also for batch jobs.

On MVS, you can use the generation routine that RACF provides. An example is shown below.

```
* PASSTICKET GENERATION EXAMPLE
         SETAMOD 31                 Personal macro to be in 31-bit amode
         MODESET KEY=ZERO,MODE=SUP  Become authorized
         L     R15,16               Point to CVT
         USING CVT,R15              CVT addressability
         L     R15,CVTRAC           Point to RACF CVT
         USING RCVT,R15             RCVT addressability
         L     R15,RCVTPTGN         Point to PassTicket routine
         CALL  (15),(USER,APPLIC)   Call PassTicket generator routine
         ST    R15,RC               Let's save the return code
         STM   RØ,R1,PTKTVAL        Let's store the PassTicket value
         MODESET KEY=NZERO,MODE=PROBLet's drop authorization
         SETAMOD 24                 Personal macro to be in 24-bit amode
         OC    RC,RC                Return code must be zero
         BNZ   ERROR             Error: no PTKTDATA profile or ACEE, etc
* WORK DATA
PTKTVAL  DS    D                    Generated PassTicket value
USER     DS    ØCL9                 Userid data :
         DS    AL1(7)                - number of userid characters
         DS    CL8'IBMUSER '         - value for the userid
APPLIC   DS    ØCL9                 Application data :
         DS    AL1(4)                - number of application characters
         DS    CL8'IMS1    '         - value for the application name
RC       DS    F                    Return Code
         CVT   DSECT=YES            , CVT
         ICHPRCVT                   , RACF CVT
```

The CICS Front End Programming Interface (FEPI), an integral part of CICS, can also be used to generate a PassTicket:

```
EXEC CICS FEPI REQUEST PASSTICKET
```

requests the external security manager to supply a PassTicket.

On other platforms, PassTickets can be generated in several ways:

- Develop your own routine, using the algorithm described in the *RACF Macros and Interfaces* documentation. Though not an easy task, this is possible (see below).
- Buy a commercial product that generates PassTickets on your platform.
- Implement a complete enterprise solution, like a single sign-on

product (the ‘Holy Grail’ of security). Many single sign-on products support RACF PassTickets.

The most common cause for technical problems with PassTicket verification stems from differences in time settings. When PassTickets are created on a Windows or Unix machine and are verified by RACF on OS/390, both machines must be at the same GMT (or UTC – Universal Time Coordinated) time.

SECURITY ASSESSMENT

In fact, of course, the PassTicket concept merely displaces the security issue so that the weak link is no longer the network, but instead the PassTicket generator, and chiefly the secure keys. The secret Secured Sign-on application keys must not be easily compromised, otherwise a hacker could generate PassTickets on your behalf and use your userid at will (and changing your regular password won’t help here!).

One possible solution is to encipher the keys, or to store them on a trusted server, or even on a diskette that you keep in security (if you use a personal generator like the one I wrote). If they are compromised, you must immediately change their value on MVS by a RACF command.

Whether generation is done on the mainframe or on a remote site, I think PassTicket usage should be audited. SMF records type 80 are cut for event code 1 (RACINIT : job initiation, TSO logon or logoff) with event code qualifiers 32 (‘successful initiation using passticket’) and 33 (‘attempted replay of passticket’).

RACF IMPLEMENTATION

The first step in implementing a PassTicket is to activate the passticket class :

```
SETROPTS CLASSACT(PTKTDATA)
SETROPTS RACLIST(PTKTDATA)
```

You then define a profile for the userid + application:

```
RDEFINE  PTKTDATA  appl.group.userid
SSIGNON(KEYMASKED(Ø123456789abcdef))
```

where appl is to be replaced by the application name (CICS, TSO+SMFID, etc). Group and userid are the RACF group and userid, and 0123456789abcdef (the Secured Sign-on application key) should be replaced by a valid 16-digit secret key. Note that appl.group.userid can also be replaced by appl.userid if the connection group doesn't matter. A more secure alternative to

```
SSIGNON(KEYMASKED(...))
```

is

```
SSIGNON(KEYENCRYPTED(...))
```

but this requires a cryptographic product to be active on the system.

Refresh the PTKTDATA class:

```
SETR REFRESH RACLIST(PTKTDATA)
```

Next, the userid, the application name, and the Secured Sign-on application key (the 'secret factor' that must be kept in security) must be known by the PassTicket generator.

Before RACF 2.2, an application was limited to a single key ('unqualified' PTKTDATA profile, with the application name only). It is preferable to use 'qualified' PassTicket profile names, with the user's group name and userid.

RACF PASSTICKET GENERATOR FOR WINDOWS

My Passticket generator for Windows, PTKTGEN, is a software-only implementation of a RACF Passticket generator. It is a DOS program written in REXX; it invokes Megacrypt/DOS (a freeware) for encryption functions. Secured Sign-on application keys are stored in a 'userid.INI' file (where userid is the RACF userid). This is sensitive information that you should protect. This is why I call PTKTGEN an 'unsecure Passticket generator'. PTKTGEN can be downloaded from:

http://os390-mvs.hypermart.net/ptkt.zip

You can use it for demonstration or educational purposes.

Thierry Falissard
etic software (France)

RACF and DFSMS

With DFSMS, ownership of data has been introduced in RACF. Previously, when a dataset was opened, all authorization checking in RACF was performed against the user who invoked the function. In a DFSMS environment, however, the ACS routines are used to control which SMS constructs are assigned to a dataset. RACF is then called to check whether the assigned management and storage class can be used in the allocation of the dataset. The authority to use a storage class and management class is checked not against the user who allocated the dataset, but against the owner of the data. This will be the user or group specified in the RESOWNER field in the dataset profile that protects the dataset; if none is specified, it will default to the user or group named by the high-level qualifier of the dataset name.

DFSMS DEFAULTS

RACF stores the DFSMS defaults in user and group profiles. The DFSMS defaults are:

- DATAAPPL
- DATACLAS
- STORCLAS
- MGMTCLAS.

The use of RACF defaults for SMS constructs can give an installation a lot of flexibility in assigning default values to the various data owners. The major benefits are that:

- It can reduce the complexity of the ACS routines.
- It can eliminate the need for frequent change.

With default values stored in RACF profiles, the storage administrator can use RACF commands to change the values for users and groups. These changes take effect immediately.

To activate the DFSMS defaults in the ACS routines, the IGDSMSnn parmlib member must be specified as follows:

```
ACSDEFAULTS=YES
```

SMS will initialize the following ACS routine variables from an additional call to RACF:

- &APPLIC
- &DEF_DATACLAS
- &DEF_MGMTCLAS
- &DEF_STORCLAS

DFSMS user defaults

For each user defined to RACF there is a user profile. Part of the user profile is the DFP segment which contains four fields reserved for the DFSMS defaults. This is shown in Figure 1.

To change or delete any of the DFSMS defaults in a user profile, the storage administrator can issue the ALTUSER RACF command as follows:

```
ALTUSER  userid     DFP(DATAAPPL(.........)
                       DATACLAS(.........)
                       MGMTCLAS(.......)
                       STORCLAS(.........))
```

To list the DFSMS defaults for a particular user, enter the following RACF command:

```
LISTUSER  userid  DFP
```

DFSMS group defaults

A RACF group profile contains information about the group and about which users are connected to the group. Part of the group profile is the DFP segment which contains four fields reserved for the DFSMS defaults. This is shown in Figure 2.

To change or delete any of the DFSMS defaults in a group profile, the storage administrator can issue the ALTGROUP RACF command as follows:

```
ALTGROUP  groupid     DFP(DATAAPPL(.........)
                       DATACLAS(.........)
```

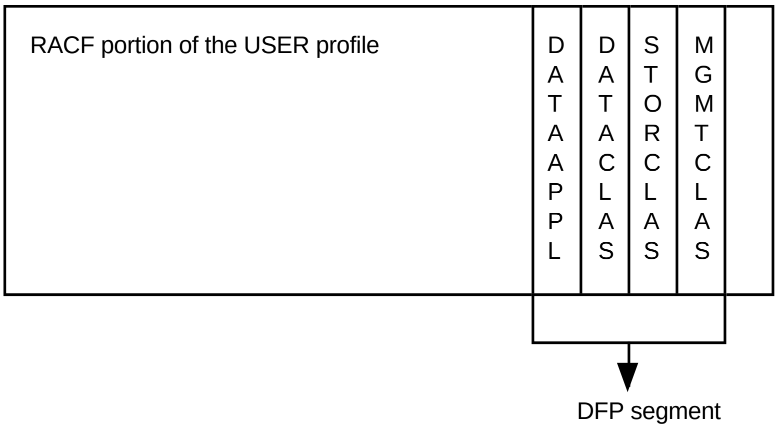

Figure 1: RACF user profile

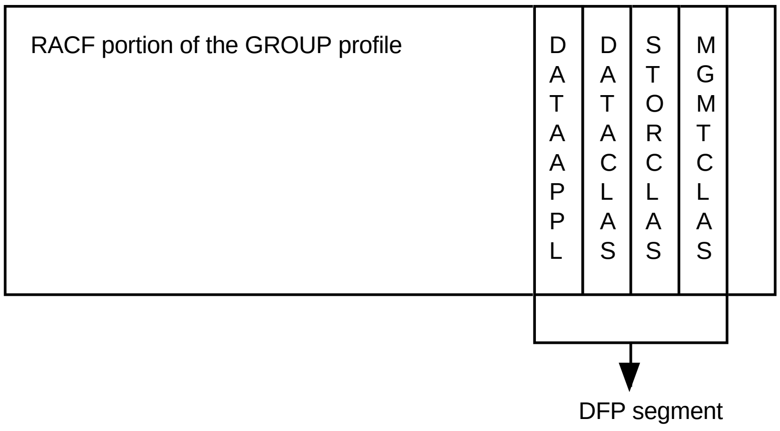

Figure 2: RACF group profile

```
                        MGMTCLAS(.......)
                        STORCLAS(........))
```

To list the DFSMS defaults for a particular group, enter the following RACF command:

```
LISTGRP  groupid  DFP
```

DATASET PROFILES

For a dataset to be protected by RACF, the HLQ must be defined to RACF as either a user profile or a group profile, and a RACF dataset profile must exist to protect the dataset. As part of the dataset profile, the DFP segment contains the RESOWNER field, in which you can

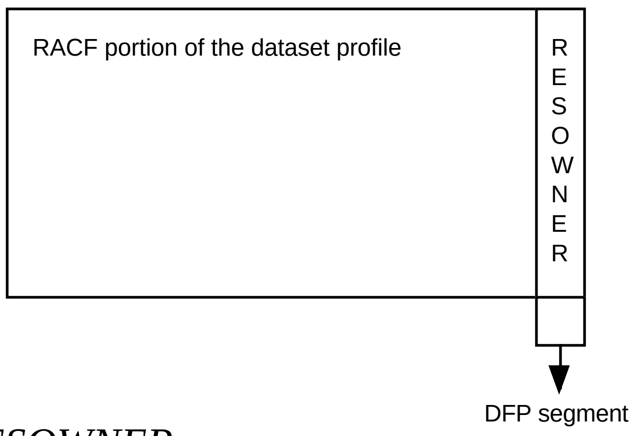

Figure 3: RESOWNER

specify the owner of a system-managed dataset protected by the profile. The owner can be a RACF-defined user or group. If there is no RESOWNER specified in this field, the HLQ of the dataset becomes the dataset owner.

It is important to note that the dataset owner is not the same as the owner of the dataset profile. The owner of the dataset profile is used for security administration purposes, whereas the dataset owner is used in checking against the STORCLAS and MGMTCLAS profiles for access when allocating a dataset. The RESOWNER field is shown in Figure 3.

