February 2004

In this issue

3  Access authority for datasets
11 Protecting OS/390 authorized libraries
16 Using IRREVX01 and the RACF database to help differentiate REVOKED userids
34 Easy LDAP program to authenticate RACF userids
43 Auditing RACF protection
55 RACF in focus – implementing OS/390 Unix controls
61 August 1995 – February 2004 index
65 RACF news

© Xephon Inc 2004
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 $290.00 in the USA and Canada; £190.00 in the UK; £196.00 in Europe; £202.00 in Australasia and Japan; and £200.50 elsewhere. The price includes postage. Individual issues, starting with the August 2000 issue, are available separately to subscribers for $72.75 (£48.50) each including postage.

Contributions
When Xephon is given copyright, articles published in RACF Update are paid for at the rate of $160 (£100 outside North America) per 1000 words and $80 (£50) per 100 lines of code for the first 200 lines of original material. The remaining code is paid for at the rate of $32 (£20) per 100 lines. To find out more about contributing an article, without any obligation, please download a copy of our Notes for Contributors from www.xephon.com/nfc.

© Xephon Inc 2004. All rights reserved. None of the text in this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior permission of the copyright owner. Subscribers are free to copy any code reproduced in this publication for use in their own installations, but may not sell such code or incorporate it in any commercial product. No part of this publication may be used for any form of advertising, sales promotion, or publicity without the written permission of the publisher.

Printed in England.
Access authority for datasets

The DSRAT REXX procedure presented here supports the following RACF activities for datasets:

- **LISTDSD (List Data Set Profile) command.** Use the LISTDSD command to list information included in tape and DASD dataset profiles. A dataset profile consists of a RACF segment and, optionally, a DFP or TME segment. The LISTDSD command provides you with the choice of listing information contained in either the entire dataset profile (all segments) or only a specific segment of it.

- **ADDS D (Add Data Set Profile) command.** Use the ADDSD command to add RACF protection to datasets with either discrete or generic profiles.

- **ALTDSD (Alter Data Set Profile) command.** Use the ALTDSD command to modify an existing discrete or generic dataset profile.

- **DELDSD (Delete Data Set Profile) command.** Use the DELDSD command to remove RACF protection from tape or DASD datasets that are protected by either discrete or generic profiles.

DSRAT runs in the ISPF environment, and the code below shows how to find datasets in ISPF 3.4

```
DSLIST - Data Sets Matching SYSADM.DBC*                       Row 1 of 6
Command ===>                                            Scroll ===> PAGE
Command - Enter "/" to select action                  Message     Volume
------------------------------------------------------------------------
SYSADM.DBC.TRAN                                          MVSØØ3
DSRAT    SYSADM.DBCLONEX                                          MVSØØ3
SYSADM.DBCLONEY                                          T14CAT
SYSADM.DBCLONE1                                          T14CAT
SYSADM.DBCLONE4                                          MVSØØ3
SYSADM.DBCLONE6                                          T14CAT

The main menu of the DSRAT procedure is as follows:

```
The parameter entry for the ADDSD command is as follows:

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>PARAMETER VALUE</th>
<th>PROMPT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Profile command</td>
<td>=&gt; ADDSD</td>
<td></td>
</tr>
<tr>
<td>Profile name</td>
<td>=&gt; 'SYSADM.DBCONEX'</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>=&gt; ______</td>
<td>Model-Tape-Generic or blank</td>
</tr>
<tr>
<td>Volume serial</td>
<td>=&gt; ______</td>
<td>If data set is not cataloged</td>
</tr>
<tr>
<td>Unit</td>
<td>=&gt; ______</td>
<td>If Volume serial specified</td>
</tr>
<tr>
<td>Password</td>
<td>=&gt; ______</td>
<td>Data set password</td>
</tr>
<tr>
<td>Owner</td>
<td>=&gt; IN1048</td>
<td>Userid or group name</td>
</tr>
<tr>
<td>Level</td>
<td>=&gt; 0</td>
<td>0-99</td>
</tr>
<tr>
<td>UACC</td>
<td>=&gt; READ</td>
<td>None-Read-Update-Control-Alter-Execute</td>
</tr>
<tr>
<td>Access attempt</td>
<td>=&gt; ALL</td>
<td>All-Failures-None-Success or blank</td>
</tr>
<tr>
<td>Access level</td>
<td>=&gt; READ</td>
<td>Read-Update-Control-Alter or blank</td>
</tr>
<tr>
<td>Indicator</td>
<td>=&gt; SET</td>
<td>Set-Noset-Only</td>
</tr>
<tr>
<td>Notify</td>
<td>=&gt; SYSADM</td>
<td>Userid</td>
</tr>
<tr>
<td>Erase on delete</td>
<td>=&gt; YES</td>
<td>Yes or blank</td>
</tr>
</tbody>
</table>

Enter parameters values for the ADDSD Profile service | PF3 Return
DSRAT has the following components, presented in turn below:

- **DSRAT** – the driver procedure
- **DSRATM** – the main menu
- **DSRATP** – list message panel
- **DSRATA** – add, alter and delete panel.

**DSRAT**

```rexx
/* REXX */
/* trace r */
parse upper arg dsn
if dsn='' then do
  say 'The procedure DSRAT valid only in ISPF member list.'
  Exit
end
TOP:
x=''
date=DATE()
time=TIME(C)
address ispexec 'addpop row(6) column(15)'
address ispexec "display panel(dsratm)"
if rc=8 then Exit
address ispexec 'rempop'
ds = 'dataset('||dsn||') all'
if x=1 then do
  q=outtrap('var.')
  address tso "listdsd" ds
  q=outtrap('off')
  Call Report
end
if x=2 then do
  profc='ADDSD'
  address ispexec "display panel(dsrata)"
  if rc<8 then do
    option='('||dsn||')'
    if rpas ¬= ' ' then option='('||dsn||' '/'||rpas||')'
    option=option||rtype||'
    if rvol ¬= ' ' then option=option||'VOLUME('||rvol||') UNIT('||runit||') '
    if rowner ¬= ' ' then option=option||'OWNER('||rowner||') '
    if rnot ¬= ' ' then option=option||'NOTIFY('||rnot||') '
    option=option||'LEVEL('||rl||') '
    option=option||'UACC('||ruacc||') '
    option=option||rinck||'
    if rasuc ¬= ' ' then
```
option=option||'AUDIT('||rasuc||'|'rafal|'))'
if rye='YES' then option=option||'ERASE'
q=outtrap('var.')
address tso "addsd" option
q=outtrap('off')
Call Report
end

if x=3 then do
prof='ALTDSD'
address ispexec "display panel(dsrata)"
if rc<8 then do
option=('|dsn|')
if rpas=' ' then option=('|dsn|'/'|rpas|')
option=option||rtype||'
if rowner=' ' then option=option||' OWNER('||rowner|')'
if rnot=' ' then
option=option||' NOTIFY('||rnot|')'
else option=option||' NONOTIFY'
if rl=' ' then rl=Ø
option=option||'LEVEL('||rl|')'
option=option||'UACC('||ruacc|')'
option=option||rindk||'
if rasuc=' ' then
option=option||'AUDIT('||rasuc||'|'rafal|')'
if rye='YES' then option=option||'ERASE'
else option=option||'NOERASE'
q=outtrap('var.')
address tso "altdsd" option
q=outtrap('off')
Call Report
end

if x=4 then do
prof='DELDSD'
address ispexec "display panel(dsrata)"
if rc<8 then do
option=('|dsn|')
option=option||rindk
if rvol=' ' then option=option||' VOLUME('||rvol|')'
q=outtrap('var.')
address tso "deldsd" option
q=outtrap('off')
Call Report
end
end
Signal TOP
Report:
item=''}
address ispexec 'tbcreate "plist" names(item)'
do i=1 to var.Ø
    item = var.i
    address ispexec 'tbadd "plist"'
end
if item='' & (x=2 | X=3) then do
    item=dsn||' DEFINED TO RACF'
    address ispexec 'tbadd "plist"'
end
if item='' & x=4 then do
    item=dsn||' DELITED FROM RACF'
    address ispexec 'tbadd "plist"'
end
address ispexec 'tbtop "plist"'
address ispexec 'tbdispl "plist" panel(DSRATP)'
address ispexec 'tbend "plist"'
Return
Exit

DSRATM

)ATTR DEFAULT(%+_)[
] TYPE (OUTPUT) INTENS(LOW) COLOR(GREEN) CAPS(OFF)
# TYPE (OUTPUT) INTENS(LOW) COLOR(WHITE) CAPS(OFF)
) TYPE (TEXT) INTENS(LOW) COLOR(WHITE) CAPS(OFF) HILITE(REVERSE)
- TYPE (INPUT) INTENS(LOW) COLOR(YELLOW) CAPS(ON) HILITE(BLINK)
| TYPE (OUTPUT) INTENS(LOW) COLOR(GREEN) CAPS(OFF)
+ TYPE (TEXT) INTENS(LOW) COLOR(GREEN)
/ TYPE (TEXT) INTENS(LOW) COLOR(TURQ)
~ TYPE (TEXT) INTENS(HIGH) COLOR(TURQUOISE)
@ TYPE (TEXT) INTENS(HIGH) COLOR(RED) CAPS(OFF) HILITE(REVERSE)
)BODY WINDOW(41,14) EXPAND ($$)
+] + Date: |date +
+] Data Set Profile + Time: |time +
+] + User: &zuser
/ ******************************************************
+
+ [ row1 +
+ [ row2 +
+ [ row3 +
+ [ row4 +
+ [ row5 +
+
/ ******************************************************
+@==>+ _X+ #msg +
+ ] PF3 End +
)INIT
&row1= '1 · L ISTDSD - List Data Set'
&row2= '2 · ADDSD - Add Data Set'
&row3= '3 - ALTDSD - Alter Data Set'
&row4= '4 - DELDSD - Delete Data Set'
IF ( &X = 1, 2, 3, 4, X)
  &msg = ''
ELSE
  ATTR (msg) = 'COLOR (RED)'
  &msg = 'Enter 1, 2, 3, 4 or X.'
IF ( &X = 1)
  .ATTR (row1) = 'COLOR (YELLOW) CAPS(ON)'
IF ( &X = 2)
  .ATTR (row2) = 'COLOR (YELLOW) CAPS(ON)'
IF ( &X = 3)
  .ATTR (row3) = 'COLOR (YELLOW) CAPS(ON)'
IF ( &X = 4)
  .ATTR (row4) = 'COLOR (YELLOW) CAPS(ON)'
) PROC
IF (.PFKEY = PFØ3) &PF3 = EXIT
) END

