32

RACF

May 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.

RACF in focus – facility class

'RACF in focus' is a regular column focusing on specific aspects of RACF. Here, we focus on profiles found in the facility class.

The RACF general resource facility class provides controls for a wide variety of functions and resources in OS/390, and IBM itself uses it to protect a range of functions, from Unix System Services to storage management.

In this sense, therefore, it's different from other general resource classes: while most protect one specific area, or one specific type of resource, this one protects a miscellaneous and assorted group of resources that don't have anything in common, or don't have a home anywhere else.

Although many of the services protected by the facility class are MVS-type resources, conceptually you can have other resources as well. In addition, at your installation, there may even be third-party products using this class to protect their own resources, not to mention home-grown applications.

The following examples illustrate the diversity of protection provided by the facility class. Note that these examples deal only with IBM-provided uses.

STORAGE ADMINISTRATION

Many storage administration functions in OS/390 require RACF protection. These are grouped under the facility class, and all have profiles beginning with STGADMIN. The second qualifier of these profiles further narrows down the specific storage product being protected. Thus,

- STGADMIN.ADR.** profiles protect DFDSS functions
- STGADMIN.IDC.** profiles protect IDCAMS functions
- STGADMIN.IGD.** profiles protect SMS functions
- And so on.

UNIX SYSTEM SERVICES

Unix System Services (previously known as Open Edition MVS) also uses facility class profiles. These all begin with BPX. For example:

- BPX.DAEMON
- BPX.FILEATTR.APF
- BPX.FILEATTR.PROGCTL
- BPX.SUPERUSER.

The profiles provide RACF protection for Unix resources.

RACF

There are some profiles in the facility class that provide protection for RACF functions. These begin with IRR. For example, the profile IRR.LISTUSER specifies who can list – without special powers such as Special or Auditor – the user profiles of other users.

The profile IRR.PASSWORD.RESET enables, say, Help Desk staff to reset user passwords. IBM provided this additional method a few years ago so that the installation didn't have to give the Help Desk staff more powerful attributes such as Special or Group Special.

OTHERS

There are many more. There's a profile in this class that provides tape BLP (Bypass Label Processing) protection. It's called ICHBLP. There are profiles that protect the Distributed Computing Environment (DCE). And so on. IBM keeps adding more profiles in this category as the need arises.

ADMINISTRATION OF FACILITY CLASS PROFILES

RACF administrators attempting to effectively manage these profiles need to understand a number of diverse areas. For

example, it's tricky to ensure that all facility class profiles are properly defined and have the appropriate access. If your installation is decentralized, you may want to consider delegating some of the administration duties to people who are closer to the function or feature that's being protected, and who are therefore more qualified to determine the access requirements. For example, the Unix System Services folk could do the administration work for their set of profiles. This could mean maintaining access lists only. Or it could even include adding or deleting profiles. The Storage Administration group could do the same for profiles related to their area of expertise.

Decentralizing by function in this way will simplify your job. But remember that, even if you choose to delegate, you'll still need to keep tabs on the overall picture by having a general understanding of what these profiles can do. This, together with adequate auditing of changes to these profiles, will help ensure you have this area under control.

DOCUMENTATION

You may wonder where all these profiles are documented. Well, because of their wide use, there is no single place, and you may have to look in several different places. The section entitled 'Planning for Profiles in the Facility Class' in the *RACF Security Administrator's Guide* is a good starting point. You may have to borrow Storage Administration manuals in order to understand STGADMIN profiles and so on.

THIRD-PARTY PRODUCTS

If you're evaluating third-party products, and security for them falls under the facility class, make sure their profiles don't conflict with conventions and standards already in place. A good way to do this is to make sure their product name is part of the profile description.

AUDITING

Auditing of facility profiles can be difficult. One thing you can do

is ensure that there are no runaway general profiles (those that have ** in them) with wide-open access that may inadvertently allow wide-ranging access. For example, it would be inadvisable to have a profile called STGADMIN.** with Universal Access of READ.

SUMMARY

We've touched on many profiles, or groups of profiles, in the facility class, but, because of their very nature, we can't cover them all. IBM may introduce new ones as it sees fit. A careful review and judicious delegation of authority is the best way to provide a secure RACF environment.

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

RACF macros and exits

*This article is intended for systems programmers and software developers who require some basic knowledge of RACF macros and exits. An earlier article (*RACF Update *Issue 31, pp 29-40) looked in detail at RACF control blocks and SAF and RACF customizable modules.*

RACF macros come in two flavours:

- *Independent system macros*, one per RACF service invoked by an SVC call. These types of RACF macro are available only to 24-bit callers. When a resource manager issues one of these macros, they invoke RACF directly and bypass the SAF interface.
- *The RACROUTE REQUEST= macro*, which is available to both 24- and 31-bit callers, and provides greater service and enhance-ments. A RACROUTE request requires a standard register save area of 18 words. Register 13 must point to this save area when issuing the RACROUTE request.

Independent system macros	RACROUTE REQUEST=
RACINIT	VERIFY & VERIFYX
RACHECK	AUTH
FRACHECK	FASTAUTH
RACDEF	DEFINE
RACLIST	LIST
RACXTRT	EXTRACT
RACSTAT	STAT
	AUDIT
	DIRAUTH
	SIGNON
	TOKENBLD
	TOKENMAP
	TOKENXTR

Figure 1: Correspondence between the two macro types

Function	Independent macro	RACROUTE
Authentication		
Create, change, or delete an ACEE. Validate a userid, password, or group	RACINIT	REQUEST=VERIFY
Authorization	RACHECK	REQUEST=AUTH
	FRACHECK	REQUEST=FASTAUTH
	RACDEF	REQUEST=DEFINE
Check RACF status	RACSTAT	REQUEST=STAT
Pre-LOAD resource profile	RACLIST	REQUEST=LIST
Extract or update files in a RACF profile	RACXTRT	REQUEST=EXTRACT
Auditing without a resource check		REQUEST=AUDIT
Create a TOKEN		REQUEST=TOKENBLD
Hash or de-hash a TOKEN		REQUEST=TOKENMAP
Copy TOKEN		REQUEST=TOKENXTR
Mandatory access checking		REQUEST=DIRAUTH
Sign-on processing		REQUEST=SIGNON

Figure 2: Macros by function

Note that it's recommended that the RACROUTE macro be issued to use the SAF int_Bface, instead of the independent macros.

This section deals with just a subset of the RACROUTE macros and parameters that are available, as they are too numerous to cover in this article. Please refer to the *RACF RACROUTE Macro Reference* for more detailed information.

Figure 1 shows the correspondence between the two macro types, while Figure 2 outlines the macros by function.

ICHEINTY ICHEACTN and ICHETEST macros

Another set of RACF macros are the ICHEINTY, ICHEACTN, and ICHETEST macros. These macros allow direct RACF database access but require a thorough knowledge of RACF database templates and database. IBM's recommendation is to use RACXTRT or RACROUTE=EXTRACT, although at times I have found these macros to be very useful and straightforward. Some excellent examples can be found in the *RACF Macros and Interfaces* manual.

SAF router parameter list

For each RACROUTE macro call, a SAF router parameter list (SAFP) is created. The SAFP precedes the RACF service parameter list and contains the offset, not the address, of the RACF service parameter list. The SAFP also contains the RACF return code and reason code and the SAF return code and reason code. The SAFP is normally initialized via

```
RACROUTE REQUEST=,.... MF=L
```

or when a copy of the MF=L is copied to a dynamic work area. The SAFP is mapped by the ICHSAFP macro.

The following code illustrates how to set up the SAFP in a re-entrant Assembler routine:

```
MVC      RACINIT_E,RACINIT_L Copy the RACINIT
RACROUTE REQUEST=VERIFY,
```

```
                              ENVIR=CREATE,
                              RELEASE=77Ø3,
                              WORKA=(R5),
                              MF=(E,RACINIT_E)

RACINIT_L      RACROUTE REQUEST=VERIFY,
                        ENVIR=CREATE,
                        RELEASE=77Ø3,
                        WORKA=*-*,
                        MF=L
RACINIT_LLEN       EQU      *-RACINIT_L

WORK       DSECT
RACINIT_E      DS       XL(RACINIT_LLEN)
```

SAF work area

For each RACROUTE call, SAF requires a work area (SAFW). It is 512 (x'200') bytes long and is set via the RACROUTE WORKA= keyword. Although the SAF work area doesn't have to be initialized, it's good programming practice to do so.

The following code illustrates how to set up the SAFW in a re-entrant Assembler routine:

```
            MVC      RACINIT_E,RACINIT_L       Copy the RACINIT
            RACROUTE REQUEST=VERIFY,
                        ENVIR=CREATE,
                        RELEASE=77Ø3,
                        WORKA=SAFWA,        <========
                        MF=(E,RACINIT_E)

RACINIT_L      RACROUTE REQUEST=VERIFY,
                        ENVIR=CREATE,
                        RELEASE=77Ø3,
                        WORKA=*-*,   <========
                        MF=L
RACINIT_LLEN       EQU      *-RACINIT_L

WORK       DSECT
RACINIT_E      DS       XL(RACINIT_LLEN)
SAFWA       DS       XL512
```

Note that on the MF=E execute form of the RACROUTE parameter, the WORKA= parameter must be specified or a 0C4 abend will occur. It's not sufficient to have the WORKA= on the MF=L forms.

Determine RACF status

One extremely useful RACROUTE macro is the RACROUTE REQUEST=STAT macro. This determines whether RACF is active, and will optionally determine whether a given resource class is defined to RACF. If a resource class is defined to RACF, the macro also determines whether the class is active.

```
         L           R8,16(Ø)          CVT @
         USING     CVTMAP,R8           @ The CVT
         L           R9,CVTRAC         @ Of the RACF CVT
         USING     RCVT,R9             @ The RCVT
         L           R7,RCVTCDTL       Length for CNST + CNSX Entry
         GETMAIN     RU,LV=(7)         Get the Required Storage
         LR           R6,R1            Storage Address
         RACROUTE     REQUEST=STAT,
                     CLASS='MYCLASS',
                     COPY=(R6),
                     COPYLEN=(R7),
                    WORKA=SAFWORK,
                    RELEASE=77Ø3,
                    MF=S
*
         USING  CNST,R2                Copy Area
         USING  CNSX,R3                @ CNSX
                     .............
         ICHPCNST                      Class Descriptor Table
```

Note that the RCVTCDTL field of the RCVT contains the length needed to hold both a CNST and a CNSX entry.

Function	RACROUTE parameter
To suppress messages	MSGSUPP=
To capture messages	MSGRTRN= The address of the ICH408I message is placed in the router parameter list.
To control the subpool used for capturing messages	MSGSP=

Figure 3: Parameters available on RACROUTE macro to control messages

RACROUTE RELEASE= Value

Always ensure that the release number specified on the RACROUTE RELEASE=..,MF=L macro creates a parmlist area large enough to handle MF=M and MF=E parameters.

RACROUTE message processing

There are a number of parameters on the RACROUTE macro to control message processing (see Figure 3), for example to:

- Suppress any message
- Suppress messages and issue your own message
- Store and then issue a message using SVC34
- Extract information from messages.

Message processing applies only to the following RACROUTE calls:

- REQUEST= VERIFY (RACINIT)
- REQUEST= AUTH (RACHECK)
- REQUEST= DEFINE (RACDEF)

The following example illustrates addressability to the RACF ICH408I message:

```
*-------------------------------------------------------------------*
        MVC      RACINIT_E,RACINIT_L    Copy the RACINIT
        RACROUTE   REQUEST=AUTH,
               ENTITY=(R6),
               CLASS=(R7),
               WORKA=(SAFWA),
               RELEASE=77Ø3,
               MSGRTRN=YES,
               MSGSUPP=YES,
               MSGSP=78,
               MF=(E,RACAUTH_E)
        LTR    R15,R15          RACROUTE OK?
        BZ     AUTHOK           y,
        LA     R4, RACAUTH_E    Pointer to router's parameter list
        USING  SAFP,R           Addressability to the SAF Parameter List
        ICM    R3,B'1111',SAFPMSAD  @ of the ICH4Ø8I Message
        BZ     NO_MESS
```

```
PROC_MSG  DS   ØH
*-------------------------------------------------------------------*
*  SAFPMSAD points to an area that contains                         *
*                                                                   *
*  First word, length of the message buffer                         *
*  Second word, address of the next message or zero                 *
*  Beginning in the third word, the message in the form of a WTO list. *
*                                                                   *
*-------------------------------------------------------------------*
            ..............
NO_MESS     DS            ØH
            ..............
AUTHOK      DS            ØH
            ..............
RETURN      DS            ØH
            .............
*-------------------------------------------------------------------*
*                      Work Areas                                   *
*-------------------------------------------------------------------*
RACAUTH_L            RACROUTE REQUEST=AUTH,
                        CLASS=*-*,
                        ENTITY=*-*,
                        RELEASE=77Ø3,
                        WORKA=*-*,
                        MF=L
RACAUTH_LLEN      EQU      *-RACAUTH_L
*
WORK       DSECT
RACINIT_E  DS      XL(RACAUTH_LLEN)
SAFWA      DS      XL512

*-------------------------------------------------------------------*
*                      Mapping Macros                               *
*-------------------------------------------------------------------*
            ICHSAFP
```

RACROUTE parameter differences

When coding RACROUTE parameters, watch out for the following coding differences between requests:

- A CLASS parameter needs a length for the REQUST=AUTH, but must not have one for REQUEST=FASTAUTH.
- A USERID parameter needs a length field for a REQUEST=VERIFY, but must not have one for REQUEST=AUTH.

- The ACEE parameter for REQUEST=VERIFY points to a fullword that contains the address of the ACEE, but REQUEST=AUTH has the ACEE parameter pointing directly to the ACEE.

RACROUTE recommendations

IBM recommends using the ENTITYX rather than ENTITY. For a full explanation for this recommendation, please refer to the *RACF RACROUTE Macro Reference* manual.

RACROUTE return codes

RACROUTE stores the RACF return and reason code in the first two words of the SAF work area. The ICHSAFP mapping macro is required to access the return codes.

- SAFPRRET – RACF return code
- SAFPRREA – RACF reason code.

When processing returns from the RACROUTE call, the return codes can be processed as follows:

- R15 contains the SAF (RACROUTE) return code.
- The RACF return and reason codes can be found in the first two words of the SAF parameter list.
- The SAF return code can also be found in the SAF parameter list.

Note that the meaning of the SAF and RACF return and reason codes is not the same for all RACROUTE calls.