To list the RESOWNER field, issue the following RACF command:

```
LISTDSD DATASET('profile name') DFP GENERIC
```

PROTECTING THE DFSMS STORCLAS AND MGMTCLAS

RACF provides the following general resource classes defined in the RACF class descriptor table for protecting SMS management classes and SMS storage classes:

- MGMTCLAS
- STORCLAS

To define a general resource profile to RACF, the RDEFINE RACF command must be issued:

```
RDEFINE MGMTCLAS 'management class'  OWNER(stgadmin) UACC(NONE)
RDEFINE STORCLAS  'storage class' OWNER(stgadmin) UACC(NONE)
```

The owner field of the profile should be the RACF group to which the storage administrator is connected. This will ensure that the storage administrators maintain control over the profile. To permit a user or group to access the resource class, use the RACF PERMIT command. For example, to give the group APG1 access to the SMS storage class STRCLAS1, enter the following command:

```
PERMIT STRCLAS1 CLASS(STORCLAS) ID(APG1) ACCESS(READ)
```

To list who is on the access list, use the AUTHUSER parameter of the RLIST RACF command as follows:

```
RLIST STORCLAS STRCLAS1 AUTHUSER
```

To activate the SMS classes, use the SETROPTS RACF command as follows:

```
SETROPTS RACLIST(STORCLAS MGMTCLAS)
```

When a general resource class is RACLISTED, the profile is available to all users, thereby eliminating the need for RACF to retrieve a profile each time a user requests access to a resource protected by that profile. As a result, when this process is activated, processing overhead is reduced.

If a new profile is added, changed, or deleted in one of the SMS classes, the in-storage profile needs to be refreshed as follows:

```
SETROPTS RACLIST(class-name) REFRESH
```

AUTHORIZATION CHECKING

The following information explains the steps that are performed when allocating a new SMS-managed dataset.

When a user allocates a new SMS-managed dataset, DFSMSdfp calls RACF and checks the RACF dataset profile for the dataset to be allocated. If a RESOWNER is specified in the RESOWNER field, this will then become the dataset owner; if not, the HLQ of the dataset becomes the dataset owner. If the SMS parmlib member IGDSMSnn contains the parameter ACSDEFAULTS=YES, DFSMSdfp then calls RACF and checks the dataset owner profile for the SMS defaults, which are contained in the DFP segment. The dataset owner profile may be either a user profile or a group profile. The SMS defaults are obtained as follows:

- When the dataset owner is a group defined to RACF, the DFP segment of the group profile is checked for each default.
- When the dataset owner is a user defined to RACF, the DFP segment of the user profile is checked for each default. If a value is not found for a certain default in the DFP segment, the DFP segment of the user's default group is checked. If a default is specified in this profile, this default is used.

The ACS routines are then invoked. The logic which is contained in the ACS routines may or may not use SMS defaults of the dataset owner. Once the ACS routines have been completed and the SMS classes have been assigned to the dataset to be allocated, RACF is called to resource-check the management class assigned.

The check is performed against the dataset owner. If the dataset owner has access to the management class, the assigned storage class is resource-checked. If the dataset owner does not have sufficient access to the storage class, the dataset allocation fails. If the dataset owner has the required access to both the management class and the storage class, dataset access checking is invoked. When dataset access checking is invoked, the user requesting the allocation is used in the access check and not the dataset owner. If the user has the required access to allocate the dataset, allocation is granted.

During authorization checking of the management class and storage class, the access list of the profile is checked. If the dataset owner has READ, UPDATE, or ALTER access to the profile, access is granted to the requested management class or storage class. If the dataset owner is on the access list with NONE or EXECUTE (less than READ), access is denied and allocation fails. If the dataset owner is a user and not a group, all the groups that the user is connected to are checked to see whether they have an access of READ, UPDATE, or ALTER to the resource. If any of these groups has this access level, access is granted to the requested management class or storage class. If no access is granted and there is a group defined on the access list with an access of NONE or EXECUTE, access is denied and allocation fails. This process applies only when the List Of Group RACF option is active. If List Of Groups checking is inactive, only the current connect group is checked for access.

Note that:

- It is advisable to use the Global Access Table to store SMS classes which anyone may use.
- Revoked USERIDs should not be used as a resource owner – this causes RACF to fail the request.
- If you specify USE_RESOWNER=NO in the IGDSMSxx member, RACF uses the execution userid instead of the resource owner to check authorization. This allows users who do not use a naming convention, userid, or group as the HLQ of dataset names to check authorization to use storage and management classes. If USE_RESOWNER=YES is specified in the IGDSMSxx member, there is no change to current processing.

PROTECTING SMS DEFAULTS

If the intention is to use the SMS defaults in the ACS routines, they should be protected by RACF. To protect these defaults, RACF general resource profiles are defined in the FIELD class. FIELD level checking in RACF can be used to control access to the fields in the DFP segment. Members of the storage administration group should be able to update all fields in all DFP segments. Individual users should be able to list all fields in their own user and dataset profiles. The only field that they should be able to update is DATACLAS in the user profile.

Control of the profiles in the FIELD class should remain with the security administrator. The storage administrator should not be given CLAUTH(FIELD) as the FIELD class contains not only profiles for the DFP segment, but also profiles for other segments unrelated to DFP, such as the TSO segment contained in user profiles.

To protect user defaults, use the following RDEFINE RACF command:

```
RDEFINE FIELD USER.DFP.dflt.name OWNER(res-owner) UACC(access authority)
```

where dflt-name is a SMS default name as follows:

- DATAAPPL
- DATACLAS
- STORCLAS
- MGMTCLAS

To protect the group defaults, use the following RDEFINE RACF command:

```
RDEFINE FIELD GROUP.DFP.dflt.name OWNER(res-owner) UACC(access
authority)
```

If all users or groups require the same access, use a generic profile to cover all the fields in a DFP segment as follows:

```
RDEFINE FIELD USER.DFP.* OWNER(res-owner) UACC(access authority)
RDEFINE FIELD GROUP.DFP.* OWNER(res-owner) UACC(access authority)
```

The required PERMIT commands must be issued:

```
PERMIT USER.DFP.dflt-name CLASS(FIELD) ID(userid/group name)
ACCESS(access-level)
PERMIT GROUP.DFP.dflt-name CLASS(FIELD) ID(userid/group name)
ACCESS(access-level)
```

Note that UPDATE authority is sufficient to change a value in a field of the DFP segment.

To RACLIST the FIELD class, enter the following RACF command:

```
SETROPTS RACLIST(FIELD)
```

To activate the FIELD general resource CLASS, enter the following RACF command:

```
SETROPTS CLASSACT(FIELD)
```

To protect the RESOWNER field contained in the DFP segment of a dataset profile, a RACF general resource profile is defined in the FIELD CLASS as follows:

```
RDEFINE FIELD DATASET.DFP.RESOWNER OWNER(res-owner) UACC(access-
authority)
```

R F Perretta
Millenium Computer Consultancy (UK)

Resetting passwords

The application presented here was designed to assist our help desk to reset callers' passwords. It was initially written to run under ISPF, but as the help desk spends a lot of time logged onto CICS (our e-mail package runs under this environment), it was ported to run there.

The application has been tested on a system running APPC, CICS/ESA Version 4, and RACF Version 2.4.

A number of hurdles had to be overcome when I ported the application:

- It's not a good idea to run authorized code under CICS.
- It's not recommended under CICS to allow your application to converse with the user.
- We couldn't allow the application to perform any worse than under TSO/ISPF.

The application is initiated under CICS by the user entering transaction HDPW from the screen. Transaction HDPW 'uses' program HDUSER, which calls up BMS HDMAP to allow the user to enter the customer id and command. HDUSER verifies that all the fields on the screen are entered, based on a selected action; if not, it will loop round until it's happy.

HDUSER then builds a commarea and links to program HDUSERI.

HDUSERI allocates an LU6.2 connection and then initiates an APPC transaction. A conversation with this transaction is undertaken, and the results are stored in the commarea storage provided by HDUSER. Control is then passed back to HDUSER, which in turn displays the results from the HDUSERI call using HDMAP.

Note the following:

- By using an APPC scheduled transaction, I could call 'authorized' programs out of the CICS environment.
- Performance was guaranteed by using multi-scheduled transactions, rather than standard.

- I used a looping mechanism in HDUSER to make the user think he was conversing with the transaction.

VTAM

The following is the VTAM ACB source I used, created in member APPLHD;

```
         VBUILD     TYPE=APPL
HDLU62   APPL       ACBNAME=HDLU62,                      X
         APPC=YES,                                       X
         AUTOSES=Ø,                                      X
         DDRAINL=NALLOW,                                 X
         DLOGMOD=LU62SYS1,                               X
         DMINWNL=3,                                      X
         DMINWNR=6,                                      X
         DRESPL=NALLOW,                                  X
         DSESLIM=9,                                      X
         EAS=1,                                          X
         MODETAB=MODELU6,                                X
         PARSESS=YES,                                    X
         SECACPT=ALREADYV,                               X
         SRBEXIT=YES,                                    X
         VPACING=2
```

Note that the dlogmod and modetab were already set up for me, but you may have to create your own.

Vary the node active by using the following console command:

```
V NET,ACT,ID=APPLHD
```

ASCH

Listed below are the ASCH parameters I used. These are added to your SYS1.PARMLIB ASCHPM*xx* member.

```
CLASSADD
     CLASSNAME(MULTI)
     MAX(25)
     MIN(1)
     RESPGOAL(1)
     MSGLIMIT(12ØØØ)
     TPDEFAULT
     REGION(48M)
     TIME(144Ø)
     MSGLEVEL(1,1)
     OUTCLASS(X)
```

To activate these new parms, issue the following console command:

```
T ASCH=xx
```

APPC

Listed below are the APPC parameters I used. These are added to your SYS1.PARMLIB APPCPM*xx* member;

```
LUADD
     ACBNAME(HDLU62)
     SCHED(ASCH)
     BASE
     TPDATA(BD.VOMVSZT.CSR.TPDATA.CLUSTER)
     TPLEVEL(SYSTEM)
```

In my case, the sideinfo dataset was previously defined; note that you may need to add your own sideinfo parameter.

You can use

```
T APPC=xx
```

to activate any changes made to the parmlib member.

Use the following IDCAMS define to create your TP dataset:

```
DEFINE CLUSTER                                              -
          (NAME(BD.VOMVSZT.CSR.TPDATA.CLUSTER)              -
          INDEXED REUSE                                     -
          SHAREOPTIONS(3 3)                                 -
          RECORDSIZE(3248 7Ø24)                             -
          KEYS(112 Ø)                                       -
          TRACKS(5Ø)                                        -
          VOLUME(SYSLØ9))
```

Once the TP file is created, you'll need to prime it with your TP data.