DSRATP
)
Attr Default( %+ _ )
| type( text)   intens(high) caps(on ) color(yellow)
$ type( output) intens(high) caps(off) color(yellow)
? type( text)   intens(high) caps(on ) color(green) hilite(reverse)
# type( text)   intens(high) caps(off) hilite(reverse)
} type( text)   intens(high) caps(off) color(white)
[ type( input) intens(high) caps(on ) just(left )
] type( input) intens(high) caps(off) just(left ) pad(' _ ')
^ type( output) intens(low ) caps(off) just(asis ) color(green)
) Body   Expand( ///
%-/-/-   ? List Data Set Profile +%-/-/-
%Command ==>_zcmd                                 / /%Scroll ==>_amt +
+Data Set Profile[dsn
+---------------------------------------------------------------
) Model
- z
+ ) Init
  .ZVARS = ' (item)'
  &amt = PAGE
  &cmd = ''
) Reinit
) Proc
) End
DSRATA

)Attr Default(%+_)
| type(text) intens(high) caps(on) color(yellow)
$ type(output) intens(high) caps(off) color(yellow)
? type(text) intens(high) caps(on) color(green) hilite(reverse)
# type(text) intens(high) caps(off) hilite(reverse)
) type(text) intens(high) caps(off) color(yellow) hilite(reverse)
[ type(input) intens(high) caps(on) color(green) pad(_)
)Body Expand(/)/
| /-/-/- Parameter Entry /-/-/
%Command ===>_zcmd
+
+
#PARAMETER #PARAMETER VALUE #PROMPT
+
+Profile command =>$prof c +
+Profile name =>$dsn +
+Type =$rtype + Model Tape Generic or blank
+Volume serial =$rvol + If data set is not cataloged
+Unit =$runit + If Volume serial specified
+Password =$rpas + Data set password
+Owner =$rowner + User id or group name
+Level =$rl + Ø-99
+UACC =$ruacc + None Read Update Control
Alter Execute
+Access attempt =$rasuc + All Failures None Success or blank
+Access level =$rafal + Read Update Control Alter or blank
+Indicator =$rindk + Set Noset Only
+Notify =$rnot + User id
+Erase on delete =$rye + Yes or blank
+$msg +
+
)Init
if (&x='2')
   $msg = 'Enter parameters values for the ADDSD Profile service |'
if (&x='3')
   $msg = 'Enter parameters values for the ALTDSD Profile service |'
if (&x='4')
   $msg = 'Enter parameters values for the DELDSD Profile service |'
if (&x='3')
   .attr (rtype) = 'type(output) color(white)'
   .attr (rvol) = 'type(output) color(white)'
   .attr (runit) = 'type(output) color(white)'
if (&x='4')
   .attr (rnot) = 'type(output) color(white)'

if (&rtype ≠ ' ')
   .attr (rtype) = 'pad(nulls)'
if (&rvol ≠ ' ')
   .attr (rvol) = 'pad(nulls)'
if (&runit ≠ ' ')
   .attr (runit) = 'pad(nulls)'
if (&rpas ≠ ' ')
   .attr (rpas) = 'pad(nulls)'
if (&rowner ≠ ' ')
   .attr (rowner) = 'pad(nulls)'
if (&rl ≠ ' ')
   .attr (rl) = 'pad(nulls)'
if (&ruacc ≠ ' ')
   .attr (ruacc) = 'pad(nulls)'
if (&rasuc ≠ ' ')
   .attr (rasuc) = 'pad(nulls)'
if (&rafal ≠ ' ')
   .attr (rafal) = 'pad(nulls)'
if (&rindk ≠ ' ')
   .attr (rindk) = 'pad(nulls)'
if (&rnot ≠ ' ')
   .attr (rnot) = 'pad(nulls)'
if (&rye ≠ ' ')
   .attr (rye) = 'pad(nulls)'
)

) Reinit
) Proc

&rt = TRUNC(&rtype, ' ')
if (&rt=' ')
   &rtype = ' '
if (&rt='M' | &rt='MO')
   &rtype = 'MODEL'
if (&rt='T' | &rt='TA')
   &rtype = 'TAPE'
if (&rt='G' | &rt='GE')
   &rtype = 'GENERIC'
&ru = TRUNC(&ruacc, ' ')
if (&ru='N' | &ru='NO')
   &ruacc = 'NONE'
if (&ru='R' | &ru='RE')
   &ruacc = 'READ'
if (&ru='U' | &ru='UP')
   &ruacc = 'UPDATE'
if (&ru='C' | &ru='CO')
   &ruacc = 'CONTROL'
if (&ru='A' | &ru='AL')
   &ruacc = 'ALTER'
if (&ru='E' | &ru='EX')
   &ruacc = 'EXECUTE'
&rf = TRUNC(&rasuc, ' ')
if (&rf='A' | &rf='AL')
   &rasuc = 'ALL'
if (&rf='N' | &rf='NO')
   &rasuc = 'NONE'
WHAT ARE OS/390 AUTHORIZED LIBRARIES?

As the name implies, OS/390 (or MVS, or z/OS) authorized libraries contain special, ‘authorized’ programs. These programs require special powers or capabilities to perform ‘authorized’ functions, and are therefore kept in authorized libraries.

All OS/390 installations need to use authorized programs. Typically, these programs are part of the operating system or are extensions of it. They routinely perform functions that are outside the realm of ordinary programs, and are kept in authorized libraries to distinguish them from other programs.

First, there are libraries that are part of the core IBM operating system. Examples in this category are SYS1.LINKLIB and
SYS1.LPALIB. IBM supplies these, and many other libraries, as part of the OS/390 operating system.

Next, you may have some libraries that are extensions of the operating system. These include software supplied by vendors other than IBM. For example, you may have a Tape Management System from a vendor other than IBM, and this vendor may have specified that their library ABC.AUTHLIB needs to be an authorized library.

You may also have ‘home-grown’ software that performs system-type functions and requires authorized programs, and therefore authorized libraries.

THE NEED TO PROTECT OS/390 AUTHORIZED LIBRARIES

Authorized libraries should be protected as much as, if not more than, say, production payroll files. However, while everyone understands and appreciates the reasons behind protecting payroll files, not many understand the real issues behind protecting authorized libraries.

In a nutshell, authorized libraries need to be protected because they contain programs that can compromise system integrity and invalidate all the work done at your installation.

The whole issue of OS/390 system integrity depends on the fact that ‘user’ programs – that is, all programs that are not authorized – have limited capabilities. They can at most damage or destroy their own data, but not other users’ data or system data.

Authorized programs, by contrast, have wide-ranging powers, since they perform system-wide functions. These programs have a legitimate need to control all other work that goes on in the system. Left unguarded, they can be inappropriately modified in order to perform unintended functions.

SPECIFIC SECURITY CONCERNS

One of the security concerns is what would happen if someone succeeded in placing a wrong program in one of the authorized
libraries. Or what if someone succeeds in modifying an existing authorized program? There may be a disgruntled employee who wants to damage your system; or a prankster with malicious intent may want to plant a virus or some other bad code into it.

These individuals will look for weak controls at your installation and try to insert an illegal program into (or modify an existing program in) one of the authorized libraries, as that is the first thing they must achieve in order to do any significant damage. Therefore, authorized libraries are the prime targets of intruders wanting to penetrate a system.

If an illegal program can make its way into an authorized library, it will be considered authorized, and can do any of the following to your system. It can cause your system to crash unexpectedly (by changing some of the system programs). It can invalidate all your back-up copies on tape (by corrupting your disk back-up program). It can cause unexpected things to happen at random (by deleting an insignificant OS/390 module or replacing it with bad code). It can even slow down your entire machine.

It’s also possible to insert computer viruses in an OS/390 system, or any one of the other harmful bugs that have found their way into personal computers.

While this may sound frightening, it’s relatively easy to prevent these things from happening. Sufficient safeguards are available, but it is up to you to implement them.

Remember, if your authorized libraries are not closely guarded, you can institute all the security controls you want yet still be vulnerable to an attack. Having security controls but leaving authorized libraries unguarded is like closing all your windows at night, but leaving the front door wide open!

PROTECTING OS/390 AUTHORIZED LIBRARIES

So what can you do to protect your system?

First, you need to identify all your authorized libraries. Although you could ask your systems programmer for this information, the best way to get a list of all your authorized libraries is to run the
Data Security Monitor (DSMON) report with the ‘Selected Data Sets Report’ option. This will identify all your authorized libraries.

Next, you must take steps to ensure that sufficient protection exists for all authorized libraries found in this report. To do this, you’ll have to work with the systems programmer at your installation.

In particular, you should do the following:

• Keep authorized libraries to a minimum. Review the list of authorized libraries, and if any are identified as belonging to obsolete software or obsolete software versions, request the systems programmer to remove them from the authorized list.

• Make sure the remaining authorized libraries are protected by ‘fully qualified’, generic RACF profiles. For example, if you find SYS1.LINKLIB as an authorized library, make sure you have a generic profile called SYS1.LINKLIB. Make sure the Universal Access (UACC) on this profile is NONE, or READ at the most. It’s not sufficient to protect it with SYS.LINK*, for instance. The protection requirements for other libraries that this profile would protect may not be as stringent as those required for authorized libraries.

• Keep to a minimum the number of people with WRITE or UPDATE access to authorized libraries. For each of the profiles protecting authorized libraries, make sure only a few individuals in the systems programming group are in the access list with UPDATE, CONTROL, or ALTER access.