RACROUTE examples

The following example validates a userid and password:

```
RACROUTE    REQUEST=VERIFY,
            ENVIR=CREATE,
            USER=USERID,
            PASSWRD=PASSWORD,
            WORKA=(R5),
```

```
                    APPL=APPLID,
                    RELEASE=77Ø3,
                    STAT=ASIS,
                    LOG=ASIS,
                    MF=(E,RACINIT_L)

USERID      DC      AL1(7),CL7'USERØØ1'
PASSWORD    DC      AL1(7),CL7'LETITBE'
APPLID      DC      CL8'APPL'
```

The following example validates a userid with no password check:

```
            RACROUTE    REQUEST=VERIFY,
                        ENVIR=CREATE,
                        USER=USERID,
                        PASSCHK=NO,
                        WORKA=(R5),
                        APPL=APPLID,
                        RELEASE=77Ø3,
                        STAT=-NONE,
                        LOG=NONE,
                        MF=(E,RACINIT_L)
```

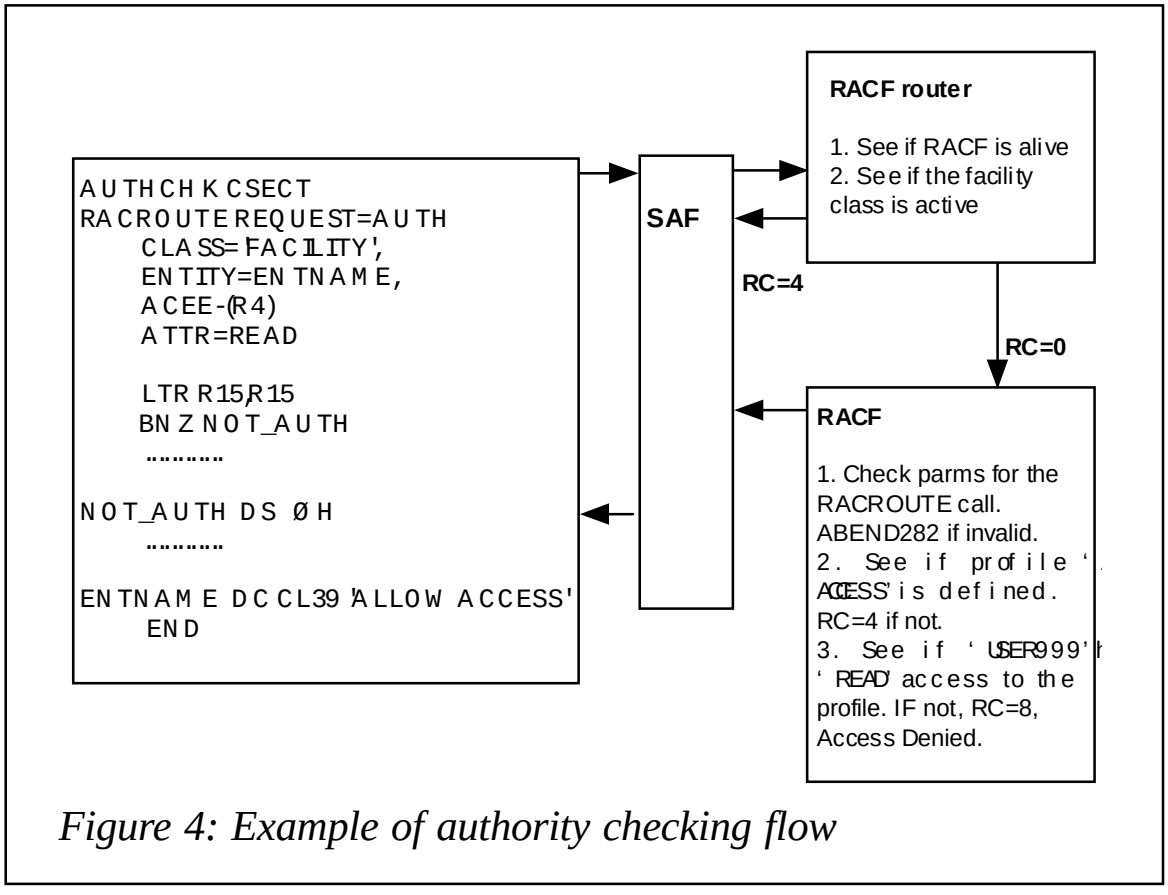

Figure 4: Example of authority checking flow

```
USERID      DC      AL1(7),CL7'USERØØ1'
APPLID      DC      CL8'APPL'
```

The following example invokes a RACF authorization check:

```
            L           R4,ASXBSENV           ACEE
            RACROUTE      REQUEST=AUTH,
                        CLASS=FAC,
                        ENTITY=ENT,
                        ACEE-(R4),
                        WORKA=(R5),
                        RELEASE=77Ø3,
                        MF=(E,RACAUTH_L)

FAC           DC      AL(L'FACILITY)
FACILITY      DC      CL8'FACILITY'
ENT           DC      AL(L'MYENT)
MYENT      DC       CL16'MY.FACILITY.TEST'
```

Example of authority checking flow

An example of authority checking flow is shown in Figure 4. The program AUTHCHK calls RACF through the SAF interface to

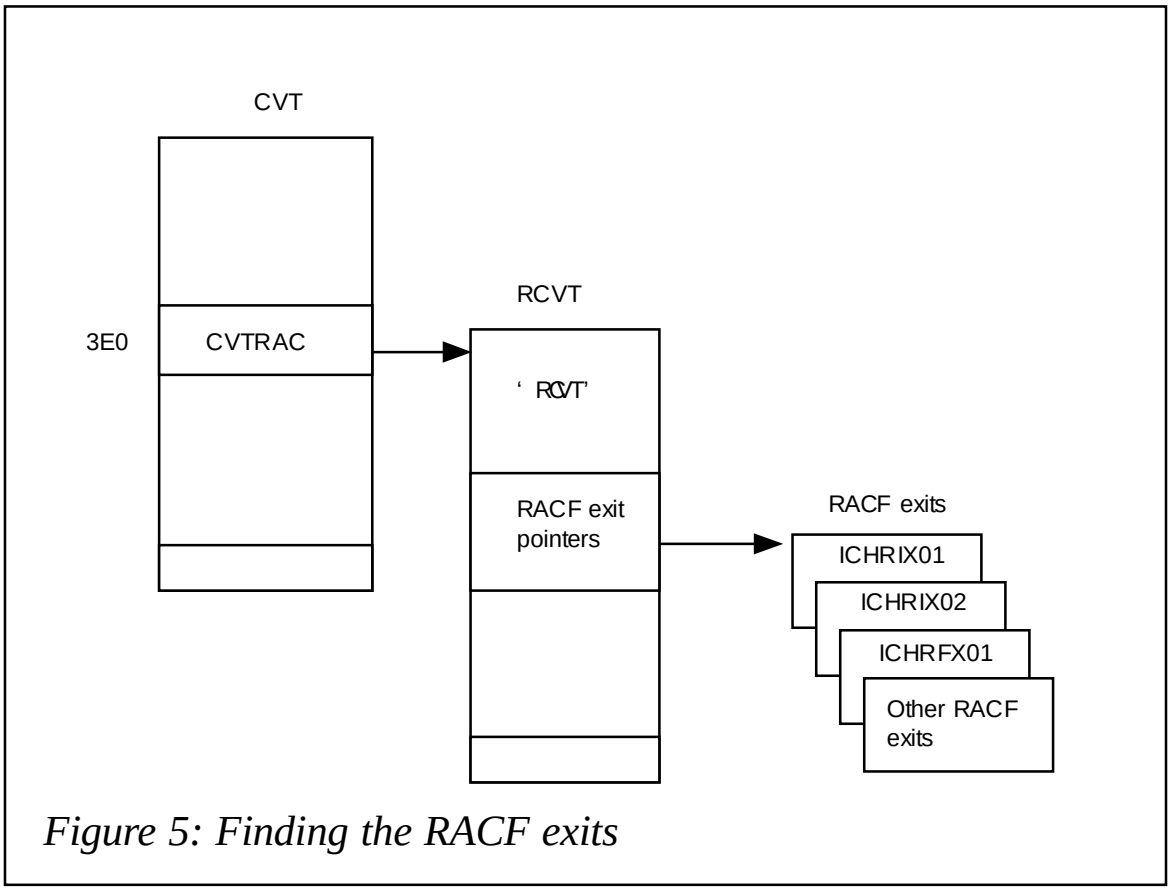

Figure 5: Finding the RACF exits

Exit	RACROUTE REQUEST=	Environment	Exit function
ICHRIX01 ICHRIX02	VERIFY VERIFYX	RACINIT	Determine that a userid and password are correct. The ICHRIX01 exit is called from one of the following environments: • LOGON • JES VERIFYX processing • Job start • Job end • CICS CSSN.
ICHRCX01 ICHRCX02	AUTH	RACHECK	Check that a user is authorized to a resource. The ICHRCX01 exit is called every time RACHECK is called. For example, ICHRCX01 could prevent unauthorized access for the RACF commands ALTDSD, PERMIT, and/or RALTER.
ICHRFX01 ICHRFX02	FASTAUTH	FRACHECK	Check authorization for RACLISTED-created profiles.
ICHRDX01 ICHRDX02	DEFINE	RACDEF	Define, change, or delete profiles. The RACDEF exit gets control when the RACDEF SVC is called, when ALLOCATE creates new datasets, and when RACDEF not RACHECK is called to determine whether a resource is RACF protected.
ICHPWX01		RACINIT ALTUSER PASSWORD	New password processing exit. This exit can be used to enforce installation standards for passwords. The exit is called from: • RACINIT • ALTUSER command • PASSWORD command.
ICHRLX01 ICHRLX02	LIST	RACLIST	Build in-storage copies of resource profiles.

Figure 6: Pre-processing and post-processing RACF exits

see whether USER999 can access a sensitive resource or sensitive KEY 0 storage.

RACF EXITS

RACF provides a number of exits that allow an installation to meet special requirements without making direct modifications to RACF. RACF exits are provided with the following RACF functions:

- User identification
- User verification
- Automatic profile creation and deletion
- Resource command manipulation
- User command manipulation
- Group profile command manipulation.

RACF exits must be located in the LPA and must be re-entrant and refreshable. If an exit is modified, the system must be re-IPL'd for the modifications to take effect. RACF currently has no mechanism for enabling or disabling exits. Several RACF exits have a pre-processing and a post-processing entry, which provides examination of the input before RACF processes it, and after RACF has finished, so that the exit can override the RACF result.

During RACF initialization, RACF loads the exit routines and places the address of each exit routine in the RACF Communication Vector Table (RCVT).

Figure 5 shows how to find the RACF exits, while Figure 6 lists the most-often-used pre-processing and post-processing RACF exits.

RACF exit flow

When a RACROUTE REQUEST=VERIFY macro is issued, the SAF interface module ICHSFR00 will get control. It will then pass

control to the RACF router module ICHRFR00, which will in turn pass control to the RACINIT RACF module. The RACINIT module will pass control to the RACINIT exits ICHRIX01 and ICHRIX02.

RACF exit considerations

An important consideration when coding RACF exits is not to use a WAIT macro or call a routine that uses the WAIT macro. Any code that has a WAIT macro may cause the issuing task to stop functioning.

R F Perretta
Millenium Computer Consultancy (UK)

Validating RACF information

Information in the RACF database, such as userids, groups, and access lists, needs to be reviewed periodically. This article describes a review process that will eliminate many of the concerns of security managers and internal auditors. The process involves preparing reports that are sent out to appropriate individuals, in most cases managers and supervisors, for review and validation. Once the information is validated, any changes required are carried out by the security administrators.

Note that the reporting methods described here are based on the assumption that your site has no RACF add-on products; note also that the reporting capabilities of these products may simplify some of the reporting required for this project.

WHY VALIDATE RACF INFORMATION?

During the normal business day, security administrators are busy performing the normal functions of granting and removing security access and don't have time to do any clean-up activities. At the same time, management may be looking at the bottom line of getting the work done with as few resources as possible. This can lead to several undesirable scenarios, not to mention security exposures.

Here are the main reasons why RACF information needs to be validated:

- *Obsolete userids*. Once-valid userids defined in the RACF database may now be obsolete because the person has left the company, or because the person doesn't need a userid anymore. This can happen even when there's a mechanism in place for the security department to be notified of employee terminations, transfers, and resignations. Usually, in a busy day, a security administrator's priority is to grant new accesses and create new userids, not delete obsolete ones. New userids that aren't created will draw immediate attention,

whereas old userids not deleted will seldom be noticed!

- *Obsolete groups*. RACF groups created at one time may no longer be required, either because the company has been re-organized, or because the special reason the group was created no longer applies.

- *Obsolete profiles*. Entire RACF profiles can become obsolete over time. This can happen when new dataset naming standards are introduced; datasets and their underlying RACF profiles may no longer be required because the project they belonged to has been cancelled or has been completed; or software products may have been retired from use. Yet another instance is when profiles may have started out more granular, but can now be simplified.

  An example of this situation is as follows. Imagine that the following profiles exist for the PAYROLL application:

  1 PAYROLL.PROJ1.**

  2 PAYROLL.PROD.DEPT1.**

  3 PAYROLL.PROD.**

  4 PAYROLL.PRD1.**

  5 PAYROLL.JAN11.SNAP.**

  This level of granularity may not be required any more, and profile numbers 2 and 4 may have become obsolete.

  CICS transactions defined to RACF can also become obsolete when CICS regions are retired for whatever reason.

  In all these cases, the underlying RACF profiles become obsolete. But they may not have been cleaned up, either because no-one in the security department was notified, or because things ‘fell through the cracks’ because of on-going administration activities.

- *Inappropriate access*. Access specified as per access lists in RACF profiles may not be valid any more. Access may

have been granted at one point – with due approvals, no doubt – to some sensitive data because it was needed for a specific reason, and for a specific time only. This may not be required any more because the person no longer performs that function. Alternatively, the person may now have moved to a new position in the company. Unless you intervene, this situation can introduce serious security exposures.

A periodic review of the contents of the RACF database also offers the opportunity to consolidate some of the profiles and access lists so that the information is specified in a simple, concise manner. For example, imagine that several individuals belonging to the same group have been granted access to a profile. It may now make more sense, still keeping security in mind of course, to provide access at the group level. This would replace several individual entries in the access list with a single entry for the group. This will reduce the security administration effort on an on-going basis.

An indirect benefit of this project is that it will help better identify and validate owners and reviewers for various pieces of RACF information. Inevitably, some reports will come back marked 'not my area'! For these, identifying correct owners will be helpful on an on-going basis, when the time comes for granting access and receiving approvals.

STAFF RESOURCES REQUIRED

The security administrator will play an important role, both in defining the reporting requirements and in performing the additional security clean-up activities that such projects inevitably generate.

A clerical person or a junior security administrator will be needed. This person will do the numerous mailings and track the progress of reports returned, send out follow-up mail, and so on.

Most importantly, someone with report writing skills will be required to produce the necessary reports.