The following JCL adds a TPNAME of HDUSER, and inserts the required JCL to run the transaction under ASCH.

```
//BDCSRT     JOB (,IS),'CALUM',CLASS=A,MSGCLASS=X,
//     NOTIFY=&SYSUID
//STEPØØØ2     EXEC PGM=ATBSDFMU
//SYSPRINT     DD SYSOUT=*
//SYSSDOUT     DD SYSOUT=*
```

```
//SYSSDLIB    DD DISP=SHR,
//            DSN=BD.VOMVSZT.CSR.TPDATA.CLUSTER
//SYSIN       DD DATA,DLM=QQ
  TPADD
     TPNAME(HDUSER)
     SYSTEM
     ACTIVE(YES)
     TPSCHED_DELIMITER(###)
     TAILOR_SYSOUT(NO)
     TAILOR_ACCOUNT(NO)
     CLASS(MULTI)
     TPSCHED_TYPE(MULTI_TRANS)
     GENERIC_ID(BDCICPG)
     JCL_DELIMITER(END_OF_JCL)
//BDUSRTA     JOB (,IS),'APPC/HDUSER',MSGCLASS=X
//STEPØØØ1    EXEC PGM=IKJEFTØ1,PARM='ZHDUSER'
//SYSPROC     DD DISP=SHR,DSN=BD.COMMON.CLISTT
//            DD DISP=SHR,DSN=BD.COMMON.CLISTS
//            DD DISP=SHR,DSN=BD.COMMON.CLIST
//SYSTSPRT    DD SYSOUT=*
//SYSPRINT    DD SYSOUT=*
//SYSTERM     DD SYSOUT=*
//SYSHELP     DD DSN=SYS1.HELP,DISP=SHR
//SYSIN       DD DUMMY
//SYSTSIN     DD DUMMY
END_OF_JCL
               KEEP_MESSAGE_LOG(ERROR)
               MESSAGE_DATA_SET(
               BD.&SYSUID.&TPDATE.&TPTIME.HDLOG
               )
               DATASET_STATUS(NEW)
###
QQ
```

CICS DEFINITIONS

The required CICS definitions for HDUSER are shown in Figures 1 to 4.

REXX EXECS

I've used two REXX execs, ZHDUSER and XHDUSER. The first is 'wrapper' code, which allows multi-scheduling. These execs are run from the ASCH-initiated JCL.

```
Connection                  :    HD00
     Group                  :    HDUSER
     Description            :    Connection used by HDUSERI
CONNECTION IDENTIFIERS
     Netname                :    HDLU62
     INDsys                 :
REMOTE ATTRIBUTES
     REMOTESYSTem           :
     REMOTEName             :
     REMOTESYSNet           :
CONNECTION PROPERTIES
     Accessmethod           :    Vtam
     Protocol               :    Appc
     Conntype               :
     Singlesess             :    No
     Datastream             :    User
     RECordformat           :    U
     Queuelimit             :    No
     Maxqtime               :    No
OPERATIONAL PROPERTIES
     Autoconnect            :    Yes
     INService              :    Yes
SECURITY
     Securityname           :
     Attachsec              :    Local
     BINDPassword           :
     BINDSecurity           :    No
     Usedfltuser            :    No
RECOVERY
     Psrecovery             :    Sysdefault


Mapset                      :    HDMAP
     Group                  :    HDUSER
     Description            :    Mapset for HDUSER
     Resident               :    No
     USAge                  :    Normal
     USElpacopy             :    No
     Status                 :    Enabled
     Rsl                    :    00


PARTNer                     :    HDUSER
     Group                  :    HDUSER
     Description            :    TPNAME definition for HDUSER
REMOTE LU NAME
     NETName                :    HDLU62
     NETWork                :
SESSION PROPERTIES
     Profile                :    DFHCICSA
REMOTE TP NAME
     Tpname                 :    HDUSER
     Xtpname                :
```

Figure 1: CICS definitions for HDUSER

```
PROGram                 :   HDUSER
    Group               :   HDUSER
    Description         :   Main program
    Language            :   Assembler
    RELoad              :   No
    RESident            :   No
    USAge               :   Normal
    USElpacopy          :   No
    Status              :   Enabled
    Rsl                 :   00
    Cedf                :   Yes
    Datalocation        :   Any
    EXECKey             :   User
REMOTE ATTRIBUTES
    REMOTESystem        :
    REMOTEName          :
    Transid             :
    EXECUtionset        :   Fullapi

PROGram                 :   HDUSERI
    Group               :   HDUSER
    Description         :   LU62 comms program
    Language            :   Assembler
    RELoad              :   No
    RESident            :   No
    USAge               :   Normal
    USElpacopy          :   No
    Status              :   Enabled
    Rsl                 :   00
    Cedf                :   Yes
    Datalocation        :   Any
    EXECKey             :   User
REMOTE ATTRIBUTES
    REMOTESystem        :
    REMOTEName          :
    Transid             :
    EXECUtionset        :   Fullapi
```

Figure 2: CICS definitions for HDUSER (continued)

ZHDUSER

ZHDUSER is the wrapper code.

```
/* REXX
trace i */

cc = 0

address tso "atbgtrn returnc"
cc = rc

if cc ¬= 0 then signal exit_point_zhduser

do forever

  call xhduser
```

```
Sessions                        :       HD00SESS
        Group                   :       HDUSER
        Description             :       Session used by HDUSER
SESSION IDENTIFIERS
        Connection              :       HD00
        SESSName                :
        NETnameq                :
        Modename                :       LU62SYS1
SESSION PROPERTIES
        Protocol                :       Appc
        Maximum                 :       010, 008
        RECEIVEPfx              :
        RECEIVECount            :
        SENDPfx                 :
        SENDCount               :
        SENDSize                :       04096
        RECEIVESize             :       04096
        SESSPriority            :       000
        Transaction             :
OPERATOR DEFAULTS
        OPERId                  :
        OPERPriority            :       000
        OPERRsl                 :       0
        OPERSecurity            :       1
PRESET SECURITY
        USERId                  :
OPERATIONAL PROPERTIES
        Autoconnect             :       Yes
        Inservice               :
        Buildchain              :       Yes
        USERArealen             :       000
        Ioarealen               :       00000, 00000
        RELreq                  :       No
        Discreq                 :       No
        NEPclass                :       000
RECOVERY
        RECOVOption             :       Sysdefault
        RECOVNotify             :       None
```

Figure 3: CICS definitions for HDUSER (continued)

```
  address tso "atbgtrn returnc"
  cc = rc

  if cc ¬= 0 then leave

end

exit_point_zhduser:
exit cc
```

XHDUSER

XHDUSER is the main routine. It communicates with the CICS user who initiated the conversation, and also processes the RACF commands to reset user passwords.

```
TRANSaction          :   HDPW
    Group            :   HDUSER
    Description      :   Main transaction
    PROGram          :   HDUSER
    Twasize          :   00000
    PROFile          :   DFHCICST
    Partitionset     :
    STAtus           :   Enabled
    PRIMedsize       :   00000
    TASKDATALoc      :   Any
    TASKDATAKey      :   User
    STOrageclear     :   No
    Runaway          :   System
    Shutdown         :   Disabled
    Isolate          :   Yes
REMOTE ATTRIBUTES
    Dynamic          :   No
REMOTESystem         :
    REMOTEName       :
    TRProf           :
    Localq           :
SCHEDULING
    PRIOrity         :   001
    Tclass           :   No
    TRANClass        :   DFHTCL01
ALIASES
    Alias            :
    TASKReq          :
    XTRanid          :
    TPName           :
    XTPname          :
```

Figure 4: CICS definitions for HDUSER (continued)

```
/* REXX
trace e */

userid = copies(' ',8)
dummy_cc = copies('0',8)
numvars = 14
cc = 0
yy = 19

recbuf = copies(' ',24)
reqlen = length(recbuf)

message = copies(' ',1978)
message_length = left('07ba'x,2)
sndlen = length(message)

cvt        = c2x(storage(10,4))
ascb       = c2x(storage(224,4))
asxb       = c2x(Storage(D2x(X2d(ascb)+108),4))
acee       = c2x(Storage(D2x(X2d(asxb)+200),4))
```

```
calluser  =     Storage(D2x(X2d(acee)+21),8)
calltime  = time()
calldate  = date('e')


address cpicomm "cmaccp convid r_c"
if r_c > Ø then call error CMACCP r_c

address cpicomm ,
         "cmrcv convid recbuf reqlen datarec reclen status rtsr r_c"
if r_c > Ø then call error CMrRV r_c

parse var recbuf 1 length 3 action 9 userid2 17 newpass 25 .

userid = overlay(userid2,userid,1,8,' ')

say 'CallUser' calluser 'CallTime' calltime 'CallDate' calldate ,
    'CallType' action

select
  when (action = 'LIST')   then call action_list_user
  when (action = 'RESET')  then call action_reset_user
  when (action = 'RESUME') then call action_resume_user
  otherwise  message = overlay('selection not available yet',message)
end

message = message_length||message

ptype = 3
address cpicomm "cmsst convid ptype r_c"
if r_c > Ø then call error CMSST r_c

sndbuf = message
address cpicomm "cmsend convid sndbuf sndlen rtsr r_c"
if r_c > Ø then call error CMSEND r_c

address cpicomm ,
         "cmrcv  convid recbuf reqlen datarec reclen status rtsr r_c"
if (r_c > Ø) & (r_c ¬= 18) then call error CMRCV r_c

exit Ø
error:nop
arg rtn retcode
errmsg = 'Error in 'rtn'; return code is 'retcode
message = overlay(errmsg,message)
address cpicomm "cmdeal convid r_c"
exit 8

action_list_user:
```

```
call get_storage

call set_parms

program = lmvuseri

call call_racf_program

call variable_tab

call retrieve_program_diags

if progrc *= dummy_cc then do
  message = overlay('Bad rc, check userid entered ok.',message)
  say 'Userid ' userid1
  say 'ProgRc ' progrc
  say 'RACFRc ' racfrc
  say 'RACFRSN' racfrsn
  say 'SAFRc  ' safrc
  say 'SAFRSN ' safrsn
  say 'SAFmsg ' safmsg
  signal action_list_user_end
end

offset = 56

do loop = 1 to numvars

  working_addr = d2x(c2d(addr)+offset)
  info_length = get_var(working_addr,4)
  info_length = c2d(info_length)
  offset = offset + 4
  working_addr = d2x(c2d(addr)+offset)
  varz.loop = get_var(working_addr,info_length)

  var_type = substr(var.loop,1,1)
  var_len  = substr(var.loop,2,2)
  var_flag = substr(var.loop,4,4)
  vart.loop = substr(var.loop,8)

  select
   when (var_type = 'C') then,
        varz.loop = substr(varz.loop,1,var_len)
   when (var_type = 'F') then,
        varz.loop = substr(c2d(varz.loop),1,var_len)
   when (var_type = 'X') then,
        varz.loop = substr(c2x(varz.loop),1,var_len)
   otherwise nop
  end
  select
   when (var_flag = 'no') then nop
   when (var_flag = 'grp') then dflt_group = varz.loop
   when (var_flag = 'flag') then do
                                 varx = varz.loop
                                 varz.loop = flag.varx
                                 end
   when (var_flag = 'date') then do
                                 if varz.loop = 'ØØØØØ' then,
                                    varz.loop = ' '
                                 else,
                                 if varz.loop = 'FFFFF' then,
                                    varz.loop = ' '
                                 else do
                                    cyy = substr(varz.loop,1,2)
                                    if cyy < 7Ø then yy = 2Ø
                                    varz.loop = datex(j,e,yy||varz.loop)
                                 end
                                 end
   otherwise nop
  end

  offset = offset + info_length
end

call build_screen

message = overlay(screen,message)

action_list_user_end:

call free_storage

return

retrieve_program_diags:
working_addr = c2x(addr)
userid1= get_var(working_addr,8)

working_addr = d2x(c2d(addr)+8)
progrc = get_var(working_addr,8)

working_addr = d2x(c2d(addr)+16)
racfrc = get_var(working_addr,8)

working_addr = d2x(c2d(addr)+24)
racfrsn= get_var(working_addr,8)

working_addr = d2x(c2d(addr)+32)
safrc  = get_var(working_addr,8)
```

```
working_addr = d2x(c2d(addr)+40)
safrsn = get_var(working_addr,8)

working_addr = d2x(c2d(addr)+48)
safmsg = get_var(working_addr,8)
return

get_storage:
command = get
addr    = left('00000000'x,4)
length = 1024
address linkmvs "storage command addr length"
cc = rc
return

free_storage:
command = free
address linkmvs "storage command addr length"
return

get_var: return storage(arg(1),arg(2))

call_racf_program:
"tsoexec call 'bdmx.mnlodzsx.auth("program")'" "'"addr"' asis"
cc = rc
return

set_parms:
working_addr = c2x(addr)
save_var = storage(working_addr,8,userid)
working_addr = d2x(c2d(addr)+8)
save_var = storage(working_addr,8,dummy_cc)
return

get_var:
working_addr = arg(1)
len = arg(2)
getstor = storage(working_addr,len)
return getstor

variable_tab:
var.1 = 'C'||08||'grp '||'Default connect group'
var.2 = 'C'||20||'no  '||"User's name"
var.3 = 'X'||05||'date'||'Password last changed date'
var.4 = 'F'||02||'no  '||'Password change interval'
var.5 = 'X'||05||'date'||'Last access date'
var.6 = 'X'||04||'no  '||'Last access time (hhmm)'
var.7 = 'X'||02||'no  '||'Number password attempts'
var.8 = 'X'||02||'flag'||'Userid revoked'
var.9 = 'X'||02||'flag'||'Auditor attributes'
var.1Ø= 'X'||Ø2||'flag'||'Operations attributes'
var.11= 'X'||Ø2||'flag'||'Special attributes'
var.12= 'X'||Ø2||'flag'||'Password not required'
var.13= 'X'||Ø2||'flag'||'User being audited'
var.14= 'C'||8Ø||'no  '||'Installation data'
flag.ØØ = 'No'
flag.8Ø = 'Yes'
return

build_screen:
screen = copies(' ',78*numvars)
do loop = 1 to numvars
   z = loop - 1
   screen = overlay(vart.loop,screen,1+(z*78))
   screen = overlay(varz.loop,screen,29+(z*78))
end
return

action_resume_user:
oon = outtrap("resumeu.",'*')

"alu" userid "resume"
cc = rc

ooff = outtrap("OFF")

if cc = Ø then,
   message = overlay('Resume for user id completed ok',message)

else,
  do loop = 1 to resumeu.Ø
     offset = (((loop-1)*8Ø)+1)
     message = overlay(resumeu.loop,message,offset)
  end
return

action_reset_user:
oon = outtrap("resetu.",'*')

"alu" userid "password("newpass")"
cc = rc

ooff = outtrap("OFF")

if cc = Ø then,
   message = overlay('Password reset for user id completed ok',message)

else,
  do loop = 1 to resetu.Ø
     offset = (((loop-1)*8Ø)+1)
     message = overlay(resetu.loop,message,offset)
  end
return
```