• When an authorized library is added to the system, make sure it’s protected by a ‘fully qualified’ generic RACF profile. When an authorized library is removed, ensure that the corresponding profile is also removed.

• Use established change control procedures to add or delete an authorized library. Ad hoc or undocumented changes to these libraries should not be allowed. Make sure you have controls in place for adding or deleting programs in these libraries.
• Produce specific reports that show RACF violations against authorized libraries, and pay particular attention to any violations you find in these reports.

• Conduct periodic reviews (once every three months is sufficient) of all authorized libraries, to determine their validity and to ensure that there is proper RACF protection for them.

OTHER PRECAUTIONS
In addition to this, people with special RACF powers such as SPECIAL or OPERATIONS attribute should be advised to guard their terminals while they are logged on to the system. Otherwise, your system is still vulnerable. Here are some examples:

• A person finds that someone with update access to authorized libraries (a systems programmer, for instance) has left their terminal signed on, but unguarded. This person walks to the terminal and copies an illegal program from a private library to an authorized library. The illegal program can now do damage at will.

A solution that would make such an event less likely to occur would be to employ OS/390 controls to log off a TSO user if there is no activity on the terminal for, say, 20 minutes.

• People with special RACF powers should be careful of computer games that may secretly introduce malicious code into the system.

SUMMARY
Security attacks can be prevented by safeguarding access to authorized libraries, by conducting periodic reviews to ensure that control mechanisms are in place, and by instilling security awareness among staff members with special RACF powers.

Dinesh Dattani
Information Security Consultant, Toronto (Canada)
Dinesh123@rogers.com

© Xephon 2004
Using IRREVX01 and the RACF database to help differentiate REVOKED userids

When deciding whether or not to reactivate (RESUME) a revoked userid, it helps if you can discover why it was revoked and why system access is being prevented by RACF. The two main reasons why system access will be denied by RACF are as follows:

- Repeated attempts to access the system by a given userid were not accompanied by the correct password for that userid. After the installation-defined limit for consecutive unsuccessful log-on attempts was reached, RACF REVOKED the userid.

- The userid is manually REVOKED by a RACF administrator either immediately or as a result of a preset REVOKE date being reached.

The first case is often referred to as ‘intruder lockout’, although in practice it’s typically caused by a legitimate user forgetting their password. The second situation generally involves some extenuating circumstance, such as an employee being terminated or a time limit being reached for a userid that has been granted temporary system access.

It’s not always obvious from the information in the RACF database why a userid is currently in REVOKE status:

- Where system access is being denied because a REVOKE date has been encountered, the REVOKEDT/RESUMEDT fields in the USER BASE segment for that userid contain sufficient information.

- Things are more complicated when the userid has been REVOKED for some other reason, because the same RACF database field flag (FLAG4) is used to indicate both that a userid has been revoked because of too many invalid log-on attempts and that the userid has been revoked as a result of
an ALTUSER command with a non-date-specific REVOKE operand. In short, the same flag is used to indicate two very different revoke reasons. This makes it difficult for a RACF administrator to know whether or not to proactively RESUME a userid based on a phone call from a locked-out user. The only way to determine the difference is to post-process RACF SMF records (type 80) to determine why a userid went into REVOKE status. If there are large amounts of SMF data and/or it’s not known when the state changed, it may be difficult to determine why the userid is revoked.

This article describes a technique that uses the RACF IRREVX01 command exit to recognize ALTUSER REVOKE requests that cause a userid to be switched into REVOKE status. When this condition is recognized, a second bit in the RACF FLAG4 USER BASE segment record field is set to indicate that the userid was administratively revoked. This gives the RACF administrator enough information to know whether to RESUME a userid or investigate further.

Two programs are provided with this article. The first is the RACF IRREVX01 exit program. This provides the command capture mechanism as well as the logic to determine whether the REVOKE status has changed between the pre-call to the exit and the post-call to the exit. If the userid has gone REVOKED, the post-call to the exit extracts the current FLAG4 value, sets the x'08' bit in that field, and then replaces that field value into the RACF database. (The normal REVOKE status has FLAG4 set to x'80'. The IRREVX01 exit will cause the FLAG4 field to be set to x'88' if a userid has gone into REVOKED status between the pre- and post-exit calls.)

The second program, RVKLST, provides the interface to examine the REVOKE status of userids in the RACF database, taking into account this use of the x'08' bit in the FLAG4 field. The program can run in two ways. If no userid is passed as a program parameter, the current revoke status of all userids is dumped to an output dataset. If a userid is specified in the program parameter, the REVOKE status for only that userid is listed in the output dataset.
The use of the additional bit setting in the FLAG4 field doesn’t compromise the normal use of this field by RACF, and, when a subsequent ALTUSER command is used to RESUME a userid that has had its FLAG4 field modified as described earlier, the flag is properly reset to x'00'.

PROGRAM LINKAGE AND EXIT MANAGEMENT
Both IRREVX01 and RVKLST should be linked into an authorized library with AC(1). The following linkedit control cards should be used:

- INCLUDE OBJECT(IRREVX01)
- ENTRY IRREVX01
- SETCODE AC(1)
- NAME IRREVX01(R)
- INCLUDE OBJECT(RVKLST)
- ENTRY RVKLST
- SETCODE AC(1)
- NAME RVKLST(R)

The IRREVX01 exit is eligible to be managed by the OS/390 dynamic exit manager and, as a result, it can be dynamically enabled and disabled with the following operator commands:

SETPROG EXIT,ADD,EXITNAME=IRREVX01,MODNAME=exitname,DSNAME=catalogd.dsn
SETPROG EXIT,DELETE,EXITNAME=IRREVX01,MODNAME=exitname

where ‘exitname’ is the member name of the IRREVX01 exit that resides in the catalogued dataset specified in ‘catalogd.dsn’.

This capability is useful for testing, but, once you’re satisfied with the functionality of the exit, you may choose to place it in your system link list.
USING THE RVKLST UTILITY

After the exit has been enabled, manually REVOKE some userids and then run the RVKLST program against your RACF database. The following JCL can be used to list the REVOKE status of all the userids in your RACF database:

```
//RVKLST   EXEC PGM=RVKLST1
//STEPLIB  DD   DSN=auth.dataset,DISP=SHR
//SYSPRINT DD   SYSOUT=*  
```

Running this job should produce results in the SYSPRINT output dataset similar to the following:

```
Userid    Revoke Status
--------  ---------------------
ABARS     Not REVOKED
APPC      Not REVOKED
ASCH      REVOKED
BLSJPRMI  REVOKED
BPXOINIT  Not REVOKED
BPXROOT   Not REVOKED
USER1     Administrator REVOKED
USER2     Not REVOKED
USER3     Not REVOKED
USER4     Date REVOKED
USER5     Not REVOKED
```

Optionally, the RVKLST program can be run with a userid specified as a program parameter. For example, running the following sample JCL:

```
//RVKLST   EXEC PGM=RVKLST1,PARM='USER1'
//STEPLIB  DD   DSN=auth.dataset,DISP=SHR
//SYSPRINT DD   SYSOUT=*  
```

would produce this output in SYSPRINT:

```
Userid    Revoke Status
--------  ---------------------
USER1     Administrator REVOKED
```

This article shows only the most basic method of invoking the RVKLST program. Optionally, a CLIST could be used to invoke the program from TSO, or logic could be added to convert the RVKLST program into a command processor, but that extends beyond the scope of this article, and such extensions are left to the reader. In any event, RVKLST uses the RACROUTE macro
call and requires the AC(1) linkedit attribute. If the program is invoked from TSO (either through a CALL or as a TSO command), the IKJTSO\textit{xx} PARMLIB member will need to be updated to reflect a new authorized program or command.

As can be seen, this small modification provides a very powerful extension to using RACF. Providing this type of information to RACF administrators can help make them more productive and efficient when dealing with REVOKED RACF users.

IRREVX01 EXIT PROGRAM

IRREVX01 CSECT
IRREVX01 AMODE 31
IRREVX01 RMODE ANY

***************************************************************************
*                                                                     *
*   THIS VERSION OF THE EXIT EXAMINES THE COMMAND BUFFER IN THE        *
*   PRE CALL TO DETERMINE IF THE REVOKE KEYWORD OF AN ALU COMMAND      *
*   HAS BEEN SPECIFIED. IF IT IS PRESENT, THE CURRENT REVOKE           *
*   STATUS FOR THIS USERID IS DETERMINED VIA A RACROUTE COMMAND.        *
*                                                                     *
*   EVXWORK IS USED TO MAINTAIN A CONTROL BLOCK ADDRESS ACROSS THE     *
*   PRE AND THE POST CALL OF THE EXIT. INFORMATION IN THIS CONTROL     *
*   BLOCK INCLUDES THE USERID AND THE REVOKE STATUS OF THE USERID       *
*   DURING THE PRE CALL. IF THE STATUS OF THE USERID HAS GONE FROM      *
*   NOT REVOKED TO REVOKED (BASE SEGMENT FIELD FLAG4 =\textit{X'80}' ),  *
*   THE FLAG4 FIELD IS UPDATED TO INCLUDE \textit{X'08'} TURNED ON AS WELL. THIS *
*   INDICATES A USERID THAT HAS BEEN REVOKED BY AN ADMINISTRATOR.       *
*                                                                     *
*   THIS EXIT IS ENTERED FROM RACF IN SUPERVISOR STATE, KEY 0 SO        *
*   BE CAREFUL.                                                        *
***************************************************************************