REVIEWERS AND OWNERS/CUSTODIANS

Most installations already have lists that specify owners or custodians for RACF groups and profiles. These people would probably be managers, supervisors, or project leaders in different areas of the company. If no list of this kind exists, it's a good idea to develop one, to assign review and ownership responsibility. These lists also form the basis for approvals whenever on-going security changes are requested. The lists, of course, require periodic maintenance to reflect changed owners, and so on.

RACF INFORMATION VALIDATION PROCESS

The RACF information validation process involves listing all userids, groups, and profiles in the RACF database. These lists are then converted into rudimentary reports, and sent out to the appropriate owners with a cover letter stating the reason for the communication, the action required from them, and the date by which the report must be returned to the security department. A sample cover letter is shown in Appendix 1.

Any changes to the information must be duly signed by the owner or reviewer. It's the job of the security administrator to make sure the letters are sent out, to receive the replies, and to follow up if no response is received by the target date.

The following reports can be generated using the SAS programming language, or any other language at the disposal of the report writer. In some cases, ISPF edit functions will suffice to produce a presentable report.

The userid report

The userid report lists each personal (as opposed to functional or batch) userid, together with information such as the userid's special attributes (operations, special, audit) and all the groups to which the userid belongs. These reports are then sent out to the managers to whom the individuals report. The managers need to verify that all the userids are valid, that they work for them, and that the special attributes, if any, are still valid. If there

are any changes, these should be marked on the reports and returned to the security administrator.

The userid report can be created by listing all userids in batch and 'messaging' the data using either ISPF edit or some programming language.

The RACF command is:

```
LISTUSER *
```

Make sure you have sufficient RACF authority to list all users.

A sample userid report is shown in Appendix 2.

The group report

For each RACF group, a report is generated to show which userids belong to that group. The 'owner' of each group verifies the accuracy of the userids belonging to their respective groups.

One way to create the group report is to use the RACF unload database as input to a program. Another way is to list all groups in batch and input this list to a program.

The RACF command is:

```
LISTGRP  *
```

Make sure you have sufficient RACF authority to list all groups.

A sample group report is shown in Appendix 3.

The dataset profile report

For each dataset profile, the dataset profile report identifies the groups and userids in its access list, the level of access (none, execute, read, update, alter), and general information related to the profile, such as Universal Access and logging information for the profile.

The report can be created by listing all dataset profiles and 'messaging the listing', using ISPF editor, or by extracting the dataset profile information from the RACF unload database using a program.

A sample dataset profile report is shown in Appendix 4.

The resource profile report

The resource profile report is very similar to the dataset profile report, the major difference being that the resource class to which the profile belongs is also shown.

A sample resource profile report is shown in Appendix 5.

The started procedures report

The started procedures report will probably be a single report, to be reviewed by a single person such as the resident system programmer. In some installations, the report may need to be reviewed by more than one person. It will identify all started tasks defined to RACF. The special attributes granted to each started procedure need to be listed and validated. In addition, since started procedures can create security exposures, they should be reviewed for their existence in the first place (that is, do we still require this started procedure?).

The started procedures report can be produced by listing all members in the STARTED class.

A sample started procedures report is shown in Appendix 6.

The batch userid report

The batch userid report is similar to the userid report, except that these are batch userids, used by batch jobs (test, development, and production). Since they may have wide-ranging security powers, and since some of them handle critical production batch processes, they need to be tightly controlled. The in-house scheduler package often uses these userids to submit batch jobs.

In most installations, they will be reviewed by the Manager of the Production Support Department.

A sample batch userid report is shown in Appendix 7.

SENDING OUT THE REPORTS AND TRACKING PROGRESS

It's extremely important to track the progress of the reports. Inevitably, not all reports will come back duly signed, and some may require follow-up action. Without proper tracking, the security administrator will lose sight of which reports have come back and which have not. Also, as reports come in, the security administrator needs to make the changes indicated by the reviewers.

Ideally, the various reports should not be sent out all at once, so as not to inundate the reviewers (and security administrators) with tons of paperwork. Security administrators should also be prepared to answer any questions from the report reviewers.

HOW OFTEN TO REVIEW

How often you review depends on factors such as company policy, the nature of the company's business, the resources available for such a project, and the pressure exerted by the internal auditors. In most cases, an annual review should be adequate.

CONCLUSION

Once this project is completed, the information contained in the RACF database will more accurately reflect the organization's requirements. The information will be accurate for a while, and gradually inaccuracies will creep in. That's when it's time to do another review!

Appendices

APPENDIX 1 – SAMPLE COVER LETTER

This is a sample cover letter, to be sent out with all reports. Since there's just one cover letter for all reports, the security administrator

will need to circle the number to which the letter applies.

To: Manager
Payroll Department 14 May 2003
Re: RACF Information Validation Project
Information Security Department is conducting its yearly review and validation of all information contained in the RACF database. Attached please find:
1. The Userid Reports for users that belong in your department.
2. The Group Reports for RACF groups for which you are responsible.
3. The Dataset Profile Reports for dataset profiles under your ownership.
4. The Resource Profile Reports for resource profiles under your ownership.
5. The Started Procedures Report for which you are responsible.
6. The Batch Userid Report for batch userids for which you are responsible.
Please review these reports, and return them to the security department by: 20 June 2003. Your co-operation in this matter will be very much appreciated.

APPENDIX 2 – SAMPLE USERID REPORT

RACF Information Validation Project: Userid Report
14 May 2003

Userid: USER099 Name: John Doe
Special Attributes: OPERATIONS, GROUP-SPECIAL
Default Group: PAYROLL
Other Groups Connected to: ACCT, SYSPROG

APPENDIX 3 – SAMPLE GROUP REPORT

RACF Information Validation Project: Group Report
14 May 2003

Group Name: PAYROLL
Superior Group: ADMIN Owner: PAYMGR
Userids belonging to this group:
USER090
USER123
.....

APPENDIX 4 – SAMPLE DATASET PROFILE REPORT

RACF Information Validation Project: Dataset Profile Report
14 May 2003

Dataset Profile: PAYROLL.*.NEW.**
Universal Access: READ
Logging Options: Log all updates
Access List:

Group/Userid	Access Level
PAYROLL	UPDATE
USER300	READ
USER001	NONE

APPENDIX 5 – SAMPLE RESOURCE PROFILE REPORT

RACF Information Validation Project: Resource Profile Report
14 May 2003

Resource Class: CICSTRN1 Profile Name: TRN1
Universal Access: NONE
Logging Options: No Logging
Access List:

Group/Userid	Access Level
PAYROLL	READ
USER300	UPDATE
USER001	ALTER

APPENDIX 6 – SAMPLE STARTED PROCEDURES REPORT

RACF Information Validation Project: Started Procedures Report
14 May 2003

Started Procedure Name:	Special Attributes:
NETPN	OPERATIONS
SCHEDPRD	NONE
DFHSM	OPERATIONS
.....	.....

APPENDIX 7 – SAMPLE BATCH USERID REPORT

RACF Information Validation Project: Batch Userid Report

14 May 2003

Batch Userid: PAYPROD Name: Production Payroll Batch

Special Attributes: GROUP-OPERATIONS

Default Group: PAYROLL Other Groups Connected to: FINANCE

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

Displaying the status of RACF userids

This article follows on from the previous one in this issue (pp 18-26), and looks specifically at ways in which common userid maintenance tasks can be accomplished. The program presented provides a comprehensive userid status report to help you keep your RACF database properly maintained.

Maintaining the contents of a RACF database is a constant challenge. Some common userid maintenance tasks include:

- Adding a userid for a new employee.
- Deleting a userid (and other access related data) for a departing employee.
- Suspending a userid for an employee on temporary leave.
- Creating a short-term userid for a consultant or an employee temporarily assigned to another job function.
- Altering userid access capabilities because an employee has changed departments.

But what happens to your RACF database if the employee on temporary leave doesn't return to active employment? What happens to the userid for a departing employee that was revoked when they left, but wasn't immediately deleted? What happens ...?

Over time, it's not hard to see how some things could get lost in the shuffle. It would clearly be helpful to be able to get a comprehensive userid status report on a regular basis so that a cross-check could be made against a userid and its assigned access capabilities. The program presented provides at least some of this useful information.

The USERLIST program reads through an entire RACF database and provides an output record for each defined userid that indicates the following:

- The userid (eg JSMITH).
- The user name (eg John Smith).
- The assigned attributes (eg ADSP, SPEC, OPER, GRPACC, AUD, PROT, OID).
- The password expiry status (eg CURRENTLY EXPIRED, NOT REQUIRED, EXPIRES IN *nn* DAYS, NEVER EXPIRES).
- The userid revoke status (eg REVOKED, NOT REVOKED).
- The defined RACF segments (eg CI, DC, DF, LA, NV, OM, OP, OV, TS, WA).

The information in this report allows a RACF administrator to ask questions like the following:

- Why does a userid have a defined TSO segment, but no password requirement?
- Should a userid with a CICS or TSO segment have a password that never expires?
- Should a particular userid have AUDIT, OPERATOR, or SPECIAL attributes?
- That person left months ago – why is the userid still in the database?

CREATING THE LOAD MODULE

The USERLIST program needs to be assembled with your Assembler H or High Level Assembler. The resulting object module should be linkedited using the following job:

```
//jobname  JOB  ...
//IEWL     EXEC PGM=HEWLHØ96,PARM='XREF,LIST,MAP'
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   UNIT=SYSDA,SPACE=(CYL,(2,1))
//OBJECT   DD   DSN=your.object.pds,DISP=SHR
//SYSLMOD  DD   DSN=an.auth.dataset,DISP=SHR
//SYSLIN   DD   *
   INCLUDE OBJECT(USERLIST)
   ENTRY   USERLIST
   SETCODE AC(1)
   NAME    USERLIST(R)
```

Note the requirement for APF authorization.

RUNNING THE USERLIST UTILITY

Once you've assembled and linked the USERLIST utility, you're ready to display the status of the userids defined to your RACF database. The following sample job can be used to run the USERLIST utility:

```
//jobname  JOB  ...
//USERLIST EXEC PGM=USERLIST
//STEPLIB  DD   DSN=an.auth.dataset,DISP=SHR
//SYSPRINT DD   SYSOUT=*
```

The USERLIST program can run in two modes. It can be used to list every userid defined to the RACF database. This is accomplished by specifying no program parameter, as in the above sample job. It can also be used to list only those userids that belong to a particular RACF group. You would do this by specifying a group name parameter value. For example:

```
//jobname  JOB  ...
//USERLIST EXEC PGM=USERLIST,PARM='GROUPNAME(SYS1)'
//STEPLIB  DD   DSN=an.auth.dataset,DISP=SHR
//SYSPRINT DD   SYSOUT=*
```

In the above example, USERLIST would produce a report for all userids connected to the SYS1 RACF group.

The USERLIST utility will produce output in the SYSPRINT DD. Example output will look something like the following:

```
Userid  User Name  Attributes PWD Expiry Status  Revoke Status Defined Segments
------- ---------- ---------- ----------------  ------------ --------------
AUDIT1  AUDITOR 1   AUD       NEVER EXPIRES      NOT REVOKED  TS
OPER1   OPERATOR 1  OPER      EXPIRES IN 45 DAYS NOT REVOKED  CI,NV,OP,TS
OPER2   OPERATOR 2  OPER      CURRENTLY EXPIRED  REVOKED      CI,NV,OP,TS
TEST1   TEST USER1  OPER,SPEC EXPIRES IN 15 DAYS NOT REVOKED  CI,OM,TS
TEST2   TEST USER2  NONE      CURRENTLY EXPIRED  NOT REVOKED  CI,DC,DF,OM,OP+
```

USERLIST will check for the existence of all RACF segments except the KERBEROS segment, the Lotus Notes segment, and the NDS segment. As space is limited in the output record, if a userid has more segment definitions than space allows, a plus sign (+) will be displayed at the end of the defined segment list for that userid.

The USERLIST utility can be run on a regular basis to make sure that RACF userid maintenance is not falling through the cracks.