ASSEMBLER CODE

The following code is presented below:

- HDUSER, the main CICS program for the application.
- HDUSERI, LU6.2 communication.
- HDMAP, CICS BMS.
- LMVUSERI, which retrieves RACF information for userid.

HDCOMM and HDUSER

This code should be link'd Rmode ANY, Amode 31, and RENT.

HDCOMM

```
COMMAREA DSECT
RETURN_MESSAGE DS CL2ØØØ
         ORG RETURN_MESSAGE
ACTION   DS    CL6
USER_ID  DS    CL8
NEWPASS  DS    CL8
COMMDATA_L DS  H
DATA_POS EQU   *-COMMAREA
COMMDATA DS    CL(L'RETURN_MESSAGE-DATA_POS)
         ORG   ,
COMMAREA_LENGTH EQU *-COMMAREA
*
```

HDUSER

```
HDUSER   DFHEIENT CODEREG=(12),DATAREG=(13),EIBREG=(11)
*
         MVC   TRANS_,EIBTRNID
*
         EXEC CICS HANDLE CONDITION MAPFAIL(SEND_MAP_SCRATCH)
*
         EXEC CICS HANDLE AID PF3(DISPLAY_RESULTS_END)
*
CHECK_FOR_INFORMATION_SCREEN EQU *
         EXEC CICS RECEIVE MAPSET('HDMAP') MAP('HDMAP')
*
```

```
CHECK_FOR_USERID EQU *
         CLI   IUSERIDO,C' '
         BE    GET_USERID
         CLI   IUSERIDO,X'ØØ'
         BNE   CHECK_FOR_COMMAND
*
GET_USERID EQU *
         MVC   IUSERIDL,=H'-1'
         B     SEND_MAP
*
CHECK_FOR_COMMAND EQU *
         MVC   USER_ID_,IUSERIDO
         CLI   ICOMMO,C' '
         BE    GET_COMMAND
         CLI   ICOMMO,X'ØØ'
         BNE   CHECK_COMMAND_TYPE
*
GET_COMMAND EQU *
         MVC   ICOMML,=H'-1'
         B     SEND_MAP
*
CHECK_COMMAND_TYPE EQU *
         CLI   ICOMMO,C'L'
         BE    SETUP_LIST
*
         CLI   ICOMMO,C'R'
         BE    SETUP_RESUME
*
         CLI   ICOMMO,C'P'
         BE    CHECK_FOR_NEWPASS
*
         MVI   ICOMMO,C' '
         MVC   ICOMML,=H'-1'
         B     SEND_MAP
*
CHECK_FOR_NEWPASS EQU *
         CLI   INEWPASO,C' '
         BE    GET_NEWPASS
         CLI   INEWPASO,X'ØØ'
         BNE   CHECK_FOR_CONFIRM
*
GET_NEWPASS EQU *
         MVC   INEWPASL,=H'-1'
         B     SEND_MAP
*
CHECK_FOR_CONFIRM EQU *
         CLI   ICONFO,C' '
         BE    GET_CONFIRM
         CLI   ICONFO,X'ØØ'
         BNE   X_CHECK_PASSWORDS
*
```

```
GET_CONFIRM EQU *
         MVC   ICONFL,=H'-1'
         B     SEND_MAP
*
X_CHECK_PASSWORDS EQU *
         CLC   INEWPASO,ICONFO
         BE    SETUP_RESET
         MVC   ICONFL,=H'-1'
         MVI   ICONFO,C' '
         MVC   ICONFO+1(L'ICONFO-1),ICONFO
         B     SEND_MAP
*
SETUP_RESET EQU *
         MVC   ACTION_,=CL8'RESET'
         MVC   NEWPASS_,INEWPASO
         B     CALL_HDUSERI
*
SETUP_LIST EQU *
         MVC   ACTION_,=CL8'LIST'
         B     CALL_HDUSERI
*
SETUP_RESUME EQU *
         MVC   ACTION_,=CL8'RESUME'
*
CALL_HDUSERI EQU *
         EXEC CICS GETMAIN SET(4) FLENGTH(=A(COMMAREA_LENGTH))         X
               INITIMG(ZERO)
         USING COMMAREA,4
         ST    4,SAVE_COMMAREA_PTR
         MVC   ACTION,ACTION_
         MVC   USER_ID,USER_ID_
         MVC   NEWPASS,NEWPASS_
*
         L     1,=A(COMMAREA_LENGTH)
         STH   1,COMMAREA_H
         EXEC CICS LINK PROGRAM('HDUSERI') COMMAREA(COMMAREA)          X
               LENGTH(COMMAREA_H)
*
         MVC   ILINE1O,COMMDATA
         MVC   ILINE2O,COMMDATA+L'ILINE1O
         MVC   ILINE3O,COMMDATA+(L'ILINE1O*2)
         MVC   ILINE4O,COMMDATA+(L'ILINE1O*3)
         MVC   ILINE5O,COMMDATA+(L'ILINE1O*4)
         MVC   ILINE6O,COMMDATA+(L'ILINE1O*5)
         MVC   ILINE7O,COMMDATA+(L'ILINE1O*6)
         MVC   ILINE8O,COMMDATA+(L'ILINE1O*7)
         MVC   ILINE9O,COMMDATA+(L'ILINE1O*8)
         MVC   ILINE1ØO,COMMDATA+(L'ILINE1O*9)
         MVC   ILINE11O,COMMDATA+(L'ILINE1O*1Ø)
         MVC   ILINE12O,COMMDATA+(L'ILINE1O*11)
         MVC   ILINE13O,COMMDATA+(L'ILINE1O*12)
         MVC   ILINE14O,COMMDATA+(L'ILINE1O*13)
         MVC   ILINE15O,COMMDATA+(L'ILINE1O*14)
*
FREE_COMMAREA EQU *
         L     4,SAVE_COMMAREA_PTR
         EXEC CICS FREEMAIN DATAPOINTER(4)
         MVC   ICOMML,=H'-1'
         MVI   ICOMMO,C' '
         MVI   ICONFO,C' '
         MVC   ICONFO+1(L'ICONFO-1),ICONFO
         MVI   INEWPASO,C' '
         MVC   INEWPASO+1(L'INEWPASO-1),INEWPASO
*
SEND_MAP EQU   *
         EXEC CICS SEND MAPSET('HDMAP') MAP('HDMAP') FREEKB            X
               CURSOR
         B     CALL_TRANS
*
SEND_MAP_SCRATCH EQU   *
         EXEC CICS SEND MAPSET('HDMAP') MAP('HDMAP') ERASE             X
               MAPONLY
*
CALL_TRANS EQU *
         EXEC CICS RETURN TRANSID(TRANS_)
*
DISPLAY_RESULTS_END EQU *
         MVC   SEND_MESSAGE,END_OF_DIALOG
*
OUTPUT_TO_TERMINAL EQU *
         EXEC CICS SEND CONTROL ERASE
         EXEC CICS SEND FROM(SEND_MESSAGE)
*
EXIT_POINT DS ØH
         EXEC  CICS RETURN
*
START_OF_LITERALS DC CL8'########'
*
ZERO       DC    X'ØØ'
END_OF_DIALOG DC CL(L'SEND_MESSAGE)'.HDPW COMPLETED'
ERROR_FROM_HDUSERI DC CL(L'SEND_MESSAGE)'.HDPW INVALID USERID, PLEASE CX
               HECK AND RE-ENTER'
*
         LTORG ,
*
END_OF_LITERALS DC CL8'########'
*
*
         DFHEISTG
DATA_LENGTH DS H
SAVE_COMMAREA_PTR DS F
STATE_CHECK DS F
```

```
SEND_MESSAGE DS CL6Ø
TRANS_   DS   CL4
ACTION_  DS    CL6
USER_ID_ DS    CL8
NEWPASS_ DS    CL8
COMMAREA_H DS H
*
HDMAPSTA EQU  *
         COPY HDMAP
HDMAPLEN EQU  *-HDMAPSTA
         COPY DFHBMSCA
         COPY DFHAID
END_OF_DFHEISTG DS CL8
*
         COPY HDCOMM
*
         END
```

HDUSERI

The HDUSERI code should be link'd Rmode ANY, Amode 31, and RENT.