STM   R14,R12,12(R13)       SAVE INCOMING REGISTERS
LR    R12,R15               COPY MODULE ADDRESS
USING IRREVX01,R12           SET ADDRESSABILITY
LR    R2,R1                 SAVE INCOMING PARM ADDRESS
LR    R11,R13               SAVE OLD SAVEAREA ADDRESS
STORAGE OBTAIN,LENGTH=WORKLEN,LOC=BEBLOW
LR    R13,R1                 GET NEW SAVEAREA ADDRESS
LR    R0,R1                  COPY ADDRESS
LR    R14,R1                 AGAIN
L     R1,=A(WORKLEN)         GET LENGTH
XR    R15,R15                SET FILL BYTE
MVCL  R0,R14                 CLEAR THE STORAGE
USING WORKAREA,R13           SET ADDRESSABILITY
ST R11, SAVEAREA+4          SAVE OLD SAVEAREA ADDRESS
**********************************************************************
USING EVXPL, R2             SET PARAMETER ADDRESSABILITY
**********************************************************************
L   R3, EVXFLAGS            GET FLAG POINTER
TM   Ø(R3), EVXPRE          PREPROCESSING CALL?
BO   PRECALL               YES - ISSUE WTO
TM   Ø(R3), EVXPOST         POSTPROCESSING CALL?
BO   POSTCALL              YES - ISSUE WTO
B     RETURN                WE'RE DONE
PRECALL EQU *               

L   R3, EVXCALLR            GET FUNCTION CODE BYTE ADDRESS
CLI   Ø(R3), EVXALTUS       ALTUSER COMMAND?
BE   PREALTUS               YES - GO PROCESS
B     RETURN                NO - JUST RETURN
PREALTUS EQU *              
BAL   R14, REVOKECHK        CHK IF CMD BUFF CONTAINS 'REVOKE'
LTR   R15, R15              A 'REVOKE'? 
BZ    PRERVK                YES - GO PROCESS
B     RETURN
PRERVK EQU *
BAL   R14, GETUSRID         GO ISOLATE THE USERID
XC   RACWORK(256), RACWORK
XC   RACWORK+256(256), RACWORK+256
MVC   ROUTWRK1(ROUTLEN1), RACROUT1
RACROUTE REQUEST=EXTRACT, TYPE=EXTRACT, ENTITY=USERIDSAV, RELEASE=1.9.2, FIELDS=FLDLIST1, SUBPOOL=1, WORKA=RACWORK, MF=(E, ROUTWRK1)
LTR   R15, R15              EXTRACT OK?
BNZ   RETURN                NO - BUG OUT
LR    R6, R1                COPY THE EXTRACT AREA ADDRESS
**********************************************************************
XR   R15, R15               CLEAR R15
ICM   R15, B'Ø011', 4(R6)   GET DATA OFFSET
AR   R15, R6                POINT TO FLAG4 DATA AREA
MVC   FLAG4SAV(1), 4(R15)   SAVE THE FLAG DATA
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)
**********************************************************************
STORAGE OBTAIN, LENGTH=EVXWLEN, LOC=ANY
XC   Ø(16, R1), Ø(R1)       CLEAR THE STORAGE
L   R3, EVXWORK             GET ADDRESS OF WORK WORD

ST R1,Ø(R3)          SAVE THE WORK ADDRESS  
LR R5, R1          COPY THE STORAGE ADDRESS  
USING EVXWAREA, R5  
MVC EVXWID D(4), =C'EVXW' MOVE IN THE EYECATCHER  
MVC EVXWUID D(8), USRIDSAAV SAVE THE USERID  
MVC EVXWFLG1(1), FLAG4SAV COPY FLAG4  
DROP R5  
B RETURN  
***********************************************************************  
POSTCALL EQU *  
L R4, EVXCMDRC GET ADDRESS OF RETURN CODE  
CLC Ø(R4), =F'Ø' RETURN CODE Ø?  
BNE CMDFAIL NO - ISSUE FAILURE WTO  
L R4, EVXFLAGS GET ADDRESS OF FLAG AREA  
TM Ø(R4), EVXABND WAS THERE AN ABEND?  
BO CMDFAIL YES - ISSUE FAILURE WTO  
***********************************************************************  
L R3, EVXCALLR GET FUNCTION CODE BYTE ADDRESS  
CLI Ø(R3), EVXALTUS ALTUSER COMMAND?  
BE PSTALTUS YES - GO PROCESS  
B RETURN NO - JUST RETURN  
PSTALTUS EQU *  
***********************************************************************  
L R3, EVXWORK GET ADDRESS OF WORK WORD  
LTR R3, R3 ANY WORK WORD?  
BZ RETURN NO WE'RE DONE  
L R5, Ø(R3) GET WORK AREA BUFFER ADDRESS  
LTR R5, R5 ANY BUFFER?  
BNZ PSTRVK YES - REVOKE TO PROCESS  
B RETURN WE'RE DONE  
PSTRVK EQU *  
***********************************************************************  
USING EVXWAREA, R5  
CLC EVXWID D(4), =C'EVXW' OUR EYECATCHER?  
BNE RETURN NO - BYPASS  
***********************************************************************  
XC RACWORK (256), RACWORK  
XC RACWORK +256 (256), RACWORK +256  
MVC ROUTWRK1 (ROUTLEN1), RACROUT1  
RACROUTE REQUEST=EXTRACT, TYPE=EXTRACT, ENTITY=EVXWUID, RELEASE=1.9.2, FIELDS=FLDLIST1, SUBPOOL=1, WORKA=RACWORK, MF=(E, ROUTWRK1)  
LTR R15, R15 EXTRACT OK?  
BNZ RVKEND NO - BUG OUT  
LR R6, R1 COPY THE EXTRACT AREA ADDRESS  
***********************************************************************  
XR R15, R15 CLEAR R15
ICM R15, B'0011', 4(R6)   GET DATA OFFSET
AR R15, R6       POINT TO FLAG4 DATA AREA
MVC FLAG4SAV(1), 4(R15)   SAVE THE FLAG DATA
XR R8, R8        CLEAR R8
XR R9, R9        CLEAR R9
IC R9, Ø(, R6)    SAVE THE SUBPOOL VALUE
ICM R8, B'0111', 1(R6)   SAVE W/A LENGTH
STORAGE RELEASE, LENGTH=(R8), ADDR=(R6), SP=(R9)
***********************************************************************
TM FLAG4SAV, X'80'    USERID IS REVOKED?
BZ RVKEND            NO - THEN WE'RE DONE
OI FLAG4SAV, X'08'   SET ADMIN REVOKE FLAG
MVC FLAG4LEN(4), =F'1' SET LENGTH
***********************************************************************
XC RACWORK(256), RACWORK
XC RACWORK+256(256), RACWORK+256
MVC ROUTWRK1(ROUTLEN1), RACROUT1
RACROUTE REQUEST=EXTRACT, TYPE=REPLACE, ENTITY=EVXWUID, RELEASE=1.9.2, FIELDS=FLDLIST1, SEGDATA=SEGLST1, WORKA=RACWORK, MF=(E, ROUTWRK1)
LTR R15, R15      REPLACE OK?
BNZ REPFAIL       NO - ISSUE MESSAGE
MVC WTOWRK(WTOLN), WTOLST MOVE IN WTO MODEL
MVC WTOWRK+4(OKMSGL), OKMSG MOVE IN THE MESSAGE MODEL
MVC WTOWRK+49(8), EVXWUID COPY IN USERID
WTO MF=(E, WTOWRK) ISSUE THE WTO
B RVKEND          GO FINISH UP
REPFAIL EQU *
MVC WTOWRK(WTOLN), WTOLST MOVE IN WTO MODEL
MVC WTOWRK+4(FAILMSGL), FAILMSG MOVE IN THE MESSAGE MODEL
MVC WTOWRK+4+FAILUID-FAILMSG(8), EVXWUID COPY THE USERID
ST R15, DBL2       SAVE THE SAF RC
UNPK DBL1(9), DBL2(5) UNPACK IT
NC DBL1(8), =8X'0F' TURN OFF HIGH NIBBLES
TR DBL1(8), =C'Ø123456789ABCDEF' MAKE IT READABLE
MVC WTOWRK+4+FAILR1C1-FAILMSG+8(2), DBL1+6 COPY SAF RC
MVC DBL2(4), ROUTWRK1 COPY THE RACF RC
UNPK DBL1(9), DBL2(5) UNPACK IT
NC DBL1(8), =8X'0F' TURN OFF HIGH NIBBLES
TR DBL1(8), =C'Ø123456789ABCDEF' MAKE IT READABLE
MVC WTOWRK+4+FAILR2C2-FAILMSG+9(2), DBL1+6 COPY RACF RC
MVC DBL2(4), ROUTWRK1+4 COPY THE RACF RSN
UNPK DBL1(9), DBL2(5) UNPACK IT
NC DBL1(8), =8X'0F' TURN OFF HIGH NIBBLES
TR DBL1(8), =C'Ø123456789ABCDEF' MAKE IT READABLE
MVC WTOWRK+4+FAILR3N-FAILMSG+10(2), DBL1+6 COPY RACF RSN
WTO   MF=(E, WTOWK)     ISSUE THE WTO
B    RVKEND          GO FINISH UP

*******************************************************************************
RVKEND EQU *
*******************************************************************************
STORAGE RELEASE, LENGTH=EVXWLEN, ADDR=(R5)
DROP R5
B    RETURN          WE'RE DONE

*******************************************************************************
CMDFAIL EQU *
*******************************************************************************
WTO 'IRREVXØ1 - RACF COMMAND FAILED', ROUTCDE=(1), DESC=(6)
B    RETURN          WE'RE DONE

RETURN EQU *
LR    R1, R13        GET WORKAREA ADDRESS
L     R2, SAVEAREA+4  SAVE OLD SAVEAREA ADDRESS
STORAGE RELEASE, LENGTH=WORKLEN, ADDR=(R1)
LR    R13, R2        COPY OLD SAVEAREA ADDRESS
LM    R14, R12, 12(R13)  RESTORE REGISTERS
XR    R15, R15      SET RETURN CODE
BR    R14           RETURN

*******************************************************************************
GETUSRID EQU *
*******************************************************************************
ST    R14, R14SAVE   SAVE RETURN ADDRESS
L     R4, EVXCMBUF   GET COMMAND BUFFER ADDRESS
XR    R5, R5         CLEAR R5
LA    R7, 4(R4)     GET COMMAND ADDRESS
ICM   R5, B'ØØ11', Ø(R4)  GET BUFFER LENGTH
C     R5, =F'4'    ANY BUFFER?
BL    RETURN        NO - WE'RE DONE
S     R5, =F'4'    REDUCE BY HEADER LENGTH