USERLIST

```
USERLIST CSECT
USERLIST AMODE 31
USERLIST RMODE 24                        DCB'S NEED 24-BIT ADDRESSES
*********************************************************************
*                                                                   *
*   The USERLIST utility can be used to list all userids defined to *
*   your racf database.  Optionally, it can be used to list only    *
*   those userids that belong to the group specified in the         *
*   GROUPNAME program parm.                                         *
*                                                                   *
*   USERLIST directs its output to the SYSPRINT DD which has DCB    *
*   characteristics of LRECL=133, DSORG=PS, RECFM=F(B),             *
*   BLKSIZE=multiple of 133.                                        *
*                                                                   *
*   USERLIST should be linkedited into an authorized library with   *
*   AC(1).                                                          *
*                                                                   *
*   USERLIST can end with the following return codes:               *
*                                                                   *
*    Ø:  USERID LIST HAS BEEN CREATED                               *
*    4:  THE RCVT FOR RACF WAS NOT LOCATED                          *
*    8:  THE SPECIFIED GROUPNAME IS NOT DEFINED TO RACF             *
*   12:  A PROGRAM PARM WAS DETECTED BUT NOT RECOGNIZED AS          *
*        GROUPNAME(grpname)                                         *
```

```
*   16:  A PROGRAM PARM WAS DETECTED BUT THE LENGTH DID NOT          *
*        CORRESPOND TO A VALID GROUPNAME(grpname) PARAMETER          *
*                                                                    *
**********************************************************************
         PRINT GEN
         STM   R14,R12,12(R13)        SAVE THE REGISTERS
         LR    R12,R15                COPY MODULE BASE ADDRESS
         LA    R11,4095(,R12)         SET SECOND BASE ...
         LA    R11,1(,R11)               REGISTER ADDRESS
         USING USERLIST,R12,R11       SET ADDRESSABILITY
         LR    R10,R13                SAVE OLD SAVEAREA ADDRESS
         LR    R2,R1                  SAVE INCOMING PARM ADDRESS
         STORAGE OBTAIN,LENGTH=WALEN  GET SOME WORKING STORAGE
         LR    R13,R1                 COPY THE ADDRESS
         LR    R0,R13                 AGAIN
         L     R1,=A(WALEN)           SET THE LENGTH
         LR    R14,R13                SET SOURCE ADDRESS TO TARGET
         XR    R15,R15                SET FILL BYTE
         MVCL  R0,R14                 CLEAR THE STORAGE
         ST    R10,4(,R13)            SAVE OLD SAVEAREA ADDRESS
         USING WORKAREA,R13           WORKING STORAGE ADDRESSABILITY
**********************************************************************
         L     R1,16                  GET CVT ADDRESS
         USING CVT,R1                 SET ADDRESSABILITY
         L     R6,CVTRAC              GET RCVT ADDRESS
         USING RCVT,R6                SET ADDRESSABILITY
         CLC   RCVTID(4),=C'RCVT'     RACF?
         BNE   RETURN4                NO - RETURN
         MVC   SAVEPINV(1),RCVTPINV   SAVE THE RACF PASSWORD INTERVAL
         DROP  R1,R6
         LR    R1,R2                  RELOAD INCOMING PARM ADDRESS
**********************************************************************
*                                                                    *
*   EXTRACT INCOMING PARM DATA.  VALID PARM DATA IS AS FOLLOWS:      *
*                                                                    *
*   - NO PARAMETER                                                   *
*     - LIST ALL USERIDS ALPHABETICALLY (INCLUDING USER NAME)        *
*     - FOR EXAMPLE:                                                 *
*       //USERLIST EXEC PGM=USERLIST                                 *
*       //STEPLIB  DD   DSN=authorized.load.library,DISP=SHR         *
*       //SYSPRINT DD   SYSOUT=*                                     *
*                                                                    *
*   - 'GROUPNAME(GRPNAME)'                                           *
*     - LIST ALL USERIDS (INCLUDING USER NAME) THAT ARE DEFINED      *
*       TO GROUP 'GRPNAME'                                           *
*     - FOR EXAMPLE:                                                 *
*       //USERLIST EXEC PGM=USERLIST,PARM='GROUPNAME(MVSGRP)'        *
*       //STEPLIB  DD   DSN=authorized.load.library,DISP=SHR         *
*       //SYSPRINT DD   SYSOUT=*                                     *
*                                                                    *
```

```
*********************************************************************
         LR    R8,R1                    SAVE PARM ADDRESS
         L     R9,Ø(,R8)                GET ADDRESS OF PARM
         CLC   Ø(2,R9),=H'Ø'            ANY PARAMETER?
         BE    FULLLIST                 NO - DO A FULL LIST
         CLC   Ø(2,R9),=H'12'           IS TOTAL LENGTH OK?
         BL    RETURN16                 NO - SET RETURN CODE & EXIT
         CLC   Ø(2,R9),=H'19'           IS TOTAL LENGTH OK?
         BH    RETURN16                 NO - SET RETURN CODE & EXIT
         XR    R7,R7                    CLEAR COUNT REGISTER
         ICM   R7,B'ØØ11',Ø(R9)         SAVE THE LENGTH
         LA    R9,2(,R9)                POINT TO DATA
         LA    R8,Ø(R7,R9)              POINT PAST DATA
         BCTR  R8,Ø                     POINT TO LAST BYTE
         CLC   Ø(1Ø,R9),=C'GROUPNAME('  GROUPNAME PARAMETER?
         BNE   RETURN12                 NO - RETURN
         CLI   Ø(R8),C')'               PROPER FORMAT?
         BNE   RETURN12                 NO - RETURN
         S     R7,=F'11'                SUBTRACT PARAPHANALIA LENGTH
         STH   R7,GRPNMLEN              SAVE LENGTH
         BCTR  R7,Ø                     REDUCE BY ONE FOR EX
         MVC   GRPNAME(8),=8C' '        INITIALIZE GROUP NAME FIELD
         LA    R9,1Ø(,R9)               POINT TO GROUP NAME
         EX    R7,GRPNMMVC              SAVE THE GROUP NAME
         B     GRPLIST                  LIST USERIDS IN GROUP
FULLLIST EQU   *
*********************************************************************
*                                                                   *
*   NO PROGRAM PARM HAS BEEN DETECTED SO WE WILL PERFORM A SEARCH   *
*   FOR EACH USERID DEFINED TO THE RACF DATABASE.                   *
*                                                                   *
*********************************************************************
         OPEN  (SYSPRINT,OUTPUT),MODE=31 OPEN OUTPUT DATASET
         PUT   SYSPRINT,HDR1A           WRITE HEADER1
         PUT   SYSPRINT,HDR2A           WRITE HEADER2
         XC    XUID(4),XUID             CLEAR XUID LENGTH AREA
         MVC   XUID(2),=H'8'            SET DATA LENGTH
         MVC   XUID+4(8),=8C' '         SET STARTING UID
         MODESET MODE=SUP,KEY=ZERO
UIDLOOP  EQU   *
         XC    RACWORK(256),RACWORK     CLEAR RACROUTE ...
         XC    RACWORK+256(256),RACWORK+256 WORKAREA STORAGE
         MVC   ROUTWRK1(ROUTLEN1),RACROUT1 COPY RACROUTE PARM MODEL
         RACROUTE REQUEST=EXTRACT,                                  X
               TYPE=EXTRACTN,                                       X
               ENTITYX=XUID,                                        X
               FIELDS=FLDLIST1,                                     X
               RELEASE=1.9.2,                                       X
               SUBPOOL=1,                                           X
               WORKA=RACWORK,                                       X
```

```
              MF=(E,ROUTWRK1)
         LTR  R15,R15                  EXTRACT OK?
         BZ   LISTOK                   YES - PROCESS DATA
         ST   R15,RETCODE              SAVE THE RETURN CODE
         MVC  RACF_RC(8),ROUTWRK1      COPY RACF RTN/RSN CODES
         MODESET MODE=PROB,KEY=NZERO
         B    LISTDONE                 WE'RE DONE
LISTOK   EQU  *
*********************************************************************
*                                                                   *
*   A USERID WAS EXTRACTED.  MOVE THE DATA INTO AN OUTPUT BUFFER AND  *
*   WRITE THE RECORD.                                               *
*                                                                   *
*   USE THE DOOUTPUT ROUTINE FOR THIS PURPOSE.  R1 SHOULD CONTAIN     *
*   THE EXTRACT BUFFER ADDRESS.  THE BUFFER IS RELEASED BY DOOUTPUT.  *
*                                                                   *
*********************************************************************
         MVC  SAVEUID(8),XUID+4        COPY THE USERID
         BAL  R14,DOOUTPUT             CREATE NECESSARY OUTPUT
         B    UIDLOOP                  PROCESS NEXT USERID
LISTDONE EQU  *
         MODESET MODE=PROB,KEY=NZERO
         CLOSE (SYSPRINT),MODE=31      CLOSE OUTPUT DATASET
         B    RETURNØ                  WE'RE DONE
GRPLIST  EQU  *
*********************************************************************
*                                                                   *
*   A 'GROUPNAME(grpname)' PROGRAM PARM WAS DETECTED.  ATTEMPT TO     *
*   FIRST LOCATE THE USERID INFORMATION FOR USERIDS DEFINED TO THIS   *
*   GROUP.                                                          *
*                                                                   *
*********************************************************************
         OPEN  (SYSPRINT,OUTPUT),MODE=31 OPEN OUTPUT DATASET
         MODESET MODE=SUP,KEY=ZERO
         XC   RACWORK(256),RACWORK     CLEAR RACROUTE ...
         XC   RACWORK+256(256),RACWORK+256 WORKAREA STORAGE
         MVC  ROUTWRK3(ROUTLEN3),RACROUT3 COPY RACROUTE PARM MODEL
         RACROUTE REQUEST=EXTRACT,                                  X
               TYPE=EXTRACT,                                        X
               ENTITY=GRPNAME,                                      X
               FIELDS=FLDLIST2,                                     X
               RELEASE=1.9.2,                                       X
               SUBPOOL=1,                                           X
               WORKA=RACWORK,                                       X
               MF=(E,ROUTWRK3)
         LTR  R15,R15                  EXTRACT OK?
         BZ   GROUPOK                  YES - PROCESS USERIDS
         ST   R15,RETCODE              SAVE SAF RETURN CODE
         MVC  RACF_RC(8),ROUTWRK3      COPY RACF RTN/RSN CODES
         MODESET MODE=PROB,KEY=NZERO
```

```
         MVC   GRPMSG+6(8),GRPNAME       MOVE GRP NM INTO OUTPUT RECORD
         PUT   SYSPRINT,GRPMSG           WRITE THE RECORD
         CLOSE (SYSPRINT),MODE=31        CLOSE OUTPUT DATASET
         B     RETURN8                   GROUP WASN'T LOCATED
GROUPOK  EQU   *
         ST    R1,EXTSAVE                SAVE EXTRACT AREA ADDRESS
         PUT   SYSPRINT,HDR1A            WRITE HEADER1
         PUT   SYSPRINT,HDR2A            WRITE HEADER2
*************************************************************************
*                                                                       *
*   THE SPECIFIED GROUP WAS LOCATED.  WE NOW NEED TO EXTRACT THE        *
*   INFORMATION ABOUT EACH OF THE USERIDS.                              *
*                                                                       *
*************************************************************************
         L     R6,EXTSAVE                GET EXTRACT AREA ADDRESS
         USING EXTWKEA,R6                SET ADDRESSABILITY
         XR    R5,R5                     CLEAR R5
         ICM   R5,B'ØØ11',EXTWOFF        GET DATA OFFSET
         LA    R5,Ø(R5,R6)               SET ADDRESS
         CLC   Ø(4,R5),=F'Ø'             ANYTHING TO DO?
         BE    RELEXTA1                  NO - RELEASE EXTRACT W/A
         L     R9,Ø(,R5)                 GET DATA LENGTH
         LA    R9,4(R9,R5)               POINT PAST END OF DATA
         LA    R5,4(,R5)                 POINT TO FIRST USERID ENTRY
*************************************************************************
*                                                                       *
*   SORT THE USERID LIST.                                               *
*                                                                       *
*************************************************************************
         ST    R5,UIDSTART               SAVE STARTING ADDRESS
         ST    R9,UIDEND                 SAVE ENDING ADDRESS
         LR    R7,R5                     COPY STARTING ADDRESS
         LR    R8,R5                     COPY STARTING ADDRESS
SRTOUTLP EQU   *
         CR    R7,R9                     END OF DATA?
         BNL   SRTEND                    YES - SORT IS DONE
SRTINLP  EQU   *
         CR    R8,R9                     END OF DATA?
         BNL   SRTNEXTO                  YES - SORT NEXT ENTRY
         CLC   Ø(12,R7),Ø(R8)            ENTRY SWAP REQUIRED?
         BNH   SRTNEXTI                  NO - LOOK AT NEXT ENTRY
         MVC   TEMPENT(12),Ø(R7)         SAVE CURRENT ENTRY
         MVC   Ø(12,R7),Ø(R8)            MOVE OVER LOWER ENTRY
         MVC   Ø(12,R8),TEMPENT          COPY BACK MOVED ENTRY
SRTNEXTI EQU   *
         LA    R8,12(,R8)                POINT TO NEXT ENTRY
         B     SRTINLP                   CHECK IT OUT
SRTNEXTO EQU   *
         LA    R7,12(,R7)                POINT TO NEXT ENTRY
         LR    R8,R7                     COPY IT
```

```
         B     SRTOUTLP                  CHECK IT OUT
SRTEND   EQU   *
*********************************************************************
         L     R5,UIDSTART               LOAD STARTING ADDRESS
         L     R9,UIDEND                 LOAD ENDING ADDRESS
GRPUSRL1 EQU   *
         CR    R5,R9                     END OF DATA?
         BNL   RELEXTA1                  YES - RELEASE EXTRACT W/A
         LA    R7,4(,R5)                 POINT TO USERID
*********************************************************************
*                                                                   *
*   EXTRACT THE USERID SPECIFIC INFORMATION.                        *
*                                                                   *
*********************************************************************
         XC    RACWORK(256),RACWORK      CLEAR RACROUTE ...
         XC    RACWORK+256(256),RACWORK+256 WORKAREA STORAGE
         MVC   ROUTWRK2(ROUTLEN2),RACROUT2 COPY RACROUTE PARM MODEL
         RACROUTE REQUEST=EXTRACT,                                  X
               TYPE=EXTRACT,                                        X
               ENTITY=(R7),                                         X
               FIELDS=FLDLIST1,                                     X
               RELEASE=1.9.2,                                       X
               SUBPOOL=1,                                           X
               WORKA=RACWORK,                                       X
               MF=(E,ROUTWRK2)
         LTR   R15,R15                   EXTRACT OK?