HDUSERI

```
HDUSERI  DFHEIENT CODEREG=(12),DATAREG=(13),EIBREG=(11)
*
         USING COMMAREA,2
         L    2,DFHEICAP
         LTR  2,2
         BZ   EXIT_POINT
*
         EXEC CICS GDS ALLOCATE SYSID(CONNECTION)                      X
               STATE(STATE_CHECK)                                      X
               CONVID(CONVERSATION_ID)                                 X
               RETCODE(RETURN_CODE)
         CLC   =F'Ø',RETURN_CODE
         BE    STARTUP_PARTNER
         MVC   COMMDATA(L'ERROR_GDS_ALLOCATE),ERROR_GDS_ALLOCATE
         B     EXIT_POINT
*
STARTUP_PARTNER EQU *
         EXEC CICS GDS CONNECT PROCESS CONVID(CONVERSATION_ID)         X
               CONVDATA(CONVERSATION_DATA)                             X
               STATE(STATE_CHECK)                                      X
               PARTNER(HD_PARTNER)                                     X
               SYNCLEVEL(Ø)                                            X
               RETCODE(RETURN_CODE)
         CLC   =F'Ø',RETURN_CODE
         BE    SEND_MESSAGE_TO_PARTNER
         MVC   COMMDATA(L'ERROR_CONNECT_PROCESS),ERROR_CONNECT_PROCESS
         B     EXIT_POINT
*
SEND_MESSAGE_TO_PARTNER EQU *
         MVC   ACTION_,ACTION
         MVC   USERID_,USER_ID
         MVC   NEWPASS_,NEWPASS
         LA    1,L'SEND_MESSAGE+L'SEND_MESSAGE_HEADER
         STH   1,SEND_MESSAGE_HEADER
         ST    1,SEND_MESSAGE_LENGTH
         EXEC CICS GDS SEND CONVID(CONVERSATION_ID)                    X
               CONVDATA(CONVERSATION_DATA)                             X
               STATE(STATE_CHECK) WAIT INVITE                          X
               FROM(SEND_MESSAGE_HEADER) FLENGTH(SEND_MESSAGE_LENGTH)  X
               RETCODE(RETURN_CODE)
         CLC   =F'Ø',RETURN_CODE
         BE    RECEIVE_MESSAGE_FROM_PARTNER
         MVC   COMMDATA(L'ERROR_SEND_MESSAGE_START),ERROR_SEND_MESSAGE_X
               START
         B     EXIT_POINT
*
RECEIVE_MESSAGE_FROM_PARTNER EQU *
         L     1,=A(L'COMMDATA+1Ø)
         ST    1,SEND_MESSAGE_LENGTH
         EXEC CICS GDS RECEIVE CONVID(CONVERSATION_ID)                 X
               CONVDATA(CONVERSATION_DATA)                             X
               STATE(STATE_CHECK) BUFFER                               X
               INTO(COMMDATA_L) FLENGTH(SEND_MESSAGE_LENGTH)           X
               MAXFLENGTH(SEND_MESSAGE_LENGTH)                         X
               RETCODE(RETURN_CODE)
         CLC   =F'Ø',RETURN_CODE
         BE    CLOSE_CONNECTION
         MVC   COMMDATA(L'ERROR_RECEIVE_MESSAGE_1),ERROR_RECEIVE_MESSAGX
               E_1
         B     EXIT_POINT
*
CLOSE_CONNECTION EQU *
         EXEC CICS GDS SEND CONVID(CONVERSATION_ID)                    X
               CONVDATA(CONVERSATION_DATA)                             X
               STATE(STATE_CHECK)                                      X
               LAST WAIT                                               X
               RETCODE(RETURN_CODE)
*
FREE_CONNECTION EQU *
         EXEC CICS GDS FREE CONVID(CONVERSATION_ID)                    X
               STATE(STATE_CHECK)                                      X
               CONVDATA(CONVERSATION_DATA)                             X
               RETCODE(RETURN_CODE)
*
```

```
EXIT_POINT DS ØH
         EXEC  CICS RETURN
*
STATE_CHECK_VALUE EQU *
         CLC   STATE_CHECK,DFHVALUE(ALLOCATED)
         CLC   STATE_CHECK,DFHVALUE(CONFFREE)
         CLC   STATE_CHECK,DFHVALUE(CONFRECEIVE)
         CLC   STATE_CHECK,DFHVALUE(CONFSEND)
         CLC   STATE_CHECK,DFHVALUE(FREE)
         CLC   STATE_CHECK,DFHVALUE(PENDFREE)
         CLC   STATE_CHECK,DFHVALUE(PENDRECEIVE)
         CLC   STATE_CHECK,DFHVALUE(RECEIVE)
         CLC   STATE_CHECK,DFHVALUE(ROLLBACK)
         CLC   STATE_CHECK,DFHVALUE(SEND)
         CLC   STATE_CHECK,DFHVALUE(SYNCFREE)
         CLC   STATE_CHECK,DFHVALUE(SYNCRECEIVE)
         CLC   STATE_CHECK,DFHVALUE(SYNCSEND)
STATE_CHECK_VALUE_END EQU *
*
MOVE_CHAR  MVC  Ø(Ø,4),USER_ID
*
START_OF_LITERALS DC CL8'########'
*
ERROR_GDS_ALLOCATE DC C'ERROR ALLOCATING LU62 CONNECTION.'
ERROR_CONNECT_PROCESS DC C'ERROR TRYING TO START TRIGGER PROCESS.'
ERROR_SEND_MESSAGE_START DC C'ERROR SENDING FIRST MESSAGE.'
ERROR_RECEIVE_MESSAGE_1 DC C'ERROR RECEIVING FIRST MESSAGE.'
ERROR_RECEIVE_MESSAGE_2 DC C'ERROR RECEIVING SECOND MESSAGE.'
*
HD_PARTNER DC CL8'HDUSER'
CONNECTION DC  CL4'HDØØ'
         LTORG ,
*
END_OF_LITERALS DC CL8'########'
*
*
         DFHEISTG
CONVERSATION_ID DS CL4
CONVERSATION_DATA DS CL24
DATA_LENGTH DS F
          DS   ØF
RETURN_CODE DS CL6
          DS   ØF
SEND_MESSAGE_HEADER DS H
SEND_MESSAGE DS CL22
          ORG  SEND_MESSAGE
ACTION_   DS   CL6
USERID_   DS   CL8
NEWPASS_  DS   CL8
          ORG  ,
SEND_MESSAGE_LENGTH DS F
STATE_CHECK DS F
END_OF_DFHEISTG DS CL8
*
         COPY  HDCOMM
*
         ORG   ,
         END
```

HDMAP

This code should be link’d Rmode ANY, Amode 31.

```
PRINT ON,NOGEN
HDMAP    DFHMSD TYPE=MAP,LANG=ASM,MODE=INOUT,SUFFIX=
         TITLE 'BMS: HDMAP    HDMAP             '
HDMAP    DFHMDI SIZE=(24,8Ø),CTRL=(PRINT,FREEKB),COLUMN=SAME,LINE=NEXT,*
               DATA=FIELD,TIOAPFX=YES,OBFMT=NO
         DFHMDF POS=(1,1),LENGTH=5,INITIAL='+HDPW',ATTRB=(PROT,BRT)
         DFHMDF POS=(1,7),LENGTH=2Ø,INITIAL='Customer''s RACF id :',   *
               ATTRB=(PROT,NORM)
* IUSERID                         IUSERID
IUSERID  DFHMDF POS=(1,28),LENGTH=8,JUSTIFY=(LEFT,BLANK),ATTRB=(UNPROT,*
               BRT,IC,FSET)
         DFHMDF POS=(1,37),LENGTH=13,INITIAL='    Command :',          *
               ATTRB=(PROT,NORM)
* ICOMM                           ICOMM
ICOMM    DFHMDF POS=(1,52),LENGTH=1,JUSTIFY=(LEFT,BLANK),ATTRB=(UNPROT,*
               BRT,FSET)
         DFHMDF POS=(1,54),LENGTH=22,INITIAL='  P for Password reset', *
               ATTRB=(PROT,NORM)
         DFHMDF POS=(2,56),LENGTH=12,INITIAL='R for Resume',           *
               ATTRB=(PROT,NORM)
         DFHMDF POS=(3,56),LENGTH=1Ø,INITIAL='L for List',ATTRB=(PROT,N*
               ORM)
         DFHMDF POS=(5,1),LENGTH=14,INITIAL='New password :',          *
               ATTRB=(PROT,NORM)
* INEWPAS                         INEWPAS
INEWPAS  DFHMDF POS=(5,16),LENGTH=8,JUSTIFY=(LEFT,BLANK),ATTRB=(UNPROT,*
               BRT,FSET)
         DFHMDF POS=(5,25),LENGTH=19,INITIAL=' Confirm password :',    *
               ATTRB=(PROT,NORM)
* ICONF                           ICONF
ICONF    DFHMDF POS=(5,45),LENGTH=8,JUSTIFY=(LEFT,BLANK),ATTRB=(UNPROT,*
               DRK,FSET)
         DFHMDF POS=(5,54),LENGTH=Ø,ATTRB=(PROT,NORM)
* ILINE1                          ILINE1
ILINE1   DFHMDF POS=(7,1),LENGTH=78,JUSTIFY=(LEFT,BLANK),ATTRB=(UNPROT,*
               NORM)
         DFHMDF POS=(7,8Ø),LENGTH=Ø,ATTRB=(PROT,NORM)
```

```
* ILINE2                          ILINE2
ILINE2   DFHMDF POS=(8,1),LENGTH=78,JUSTIFY=(LEFT,BLANK),ATTRB=(UNPROT,*
               NORM)
         DFHMDF POS=(8,8Ø),LENGTH=Ø,ATTRB=(PROT,NORM)
* ILINE3                          ILINE3
ILINE3   DFHMDF POS=(9,1),LENGTH=78,JUSTIFY=(LEFT,BLANK),ATTRB=(UNPROT,*
               NORM)
         DFHMDF POS=(9,8Ø),LENGTH=Ø,ATTRB=(PROT,NORM)
* ILINE4                          ILINE4
ILINE4   DFHMDF POS=(1Ø,1),LENGTH=78,JUSTIFY=(LEFT,BLANK),             *
               ATTRB=(UNPROT,NORM)
         DFHMDF POS=(1Ø,8Ø),LENGTH=Ø,ATTRB=(PROT,NORM)
* ILINE5                          ILINE5
ILINE5   DFHMDF POS=(11,1),LENGTH=78,JUSTIFY=(LEFT,BLANK),             *
               ATTRB=(UNPROT,NORM)
         DFHMDF POS=(11,8Ø),LENGTH=Ø,ATTRB=(PROT,NORM)
* ILINE6                          ILINE6
ILINE6   DFHMDF POS=(12,1),LENGTH=78,JUSTIFY=(LEFT,BLANK),             *
               ATTRB=(UNPROT,NORM)
         DFHMDF POS=(12,8Ø),LENGTH=Ø,ATTRB=(PROT,NORM)
* ILINE7                          ILINE7
ILINE7   DFHMDF POS=(13,1),LENGTH=78,JUSTIFY=(LEFT,BLANK),             *
               ATTRB=(UNPROT,NORM)
         DFHMDF POS=(13,8Ø),LENGTH=Ø,ATTRB=(PROT,NORM)
* ILINE8                          ILINE8
ILINE8   DFHMDF POS=(14,1),LENGTH=78,JUSTIFY=(LEFT,BLANK),             *
               ATTRB=(UNPROT,NORM)
         DFHMDF POS=(14,8Ø),LENGTH=Ø,ATTRB=(PROT,NORM)
* ILINE9                          ILINE9
ILINE9   DFHMDF POS=(15,1),LENGTH=78,JUSTIFY=(LEFT,BLANK),             *
               ATTRB=(UNPROT,NORM)
         DFHMDF POS=(15,8Ø),LENGTH=Ø,ATTRB=(PROT,NORM)
* ILINE1Ø                         ILINE1Ø
ILINE1Ø  DFHMDF POS=(16,1),LENGTH=78,JUSTIFY=(LEFT,BLANK),             *
               ATTRB=(UNPROT,NORM)
         DFHMDF POS=(16,8Ø),LENGTH=Ø,ATTRB=(PROT,NORM)
* ILINE11                         ILINE11
ILINE11  DFHMDF POS=(17,1),LENGTH=78,JUSTIFY=(LEFT,BLANK),             *
               ATTRB=(UNPROT,NORM)
         DFHMDF POS=(17,8Ø),LENGTH=Ø,ATTRB=(PROT,NORM)
* ILINE12                         ILINE12
ILINE12  DFHMDF POS=(18,1),LENGTH=78,JUSTIFY=(LEFT,BLANK),             *
               ATTRB=(UNPROT,NORM)
         DFHMDF POS=(18,8Ø),LENGTH=Ø,ATTRB=(PROT,NORM)
* ILINE13                         ILINE13
ILINE13  DFHMDF POS=(19,1),LENGTH=78,JUSTIFY=(LEFT,BLANK),             *
               ATTRB=(UNPROT,NORM)
         DFHMDF POS=(19,8Ø),LENGTH=Ø,ATTRB=(PROT,NORM)
* ILINE14                         ILINE14
ILINE14  DFHMDF POS=(2Ø,1),LENGTH=78,JUSTIFY=(LEFT,BLANK),             *
               ATTRB=(UNPROT,NORM)
         DFHMDF POS=(20,80),LENGTH=0,ATTRB=(PROT,NORM)
* ILINE15                          ILINE15
ILINE15  DFHMDF POS=(21,1),LENGTH=78,JUSTIFY=(LEFT,BLANK),             *
               ATTRB=(UNPROT,NORM)
         DFHMDF POS=(21,80),LENGTH=0,ATTRB=(PROT,NORM)
         DFHMDF POS=(23,1),LENGTH=75,                                  *
               INITIAL='Cmds P+R unavailable for ids with Special, Oper*
               ations or Auditor attributes',ATTRB=(PROT,NORM)
         DFHMDF POS=(24,1),LENGTH=8,INITIAL='Pfkeys :',ATTRB=(PROT,NORM*
               )
         DFHMDF POS=(24,10),LENGTH=7,INITIAL='PF3=End',ATTRB=(PROT,BRT)
         DFHMSD TYPE=FINAL
         END
```