*******************************************************************************
*  FLUSH LEADING BLANKS
PSTLPØ1 EQU *
CLI     Ø(R7), C' '  A BLANK?
BNE    PSTEND1       NO - DONE WITH LEADING BLANKS
LA     R7, 1(R7)     POINT TO NEXT BUFFER BYTE
BCT    R5, PSTLPØ1   IF MORE, GO CHECK
B     RETURN        WE'RE DONE

PSTEND1 EQU *
LA     R7, 1(R7)     SKIP PAST ENCLOSURE
BCTR   R5, Ø         REDUCE BUFFER COUNT BY ONE

PSTLPØ2 EQU *
CLI     Ø(R7), C' '  A BLANK?
BE    PSTEND2        YES - FOUND END OF PRIMARY KW
LA     R7, 1(R7)     POINT TO NEXT BUFFER BYTE
BCT    R5, PSTLPØ2   IF MORE, GO CHECK
B     RETURN        WE'RE DONE

PSTEND2 EQU *
LA     R7, 1(R7)     SKIP PAST ENCLOSURE
BCTR   R5, Ø         REDUCE BUFFER COUNT BY ONE

PSTLP03 EQU *
CL CLI Ø(R7), C' ' A BLANK?
BNE PSTEND3 NO - FOUND THE NAME START
LA R7, 1(R7) POINT TO NEXT BUFFER BYTE
BCT R5, PSTLP03 IF MORE, GO CHECK
B RETURN WE'RE DONE

PSTEND3 EQU *
CL CLI Ø(R7), C'(' ENCLOSURE?
BNE NAMESTRT NO - NAME STARTS RIGHT HERE
LA R7, 1(R7) SKIP PAST ENCLOSURE
BCTR R5, Ø REDUCE BUFFER COUNT BY ONE

NAMESTRT EQU *
LR R8, R7 SAVE STARTING ADDRESS

PSTLP04 EQU *
CL CLI Ø(R7), C' ' A BLANK?
BE NAMEEND YES - FOUND THE NAME END
CL CL Ø(R7), C' ') ENCLOSURE?
BE NAMEEND YES - FOUND THE NAME END
LA R7, 1(R7) POINT TO NEXT BUFFER BYTE
BCT R5, PSTLP04 IF MORE, GO CHECK
B RETURN WE'RE DONE

NAMEEND EQU *
MVC USRIDSAV(8), =8C' ' CLEAR THE AREA
LR R15, R7 SAVE ENDING ADDRESS
SR R15, R8 GET THE LENGTH
BCTR R15, Ø REDUCE BY ONE FOR EX
EX R15, USRIDMVC MOVE USERID INTO BUFFER
L R14, R14SAVE GET RETURN ADDRESS
BR R14 RETURN

***********************************************************************
REVOKCHK EQU *
STM RØ, R15, REGSAVE SAVE REGISTERS
L R4, EVXCMBUF GET COMMAND BUFFER ADDRESS
XR R5, R5 CLEAR R5
ICM R5, B'ØØ11', Ø(R4) GET BUFFER LENGTH
C R5, =F'4' ANY BUFFER?
BL RETNORVK NO - WE'RE DONE
LA R7, Ø(R5, R4) GET BUFFER END ADDRESS
S R7, =F'3' MAKE SURE THERE'S ENOUGH ROOM
ICM R5, B'ØØ11', 2(R4) GET OFFSET OF KEYWORD AREA
LA R4, 4(R5, R4) GET SEARCH START ADDRESS

***********************************************************************
BUFLP1 EQU *
CR R4, R7 END OF BUFFER?
BNE RETNORVK YES - 'REVOKE' NOT DETECTED
CC CL Ø(3, R4), =C' REV' REVOKE (OR SOME SHORTFORM)?
BE RETRVK YES - RETURN REVOKE DETECTED
LA R4, 1(R, R4) POINT TO NEXT BYTE
B BUFLP1 GO CHECK IT OUT

RETRVK EQU *

LR R1, R4           SAVE BUFFER ADDRESS
LM R2, R14, REGSAVE+8 RESTORE SOME REGISTERS
XR R15, R15         SET RETURN CODE TO Ø
BR R14              RETURN

RETNORVK EQU *
LM R2, R14, REGSAVE+8 RESTORE SOME REGISTERS
LA R15, 4           SET RETURN CODE TO 4
BR R14              RETURN

***********************************************************************
* EXECUTED INSTRUCTIONS *
***********************************************************************
USRIDMVC MVC USRIDSAV(*-*), Ø(R8) COPY IN THE USERID
***********************************************************************
* CONSTANTS *
***********************************************************************
WTOLST WTO ' , ROUTCDE=(1), DESC=(6), MF=L
WTOLN EQU *· WTOLST
***********************************************************************
FLDLIST1 DC F'1'
DC CL8 'FLAG4'
***********************************************************************
RACROUT1 RACROUTE REQUEST=EXTRACT, TYPE=EXTRACT, CLASS='USER', RELEASE=1.9.2, MF=L
ROUTLEN1 EQU *· RACROUT1
***********************************************************************
OKMSG DC C'IRREVXØ1 - ADMINISTRATOR REVOKE FLAG SET FOR XXXXXXXX'
OKMSGL EQU *· OKMSG
***********************************************************************
FAILMSG DC C'IRREVXØ1 - ADMINISTRATOR REVOKE FLAG FAILED TO BE ' DC C' SET FOR '
FAILUID DC C'XXXXXXXX. '
FAILRC1 DC C'SAF RC: XX '
FAILRC2 DC C'RACF RC: XX '
FAILRSN DC C'RACF RSN: XX'
FAILMSGL EQU *· FAILMSG
***********************************************************************
LTORG
WORKAREA DSECT
SAVEAREA DS 18F
REGSAVE DS 18F
R14SAVE DS F
RVKLST

RVKLST CSECT
RVKLST AMODE 31
RVKLST RMODE 24

DCB'S NEED 24-BIT ADDRESSES
***********************************************************************
* THE RVKLST PROGRAM CAN WORK WITH ANY RACF DATABASE, BUT IS *
* ESPECIALLY DESIGNED TO WORK IN CONJUNCTION WITH A IRREVXØ1 RACF *
* COMMAND EXIT THAT SETS THE X'08' BIT IN THE FLAG4 USER BASE *
* SEGMENT FIELD WHEN A USERID IS REVOKED VIA A RACF ALTUSER *
* COMMAND. *
* *
* THIS FLAG IS EXAMINED BY THE RVKLST PROGRAM AND CAN BE USED TO *
* DIFFERENTIATE BETWEEN A USERID THAT HAS BEEN REVOKED DUE TO TOO *
* MANY INVALID LOGON ATTEMPTS AND A USERID THAT HAS BEEN REVOKED *
* BY A RACF ADMINISTRATOR. *
* *
* THE RVKLST PROGRAM SHOULD RESIDE IN AN APF AUTHORIZED LIBRARY *
* AND SHOULD BE LINKEDITED AC(1). WHEN THE PROGRAM IS RUN, A *
* SYSPRINT DD SHOULD BE SPECIFIED IN THE JCL. ALL PROGRAM OUTPUT *
* IS DIRECTED TO THIS OUTPUT DATASET. FOLLOWING IS SOME SAMPLE *
* OUTPUT THAT COULD BE EXPECTED: *
* *
* Userid    Revoke Status
ABARS  Not REVOKED
APPC   Not REVOKED
ASCH   REVOKED
BLSJPRMI  REVOKED
BPXOINIT Not REVOKED
BPXROOT  Not REVOKED
USER1  Administrator REVOKED
USER2  Not REVOKED
USER3  Not REVOKED
USER4  Date REVOKED
USER5  Not REVOKED

IF NO PROGRAM PARM IS SPECIFIED, A LIST OF ALL RACF DEFINED
USERID'S IS PRODUCED. YOU CAN LIMIT THE LIST TO A SPECIFIC
USERID BY SPECIFYING THAT USERID ON THE PROGRAM PARM. FOR
EXAMPLE:

//RVKLST EXEC PGM=RVKLST,PARM='USER1'
//STEPLIB DD DSN=auth.library,DISP=SHR
//SYSPRINT DD SYSOUT=*

WOULD PRODUCE THE FOLLOWING OUTPUT IN THE SYSPRINT DATASET:

UserId Revoke Status
-------- ---------------------
USER1  Administrator REVOKED

THIS CAN BE A USEFUL TOOL FOR A RACF ADMINISTRATOR WHO MAY BE
TRYING TO DETERMINE IF THE REVOKE STATUS OF A CERTAIN USERID
SHOULD BE CHANGED TO RESUME. A USERID THAT IS IN 'REVOKED'
STATUS MAY BE A GOOD CANDIDATE TO BE RESUMED, BUT ONE THAT IS
'Date REVOKED' OR 'Administrator REVOKED' MAY REQUIRE MORE
SCRUTINY.