         BZ    USEROK                    YES - PROCESS USERID INFO
         B     GRPUSRN1                  CHECK NEXT USERID
USEROK   EQU   *
*********************************************************************
*                                                                   *
*   A USERID WAS EXTRACTED.  MOVE THE DATA INTO AN OUTPUT BUFFER AND *
*   WRITE THE RECORD.                                               *
*                                                                   *
*   USE THE DOOUTPUT ROUTINE FOR THIS PURPOSE.  R1 SHOULD CONTAIN   *
*   THE EXTRACT BUFFER ADDRESS.  THE BUFFER IS RELEASED BY DOOUTPUT. *
*                                                                   *
*********************************************************************
         MVC   SAVEUID(8),Ø(R7)          COPY THE USERID
         BAL   R14,DOOUTPUT              CREATE NECESSARY OUTPUT
*********************************************************************
GRPUSRN1 EQU   *
         LA    R5,12(,R5)                POINT TO NEXT USERID ENTRY
         B     GRPUSRL1                  GO CHECK THINGS OUT
RELEXTA1 EQU   *
*********************************************************************
*                                                                   *
*   RELEASE THE GROUP EXTRACT BUFFER.                               *
*                                                                   *
*********************************************************************
```

```
         L     R6,EXTSAVE                GET EXTRACT AREA ADDRESS
         XR    R8,R8                     CLEAR R8
         XR    R9,R9                     CLEAR R9
         IC    R9,Ø(,R6)                 SAVE THE SUBPOOL VALUE
         ICM   R8,B'Ø111',1(R6)          SAVE W/A LENGTH
         STORAGE RELEASE,LENGTH=(R8),ADDR=(R6),SP=(R9)
         MODESET MODE=PROB,KEY=NZERO
         CLOSE (SYSPRINT),MODE=31        CLOSE OUTPUT DATASET
         B     RETURNØ                   WE'RE DONE
*********************************************************************
RETURNØ  EQU   *
         L     R1Ø,4(,R13)
         LR    R1,R13
         STORAGE RELEASE,LENGTH=WALEN,ADDR=(R1)
         LR    R13,R1Ø
         LM    R14,R12,12(R13)
         XR    R15,R15
         BR    R14
*********************************************************************
RETURN4  EQU   *
         L     R1Ø,4(,R13)
         LR    R1,R13
         STORAGE RELEASE,LENGTH=WALEN,ADDR=(R1)
         LR    R13,R1Ø
         LM    R14,R12,12(R13)
         LA    R15,4
         BR    R14
*********************************************************************
RETURN8  EQU   *
         MVC   DBL2(4),RETCODE           GET SAF RETURN CODE
         BAL   R14,HEXCNVT               CONVERT TO READABLE
         MVC   ERRMSG1+16(8),DBL1        MOVE INTO MESSAGE
         MVC   DBL2(4),RACF_RC           GET RACF RETURN CODE
         BAL   R14,HEXCNVT               CONVERT TO READABLE
         MVC   ERRMSG1+35(8),DBL1        MOVE INTO MESSAGE
         MVC   DBL2(4),RACF_RSN          GET RACF REASON CODE
         BAL   R14,HEXCNVT               CONVERT TO READABLE
         MVC   ERRMSG1+55(8),DBL1        MOVE INTO MESSAGE
ERRMSG1  WTO   'SAF RC: XXXXXXXX  RACF RC: XXXXXXXX  RACF RSN: XXXXXXXXX
                   ',ROUTCDE=(1),DESC=(6)
         L     R1Ø,4(,R13)
         LR    R1,R13
         STORAGE RELEASE,LENGTH=WALEN,ADDR=(R1)
         LR    R13,R1Ø
         LM    R14,R12,12(R13)
         LA    R15,8
         BR    R14
*********************************************************************
RETURN12 EQU   *
         L     R1Ø,4(,R13)
```

```
         LR    R1,R13
         STORAGE RELEASE,LENGTH=WALEN,ADDR=(R1)
         LR    R13,R1Ø
         LM    R14,R12,12(R13)
         LA    R15,12
         BR    R14
*********************************************************************
RETURN16 EQU   *
         L     R1Ø,4(,R13)
         LR    R1,R13
         STORAGE RELEASE,LENGTH=WALEN,ADDR=(R1)
         LR    R13,R1Ø
         LM    R14,R12,12(R13)
         LA    R15,16
         BR    R14
*********************************************************************
HEXCNVT  EQU   *
         UNPK  DBL1(9),DBL2(5)          UNPACK THE VALUE
         NC    DBL1(8),=8X'ØF'          TURN OFF HIGH NIBBLE
         TR    DBL1(8),=C'Ø123456789ABCDEF' MAKE THINGS READABLE
         BR    R14                      RETURN
*********************************************************************
DOOUTPUT EQU   *
         STM   RØ,R15,SVAREAØ2           SAVE REGISTERS
         XC    SEGDATA(SEGDATAL),SEGDATA CLEAR SEGMENT DATA SAVE AREA
         XR    R6,R6                     CLEAR R6
         ICM   R6,B'ØØ11',4(R1)          GET DATA OFFSET
         AR    R6,R1                     CALCULATE DATA ADDRESS
SVFLD1   EQU   *
         ICM   R15,B'1111',Ø(R6)         GET DATA LENGTH
         MVC   SAVENAME(2Ø),4(R6)        COPY THE USERNAME
SVFLD2   EQU   *
         LA    R6,4(R15,R6)              POINT TO FLAG1
         ICM   R15,B'1111',Ø(R6)         GET DATA LENGTH
         LTR   R15,R15                   ANY DATA?
         BZ    SVFLD3                    NO - CHECK NEXT FIELD
         MVC   SAVEFLG1(1),4(R6)         SAVE FLAG1
SVFLD3   EQU   *
         LA    R6,4(R15,R6)              POINT TO FLAG2
         ICM   R15,B'1111',Ø(R6)         GET DATA LENGTH
         LTR   R15,R15                   ANY DATA?
         BZ    SVFLD4                    NO - CHECK NEXT FIELD
         MVC   SAVEFLG2(1),4(R6)         SAVE FLAG2
SVFLD4   EQU   *
         LA    R6,4(R15,R6)              POINT TO FLAG3
         ICM   R15,B'1111',Ø(R6)         GET DATA LENGTH
         LTR   R15,R15                   ANY DATA?
         BZ    SVFLD5                    NO - CHECK NEXT FIELD
         MVC   SAVEFLG3(1),4(R6)         SAVE FLAG3
SVFLD5   EQU   *
```

```
         LA    R6,4(R15,R6)            POINT TO FLAG4
         ICM   R15,B'1111',Ø(R6)       GET DATA LENGTH
         LTR   R15,R15                 ANY DATA?
         BZ    SVFLD6                  NO - CHECK NEXT FIELD
         MVC   SAVEFLG4(1),4(R6)       SAVE FLAG4
SVFLD6   EQU   *
         LA    R6,4(R15,R6)            POINT TO FLAG5
         ICM   R15,B'1111',Ø(R6)       GET DATA LENGTH
         LTR   R15,R15                 ANY DATA?
         BZ    SVFLD7                  NO - CHECK NEXT FIELD
         MVC   SAVEFLG5(1),4(R6)       SAVE FLAG5
SVFLD7   EQU   *
         LA    R6,4(R15,R6)            POINT TO FLAG6
         ICM   R15,B'1111',Ø(R6)       GET DATA LENGTH
         LTR   R15,R15                 ANY DATA?
         BZ    SVFLD8                  NO - CHECK NEXT FIELD
         MVC   SAVEFLG6(1),4(R6)       SAVE FLAG6
SVFLD8   EQU   *
         LA    R6,4(R15,R6)            POINT TO FLAG7
         ICM   R15,B'1111',Ø(R6)       GET DATA LENGTH
         LTR   R15,R15                 ANY DATA?
         BZ    SVFLD9                  NO - CHECK NEXT FIELD
         MVC   SAVEFLG7(1),4(R6)       SAVE FLAG7
SVFLD9   EQU   *
         LA    R6,4(R15,R6)            POINT TO FLAG8
         ICM   R15,B'1111',Ø(R6)       GET DATA LENGTH
         LTR   R15,R15                 ANY DATA?
         BZ    SVFLD1Ø                 NO - CHECK NEXT FIELD
         MVC   SAVEFLG8(1),4(R6)       SAVE FLAG8
SVFLD1Ø  EQU   *
         LA    R6,4(R15,R6)            POINT TO PASSINT
         ICM   R15,B'1111',Ø(R6)       GET DATA LENGTH
         LTR   R15,R15                 ANY DATA?
         BZ    SVFLD11                 NO - CHECK NEXT FIELD
         MVC   SAVEPWDI(1),4(R6)       SAVE PASSINT
SVFLD11  EQU   *
         LA    R6,4(R15,R6)            POINT TO PASSDATE
         ICM   R15,B'1111',Ø(R6)       GET DATA LENGTH
         LTR   R15,R15                 ANY DATA?
         BZ    SVFLD12                 NO - CHECK NEXT FIELD
         MVC   SAVEPWDD(3),4(R6)       SAVE PASSDATE
SVFLD12  EQU   *
         LA    R6,4(R15,R6)            POINT TO LJDATE
         ICM   R15,B'1111',Ø(R6)       GET DATA LENGTH
         LTR   R15,R15                 ANY DATA?
         BZ    SVFLD13                 NO - CHECK NEXT FIELD
         MVC   SAVELJD(3),4(R6)        SAVE LJDATE
SVFLD13  EQU   *
*********************************************************************
         XR    R7,R7                   CLEAR R7
```

```
         XR    R8,R8                     CLEAR R8
         ICM   R7,B'Ø111',1(R1)          GET STORAGE LENGTH
         ICM   R8,B'ØØØ1',Ø(R1)          GET SUBPOOL
         STORAGE RELEASE,LENGTH=(R7),ADDR=(R1),SP=(R8)
         MVI   OUTREC,C' '               SET FILL CHARACTER
         MVC   OUTREC+1(132),OUTREC      CLEAR THE AREA
         MVC   OUTREC+UIDOFF(8),SAVEUID  COPY THE USERID
         MVC   OUTREC+UNMOFF(2Ø),SAVENAME COPY THE USER NAME
         MVI   PREVATTR,OFF              SET ATTRIBUTE FLAG OFF
         LA    R2,OUTREC+ATTOFF          SET OUTPUT AREA ADDRESS
*********************************************************************
ATTR1    EQU   *
         TM    SAVEFLG1,X'8Ø'            ADSP?
         BZ    ATTR2                     NO - CHECK NEXT ATTRIBUTE
         TM    PREVATTR,ON               PREVIOUS ATTRIBUTE?
         BZ    NOSEP1                    NO - NO SEPARATOR REQUIRED
         MVI   Ø(R2),C','                SET SEPARATOR VALUE
         LA    R2,1(,R2)                 POINT PAST SEPARATOR
NOSEP1   EQU   *
         MVC   Ø(4,R2),=C'ADSP'          SET ATTRIBUTE VALUE
         LA    R2,4(,R2)                 POINT PAST VALUE
         MVI   PREVATTR,ON               SET FLAG ON
*********************************************************************
ATTR2    EQU   *
         TM    SAVEFLG2,X'8Ø'            SPECIAL?
         BZ    ATTR3                     NO - CHECK NEXT ATTRIBUTE
         TM    PREVATTR,ON               PREVIOUS ATTRIBUTE?
         BZ    NOSEP2                    NO - NO SEPARATOR REQUIRED
         MVI   Ø(R2),C','                SET SEPARATOR VALUE
         LA    R2,1(,R2)                 POINT PAST SEPARATOR
NOSEP2   EQU   *
         MVC   Ø(4,R2),=C'SPEC'          SET ATTRIBUTE VALUE
         LA    R2,4(,R2)                 POINT PAST VALUE
         MVI   PREVATTR,ON               SET FLAG ON
*********************************************************************
ATTR3    EQU   *
         TM    SAVEFLG3,X'8Ø'            OPERATIONS?
         BZ    ATTR4                     NO - CHECK NEXT ATTRIBUTE
         TM    PREVATTR,ON               PREVIOUS ATTRIBUTE?
         BZ    NOSEP3                    NO - NO SEPARATOR REQUIRED
         MVI   Ø(R2),C','                SET SEPARATOR VALUE
         LA    R2,1(,R2)                 POINT PAST SEPARATOR
NOSEP3   EQU   *
         MVC   Ø(4,R2),=C'OPER'          SET ATTRIBUTE VALUE
         LA    R2,4(,R2)                 POINT PAST VALUE
         MVI   PREVATTR,ON               SET FLAG ON
*********************************************************************
ATTR4    EQU   *
         TM    SAVEFLG5,X'8Ø'            GRPACC?
         BZ    ATTR5                     NO - CHECK NEXT ATTRIBUTE
```

```
         TM    PREVATTR,ON               PREVIOUS ATTRIBUTE?
         BZ    NOSEP4                    NO - NO SEPARATOR REQUIRED
         MVI   Ø(R2),C','                SET SEPARATOR VALUE
         LA    R2,1(,R2)                 POINT PAST SEPARATOR
NOSEP4   EQU   *
         MVC   Ø(6,R2),=C'GRPACC'        SET ATTRIBUTE VALUE
         LA    R2,6(,R2)                 POINT PAST VALUE
         MVI   PREVATTR,ON               SET FLAG ON
*********************************************************************
ATTR5    EQU   *
         TM    SAVEFLG6,X'8Ø'            AUDITOR?
         BZ    ATTR6                     NO - CHECK NEXT ATTRIBUTE
         TM    PREVATTR,ON               PREVIOUS ATTRIBUTE?
         BZ    NOSEP5                    NO - NO SEPARATOR REQUIRED
         MVI   Ø(R2),C','                SET SEPARATOR VALUE
         LA    R2,1(,R2)                 POINT PAST SEPARATOR
NOSEP5   EQU   *
         MVC   Ø(3,R2),=C'AUD'           SET ATTRIBUTE VALUE
         LA    R2,3(,R2)                 POINT PAST VALUE
         MVI   PREVATTR,ON               SET FLAG ON
*********************************************************************
ATTR6    EQU   *
         TM    SAVEFLG7,X'4Ø'            PROTECTED?
         BZ    ATTR7                     NO - CHECK NEXT ATTRIBUTE
         TM    PREVATTR,ON               PREVIOUS ATTRIBUTE?
         BZ    NOSEP6                    NO - NO SEPARATOR REQUIRED
         MVI   Ø(R2),C','                SET SEPARATOR VALUE
         LA    R2,1(,R2)                 POINT PAST SEPARATOR
NOSEP6   EQU   *
         MVC   Ø(4,R2),=C'PROT'          SET ATTRIBUTE VALUE
         LA    R2,4(,R2)                 POINT PAST VALUE
         MVI   PREVATTR,ON               SET FLAG ON
*********************************************************************
ATTR7    EQU   *
         TM    SAVEFLG7,X'8Ø'            OID?
         BZ    ATTR8                     NO - CHECK IF ANY ATTRIBUTE
         TM    SAVEFLG8,X'8Ø'            OID?
         BZ    ATTR8                     NO - CHECK IF ANY ATTRIBUTE
         TM    PREVATTR,ON               PREVIOUS ATTRIBUTE?
         BZ    NOSEP7                    NO - NO SEPARATOR REQUIRED
         MVI   Ø(R2),C','                SET SEPARATOR VALUE
         LA    R2,1(,R2)                 POINT PAST SEPARATOR
NOSEP7   EQU   *
         MVC   Ø(3,R2),=C'OID'           SET ATTRIBUTE VALUE
         LA    R2,3(,R2)                 POINT PAST VALUE
         MVI   PREVATTR,ON               SET FLAG ON
*********************************************************************
ATTR8    EQU   *
         TM    PREVATTR,ON               ANY ATTRIBUTES?
         BO    PWDEXP                    YES - CHECK PWD EXPIRY
```

```
         MVC   Ø(4,R2),=C'NONE'          SET ATTRIBUTE VALUE
*********************************************************************
PWDEXP   EQU   *
         TM    SAVEFLG7,X'CØ'            PASSWORD REQUIRED?
         BZ    PWDEXPØ                   YES - CHECK OUT VALUES
         MVC   OUTREC+EXPOFF(9),=C'PROTECTED' SET A DEFAULT
         TM    SAVEFLG8,X'8Ø'            OID?
         BO    PWDEXP3                   YES - PASSWORD NOT REQUIRED
         B     RVKCHK                    CHECK REVOKED STATUS
PWDEXPØ  EQU   *
         CLI   SAVEPWDI,X'FF'            PASSWORD NEVER EXPIRES?
         BE    PWDEXP2                   YES - SET NEVER EXPIRES MSG
         CLC   SAVEPWDD(3),=X'ØØØØØF'    PASSWORD EXPIRED?
         BE    PWDEXP1                   YES - SET PASSWORD EXPIRED MSG
         MVC   PWDINTVL(1),SAVEPWDI      SAVE PASSWORD INTERVAL
         CLI   SAVEPWDI,X'ØØ'            ANYTHING IN BASE SEGMENT?
         BE    GBLEXP                    NO - USE RACF GLOBAL EXPIRY
         CLC   SAVEPINV(1),SAVEPWDI      GLOBAL VALUE LESS?
         BL    GBLEXP                    YES - USE GLOBAL EXPIRY
         B     GETDATE                   GET THE CURRENT DATE
GBLEXP   EQU   *
         MVC   PWDINTVL(1),SAVEPINV      SET INTERVAL TO GLOBAL VALUE
GETDATE  EQU   *
         TIME  DEC                       GET CURRENT TIME AND DAY
         STCM  R1,B'Ø111',CURRDATE       SAVE IT
*********************************************************************
         LA    R15,YRTBL                 GET YEAR TABLE ADDRESS