LMVUSERI

This code should be link'd Rmode ANY, Amode 31, and AC=1.

LMVUSERI should be placed in an APF authorized library. An entry should also be placed in SYS1.PARMLIB member IKJTSO*xx*, under AUTHPGM. This allows the authorized code to be called within a TSO environment.

```
LMVUSERI AMODE 31
LMVUSERI RMODE ANY
LMVUSERI CSECT
         DS    0H
         B     BEGIN-LMVUSERI(,15)
         DC    C'LMVUSERI: '
         DC    C'&SYSDATE &SYSTIME'
         DS    0H
BEGIN    EQU   *
         BAKR  14,0
         LR    12,15
         LR    10,1
         USING LMVUSERI,12
         USING WORKAREA,11
*
START    EQU   *
         L     2,=A(WORK_AREA_LENGTH)
         STORAGE OBTAIN,LENGTH=(2)
         LR    11,1
         ST    11,GETMAIN_ADDRESS
         LA    13,SAVEAREA
         MVC   SAVEAREA+4(4),=C'F1SA'
         MVC   DSS_EYE,=CL8'LMVUSERI'
         MVC   RACROUT0(CHEC0_LEN),RAC_CHEK
         LR    1,10                    SAVE FOR DUMP
         L     10,0(,10)
         LR    2,1Ø                SAVE FOR DUMP
         LH    9,Ø(,1Ø)
         L     1Ø,2(,1Ø)
*
         ST    1Ø,WORKS
         LA    15,CONVERT1
         BALR  14,15
         MVC   WTO_DYN,WTO_STAT
         MVC   DYN_MSG,STAT_MSG
         MVC   DYN_MSG+13(8),WORK_VAR+Ø
         L     2,=A(STAT_MSG_LEN)
         STH   2,WTO_PARM
         LA    2,WTO_PARM
*        WTO   TEXT=((2)),MF=(E,WTO_DYN)
*
*        DC    F'Ø'                FORCE ABENDØC1
*
         USING USERD_DSECT,1Ø
         USING SAFP,6
         MVC   RACFPARM,USERD_ID
         MVC   ENTITY_1,=H'8'
         LA    7,RACROUTE_WORK
         LA    6,RACROUTØ
*
SET_AUTH_ON EQU   *
         MVC   DYN_MODE_PROB,LIST_MODE_PROB
         MVC   DYN_MODE_SUP,LIST_MODE_SUP
         MODESET ,MF=(E,DYN_MODE_SUP)
*
RACF_CALL EQU  *
         RACROUTE REQUEST=EXTRACT,WORKA=(7),RELEASE=1.9,               X
               TYPE=EXTRACT,                                           X
               ENTITYX=ENTITY_1,                                       X
               FIELDS=FIELD_LISTØ,                                     X
               MF=(E,RACROUTØ)
         LR    4,1
         LR    5,15
         LTR   15,15
         BNZ   SET_AUTH_OFF
         USING EXTWKEA,4
         LR    3,4
         AH    3,EXTWOFF
         MVC   USERD_DATA+Ø(1),=C' '
         MVC   USERD_DATA+1(L'USERD_DATA-1),USERD_DATA
         MVC   USERD_DATA,Ø(3)
*
CLEANUP  EQU   *
         SR    2,2
         SR    3,3
         ICM   2,B'Ø111',EXTWLN
         ICM   3,B'ØØØ1',EXTWSP
         DROP  4
         STORAGE RELEASE,ADDR=(4),LENGTH=(2),SP=(3)
*
SET_AUTH_OFF EQU *
         MODESET ,MF=(E,DYN_MODE_PROB)
*
SAVE_DATA EQU  *
         MVC   WORKS,SAFPRRET
         LA    15,CONVERT1
         BALR  14,15
         MVC   USERD_RACF_RSN,WORK_VAR+Ø
         MVC   WORKS,SAFPRREA
         LA    15,CONVERT1
         BALR  14,15
         MVC   USERD_RACF_RC,WORK_VAR+Ø
         MVC   WORKS,SAFPSFRC
         LA    15,CONVERT1
         BALR  14,15
         MVC   USERD_SAF_RC,WORK_VAR+Ø
         MVC   WORKS,SAFPSFRS
         LA    15,CONVERT1
         BALR  14,15
         MVC   USERD_SAF_RSN,WORK_VAR+Ø
         MVC   WORKS,SAFPMSAD
         LA    15,CONVERT1
         BALR  14,15
         MVC   USERD_MSGAD,WORK_VAR+Ø
         MVC   USERD_SAFP,Ø(7)
*
SAVE_RETURN_CODE EQU *
         ST    5,WORKS
         LA    15,CONVERT1
         BALR  14,15
         MVC   USERD_RC+Ø(L'USERD_RC),WORK_VAR+Ø
*
ENDIT    EQU   *
         L     2,=A(WORK_AREA_LENGTH)
         L     3,GETMAIN_ADDRESS
         STORAGE RELEASE,LENGTH=(2),ADDR=(3)
         SR    15,15
         PR    ,
*
*
CONVERT1 EQU   *
         UNPK  WORK_VAR(9),WORKS(5)
         MVZ   WORK_VAR,=XL8'ØØ'
         TR    WORK_VAR,TABLE
         XC    WORKS,WORKS
         BR    14
*
TABLE    DC    C'Ø123456789ABCDEF'
```

```
*
WTO_STAT WTO   TEXT=,MF=L
WTO_STAT_LEN EQU *-WTO_STAT
STAT_MSG DC    C'PARM ADDRESS XXXXXXXX'
STAT_MSG_LEN EQU *-STAT_MSG
*
LIST_MODE_SUP MODESET MODE=SUP,KEY=ZERO,MF=L
LIST_MODE_PROB MODESET MODE=PROB,KEY=NZERO,MF=L
*
RAC_CHEK RACROUTE REQUEST=EXTRACT,WORKA=*-*,RELEASE=1.9,              X
               TYPE=EXTRACT,                                          X
               CLASS='USER',                                          X
               ENTITYX=*-*,                                           X
               FIELDS=*-*,                                            X
               SEGMENT='BASE',                                        X
               MF=L
CHECØ_LEN EQU  *-RAC_CHEK
*
FIELD_LISTØ DC A(14)
         DC    CL8'DFLTGRP'
         DC    CL8'PGMRNAME'
         DC    CL8'PASSDATE'
         DC    CL8'PASSINT'
         DC    CL8'LJDATE'
         DC    CL8'LJTIME'
         DC    CL8'REVOKECT'
         DC    CL8'FLAG4'
         DC    CL8'FLAG6'
         DC    CL8'FLAG3'
         DC    CL8'FLAG2'
         DC    CL8'FLAG7'
         DC    CL8'UAUDIT'
         DC    CL8'INSTDATA'
*
WORKAREA DSECT
SAVEAREA DS    18F
GETMAIN_ADDRESS DS    F
DSS_EYE  DS    CL8
ENTITY_1 DS    H
ENTITY_2 DS    H
RACFPARM DS    CL8
WORKS    DS    CL4,C
WORK_VAR DS    CL8,C
*
RACROUTE_WORK DS    CL512
         DS    ØF
RACROUTØ DS    CL(CHECØ_LEN)
*
         DS    ØF
DYN_MODE_PROB DS CL(L'LIST_MODE_PROB)
         DS    ØF
```

```
DYN_MODE_SUP DS CL(L'LIST_MODE_SUP)
*
WTO_DYN  DS    CL(WTO_STAT_LEN)
WTO_PARM DS    H
DYN_MSG  DS    CL(STAT_MSG_LEN)
*
WORK_AREA_LENGTH EQU *-WORKAREA
*
USERD_DSECT DSECT
USERD_ID DS    CL8
USERD_RC DS    CL8
USERD_RACF_RC DS CL8
USERD_RACF_RSN DS CL8
USERD_SAF_RC DS CL8
USERD_SAF_RSN DS CL8
USERD_MSGAD DS CL8
USERD_DATA DS CL256
USERD_SAFP DS CL(CHECØ_LEN)
USERD_DATA_LENGTH EQU *-USERD_DATA
*
         PRINT OFF
         IRRPRXTW
         ICHSAFP
         END
```

RACF

I've used the "new" FACILITY class profile IRR.PASSWORD.RESET to allow help desk personnel to issue ALU userid PASSWORD and ALU userid RESUME commands. I've added the help desk RACF group to the access list with UPDATE access. This still means that they can't ALU ids with special, operations, or auditor attributes, and I've inserted a note in HDMAP to remind them of this. To allow them to list ANY userid, I've had to use my own code (LMVUSERI), rather than use the LU command.

I've also created a profile in TCICSTRN, called HDPW, to protect the HDPW transaction. I've added the same help desk group to the access list.

SCREEN SHOTS

The following shows the result after the list command has been entered.

```
+HDPW Customer's RACF id : SL452A     Command :    P for Password reset

R for Resume

L for List

New password :           Confirm password :

Default connect group         HOSYSGC1
User's name                   C REID
Password last changed date    23/Ø3/1999
Password change interval      31
Last access date              Ø6/Ø4/1999
Last access time (hhmm)       1416
Number password attempts      ØØ
Userid revoked                No
Auditor attributes            No
Operations attributes         Yes
Special attributes            Yes
Password not required         No
User being audited            No
Installation data


Cmds P+R unavailable for ids with Special, Operations or Auditor
attributes
Pfkeys : PF3=End
```

Calum Reid
Systems Programmer (UK)

Code from *RACF Update* articles

As a free service to subscribers and to remove the need to rekey the scripts, code from individual articles of *RACF Update* can be accessed on our Web site, at

http://www.xephon.com/racfupdate.html

You will need the user-id shown on your address label.