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 RVKLST,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
***********************************************************************
OPEN (SYSPRINT, OUTPUT), MODE=31 OPEN OUTPUT DATASET
PUT SYSPRINT, HDR1          WRITE FIRST HEADER
PUT SYSPRINT, HDR2          WRITE SECOND HEADER
***********************************************************************
L R8, Ø(, R2)               GET PARAMETER ADDRESS
CLC Ø(2, R8), =H'Ø'         ANY PARM DATA?
BE USRIDLST                 NO - IT'S A FULL LIST
CLC Ø(2, R8), =H'8'         TOO MUCH DATA FOR A USERID?
BH RETURN4                  YES - SET RC=4
B ONEUSRID                  PROCESS ONE USERID
***********************************************************************
USRIDLST EQU *              *
XC XUID(4), XUID             CLEAR XUID LENGTH AREA
MVC XUID(2), =H'8'          SET DATA LENGTH
MVC XUID+4(8), =8C' '       SET STARTING UID
UIDLOOP EQU *               *
XC RACWORK(256), RACWORK     CLEAR RACROUTE ...
XC RACWORK=256(256), RACWORK+256 WORKAREA STORAGE
MVC ROUTWRK1(ROUTLEN1), RACRT1 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
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 *              *
CLOSE (SYSPRINT), MODE=31   CLOSE OUTPUT DATASET

ONEUSRID EQU *

MVC SAVEUID(8),=8'C' ' CLEAR THE TARGET USERID FIELD
XR R15, R15 CLEAR R15
ICM R15, B'0011', 0(R8) GET THE USERID LENGTH
BCTR R15, 0 REDUCE BY ONE FOR EX
EX R15, USRI D MVC COPY THE USERID
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=SAVEUID, X
   FIELDS=FLDLIST1, X
   RELEASE=1.9.2, X
   SUBPOOL=1, X
   WORKA=RACWORK, X
   MF=(E, ROUTWRK2)
LTR R15, R15 EXTRACT OK?
BZ DOUSRID YES - PROCESS DATA
MVC OUTREC(133), =133'C' ' CLEAR THE OUTPUT RECORD
MVC OUTREC(8), SAVEUID COPY THE USERID
MVC OUTREC+10(11), =C'Not defined' USERID NOT IN DATABASE
PUT SYSPRINT,OUTREC WRITE THE OUTPUT RECORD
B LISTDONE GO FINISH UP

DOUSRID EQU *

***********************************************************************

***********************************************************************

RETURNØ EQU *

****************************************************************************

* * THE USERID DATA 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. *
****************************************************************************

BAL R14, DOOUTPUT CREATE NECESSARY OUTPUT
B LISTDONE GO FINISH UP

**************************************************************************

RETURN4 EQU *

****************************************************************************

****************************************************************************
STORAGE RELEASE, LENGTH=WALEN, ADDR=(R1)
LR R13, R10
COPY INCOMING SAVEAREA ADDR
LM R14, R12, 12(R13)
RESTORE REGISTERS
LA R15, 4
SET RETURN CODE
BR R14
RETURN
*******************************************************************************
DOOUTPUT EQU *
STM R0, R15, SVAREA02
SAVE REGISTERS
XC SEGDATA(SEGDATAL), SEGDATA CLEAR SEGMENT DATA SAVE AREA
XR R6, R6
CLEAR R6
ICM R6, B'0011', 4(R1)
GET DATA OFFSET
AR R6, R1
CALCULATE DATA ADDRESS
SVFLD1 EQU *
ICM R15, B'1111', 0(R6)
GET DATA LENGTH
MVC SAVEFLG4(1), 4(R6)
COPY FLAG4
SVFLD2 EQU *
LA R6, 4(R15, R6)
POINT TO REVOKEDT
ICM R15, B'1111', 0(R6)
GET DATA LENGTH
LTR R15, R15
ANY DATA?
BZ SVFLD3
NO - CHECK NEXT FIELD
MVC SAVERVDT(3), 4(R6)
SAVE REVOKEDT
SVFLD3 EQU *
LA R6, 4(R15, R6)
POINT TO RESUMEDT
ICM R15, B'1111', 0(R6)
GET DATA LENGTH
LTR R15, R15
ANY DATA?
BZ SVFLD4
NO - CHECK NEXT FIELD
MVC SAVERSDT(3), 4(R6)
SAVE RESUMEDT
SVFLD4 EQU *
*******************************************************************************
XR R7, R7
CLEAR R7
XR R8, R8
CLEAR R8
ICM R7, B'0111', 1(R1)
GET STORAGE LENGTH
ICM R8, B'0001', 0(R1)
GET SUBPOOL
STORAGE RELEASE, LENGTH=(R7), ADDR=(R1), SP=(R8)
*******************************************************************************
RVKCHK EQU *
MVC OUTREC(133), =133C' ' CLEAR THE OUTPUT RECORD
MVC OUTREC(8), SAVEUID COPY THE USERID
TIME DEC GET CURRENT DATE/TIME
STCM R1, B'0111', CURRDATE SAVE CURRENT DATE
TM SAVEFLG4, X'08' ADMIN REVOKED?
BZ CHKLGRVK NO - CHK INVALID PWD REVOKED
CLC CURRDATE(3), SAVERSDT CURRDATE >= RESUMEDT?
BL SETRVK1 NO - REVOKED
CLC SAVERSDT(3), =3X'00' A 'REAL' RESUMEDT?
BNE NORVK YES - NOT REVOKED
SETRVK1 EQU *
MVC OUTREC+10(21), =C'Administrator REVOKED' SET RVK STATUS
B RVKCHKDN WE'RE DONE THIS USERID
CHKLGRVK EQU *
*******************************************************************************
TM SAVEFLG4, X'80' REVOKE?
BZ DTRVKCHK NO - CHECK IF DATE REVOKED
CLC CURRDATE(3), SAVERSDT CURRDATE >= RESUMEDT?
BL SETRVK2 NO - REVOKED
CLC SAVERSDT(3), =X'00' A 'REAL' RESUMEDT?
BNE NORVK YES - NOT REVOKED

SETRVK2 EQU *
MVC OUTREC+10(7), =C' REVOKE' SET REVOKE STATUS
B RVKCHKD N WE' RE DONE THIS USER ID

DTRVKCHK EQU *
CLC SAVERSDT(3), SAVERVDT RESUMEDT < REVOKEDT?
BL RVKOUTWN YES - CURRDATE OUT OF RANGE

RVKINWN EQU *
CLC CURRDATE(3), SAVERSDT CURRDATE >= RESUMEDT?
BNL NORVK YES - NOT REVOKED
CLC CURRDATE(3), SAVERVDT CURRDATE < REVOKEDT?
BL NORVK YES - NOT REVOKED

DATERVK EQU *
MVC OUTREC+10(12), =C' Date REVOKED' SET REVOKE STATUS
B RVKCHKD N WE' RE DONE THIS USER ID

RVKOUTWN EQU *
CLC CURRDATE(3), SAVERSDT CURRDATE < RESUMEDT?
BL DATERVK YES - REVOKED
CLC CURRDATE(3), SAVERVDT CURRDATE >= REVOKEDT?
BNL DATERVK YES - REVOKED
B NORVK OUT OF WINDOW, NO REVOKE

NORVK EQU *
MVC OUTREC+10(11), =C' Not REVOKED' SET REVOKE STATUS
B RVKCHKD N WE' RE DONE THIS USER ID

RVKCHKDN EQU *
PUT SYSPRINT, OUTREC WRITE THE OUTPUT RECORD
**********************************************************************
LM R0, R15, SVAREAØ2 LOAD REGISTERS
BR R14 RETURN
**********************************************************************
USRIDMVC MVC SAVEUID(*-*), 2(R8) COPY THE USER ID
**********************************************************************
SYSPRINT DCB MACRF=(PM), LRECL=133, DSORG=PS, DDNAME=SYSPRINT
**********************************************************************
HDR1 DC CL133'User id Revoke Status'
HDR2 DC CL133'-------- ---------------------'
**********************************************************************
RACROUT1 RACROUTE REQUEST=EXTRACT, TYPE=EXTRACTN,
CLASS='USER', RELEASE=1.9.2,
MF=L
ROUTLEN1 EQU * RACROUT1
**********************************************************************
RACROUT2 RACROUTE REQUEST=EXTRACT,
ROUTLEN2 EQU 'ROUTLEN2'

FLLDLIST DC F'3'
DC C'FLAG4' REVOKE
DC C'REVOKEDT' REVOKE DATE
DC C'RESUMEDT' RESUME DATE

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 ØD, CL(ROUTLEN1)
ROUTWRK2 DS ØD, CL(ROUTLEN2)
XUID DS ØD
DS F
DS CL8

RACWORK DS ØD, CL512
DBL1 DS 2D
DBL2 DS 2D
SAVEUID DS CL8
SEGDATA DS ØD
SAVEFLG4 DS XL1
SAVERVDT DS XL3
SAVERSDT DS XL3
SEGDATAL EQU '* SEGDATA
OUTREC DS CL133
CURRDATE DS XL3
WALEN EQU '* WORKAREA
RØ EQU 0
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
R10 EQU 10
R11 EQU 11
Easy LDAP program to authenticate RACF userids

Some time ago, we installed a Web application on an NT server that accessed data on our production CICS region. Our server administrator had to maintain on his server a list of userids (defined and active with the same name on our host) that were supposed to authenticate on the server before using the Web application. The password was not the same as on the host, and the authentication phase was restricted to the NT server. We decided to try using LDAP to move the authentication phase to our host, eliminating the administrative overhead and using duplicate userids and passwords.

LDAP is a well-known protocol, and on the mainframe it can be configured to use RACF as its database. We have OS/390 2.7 and it supports LDAP level 2, but with the following ptf applied (uw84008) it obtains some password management functionalities that are typical of level 3.

There are a couple of call-to-LDAP APIs that are designed to implement the authentication function: ldap_open (or ldap_init) and ldap_bind. If the latter ends successfully, the sign-on succeeds, otherwise it returns the error ‘invalid credentials’ and then a reason code to help you work out what has happened (eg invalid password, invalid new password, userid not defined, userid revoked, etc).
The LDAP API can be used in many languages; what you choose depends to a large degree on the location of your Web server. I experimented with it in C and Java on the host (note that IBM recommends using JNDI, which is a higher level interface of Java to general directory servers).

Anyway, at our site the Web server resides on an NT server and we had some compatibility problems with Java from Sun (the Web server is Microsoft). In fact, we wanted to call our little program from an ASP. We found that Microsoft supports ADSI, which is the equivalent of JNDI and can be used for programs written in C, C++, and VBSCRIPT. The problem is that, although ADSI works very well with level 3 LDAP servers, the same is not true for level 2. In the end, we decided to write a small C++ program using the native LDAP API, made a COM+ application, and registered it on the NT server in question. The method of that application is invoked in our VBSCRIPT code inside an ASP, and it works! However, if the LDAP server is level 2, the reason codes in the case of unsuccessful sign-on are not taken into account, even if the server issues them (I think the reason is that natively the level 2 LDAP server doesn’t provide them).