         LA    R14,YRTBLEND              GET YEAR TABLE ADDRESS END
YRLOOP   EQU   *
         CR    R15,R14                   END OF TABLE?
         BNL   PWDEXP4                   YES - SOMETHINGS GOOFY
         CLC   Ø(1,R15),CURRDATE         A YEAR MATCH?
         BE    YRMATCH                   YES - GO ON
         LA    R15,3(,R15)               GET ADDR OF NEXT YEAR ENTRY
         B     YRLOOP                    TRY AGAIN
YRMATCH  EQU   *
         ST    R15,YRTBLENT              SAVE YEAR TABLE ENTRY ADDRESS
         MVI   DATEWRK1,2Ø               ASSUME 21ST CENTURY
         MVI   DATEWRK2,2Ø               ASSUME 21ST CENTURY
         MVC   DATEWRK1+1(1),CURRDATE    COPY CURRENT YEAR
         MVC   DATEWRK2+1(1),SAVEPWDD    COPY PASSWORD CHANGE YEAR
         CLI   DATEWRK1+1,7Ø             YEAR GREATER THAN 7Ø?
         BNH   CHKPWDYR                  NO - CHECK PASSWORD YEAR
         MVI   DATEWRK1,19               MOVE BACK TO 2ØTH CENTURY
CHKPWDYR EQU   *
         CLI   DATEWRK2+1,7Ø             YEAR GREATER THAN 7Ø?
         BNH   YRCOMPR                   NO - COMPARE THE YEAR VALUES
         MVI   DATEWRK2,19               MOVE BACK TO 2ØTH CENTURY
YRCOMPR  EQU   *
         CLC   DATEWRK1(2),DATEWRK2      SAME YEAR?
```

```
         BE    SAMEYEAR                  YES - PROCESS SAME YEAR
         CLC   DATEWRK1(2),DATEWRK2      PREVIOUS YEAR?
         BH    PREVYEAR                  YES - PWD LAST ALTERED DIF YR
         B     PWDEXP4                   WORKING WITH TIME FORWARD DATA
SAMEYEAR EQU   *
         MVC   DBL2(3),CURRDATE          COPY CURRENT DATE
         SP    DBL2(3),SAVEPWDD(3)       SUBTRACT PWD CHANGE DATE
         B     CALCEXP                   GO CALCULATE EXPIRY DAYS
PREVYEAR EQU   *
         L     R15,YRTBLENT              GET YEAR TABLE ENTRY ADDRESS
         S     R15,=F'3'                 BACK UP ONE YEAR
         C     R15,=A(YRTBL)             STILL A VALID TABLE ENTRY?
         BL    PWDEXP4                   NO - ISSUE A MESSAGE
         CLC   Ø(1,R15),SAVEPWDD         A YEAR MATCH?
         BNE   PWDEXP1                   NO - PASSWORD IS EXPIRED
         MVC   DBL2(3),Ø(R15)            COPY YEAR END DAY
         SP    DBL2(3),SAVEPWDD(3)       SUBTRACT PWD CHANGE DATE
         MVC   DBL1(2),CURRDATE+1        COPY CURRENT YEAR DAY
         AP    DBL2(3),DBL1(2)           ADD TOGETHER
CALCEXP  EQU   *
         XC    DBL1(8),DBL1              CLEAR A WORK AREA
         MVC   DBL1+5(3),DBL2            # OF DAYS SINCE PWD CHANGE
         CVB   R1,DBL1                   CONVERT TO BINARY
         C     R1,=F'256'                MORE THAN 256?          Ø2Ø523
         BNL   PWDEXP1                   YES - PWD IS EXPIRED    Ø2Ø523
         STCM  R1,B'ØØØ1',PWDDAYS        SAVE PWD CHANGE DAYS
         CLC   PWDDAYS(1),PWDINTVL       EXPIRED?
         BNL   PWDEXP1                   YES - ISSUE A MESSAGE
         XR    R1,R1                     CLEAR R1
         XR    R14,R14                   CLEAR R14
         IC    R1,PWDINTVL               GET PWD CHANGE INTERVAL
         IC    R14,PWDDAYS               GET # OF DAYS SINCE PWD CHANGE
         SR    R1,R14                    CALCULATE DAYS 'TIL EXPIRY
         CVD   R1,DBL2                   CONVERT TO DECIMAL
         UNPK  DBL1(8),DBL2(8)           UNPACK THE VALUE
         OI    DBL1+7,X'FØ'              MAKE IT READABLE
         MVC   OUTREC+EXPOFF(19),=C'EXPIRES IN xxx DAYS'
         MVC   OUTREC+EXPOFF+11(3),DBL1+5 MOVE IN # OF DAYS
         B     RVKCHK                    CHECK REVOKED STATUS
*********************************************************************
PWDEXP1  EQU   *
         MVC   OUTREC+EXPOFF(19),=C'CURRENTLY EXPIRED  '
         B     RVKCHK                    CHECK REVOKED STATUS
PWDEXP2  EQU   *
         MVC   OUTREC+EXPOFF(19),=C'NEVER EXPIRES      '
         B     RVKCHK                    CHECK REVOKED STATUS
PWDEXP3  EQU   *
         MVC   OUTREC+EXPOFF(19),=C'NOT REQUIRED       '
         B     RVKCHK                    CHECK REVOKED STATUS
PWDEXP4  EQU   *
```

```
         MVC   OUTREC+EXPOFF(19),=C'UNKNOWN           '
         B     RVKCHK                    CHECK REVOKED STATUS
*********************************************************************
RVKCHK   EQU   *
         MVC   OUTREC+RVKOFF(11),=C'NOT REVOKED'
         TM    SAVEFLG4,X'8Ø'            REVOKED?
         BNO   NORVK                     NO - DON'T CHANGE MESSAGE
         MVC   OUTREC+RVKOFF(11),=C'REVOKED    '
NORVK    EQU   *
*********************************************************************
         LA    R5,SEGTBL                 GET SEGMENT TABLE ADDRESS
         LA    R2,OUTREC+SEGOFF          SET OUTPUT AREA ADDRESS
         MVI   PREVATTR,OFF              SET FLAG OFF
         MODESET MODE=SUP,KEY=ZERO
SEGLOOP  EQU   *
         C     R5,=A(SEGTBLE)            END OF TABLE?
         BNL   SEGLOOPE                  YES - END OF SEGMENT LOOP
*********************************************************************
*                                                                   *
*   DETERMINE DEFINED SEGMENTS FOR THIS USERID.                     *
*                                                                   *
*********************************************************************
         XC    RACWORK(256),RACWORK      CLEAR RACROUTE ...
         XC    RACWORK+256(256),RACWORK+256 WORKAREA STORAGE
         MVC   ROUTWRK2(ROUTLEN2),RACROUT2 COPY RACROUTE PARM MODEL
         L     R7,Ø(,R5)                 GET FIELD LIST ADDRESS
         LA    R8,4(,R5)                 GET SEGMENT NAME ADDRESS
         RACROUTE REQUEST=EXTRACT,                                  X
               TYPE=EXTRACT,                                        X
               ENTITY=SAVEUID,                                      X
               FIELDS=(R7),                                         X
               SEGMENT=(R8),                                        X
               RELEASE=1.9.2,                                       X
               SUBPOOL=1,                                           X
               WORKA=RACWORK,                                       X
               MF=(E,ROUTWRK2)
         LTR   R15,R15                   EXTRACT OK?
         BNZ   SEGNEXT                   NO - CHECK NEXT SEGMENT
         LR    R6,R1                     SAVE EXTRACT AREA ADDRESS
         LA    R15,OUTREC+131            SET MAX ADDRESS
         CR    R2,R15                    AT THE END?
         BH    NOSEG                     YES - DON'T DO ANYTHING
         TM    PREVATTR,ON               PREVIOUS SEGMENT DETECTED?
         BNO   NOSEP                     NO - SEPARATOR NOT REQUIRED
         MVI   Ø(R2),C','                SET SEPARATOR
         LA    R2,1(,R2)                 SKIP PAST SEPARATOR
NOSEP    EQU   *
         OI    PREVATTR,ON               SET PREVIOUS SEGMENT FLAG ON
         CR    R2,R15                    AT THE END?
         BH    PLUSSIGN                  YES - INDICATE MORE SEGMENTS
```

```
         MVC   Ø(2,R2),12(R5)           MOVE IN SEGMENT INDICATOR
         LA    R2,2(,R2)                SKIP PAST
         B     NOSEG                    KEEP GOING
PLUSSIGN EQU   *
         MVI   Ø(R2),C'+'               SET MORE SEGMENTS INDICATOR
         LA    R2,1(,R2)                SKIP PAST SEPARATOR
NOSEG    EQU   *
         XR    R8,R8                    CLEAR R8
         XR    R9,R9                    CLEAR R9
         IC    R9,Ø(,R6)                SAVE THE SUBPOOL VALUE
         ICM   R8,B'Ø111',1(R6)         SAVE W/A LENGTH
         STORAGE RELEASE,LENGTH=(R8),ADDR=(R6),SP=(R9)
SEGNEXT  EQU   *
         LA    R5,16(,R5)               NEXT SEGMENT TABLE ENTRY
         B     SEGLOOP                  GO CHECK IT OUT
SEGLOOPE EQU   *
         MODESET MODE=PROB,KEY=NZERO
         PUT   SYSPRINT,OUTREC          WRITE A RECORD
         LM    RØ,R15,SVAREAØ2          LOAD REGISTERS
         BR    R14                      RETURN
*********************************************************************
GRPNMMVC MVC   GRPNAME(*-*),Ø(R9)
*********************************************************************
SYSPRINT DCB   MACRF=(PM),LRECL=133,DSORG=PS,DDNAME=SYSPRINT
*********************************************************************
HDR1A    DC    C'Userid    User Name             Attributes          '
HDR1B    DC    C'               PWD Expiry Status    Revoke Status  De'
HDR1C    DC    CL(133-(L'HDR1A+L'HDR1B))'fined Segments '
HDR2A    DC    C'--------  -------------------  --------------------'
HDR2B    DC    C'------------  ------------------  ------------  --'
HDR2C    DC    CL(133-(L'HDR2A+L'HDR2B))'--------------------------'
UIDOFF   EQU   Ø
UNMOFF   EQU   1Ø
ATTOFF   EQU   32
EXPOFF   EQU   68
RVKOFF   EQU   89
SEGOFF   EQU   1Ø4
GRPMSG   DC   CL133'Group xxxxxxxx not located in security product DB.'
*********************************************************************
RACROUT1 RACROUTE REQUEST=EXTRACT,                                    X
               TYPE=EXTRACTN,                                         X
               CLASS='USER',                                          X
               RELEASE=1.9.2,                                         X
               MF=L
ROUTLEN1 EQU   *-RACROUT1
*********************************************************************
RACROUT2 RACROUTE REQUEST=EXTRACT,                                    X
               TYPE=EXTRACT,                                          X
               CLASS='USER',                                          X
               RELEASE=1.9.2,                                         X
```

```
               MF=L
ROUTLEN2 EQU   *-RACROUT2
*********************************************************************
RACROUT3 RACROUTE REQUEST=EXTRACT,                                    X
               TYPE=EXTRACT,                                          X
               CLASS='GROUP',                                         X
               RELEASE=1.9.2,                                         X
               MF=L
ROUTLEN3 EQU   *-RACROUT3
*********************************************************************
FLDLIST1 DC    F'12'
         DC    C'PGMRNAME'
         DC    C'FLAG1   '           ADSP
         DC    C'FLAG2   '           SPECIAL
         DC    C'FLAG3   '           OPERATIONS
         DC    C'FLAG4   '           REVOKE
         DC    C'FLAG5   '           GRPACC
         DC    C'FLAG6   '           AUDITOR
         DC    C'FLAG7   '           bit 1 on - PROTECTED
         DC    C'FLAG8   '           OID
         DC    C'PASSINT '           USERS PASSWORD CHANGE INTERVAL
         DC    C'PASSDATE'           USERS LAST PASSWORD CHANGE DATE
         DC    C'LJDATE  '           USERS LAST LOGON DATE Ø=PWDEXP
*********************************************************************
FLDLIST2 DC    F'1'
         DC    C'USERID  '
*********************************************************************
SEGTBL   DS    ØD
         DC    A(FLDLSTØ1),CL8'CICS',CL4'CI'
         DC    A(FLDLSTØ2),CL8'DCE',CL4'DC'
         DC    A(FLDLSTØ3),CL8'DFP',CL4'DF'
         DC    A(FLDLSTØ5),CL8'LANGUAGE',CL4'LA'
         DC    A(FLDLSTØ7),CL8'NETVIEW',CL4'NV'
         DC    A(FLDLSTØ9),CL8'OMVS',CL4'OM'
         DC    A(FLDLST1Ø),CL8'OPERPARM',CL4'OP'
         DC    A(FLDLST11),CL8'OVM',CL4'OV'
         DC    A(FLDLST12),CL8'TSO',CL4'TS'
         DC    A(FLDLST13),CL8'WORKATTR',CL4'WA'
SEGTBLE  EQU   *
*********************************************************************
FLDLSTØ1 DC    F'5',CL8'OPIDENT',CL8'OPCLASS',CL8'OPPRTY'
         DC    CL8'TIMEOUT',CL8'XRFSOFF'
FLDLSTØ2 DC    F'5',CL8'DCENAME',CL8'HOMECELL',CL8'HOMEUUID'
         DC    CL8'UUID',CL8'DCEFLAG'
FLDLSTØ3 DC    F'4',CL8'DATAAPPL',CL8'DATACLAS',CL8'MGMTCLAS'
         DC    CL8'STORCLAS'
FLDLSTØ4 EQU   *                     RESERVED FOR KERBEROS SEGMENT
FLDLSTØ5 DC    F'2',CL8'USERNL1',CL8'USERNL2'
FLDLSTØ6 EQU   *                     RESERVED FOR LOTUS NOTES SEGMENT
FLDLSTØ7 DC    F'8',CL8'IC',CL8'CONSNAME',CL8'CTL',CL8'MSGRECVR'
```

```
         DC    CL8'OPCLASS',CL8'DOMAINS',CL8'NGMFADMN',CL8'NGMFVSPN'
FLDLSTØ8 EQU   *                    RESERVED FOR NDS SEGMENT
FLDLSTØ9 DC    F'9',CL8'UID',CL8'HOME',CL8'PROGRAM',CL8'ASSIZE'
         DC    CL8'CPUTIME',CL8'FILEPROC',CL8'MMAPAREA',CL8'PROCUSER'
         DC    CL8'THREADS'
FLDLST1Ø DC    F'15',CL8'OPERALTG',CL8'OPERAUTH',CL8'OPERAUTO'
         DC    CL8'OPERCMDS',CL8'OPERDOM',CL8'OPERKEY',CL8'OPERLEVL'
         DC    CL8'OPERLOGC',CL8'OPERMFRM',CL8'OPERMGID',CL8'OPERMON'
         DC    CL8'OPERMSCP',CL8'OPERROUT',CL8'OPERSTOR',CL8'OPERUD'
FLDLST11 DC    F'4',CL8'UID',CL8'HOME',CL8'PROGRAM',CL8'FSROOT'
FLDLST12 DC    F'14',CL8'TACCNT',CL8'TCOMMAND',CL8'TDEST',CL8'THCLASS'
         DC    CL8'TJCLASS',CL8'TLPROC',CL8'TLSIZE',CL8'TMCLASS'
         DC    CL8'TMSIZE',CL8'TSCLASS',CL8'TUDATA',CL8'TUNIT'
         DC    CL8'TSOSLABL',CL8'TRBA'
FLDLST13 DC    F'9',CL8'WAACCNT',CL8'WAADDR1',CL8'WAADDR2',CL8'WAADDR3'
         DC    CL8'WAADDR4',CL8'WABLDG',CL8'WADEPT',CL8'WANAME'
         DC    CL8'WAROOM'
*********************************************************************
YRTBL    DS    ØD
YR1997   DC    P'97365'
YR1998   DC    P'98365'
YR1999   DC    P'99365'
YR2ØØØ   DC    P'Ø366'
YR2ØØ1   DC    P'1365'
YR2ØØ2   DC    P'2365'
YR2ØØ3   DC    P'3365'
YR2ØØ4   DC    P'4366'
YR2ØØ5   DC    P'5365'
YR2ØØ6   DC    P'6365'
YR2ØØ7   DC    P'7365'
YR2ØØ8   DC    P'8366'
YR2ØØ9   DC    P'9365'
YR2Ø1Ø   DC    P'10365'
YR2Ø11   DC    P'11365'
YR2Ø12   DC    P'12366'
YR2Ø13   DC    P'13365'
YR2Ø14   DC    P'14365'
YR2Ø15   DC    P'15365'
YR2Ø16   DC    P'16366'
YR2Ø17   DC    P'17365'
YR2Ø18   DC    P'18365'
YR2Ø19   DC    P'19365'
YR2Ø2Ø   DC    P'2Ø366'
YR2Ø21   DC    P'21365'
YR2Ø22   DC    P'22365'
YR2Ø23   DC    P'23365'
YR2Ø24   DC    P'24366'
YRTBLEND DC    3X'FF'
*********************************************************************
         LTORG ,
```

```
*********************************************************************
WORKAREA DSECT
SAVEAREA DS    18F
SVAREAØ2 DS    18F
RETCODE  DS    F
RACF_RC  DS    F
RACF_RSN DS    F
EXTSAVE  DS    F
ROUTWRK1 DS    CL(ROUTLEN1)
ROUTWRK2 DS    CL(ROUTLEN2)
ROUTWRK3 DS    CL(ROUTLEN3)
XUID     DS    ØD
         DS    F
         DS    CL8
*********************************************************************
RACWORK  DS    ØD
         DS    CL512
DBL1     DS    2D
DBL2     DS    2D
UIDSTART DS    F
UIDEND   DS    F
TEMPENT  DS    CL12
SAVEUID  DS    CL8
SEGDATA  DS    ØD
SAVENAME DS    CL2Ø
SAVEFLG1 DS    XL1
SAVEFLG2 DS    XL1
SAVEFLG3 DS    XL1
SAVEFLG4 DS    XL1
SAVEFLG5 DS    XL1
SAVEFLG6 DS    XL1
SAVEFLG7 DS    XL1
SAVEFLG8 DS    XL1
SAVEPWDI DS    XL1
SAVEPWDD DS    XL3
SAVELJD  DS    XL3
SEGDATAL EQU   *-SEGDATA
PWDINTSV DS    F
YRTBLENT DS    F
SAVEPINV DS    XL1
PWDINTVL DS    XL1
PWDDAYS  DS    XL1
CURRDATE DS    XL3
DATEWRK1 DS    XL2
DATEWRK2 DS    XL2
PREVATTR DS    XL1
ON       EQU   1
OFF      EQU   Ø
OUTREC   DS    CL133
GRPNAME  DS    CL8
```

```
GRPNMLEN DS    H
WALEN    EQU   *-WORKAREA
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
         IRRPRXTW
         ICHPRCVT
         CVT   DSECT=YES
         END
```