Using ICHRCX02 after PROTECT ALL – revisited

This article is an update to ‘Using ICHRCX02 after PROTECT ALL’, which appeared in Issue 14 of *RACF Update* (November 1998).

The exit ICHRCX02 as coded in the November 1998 article was intended as a tool to allow systems programmers access to unprotected RACF resources (such as external/vendor tapes) once PROTECTALL had been turned on in RACF. However, the fact that the list of individuals that are allowed access is hard-coded into the ICHRCX02 program can create problems – ICHRCX02 is loaded by RACF during IPL and is only refreshable via IPL (or by using an OEM LPA module replace/refresh functional product such as TMON, which will also refresh the RACF exit address pointer). This is problematic for a shop that needs both continued OS/390 availability and the ability to change the access list.

ICHRCX02 has therefore been modified as follows:

- The access list has been removed and assembled/link-edited into its own module, ICHRCXTB in SYS1.LINKLIB.
- ICHRCX02 has been changed to do a LOAD on the ICHRCXTB module and use the returned entry point as the starting point for the access list.
- The rest of the ICHRCX02 program, once the userid in the ACEE has been validated against the access list, remains basically the same.

Now, in order to change the access list, you simply need to alter the source to ICHRCXTB, assemble and link-edit it into SYS1.LINKLIB, and refresh the LLA.

ICHRCX02

```
//ICHRCXØ2 JOB (1Ø2331Ø),'CSH ICHRCTØ2',CLASS=1,MSGCLASS=H,
//  MSGLEVEL=(1,1),NOTIFY=&SYSUID,TIME=144Ø,REGION=8M
//*
```

```
//*********************************************************************
//**  AUTHOR:    JACK HWANG   CSHWANG@HOTMAIL.COM                      **
//**  MODIFIED:  JACK HWANG                          12/14/99          **
//**               USE EXTERNAL TABLE ICHRCXTB                         **
//**  OBJECTIVE:  ALLOW SYSTEMS PROGS TO READ ANY TAPE DATASET         **
//**              RESOURCE THAT HAS NOT BEEN DEFINED.  THIS WILL       **
//**              PROVIDE FOR PRODUCT TAPES WITH MISC DATA SET         **
//**              NAMES.                                               **
//*********************************************************************
//**  MODULE : ICHRCXØ2, RE-ENTRANT, AUTHORIZED                        **
//**  RACF RACHECK POST PROCESSING EXIT                                **
//**  USED TO MODIFY STANDARD TAPEDSN PROCESSING TO PERMIT TECH        **
//**  SUPPORT READ ACCESS TO TAPE UNDEFINED RESOURCE.                  **
//**                                                                   **
//**                                                                   **
//*********************************************************************
//*
//ASM    EXEC  PGM=ASMA9Ø,PARM='OBJECT,XREF(SHORT),RENT'
//SYSLIB   DD  DISP=SHR,DSN=SYS1.MACLIB
//         DD  DISP=SHR,DSN=SYS1.MODGEN
//SYSUT1   DD  UNIT=SYSALLDA,SPACE=(CYL,(1Ø,5)),DSN=&SYSUT1
//SYSPUNCH DD  DUMMY
//SYSPRINT DD  SYSOUT=*
//SYSLIN   DD  DISP=(,PASS),UNIT=SYSALLDA,SPACE=(CYL,(5,5,Ø)),       *
//         DCB=(BLKSIZE=4ØØ),DSN=&&LOADSET
//SYSIN    DD  *
TITLE 'ICHRCXØ2 RACHECK POST PROCESSING EXIT       CSH &SYSDATE'
*
* REGISTER USAGE
*
********  CHORNG S. (JACK) HWANG 6/1/98
*         HSA SYSTEMS INC
*         CSHWANG@HOTMAIL.COM
*
*
* R1  - WORK
* R2  - WORK
* R3  - WORK
* R4  - WORK
* R5  - ACEE ADDRESS
* R6  - WORK
* R6  - WORK
* R1Ø - BASE FOR RCXPL
* R11 - BASE FOR WTO
* R12 - BASE FOR CODE
*
ICHRCXØ2 CSECT
STM   14,12,12(13)
LR    12,15
```

```
USING ICHRCXØ2,12
LR    1Ø,1               SAVE ADDRESS OF RCXPL
USING RCXPL,1Ø           ADDRESS RCXPL
*
L     1,RCXRCODE         GET ADDRESS OF RETURN CODE
CLC   2(2,1),=H'4'       RESOURCE NOT DEFINED?
BE    RCOK               YES, CONTINUE PROCESSING
CLC   2(2,1),=H'8'       ACCESS VIOLATION?
BE    RCOK               YES, CONTINUE PROCESSING
B     EXIT               NEITHER, EXIT
*
RCOK     DS    ØH
USING PSA,Ø              ADDR PSA
L     1,PSAAOLD          GET ASCB ADDRESS
USING ASCB,1             ADDR ASCB
L     1,ASCBASXB         GET ASXB ADDRESS
USING ASXB,1             ADDR ASXB
L     5,ASXBSENV         GET ACEE ADDRESS
GETMAIN RU,LV=LOADLL     ACQ LOAD LIST AREA
LR    4,1                ACQUIRED ADDR
LOAD  EP=ICHRCXTB,SF=(E,(4)) LOAD TECH USER TABLE ADDR
LR    2,Ø                SAVE LOADED ADDRESS
LR    7,15               SAVE RETURN CODE
FREEMAIN R,LV=LOADLL,A=(4)
LTR   7,7                TEST LOAD RETURN CODE
BNZ   EXIT               NOT LOADED, FREEMAIN
USERIDLP DS    ØH
CLI   Ø(2),X'Ø7'         TEST LENGTH
BH    EXIT               END REACHED - EXIT
XR    3,3                CLEAR R3
IC    3,Ø(2)             GET LENGTH
EX    3,CLCUID           COMPARE UID
USING ACEE,5             ADDR ACEE
*CLCUID  CLC   1(Ø,2),ACEEUSRI   COMPARE UID
BE    IDOK               ID IS OK, CONTINUE
LA    2,2(3,2)           GO TO NEXT ENTRY
B     USERIDLP
CLCUID   CLC   1(Ø,2),ACEEUSRI   COMPARE UID
DROP  1
*
IDOK     DS    ØH
*
*        TEST FOR RESOURCE NOT DEFINED
*
L     1,RCXRCODE         GET ADDRESS OF RETURN CODE
CLC   2(2,1),=H'4'       RESOURCE NOT DEFINED?
BNE   TESTTAPE           NO, GO CHECK FOR TAPE DATASET
*
```

```
B     CONTINUE          CONTINUE WITH PROCESSING
*
*        TEST FOR TAPE DSN
*
TESTTAPE DS    ØH
L     1,RCXRCODE        GET ADDRESS OF RETURN CODE
         CLC   2(2,1),=H'8'      NOT AUTH?
BNE   EXIT              NO, EXIT EXIT
L     1,RCXFLAG3        GET FLAG3 ADDRESS
TM    Ø(1),RCXDTYPT     DSTYPE=T?
BNO   EXIT              NO, EXIT
L     1,RCXFLAG         GET FLAG3 ADDRESS
TM    Ø(1),RCXLGNOS     LOG=NOFAIL OR NOLOG?
BZ    CONTINUE          NO, CONTINUE
L     1,RCXFLAG2        GET FLAG2 ADDRESS
TM    Ø(1),RCXATTAL     ALTER ATTEMPT?
BO    CONTINUE          NO, CONTINUE
B     EXIT
CONTINUE DS    ØH
*
GETMAIN RU,LV=WTOL      GET WORKAREA
LR    11,1              SAVE WTO WORKAREA ADDRESS
MVC   Ø(WTOL,11),WTO    MOVE WTO MESSAGE
L     1,PSAAOLD         GET ASCB ADDRESS
USING ASCB,1            ADDR ASCB
L     1,ASCBASXB        GET ASXB ADDRESS
USING ASXB,1            ADDR ASXB
L     1,ASXBSENV        GET ACEE ADDRESS
USING ACEE,1            ADDR ACEE
MVC   13(8,11),ACEEUSRI MOVE USERID INTO WTO
DROP  1                 CLEAR ADDRESSING
L     1,RCXENORP        GET PROFILE ADDRESS
MVC   41(44,11),Ø(1)    MOVE PROFILE
L     1,RCXRCODE        GET ADDRESS OF RETURN CODE
*
CLC   2(2,1),=H'4'      RESOURCE NOT DEFINED?
BNE   TAPEDSN           NO, GO MOVE TAPE DSN REQUESTS
MVC   22(18,11),=CL18'SECURITY BYPASS ON'
B     DOWTO             GO DO WTO
*
TAPEDSN  DS    ØH
L     1,RCXFLAG2        GET FLAG2 ADDRESS
TM    Ø(1),RCXATTRE     READ ATTEMPTED?
BNO   NEXT1             NO, NEXT 1
MVC   22(7,11),=CL7'READ' SPECIFY READ
B     DOACCAL
NEXT1    DS    ØH
TM    Ø(1),RCXATTUP     UPDATE ATTEMPTED?
```

```
BNO   NEXT2             NO, NEXT 1
MVC   22(7,11),=CL7'UPDATE' SPECIFY UDPATE
B     DOACCAL
NEXT2    DS    ØH
TM    Ø(1),RCXATTCO     CONTROL ATTEMPTED?
BNO   NEXT3             NO, NEXT 1
MVC   22(7,11),=CL7'CONTROL' SPECIFY CONTROL
B     DOACCAL
NEXT3    DS    ØH
MVC   22(7,11),=CL7'ALTER'   SPECIFY ALTER
DOACCAL  DS    ØH
L     1,RCXACC          GET ACCESS ALLOWED FLAG
TM    Ø(1),RCXNONE      NONE ALLOWED?
BNO   ANEXTØ            NO, NEXT 1
MVC   3Ø(7,11),=CL7'NONE' SPECIFY NONE
B     DOWTO
ANEXTØ   DS    ØH
TM    Ø(1),RCXREAD      READ ALLOWED?
BNO   ANEXT1            NO, NEXT 1
MVC   3Ø(7,11),=CL7'READ' SPECIFY READ
B     DOWTO
ANEXT1   DS    ØH
TM    Ø(1),RCXUPDAT     UPDATE ALLOWED?
BNO   ANEXT2            NO, NEXT 1
MVC   3Ø(7,11),=CL7'UPDATE' SPECIFY UDPATE
B     DOWTO
ANEXT2   DS    ØH
TM    Ø(1),RCXCONTR     CONTROL ALLOWED?
BNO   ANEXT3            NO, NEXT 1
MVC   3Ø(7,11),=CL7'CONTROL' SPECIFY CONTROL
B     DOWTO
ANEXT3   DS    ØH
MVC   3Ø(7,11),=CL7'ALTER' ALTER - THIS SHOULD NEVER HAPPEN
DOWTO    DS    ØH
L     1,RCXRCODE        GET ADDRESS OF RETURN CODE
XC    Ø(4,1),Ø(1)       SET RETURN CODE TO Ø
WTO   MF=(E,(11))       DO THE WTO
FREEMAIN DS    ØH
FREEMAIN R,LV=WTOL,A=(11)
*
EXIT     DS    ØH
LM    14,12,12(13)
SR    15,15
BR    14
*
WTO WTO 'ICHRCXØ2 UUUUUUUU AAAAAAA/ZZZZZZZ ON
PPPPPPPPPPQQQQQQQQQQRRRRRX
```

```
               RRRRRSSSSSSSSSSSTTTT',MF=L
WTOL     EQU   *-WTO
LOADL    LOAD  EP=ICHRCXTB,SF=L
LOADLL   EQU   *-LOADL
LTORG
ICHRCXP
IHAACEE
IHAASCB
IHAASXB
IHAPSA
END
//
//LKED   EXEC  PGM=IEWL,PARM='MAP,LET,LIST,NCAL,AC=1,RENT',
//         COND=(Ø,LE,ASM)
//SYSLIN   DD  DSN=&&LOADSET,DISP=(OLD,DELETE)
//         DD  DDNAME=SYSIN
//SYSUT1   DD  UNIT=SYSALLDA,SPACE=(CYL,(3,2)),DSN=&SYSUT1
//SYSPRINT DD  SYSOUT=*
//SYSLMOD  DD  DISP=SHR,DSN=SYS1.LINKLIB(ICHRCXØ2)
```