Even though the world is full of examples of this kind, I found it difficult working with so many objects so far removed from the mainframe world: ASP, VISUAL C++, VBSCRIPT, COM+, etc. Hopefully, the information I collected will save you some time. What’s more, we think LDAP is the right way to move all authentication processes to the host, because we think RACF is still the best way to manage passwords.

**GETTING STARTED**

The LDAP server is started with the following JCL:

```
| / // LDAPSRV   PROC REGSIZE=64M, |
| //          PARMS=' ', |
| //          GLDHLQ='GLD', |
| //          OUTCLASS='A' |
| //GO        EXEC PGM=GLDSLAPD,REGION=&REGSIZE,TIME=1440, |
```
The GLD.SGLDLINK must be APF authorized. In /etc/ldap you must copy all the files from /usr/lpp/ldap/etc (slapd.**).

The slapd.conf should be configured along the following lines:

```bash
include /etc/ldap/slapd.at.system
include /etc/ldap/slapd.at.conf
include /etc/ldap/slapd.at.racf
include /etc/ldap/slapd.oc.system
include /etc/ldap/slapd.oc.conf
include /etc/ldap/slapd.oc.racf
port 389
securePort 636
security none
sslKeyRingFile keyfile.kyr
sslKeyRingFilePW none
sslCipherSpecs 12288
maxthreads 500
maxconnections 500
waitingthreads 20
timelimit 3600
sizelimit 500
adminDN "racfid=yourldapadm,profiletype=user,sysplex=yoursysplex,
O=yourcompany"
database sdbm GLDBSDBM
suffix "sysplex=yoursysplex"
```

You need to RACF define an LDAP administrator userid with omvs segment (uid=0).

**C++ CODE**

The C++ code is shown below. I used Visual C++ studio 6.0 and I needed the help of one colleague to create the application definition, and of another to define the COM+ application on the NT server.
// elenalag.cpp : Implementation of Celenalog
#include "stdafx.h"
#include "Elenadap.h"
#include "elenalog.h"

/////////////////////////////////////////////////////////////////////
// Celenalog

STDMETHODIMP Celenalog::elogon(BSTR usr, BSTR pwd, BSTR newpwd, long * elresult) {

   AFX_MANAGE_STATE(AfxGetStaticModuleState())

   CString CStmp=usr;
   CString CSusername;
   CSusername = "racfid=" + CStmp + ",profileType=USER,sysplex=COPLEX";
   CString CSPwd=pwd;
   CString CSnewpwd = newpwd;
   *elresult = 0;
   if (CSnewpwd.GetLength() > 1) {
         *elresult = 1;
         CSPwd = CSPwd + "/" + CSnewpwd;
   }

   //******************************************
   //Open the connection to the LDAP server
   //******************************************
   LDAP * Myldap = ldap_open( "your ldap server ip address", 389 );
   if (Myldap == NULL) {
         *elresult += 100;
         return S_OK;
   }

   //******************************************
   //Authentication to the server
   //******************************************
   ULONG bind_ok = ldap_simple_bind( Myldap,
   (char *)LPCSTR(CSusername),
   (char *)LPCSTR(CSPwd) );
   if (bind_ok < 0) {
         *elresult = 666;
         return S_OK;
   }

   LDAPMessage ** res;
   PLDAPControl ** Servercontrols;
   PCHAR * errormsg = NULL;

   //************
   //Function to authenticate to the server
   //************
   Myldap = ldap_bind( Myldap,
   (char *)LPCSTR(CSusername),
   (char *)LPCSTR(CSPwd) );
   if (Myldap == NULL) {
         *elresult += 100;
         return S_OK;
   }

   //************
   //Get the list of server controls
   //************
   Servercontrols = ldap_get_servercontrols(Myldap);
   if (Servercontrols == NULL) {
         *elresult += 100;
         return S_OK;
   }

   //************
   //Create the bind control
   //************
   LDAPControl ** bindControl = ldap_control_new(LDAP_CONTROL_BIND);
   if (bindControl == NULL) {
         *elresult += 100;
         return S_OK;
   }

   bindControl->type = LDAP_CONTROL_BIND;
   bindControl->value = htons(0);
   bindControl->length = 1;

   //************
   //Send the bind control to the server
   //************
   ldap_control_add(Servercontrols, bindControl);
   if (ldap_servercontrols(Myldap, Servercontrols) < 0) {
         *elresult += 100;
         return S_OK;
   }

   //************
   //Get the bind result
   //************
   bind_Control * bindResult = bindControl;
   bindResult->type = LDAP_CONTROL_BIND;
   bindResult->length = 1;
   bindResult->value = htons(0);

   //************
   //Release the server controls
   //************
   ldap_servercontrols_release(Servercontrols);

   //************
   //Release the bind control
   //************
   ldap_control_unref(bindControl);

   //************
   //Get the list of attributes
   //************
   LDAPMessage ** attrList = ldap_get_attributelist(Myldap,
   (char *)LPCSTR(CSusername),
   (char *)LPCSTR(CSPwd) );
   if (attrList == NULL) {
         *elresult += 100;
         return S_OK;
   }

   //************
   //Search for the bind result
   //************
   LDAPMessage ** bindResultList;
   bindResultList = ldap_search(Myldap,
   (char *)LPCSTR(CSusername),
   (char *)LPCSTR(CSPwd) );
   if (bindResultList == NULL) {
         *elresult += 100;
         return S_OK;
   }

   //************
   //Print the bind result
   //************
   ldap_search_result_free(bindResultList);
}

//************
//Function to authenticate to the server
//************
Myldap = ldap_bind( Myldap,
   (char *)LPCSTR(CSusername),
   (char *)LPCSTR(CSPwd) );
if (Myldap == NULL) {
   *elresult += 100;
   return S_OK;
}

//************
//Get the list of server controls
//************
Servercontrols = ldap_get_servercontrols(Myldap);
if (Servercontrols == NULL) {
   *elresult += 100;
   return S_OK;
}

//************
//Create the bind control
//************
LDAPControl ** bindControl = ldap_control_new(LDAP_CONTROL_BIND);
if (bindControl == NULL) {
   *elresult += 100;
   return S_OK;
}

bindControl->type = LDAP_CONTROL_BIND;
bindControl->value = htons(0);
bindControl->length = 1;

//************
//Send the bind control to the server
//************
ldap_control_add(Servercontrols, bindControl);
if (ldap_servercontrols(Myldap, Servercontrols) < 0) {
   *elresult += 100;
   return S_OK;
}

//************
//Get the bind result
//************
bind_Control * bindResult = bindControl;
bindResult->type = LDAP_CONTROL_BIND;
bindResult->length = 1;
bindResult->value = htons(0);

//************
//Release the server controls
//************
ldap_servercontrols_release(Servercontrols);

//************
//Release the bind control
//************
ldap_control_unref(bindControl);

//************
//Get the list of attributes
//************
LDAPMessage ** attrList = ldap_get_attributelist(Myldap,
   (char *)LPCSTR(CSusername),
   (char *)LPCSTR(CSPwd) );
if (attrList == NULL) {
   *elresult += 100;
   return S_OK;
}

//************
//Search for the bind result
//************
LDAPMessage ** bindResultList;
bindResultList = ldap_search(Myldap,
   (char *)LPCSTR(CSusername),
   (char *)LPCSTR(CSPwd) );
if (bindResultList == NULL) {
   *elresult += 100;
   return S_OK;
}

//************
//Print the bind result
//************
ldap_search_result_free(bindResultList);

struct l_timeval zerotime;
zertime.tv_sec = zerotime.tv_usec = 5L;

ULONG retid = ldap_result(Myldap, bind_ok, Ø, &zerotime, res);
if (retid == Ø) {
    *elresult = 777;
    return S_OK;
}
if (retid == -1) {
    *elresult = 888;
    return S_OK;
}

ULONG parseid
ldap_parse_result(Myldap, *res, &retid, NULL, errmsg, NULL,
                   Servercontrols, 1
if (parseid == LDAP_SUCCESS && retid != LDAP_SUCCESS) {
    *elresult = 333;
    return S_OK;
}

ULONG unbind_ok = ldap_unbind(Myldap);
return S_OK;
}

Note that during the link you have to include wldap32.lib.

For the sake of completeness I’ve included below the source code in both C and Java (which is simpler!).

C CODE

#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <ctype.h>
#include <ldap.h>
#define DEFSEP "="
static char *racfid = NULL;
static char *binddn = NULL;
static char *passwd = NULL;
static char *base = NULL;
static char *ldaphost = "localhost";
static int ldapport = LDAP_PORT;
static char *sep = DEFSEP;
static int verbose, not, allow_binary, vals2tmp, ldif;
static int ldapversion = LDAP_VERSION2;
main( int argc, char **argv )
{
    int rc, authmethod;
    LDAP *ld;

    racfid = argv[ 1 ];
    passwd = argv[ 2 ];
    if (( ld = ldap_open( ldaphost, ldapport )) == NULL ) {
        perror( ldaphost );
        printf( "ldap_open( %s, %d )\n", ldaphost, ldapport );
        exit( 1 );
    }

    binddn = strncat("racfid=", racfid, 8);
    binddn = strncat( binddn, ",profiletype=USER,sysplex=yoursysplex", 33 )
    authmethod = LDAP_AUTH_SIMPLE;
    if ( ldap_bind_s( ld, binddn, passwd, authmethod ) != LDAP_SUCCESS )
        ldap_perror( ld, "ldap_bind" );
        printf( "\nracfid= %s", racfid );
        exit( 1 );
    ldap_unbind( ld );
    exit( rc );
}