Pentland Utilities review: part 2 – the rest

Part 1 of this article reviewing Nigel Pentland's DOS/Windows-based RACF utilities (*RACF Update* 31, February 2003, pp 14-28) concentrated on the following categories:

- History
- Set-up and use
- General reports
- User ID reports
- Group reports.

In this second, and final, part, we focus on the remaining utilities:

- Dataset reports

- General resource profile reports
- CICS reports.

You can download a copy of these utilities from Nigel's Web site, located at http://www.nigelpentland.co.uk. You can also find them on the CBTTAPE and the CBTTAPE CD. There are currently 72 utilities in all (see Appendix A for a full list).

Before we continue the review, I'd again like to take this opportunity to thank Nigel Pentland; first, for developing these utilities and, second, for reviewing the draft of these articles to make sure there weren't any obvious mistakes.

I'd also like to reiterate that this review covers my own experience and opinion of the various utilities; your view of the Pentland Utilities may be quite different from mine, based on your own experience. But if you've never used these utilities, here's a chance to get an idea of what you're missing.

DATASET REPORTS

The 12 Pentland programs that deal with dataset profiles are as follows (note: an asterisk next to the program name means that it generates JCL):

- *RACF07*. List dataset access
- *RACF23**. List/migrate dataset profiles
- *RACF42*. List discrete profiles with ALTER access
- *RACF56**. List all DATASET and JESSPOOL with UACC > NONE
- *RACF58**. List all profiles with NOTIFY set specified ID or ALL
- *RACF59*. List all profiles with non-default audit attributes
- *RACF61**. Grant access to some profiles with prefix for ID and access

- *RACF68*. Process DSMON output to check APF DATASET profiles
- *RACF71*. List dataset access (quick with no names)
- *RACF80**. List discrete dataset profiles and JCL to convert (except DFHSM)
- *RACF89*. Report on profiles with SECURITY in installation data
- *RACF90**. List/re-create all dataset profiles.

RACF 07 is a quite detailed listing of all dataset profiles beginning with a certain prefix. You define that prefix in the command line. So, if you type the following command:

```
RACFØ7 SYS
```

you'll get a listing of every single dataset profile beginning with SYS. It also gives you the IDs, names, and access levels of each user or group attached to the profile(s).

Note that when using this kind of high-level qualifier, you'll get a very large listing and it will take quite some time to generate. However, this can be very useful, especially for system files or other sensitive programs or data stored on DASD.

RACF 23 is similar, in that you get dataset profiles starting with a prefix. However, this goes further and generates the JCL to recreate each profile, including the PERMIT commands for each user or group at their existing level. The JCL even includes the installation data (nice touch, Nigel). The HTML output isn't quite as readable as RACF07, but its true value is in the JCL generation.

RACF 42 searches out all the DISCRETE profiles with ALTER access. DISCRETE and ALTER are a bad combination, because, if a user has ALTER access, they have full administrative capabilities over the profile. This covers not only dataset profiles but general resource and CICS profiles as well.

RACF 56 lists all dataset and JESSPOOL profiles with a UACC

greater than NONE. It also creates two JCL packets. The first grants access to a user ID, which you specify in the command line, to all of the profiles at their current UACC level. The second alters the UACC of all the profiles to NONE. This can be quite helpful if you're trying to keep your entire facility at a UACC=NONE state, and simplifies the maintenance of the database. However, the [ignore] option that Nigel has included after the user ID doesn't seem to make an operational difference to the utility, so proceed with caution if using that particular statement.

(Comment from Nigel Pentland: "Guess this is a case of the ignore option not being properly explained. The ignore option only affects the raw text output file RACF56 [no suffix], which is a list of the dataset profiles. This list can then be used as input for RACFJCL or similar for further processing." Thanks for the clarification, Nigel.)

RACF 58 lists all of the profiles (dataset and general resource) with NOTIFY set, or those with a specific user ID set to notify. This is quite nice for determining who is being informed of actions against datasets, transactions, etc. It also generates JCL to remove all of the NOTIFY flags. This is a good thing to check on every once in a while, to determine whether your system is trying to notify a non-existent user ID.

RACF 59 shows all the dataset profiles with non-default audit attributes. I rather like this one because it lets me know what sensitive dataset profiles are being audited, and whether they're being audited effectively. It also has the added bonus of showing any user IDs with the UAUDIT function switched on. This lets you know who you should be keeping an eye on, and clues you in on whether you should expand that list (Note: expand that list with caution. The more UAUDITs you use, the more system resources you're consuming. Don't go wild and UAUDIT everyone.)

RACF 61 creates JCL to grant access to some profiles with a given prefix for a specified ID (user or group). It excludes from the list and the JCL any profiles where the ID already has the authority level requested. The syntax for the command is

```
RACF61 PREFIX ID ACCESS
```

and it runs quite quickly, even if you're using an HLQ such as SYS. It's very simple, but quite effective if you need to give a user or group a certain level of authority over a large number of files with the same HLQ.

RACF 68 uses two input files – the IRRDBU00 flatfile and the DSMON output file – and generates a listing of all the fully-qualified APF-authorized libraries. It then identifies all of the fully-qualified dataset profiles which match the APF list, and flags any dataset profiles that are not (yet) fully qualified. Since APF-authorized libraries need added protection, it's better to not use asterisked profiles. (You'll note that I don't say generic – it's my opinion that all profiles should be generic, even the fully-qualified ones; no profiles should be discrete.) When you review the list, you can then create fully-qualified dataset profiles for those libraries that currently lack them. Very slick.

RACF 71 is pretty much the same as RACF07, except that it lacks the user names. It's designed to run faster than RACF07, and that's just what it does. Personally, I don't mind the extra processing time if it makes a more comprehensive and understandable report, but that's just me.

RACF 80 is a personal favourite in the dataset utility realm. It seeks out all DISCRETE dataset profiles, and creates the JCL to convert them to generic. Since I don't like to see discrete profiles (see above), this is a great tool. I find that the report is especially helpful, since my first instinct is to review it for anyone with ALTER access. The JCL that's generated simply deletes the profile and recreates it as a generic, with all of the access authorities still intact. I can't begin to tell you the number of problems this little gem has averted in my operation.

RACF 89 was a utility I requested from Nigel some years back (yes, folks, he does take requests – when he's got the time and when he thinks there will be widespread appeal). It seeks out any profiles (dataset, GenRes, CICS, everything except user IDs) with the word SECURITY somewhere in the installation data, to help keep track of those profiles identified with the Security department. The only drawback to this one is that you have to

include that keyword into the profile's Installation Data (which I hope you're doing out there). Nigel, here's a thought: perhaps you could modify this utility to allow the user to specify the keyword for the search. That way, if some company used terms like 'confidential', 'secret', 'restricted', etc, they could tailor it for their own uses.

RACF 90 is a system recovery dream. It generates JCL to re-create each and every dataset profile you've got. That can be a real lifesaver, especially if you have a large system or lots of profiles. On one LPAR I worked on, that JCL ran over 64,000 lines. Do you even want to think about trying to recreate that by hand? Yikes! There's one drawback to the utility, however: it appears to truncate the installation data. However, this is a minor issue if you're trying to recreate thousands of profiles.

GENERAL RESOURCE PROFILE REPORTS

There are 20 Pentland programs that deal with general resource profiles (which can include CICS, but are more generic):

- *RACF12*. List general resource access
- *RACF19*. List general resource class pair
- *RACF20**. List/migrate (add) prefixed profile pairs
- *RACF25**. List/migrate (delete) prefixed profile pairs
- *RACF28*. List general resource class pair (initial letter)
- *RACF30*. List STARTED profiles
- *RACF32*. List all profiles with WARNING set
- *RACF48**. List/migrate (add) non-prefixed to prefixed profile pairs
- *RACF49**. List/migrate (delete) non-prefixed profile pairs
- *RACF51*. List prefixed profile pairs (friendly format)
- *RACF58**. List all profiles with NOTIFY set specified ID or ALL

- *RACF59*. List all profiles with non-default audit attributes
- *RACF62**. Grant access to some profiles with prefix for ID and access
- *RACF64**. List/migrate (add) non-prefixed to non-prefixed profile pairs
- *RACF65*. List general resource class
- *RACF67**. Set/clear NOTIFY on prefixed general resource class pair
- *RACF72*. Summary breakdown of general resource profiles
- *RACF85**. List/migrate (add) both prefixed and non-prefixed profile pairs
- *RACF89*. Report on profiles with SECURITY in installation data
- *RACF91**. List general resource class pair with JCL to re-create profile.

RACF 12 generates a list of all general resource profiles with a certain prefix. For example, if you wanted to list all the MQ queue profiles for development systems, and all of them were prefixed with 'D', you would enter the command

```
RACF12 MQQUEUE D
```

to get the listing. It does have one slightly weird quality, however, in that the HTML output seems to indent itself bit by bit from the left as you go down the report. I have no idea why. *(Comment from Nigel Pentland: "Without the strange looking indents it becomes a lot less readable, honest!")* It's still quite readable though, and gives you a list of the user IDs and names. This is similar to the RACF07 report for dataset profiles, and is quite useful.

RACF 19 lists general resource profiles for a member-class and group-class that you specify. If you were listing all CICS profiles, that would be TCICSTRN and GCICSTRN respectively. The

listing is quite detailed, and can be quite large (depending on how many profiles and transactions you have, of course), but is very readable and easy to decipher.

RACF 20 allows you to recreate profiles for a member-class, group-class, and prefix. The JCL is quite detailed (full installation data again – hooray!) and the report identifies all the user IDs and groups. It would be a bit nicer if it could include the name as well as the ID, but that's just my personal viewpoint.

RACF 25 is similar to RACF20, except that the JCL is for deleting access to the general resource profiles. This is helpful if you're cleaning up old profiles that are no longer in use. Actually, you can use RACF20 and RACF25 together. Rename the profiles in the RACF20 JCL to the new name, and then run RACF25 to remove conflicts between old and new.

RACF 28 lists general resource class-pairs and the initial letters of the member class. I like using this for CICS profiles where the transaction starts with a specific letter such as C (generally used for IBM-supplied transactions). You can extend the search parameters so that you can find profiles and access lists for specific transactions, like the ever-popular CEMT (a risky transaction to throw around).