ICHRCXTB

```
//ICHRCXTB JOB (1Ø2331Ø),'CSH ICHRCXTB',CLASS=1,MSGCLASS=H,
//  MSGLEVEL=(1,1),NOTIFY=&SYSUID,TIME=144Ø,REGION=8M
//*
//*******************************************************************
//**  AUTHOR:    JACK HWANG                                        **
//**  OBJECTIVE:  TABLE USED BY ICHRCXØ2 TO CHECK FOR VALID IDS    **
//*******************************************************************
//*
//ASM    EXEC  PGM=ASMA9Ø,PARM='OBJECT,XREF(SHORT),RENT'
//SYSLIB   DD  DISP=SHR,DSN=SYS1.MACLIB
//         DD  DISP=SHR,DSN=SYS1.MODGEN
//SYSUT1   DD  UNIT=SYSALLDA,SPACE=(CYL,(1Ø,5)),DSN=&SYSUT1
//SYSPUNCH DD  DUMMY
//SYSPRINT DD  SYSOUT=*
//SYSLIN   DD  DISP=(,PASS),UNIT=SYSALLDA,SPACE=(CYL,(5,5,Ø)),       *
//         DCB=(BLKSIZE=4ØØ),DSN=&&LOADSET
//SYSIN    DD  *
TITLE 'ICHRCXØ2 RACHECK POST PROCESSING EXIT TABLE CSH &SYSDATE'
*
* REGISTER USAGE
*
*******  CHORNG S. (JACK) HWANG 6/1/98
*         HSA SYSTEMS INC
*         CSHWANG@HOTMAIL.COM
*
```

```
ICHRCXTB CSECT
PELIST   DS    ØC
******   LENGTH IS 1 LESS FOR EX COMMAND PURPOSE
DC    AL1(5),CL6'USERØ1'
DC    AL1(5),CL6'USERØ2'
DC    XL8'FF'            END OF LIST
END
//LKED   EXEC  PGM=IEWL,PARM='MAP,LET,LIST,NCAL,AC=1,RENT',
//         COND=(8,LE,ASM)
//SYSLIN   DD  DSN=&&LOADSET,DISP=(OLD,DELETE)
//         DD  DDNAME=SYSIN
//SYSUT1   DD  UNIT=SYSALLDA,SPACE=(CYL,(3,2)),DSN=&SYSUT1
//SYSPRINT DD  SYSOUT=*
//SYSLMOD  DD  DISP=SHR,DSN=SYS1.LINKLIB(ICHRCXTB)
```

Chorng S (Jack) Hwang
HSA Systems (USA)

Information point – reviews

Where else might you go to supplement the kind of information you find in each issue of *RACF Update*? This on-going series of articles explores some of those sources, predominantly – but not exclusively – on the Internet.

SHARE

One non-Internet source is SHARE, best known for its 45 years of technical conferences. The SHARE Web site, at

http://www.share.org

lists all of the sessions at upcoming and recent conferences. Select Conferences from the left sidebar, then the conference you are interested in. From the right sidebar, under Agenda, select Online Version.

You will see some of the technical tracks listed, and can select any of them to see all of the relevant sessions available. Further down the page, you can select sessions by day, session number, or project. But, near the top of the page is the Search the Agenda box that allows you specify one or more keywords separated by AND, OR, or AND NOT.

At March’s conference in Anaheim, a search for RACF listed 14 sessions, while ‘racf OR security’ found 94. At first, a session entitled ‘Selling Ducks on the Web’ may seem off-topic, until you realize what it is about. Security and encryption are part of the discussion of Ducks Unlimited’s System/390-based Web site.

Remember that organizations, not individuals, are members of SHARE. Qualifying organizations must use an IBM computer system or operating system and must send a representative to at least one SHARE conference a year to maintain their membership. Full details are available by clicking on Membership from the left sidebar on the home page, and then on the Become a Member link.

GSE

In the US, GUIDE closed its doors and put its moral support behind SHARE. In Europe, the two merged to form G.U.I.D.E. Share Europe (GSE). GSE’s home page is at http://www.gse.org, and GSE UK has its own home page at http://www.gse.org.uk

GSE UK has a RACF Working Group with its own home page, http://www.gse.org.uk/wg/racf/racfindx.htm.

COMPUTER SECURITY INSTITUTE

Although not as old as SHARE, the Computer Security Institute (CSI) recently completed its first quarter century. CSI’s Web site at http://www.gocsi.com offers both abstracts and full text of selected articles from CSI publications, as well as press releases. Any of the three could appear when you click on a link. But it’s worth the effort given the amount of great material available here.

And it’s not just articles. The CSI Firewall Product Resource link near the top of the home page takes you to a page with links to the Firewall Search Centre and the Firewall Archives. The Search Centre allows you to get information on individual products or to compare products.

WSC

Not to be confused with the other WSC (IBM’s Washington Systems Centre), Washington Systems Consulting specializes in SMS and security. It published a quarterly *WSC Times* for a number of years, and RACF was a frequent topic. Issues are available on-line at http://www.wscinc.com/frm_html/rtim1.htm

The most recent issues are offered complete in Adobe PDF format, requiring the free Adobe reader. Earlier issues offer selected articles for direct viewing on the Web. Unfortunately, you have to look at each issue to determine its contents, but the effort is worth it. The 3Q 1995 edition, for instance, offers only one article for on-line viewing, but that article covers RACF security for hsm.

GARUG

The Atlanta-based Georgia RACF Users' Group (GARUG) has brought together a wealth of useful resources on its home page at http://www.mindspring.com/~ajc10/garug.html

The main menu, near the top of the page, can be easy to miss as it's in the form of a combo box: click on the down arrow to see the possible choices, then select one of them. Probably of greatest interest is the Program Library page. The GARUG library is SAS routines to read RACF and SMF data. Also offered as a zip file, the IRUG tape is a variety of RACF routines collected by other RACF user groups (RUGs). Finally, there's a link to Nigel's Utilities, which will be discussed in a future article in this series.

Although much of the information is in the form of links to Web sites run by others, GARUG even hosts some of the material you wouldn't expect it to, such as the newsletter of a New York-based RACF users' group.

The main menu also includes links to two GARUG-maintained pages on selected RACF and security training and conferences upcoming across the US. But the main menu doesn't provide a path to all of the available information. At the bottom of the home page itself, for example, you will find a link to GARUG's list of other RACF user groups in the US.

CONVERTING TO RACF

Even though its main purpose is to help market CONSUL Risk Management's T2R and A2R automated conversion assistants, the detailed RACF conversion plans from CA-TSS and CA-ACF/2 can be very useful on their own. Even the descriptions of T2R and A2R provide valuable insight into both the scope and functionality involved in doing it all yourself.

http://www.consulrisk.com/services.htm provides links to both conversion plans. Even though it contains only one product-specific document, http://www.consulrisk.com/whitepap.htm is worth a look for its depth of technical discussion on security auditing across multi-platforms.

SNA SERVER

Microsoft Host Integration Server 2000 is the new name for SNA Server, but it's still in beta. If you plan to use SNA Server 4.0, http://www.microsoft.com/sna is the home page. But perhaps the most useful page is found by selecting 'Feature description for SNA Server 4.0 SP3' from the right sidebar, then 'Features at a Glance' from the Section Contents near the top right corner of the page. Its detailed description of SNA Server features includes significant information on security issues in a section towards the bottom of the page titled 'Enterprise Security Integration'.

Near the top of this Features page is a row of links that includes Technical Papers. Although it links to another Microsoft Web page, the documents are written by others. As such, they vary significantly, and only some are white papers. Several cover security issues, such as single sign-on and secure remote access.

Clicking on a link for a technical paper gives you a description anywhere from a short paragraph to almost a full white paper in its own right. The Download link in the upper right hand corner, with a size listed below it, can be clicked to transfer an executable zipped copy of the paper in Microsoft Word format to your workstation. Of the two I tested, one initiated WinZIP, which I have installed on my workstation, while the other ran an embedded unzip-only copy of PKZip from a command line (I don't have PKZip installed).

A useful set of Frequently-Asked Questions (FAQs) is available by selecting Deployment & Support from the left sidebar of any of these pages, then FAQs from a row of links at the top of the page. For example, the last question asks about the bulk migration tool for host security integration feature.

To find non-Microsoft products that can expand the functionality of SNA Server, select 'Product Showcase' from the left sidebar of any of these pages, then '3rd Party Solutions' from a row of links at the top of the page and 'Third Party Resource Guide' from the 'More Resources' section near the top right corner of the page. For example, HALO SSO manages single sign-on between OS/390 and NT.

COUPLING FACILITY

CFSIZER at http://www.s390.ibm.com/cfsizer calculates structure sizes for each IBM systems software product that uses the sysplex Coupling Facility (CF). Select the product, such as RACF, from the left sidebar. A Web page will appear with fields where you need to enter the relevant values that determine the structure size.

Hit the Click Here to Size Structure button, and you'll see a list of CF structures, with function, type, name, and size indicated. The rest of the page includes a sample CFRM policy statement.

IBM

In upcoming issues, we'll cover the many sources of information available from IBM. But if you can't wait, there's one thing worth remembering: RACF is now part of the OS/390 Security Server. So if a search on RACF doesn't produce the results you're looking for, try Security Server instead.

Jon E Pearkins
(Canada)

RACF news

Release 9 of OS/390 supports the new cryptographic capabilities in System/390 G5 and G6 servers. There's also additional support for digital certificates, which lets more users of a Web application access the application with RACF but with less administration.

For further information, contact your local IBM representative, or visit the Web site at http://www.ibm.com

* * *

William Data Systems has previewed Version 1.1 of its FTPalert, promising to overcome the major integrity and control problems that arise when TCP/IP's File Transfer Protocol is used to transfer data to or from OS/390 mainframes.

The OS/390 application interfaces with both TCP/IP and the installation's security facilities, such as RACF. All FTP data transfer activity is reported as it occurs; both successful and unsuccessful FTP data transfers are logged to provide both an audit trail and a record of data transfer statistics; and user authority is checked before file transfers are permitted.

The software provides definitions for RACF and other security access facilities.

For further information, contact:
William Data Systems, 5 High Street, Old Oxted, Surrey RH8 9LN, UK
Tel: (01883) 723 999.
URL: http://www.willdata.com

* * *

LockStar has announced Beta implementations of its end-to-end security software solution for user authentication and data security, focusing on RACF and DB2 support. The solution aims to allow mainframes and other core business resources and applications to use the trust and security of digital certificates and PKI, the *de facto* standard for Internet security.

For further information, contact:
LockStar
1200 Wall Street West. 3rd floor, Lyndhurst, NJ 07071, USA.
Tel: 201 508 3000.
URL: www.lockstar.com

* * *

xephon