RACF 30 shows all STARTED class profiles that contain any STDATA details. It's similar to what you get in the DSMON 'RACF Started Procedures Table Report', but is more readable and more convenient.

RACF 32 shows all general resource profiles with the WARNING attribute set. This makes finding those (hopefully) few profiles much easier.

RACF 48 works a lot like RACF20, but for non-prefixed general resource profile pairs. The syntax for this is

```
RACF48 MEMBER-CLASS GROUP-CLASS MEMBER-PREFIX GROUP-PREFIX
```

I've not worked with this one as much as others, but it does enable you to create a new member and group prefix from an existing set of profiles.

RACF 49 is similar to RACF25, but again for non-prefixed general resource profile pairs. It generates the JCL to delete all of the profiles. However, if you're using this for CICS, it removes the profiles but not the specific CICS transactions.

RACF 51 is a kind of super-duper version of RACF20, RACF25, RACF48, and RACF49. It lists prefixed profile pairs in a much more readable format (although it doesn't generate any JCL).

RACF 58 and **RACF 59** were mentioned up in the dataset section, so we won't go over them again here.

RACF 62 grants access to some general resource profiles with a prefix for the ID and the level of access. You can do a cheat on this one and generate the report and JCL for all the profiles, as long as you put a fictitious group name here. This is similar to the former RACF26 report, which was, sadly, discontinued when Nigel upgraded the package (see my description in the 'Commentary' section below). It's a limited function utility (for special needs and situations), but it's efficient and works pretty well.

RACF 64 is similar to RACF48 and RACF51 in structure. It's generally a likeable bit of code, and its only real drawback is that it doesn't include specific transactions if you're trying to recreate CICS profiles. Ah, well, there are other utilities that do that. Also, the report is a bit sparse.

RACF 65 is a real kicker. It lists a general resource class in excellent detail. It doesn't produce any JCL, but is a great report. My only gripe (and it's a very minor one) is that it would be better if the order were reversed, so that the user/group list was at the top of the section with the member list immediately below it. Like I said, a very minor complaint. Other than that, it's a great report.

RACF 67 will either set or clear the NOTIFY flag on a series of prefixed general resource class pairs. If you include a user ID as the last parameter, the JCL will be created to set all the profiles to notify that ID. If you omit the ID, the JCL created sets NONOTIFY to all the profiles. The report isn't all that detailed, though. It just lists the profiles – nothing else.

RACF 72 is a summary report that gives you a breakdown of all the general resource profiles in your flatfile. It's a bit of an extension of RACF01, which summarizes all the records. It's a nifty bit of code to help you identify all the different classes you have within the GenRes environment.

RACF 85 is really another RACF64, but it allows you to generate the list and JCL without the need for a prefix.

RACF 89 was mentioned back in the dataset section, so we won't bother with it again here.

RACF 91 is the Rolls-Royce of general resource profile JCL generators. It gives you all the profiles, transactions, authorities, etc. The report is also very readable. It's a very good way to have a JCL back-up of your general resource profiles (and is similar in concept to what RACF90 does for dataset profiles).

You may have noticed that some of the items above seem to be repeats, or have only vague differences between them. I believe that this was intentional on Nigel's part. He apparently developed these different reports for different specific needs or requests, hence the similarities. And I must say that it's nice to have a choice of format – much better than being stuck with only one option.

CICS REPORTS

There are six Pentland programs that deal specifically with CICS resource profiles:

- *RACF16*. List CICSTRN profiles
- *RACF22*. List of TCICSTRN transactions with descriptions
- *RACF33*. List CICSTRN profiles sorted by TRANCODE
- *RACF34*. List of duplicate CICSTRN profiles
- *RACF35*. Compare prefixed CICSTRN profiles
- *RACF37**. List CICSTRN profiles and members for a group or user.

RACF 16 is a fast little CICS report generator, but it pumps out some large data. It includes all the transactions, users, groups, and even the installation data. The only problem it raises is the size of the report. Don't get me wrong, though, that's not the utility's fault. You see, I work in an organization that operates like this: "Welcome to XYZ Corp. Here's your ID badge, here's your desk, here's your terminal, and here's your own personalized CICS region". When you have more than 30 CICS regions in a single LPAR, and are covering well over 25,000 transaction profiles, the report is bound to get a little crowded! But, to Nigel's credit, the report actually makes some sense out of a really messed up CICS structure.

RACF 22 can give you a list of all the CICS profiles with installation data. Not bad as it stands, but a couple of things could be improved. First, it all comes out lower case (I've no idea why). Second, and this is a complaint I've made before, I think it would be more helpful if the output were in Courier font, and the data was parsed to look like the output on a terminal. Just my opinion, folks.

RACF 33 is one of my two favourite utilities (RACF 34 is the other one). It creates a report sorted by CICS transaction, so you can see where all of your CEMTs and CEDAs and all your other transactions are located (as well as in which CICS regions). The report includes the installation data (nice touch!). It's extremely helpful to your RACF admin staff when they're trying to look up a transaction to see in which regions it resides. Besides, this way they don't constantly have to type the command RLIST TCICSTRN regionname.transactionID RESG in TSO to search for the location of a specific transaction.

RACF 34 goes one better. It finds duplicate entries of the same transaction in the same CICS region. This is really important to know so that you don't end up with access authority conflicts on a transaction.

A word of warning about RACF33 and RACF34. They're great utilities, but they take a very long time to run. Of course, I'm

running Windows on a Celeron 300 MHz processor (does anybody out there remember when 300 MHz was considered fast?) so it takes quite a while just to start NotePad.exe. Generally, if you're going to run these jobs, do them overnight. Start them up before you leave the office, shut down every other program and/or utility, kill the screen saver, turn off the screen, and pray that they're finished by the time you return the following morning. Better yet, start them up at the beginning of your weekend. This isn't a complaint, by the way. The complexity of the reports and the sorting and recursive nature of the processing mean that they're bound to take a long time. And the results are worth the wait.

Oh, and one other thing. If your shop's like mine, and your number of CICS regions feels like infinity minus 2, expect the RACF33 report to be BIG. On one of my LPARs, the report runs about 7 megabytes. Compare that with the size of the flatfile that spawned it, which is 18 megabytes. Ouch!

RACF 35 is used to compare member/group profiles from two different systems. This is quite helpful if you want to compare prefixed profiles between development and test, or test and production. It doesn't generate any JCL, but it does give you a good idea of where the differences are between systems.

RACF 37 is our final contestant. It's a cross-reference report, similar to RACF11 for user IDs and groups, and far better than IRRUT100. It shows all of the transaction profiles (and transactions) available to a specific user ID, and at what level of access. The JCL generated is just a series of PERMIT commands for that user ID to the various profiles. This is a nice addition, especially if you want to clone that access for another ID or group.

Once again, let me remind you that you can use many of the general resource utilities in the section above to handle CICS reporting and JCL generation. These last six utilities were built specifically for CICS, however, and aren't meant to handle non-CICS profiles.

In the first article, I mentioned that Nigel jettisoned 24 of the old utilities when he upgraded his code from DOS 16-bit to Windows 32-bit. Some were repetitive, some were too situation-specific, and some just didn't work all that well. Some, however, were ones that I rather liked (and luckily, I kept a back-up copy of them – sneaky, aren't I?). I'd like to share my thoughts on them with you, in the hope that Nigel will resurrect them in a future release.

RACF17 was virtually the same as RACF16, except that it handled CCICSCMD and VCICSCMD profiles instead of TCICSTRN and GCICSTRN. Yes, I know I could run RACF16 with the different parameters, but hey, I'm lazy. I don't want to have to go and rename the files. A separate report would be rather nice to have.

RACF26 was a little gem that granted access to all profiles in a class/group-class pair. This was a really nice thing to run from time to time, just to make sure you had the JCL available to fix a major screw-up. The support staff got too much access in the short term, no doubt, but when they're in headless chicken mode, you don't want to argue.

RACF27 was the way to reverse RACF26. Remember, once the panic subsides, your support staff have to return to real life, as does your system.

RACF29 would remove redundant access authorities to dataset and general resource profiles. In other words, if the UACC was READ, it would remove any profiles that also had READ. Nigel said in his description of the utility at the time that it never managed to develop into what he intended. Well, maybe not, Nigel, but it did have the advantage of actually working.

RACF39 and **RACF40** were similar in function to RACF33 and RACF34, except once again they were for the CCICSCMD/VCICSCMD class pair instead of TCICSTRN/GCICSTRN. Same reason here for renewal as for RACF17 – I'm lazy.

RACF54 and **RACF55** would generate a combined list of user

IDs and names from two flatfiles, and then annotate them. I liked being able to show management a list of all the users at one time. If you do ever revive this one, Nigel, perhaps you can format it so that it lists a separate column for each LPAR's ID, to show where people are on LPAR-1 or LPAR-2 or both.

CONCLUSION

In general, most people look at free utilities with a jaded eye. It's the 'you get what you pay for' attitude, I suppose. And granted, these aren't the fancy, on-line, real-time, processing reports that you get with RACF add-on reporting packages. But then again, they're not meant to be. For something that's free, these utilities offer a whole lot of value. Value in saved time, saved effort, and discovery of security problems or conflicts you may not have even thought of before.

Nigel, you could probably have sold this collection of utilities and made a bit of cash. Instead, you made them freely available to the worlds of security and audit. So from all of us who use and value this collection of programming gems, a big THANK YOU!

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

Appendix A: Summary of all Pentland Utilities

U=User ID, G=Group, D=Dataset, R=General Resource, C=CICS, X=General Function

Utility	*U*	*G*	*D*	*R*	*C*	*X*	*Key*	*Key words*
RACF00						_		*Pre-processor*
RACF01	_	_	_	_	_	_		*Summary*
RACF02	_						*JCL*	*Non-existent user IDs*
RACF03		_						*Group tree*
RACF04		_					*JCL*	*All groups*

Utility	U	G	D	R	C	X	Key	Key words
RACF05	_						JCL	Expired user IDs
RACF06		_					JCL	List group
RACF07			_					Dataset (mask)
RACF08	_						TEXT	User ID(s) (not-HTML)
RACF09	_						JCL	User IDs (mask)
RACF10								Discontinued
RACF11	_	_					JCL	XREF (JCL to grant)
RACF12				_				General resources (mask)
RACF13								Discontinued
RACF14								Discontinued
RACF15								Discontinued
RACF16					_			List member/group class
RACF17								Discontinued
RACF18	_						TEXT	All user IDs (not-HTML)
RACF19				_				General resource
RACF20				_			JCL	General resource – re-create – prefixed
RACF21	_	_					JCL	XREF (JCL to remove)
RACF22					_			Member class installation data
RACF23			_				JCL	Dataset – re-create – prefixed
RACF24	_						JCL	Revoked user IDs
RACF25				_			JCL	General resource – delete – prefixed
RACF26								Discontinued
RACF27								Discontinued
RACF28				_				General resource (prefix.mask)
RACF29								Discontinued
RACF30				_				STARTED
RACF31								Discontinued
RACF32				_				WARNING
RACF33					_			Sorted member/group class pair
RACF34					_			Duplicate member/group class pair
RACF35					_			Compare member/group class pair
RACF36	_	_					JCL	Compare group
RACF37					_		JCL	XREF – member/group class pair
RACF38	_							Audit report

Utility	U	G	D	R	C	X	Key	Key words
RACF39								Discontinued
RACF40								Discontinued
RACF41								Discontinued
RACF42			_					discrete ALTER
RACF43								Discontinued
RACF44								Discontinued
RACF45								Discontinued
RACF46	_						JCL	Delete user IDs
RACF47		_					JCL	Change group
RACF48				_			JCL	General resource – re-create – non-prefixed
RACF49				_				General resource – delete – non-prefixed
RACF50		_						Connected groups
RACF51				_				General resource friendly format
RACF52	_	_						OwnerID
RACF53		_						List group
RACF54								Discontinued
RACF55								Discontinued
RACF56			_				JCL	UACC
RACF57								Discontinued
RACF58			_	_			JCL	Notify
RACF59			_	_				Audit attributes
RACF60								Discontinued
RACF61			_				JCL	Dataset – grant access – prefixed
RACF62				_			JCL	General resource – grant access –- prefixed
RACF63								Discontinued
RACF64				_			JCL	General resource – re-create – non-prefixed
RACF65				_				General resource
RACF66	_							Listuser
RACF67				_			JCL	Notify
RACF68			_					APF
RACF69	_	_					JCL	Revoked connections
RACF70		_						Count connections

Utility	U	G	D	R	C	X	Key	Key words
RACF71			_					Dataset (mask)
RACF72				_				Summary (general resources)
RACF73		_						Connected groups
RACF74								Discontinued
RACF75		_					TEXT	List group (non-HTML)
RACF76		_					JCL	List group
RACF77		_					JCL	Connections
RACF78								Discontinued
RACF79	_						JCL	User IDs
RACF80			_				JCL	Discrete dataset
RACF81								Discontinued
RACF82	_							Annotate
RACF83								Discontinued
RACF84	_							User IDs
RACF85				_			JCL	General resource
RACF86	_						JCL	LIMBO
RACF87		_						GID
RACF88	_							UID
RACF89			_	_				SECURITY
RACF90			_				JCL	Dataset
RACF91				_			JCL	General resource – re-create
RACF92	_						JCL	User IDs (detailed)
RACFAWK		_	_	_	_	_	_	Ad-hoc (non-HTML)
RACFDIAG							_	Database diagnostic
RACFJCL	_	_	_	_	_	_	JCL	JCL generator (similar to CLIST)

EKC has announced its new EKC Personal Access List (E-PAL), offering fast assignment of dataset or resource access without impacting current user access, security implementation, or established security policy.

E-PAL is designed to control additional access for special projects, or temporary access for consultants, without any modifications to IBM RACF profiles. All access granted under E-PAL may be broader, or more restrictive, than what is otherwise permitted. SMF records are produced, and a program to report successful access, as well as access denied, is provided.

URL: http://www.ekcinc.com/SalesPDF/epalprodannouncement.pdf

* * *

BMC Software has upgraded its signature access control and provisioning solution, CONTROL-SA, adding:

- More deployment and workflow options
- Extended enterprise offerings
- A virtual directory with LDAP interface.

It has also announced that its management GUI will soon be viewable as either an MMC-based console or via a Web browser.

URL: http://www.bmc.com/news_events/

* * *

CA has announced that eTrust Web Access control is now available on mainframe Linux, enabling organizations deploying Linux and open source technology to take advantage of Linux platforms for critical enterprise applications.

URL: http://www3.ca.com/press/PressRelease.asp?CID=38915

* * *

Consul and Janus Risk Management have announced a partnership that leverages Consul's security audit and management software with Janus' enterprise security methodologies to offer corporations a comprehensive security event management solution.

URL: http://www.consul.com/index.php3?cid=563

* * *