JAVA CODE

The following Java sample also makes a search on the RACF database and the equivalent of the information you see with the LISTUSER command on the host.

```java
import java.util.Properties;
import java.io.*;
import java.util.*;
import javax.naming.*;
import javax.naming.directory.*;

class Search {
    public static void main(String[] args) {
        Properties env=utils.getProps("test.properties");
        String base = env.getProperty("example.search.base");
        String filter = env.getProperty("example.search.filter");
    }
```
```java
import javax.naming.*;
import javax.naming.directory.*;
import java.io.*;
import java.util.*;

public class utils {

    // Loads properties from a file
    static public Properties getProps(String fn) {
        Properties properties = new Properties();
        File propsFile = new File(fn);
        try {
            FileInputStream in =
                new FileInputStream(propsFile);
            properties.load(new BufferedInputStream(in));
            in.close();
        } catch (Exception e) {
            System.out.println("getProps: "+e.getMessage());
            System.exit(1);
        }
        return properties;
    }

    static void printSearchEnums(NamingEnumeration results)
    throws NamingException {
        if (results == null) {
            System.out.println("Search Results are empty.");
        } else {
```
```java
while (results.hasMoreElements()) {
    SearchResult si = (SearchResult) results.next();
    System.out.println(si.getName());
    Attributes attrs = si.getAttributes();
    if (attrs == null) {
        System.out.println("No attributes");
    } else {
        for (NamingEnumeration ae = attrs.getAll();
            ae.hasMoreElements();)
            Attribute attr = (Attribute) ae.next();
        String id = attr.getID();
        for (Enumeration vals = attr.getAll();
            vals.hasMoreElements();
            System.out.println("   "+id + ": " + vals.nextElement();
        }
    }
}
}
```

where test.properties is as follows:

```properties
java.naming.factory.initial=com.ibm.jndi.LDAPCtxFactory
java.naming.provider.url=ldap://your ldap server ip address
java.naming.security.principal=racf=youruser, profileType=USER, sysplex=yoursysplex
java.naming.security.credentials=your password
java.naming.ldap.version=2
java.naming.referral=ignore
example.search.base=racf=youruser, profileType=USER, sysplex=yoursysplex
example.search.filter=(objectclass=*)
example.search.ref.filter=(objectclass=*)
example.search.ref.base=0=bogus.base
```

**ASP FILE WITH VBSCRIPT CODE**

```vbscript
<%@ Language=VBScript %>
<% 
    dim sUser, sPassword
    dim oUser, sConnectString
```
```
dim MyVar

set myldap = CreateObject("elenadap.elenalog")
sUser = request("txtUserName")
sPassword = request("txtPassword")
sNewPassword = request("txtNewPassword")
title = "LOGON BY & sUser & TO your ldap server ip address"
result = myldap.elogon(sUser, sPassword, sNewPassword)
Select Case result
   Case 0   prompt = "logon successful"
   Case 1   prompt = "logon successful, change password successful"
   Case 49   prompt = "logon failed"
   Case 777  prompt = "reached timeout, try again"
   Case Else prompt = "logon failed! Error code = " & result
End Select
Response.Write title & "<br>"
Response.Write prompt & "<br>"
If sNewPassword <> "" and result > 1 Then
   Response.Write "change password failed"
End If
%
```

**HTML**

The html you select from the browser is as follows:

```
<%@ Language=VBScript %>
<SCRIPT id=DebugDirectives runat=server language=javascript>
</SCRIPT>
<HTML>
<HEAD>
</HEAD>
<BODY>
<H1 align=center>User &nbsp;Password &nbsp;Maintenance</H1>
<P>&nbsp;</P>
<P>Please enter your user name and New password below:</P>
<P>
<FORM action="ldaplogon.asp" method=post id=frmLogin name=frmLogin>
<TABLE border=0 cellPadding=1 cellSpacing=1 width="75%">
   <TR>
      <TD>User Name</TD>
      <TD><INPUT id=txtUserName name=txtUserName style="HEIGHT: 25px; WIDTH: 365px"></TD>
   </TR>
</TABLE>
```

Auditing RACF protection

This article presents two CLISTs to help with auditing RACF protection. The first, RACFC, produces an output file named SYS.RACF.LIST.DATAR. The second, RACFUSER, shows RACF permissions for a user or a group.

RACFC
/* *************************************************************** */
/* CLIST RACFC */
/* check permits on files users and groups */
/* *************************************************************** */
CONTROL MSG NOFLUSH
ISPEXEC CONTROL DISPLAY LINE
ISPEXEC VGET ZUSER
IF &ZUSER = PSY1 THEN GOTO MESS
IF &ZUSER = PSY3 THEN GOTO MESS
IF &ZUSER = PSY4 THEN GOTO MESS
IF &ZUSER = EXPØ THEN GOTO MESS
IF &ZUSER = EXP6 THEN GOTO MESS
IF &ZUSER = EXPB THEN GOTO MESS
IF &ZUSER = EXPR THEN GOTO MESS
/* IF &ZUSER = EXPR THEN GOTO MESS */
WRITE ************************************
WRITE USER : &ZUSER NOT AUTHORIZED
WRITE ************************************
GOTO FIN
MESS:+
ISPEXEC DISPLAY PANEL(RACFP)
IF &LASTCC > Ø THEN GOTO FIN
ISPEXEC VPUT REP
IF &REP = NON THEN GOTO SUITE
DEBUT:+
ISPEXEC FTOPEN TEMP
ISPEXEC FTINCL RACFS1
ISPEXEC FTCLOSE
ISPEXEC VGET ZTEMPF
SUBMIT '&ZTEMPF'
IF &LASTCC = Ø THEN GOTO CTRLRESS
WRITE ************************************
******************************************************************************
GOTO FIN
CTRLRESS:+
IF &SYSDSN(SYS.RACF.LIST.RESOURCE) = OK THEN DO
GOTO SUITCR
END
WRITE ************************************
WRITE FILE >>>>>>>>>>>>> SYS.RACF.LIST.RESOURCE
WRITE PATIENCE.......................%DELAI 1000
GOTO CTRLRESS
SUITCR:+
%RACFCR
SUITE:+
ISPEXEC DISPLAY PANEL(RACFP1)
IF &LASTCC > Ø THEN GOTO MESS
ISPEXEC VPUT XX
IF &XX = Ø THEN GOTO OPTØ
IF &XX = 1 THEN GOTO OPT1
IF &XX = 2 THEN GOTO OPT2
IF &XX = 3 THEN GOTO OPT3
IF &XX = 4 THEN GOTO OPT4
IF &XX = 5 THEN GOTO OPT5
IF &XX = 6 THEN GOTO OPT6
IF &XX = 7 THEN GOTO OPT7
IF &XX = 9 THEN GOTO OPT9
SET &P = &SUBSTR(2:2, &XX)
IF &P = P THEN GOTO PRINT
GOTO FIN

/* ----------------------------------------------- */
OPT0: +
IF &SYSDSN(SYS.RACF.LIST.MONITOR) = OK THEN DO
  GOTO OPT0S
END

WRITE ********************************************
WRITE FILE >>>>>>>>>>>>>>>> SYS.RACF.LIST.MONITOR
WRITE ********************************************
WRITE ABSENT OR BEING CREATED
WRITE PATIENCE......................
WRITE ********************************************
GOTO SUITE
OPT0S: +
ISPEXEC BROWSE DATASET ('SYS.RACF.LIST.MONITOR')
GOTO SUITE

/* ----------------------------------------------- */
OPT1: +
IF &SYSDSN(SYS.RACF.LIST.USER) = OK THEN DO
  GOTO OPT1S
END

WRITE ********************************************
WRITE ********************************************
WRITE FILE >>>>>>>>>>>>>>>> SYS.RACF.LIST.ANOMALIE
WRITE not here
WRITE Please wait......................
WRITE ********************************************
GOTO SUITE
OPT1S: +
ISPEXEC BROWSE DATASET ('SYS.RACF.LIST.USER')
GOTO SUITE

/* ----------------------------------------------- */
OPT2: +
IF &SYSDSN(SYS.RACF.LIST.DATA) = OK THEN DO
  GOTO OPT2S
END

WRITE ********************************************
WRITE ********************************************
WRITE FILE >>>>>>>>>>>>>>>> SYS.RACF.LIST.ANOMALIE
WRITE not here
WRITE Please wait......................
GOTO SUITE
OPT2S: +
ISPEXEC BROWSE DATASET ('SYS.RACF.LIST.DATA')
GOTO SUITE

/* ----------------------------------------------- */
OPT3: +
IF &SYSDSN(SYS.RACF.LIST.DATAR) = OK THEN DO
  GOTO OPT3S

WRITE ********************************************
WRITE ********************************************
WRITE FILE >>>>>>>>>>>>>> SYS.RACF.LIST.ANOMALIE
WRITE not here
WRITE Please wait......................

WRITE FILE >>>>>>>>>>>>>> SYS.RACF.LIST.DATAR
WRITE ABSENT OR BEING CREATED
WRITE PATIENCE......................
WRITE **************************************
GOTO SUITE
OPT3S:+
ISPEXEC BROWSE DATASET ('SYS.RACF.LIST.DATAR')
GOTO SUITE
/* *-----------------------------------------------------------------*/
OPT4:+
IF &SYSDSN(SYS.RACF.LIST.DATAR.ACCESS) = OK THEN DO
    GOTO OPT4S
END
WRITE ************************************************
WRITE ********************************************
WRITE ********************************************
WRITE FILE >>>>>>>>>>>>>> SYS.RACF.LIST.ANOMALIE
WRITE not here
WRITE Please wait......................
WRITE ********************************************
GOTO SUITE
OPT4S:+
ISPEXEC BROWSE DATASET ('SYS.RACF.LIST.DATAR.ACCESS')
GOTO SUITE
/* *-----------------------------------------------------------------*/
OPT5:+
IF &SYSDSN(SYS.RACF.LIST.GROUP) = OK THEN DO
    GOTO OPT5S
END
WRITE ********************************************
WRITE ********************************************
WRITE FILE >>>>>>>>>>>>>> SYS.RACF.LIST.ANOMALIE
WRITE not here
WRITE Please wait......................
WRITE ********************************************
GOTO SUITE
OPT5S:+
ISPEXEC BROWSE DATASET ('SYS.RACF.LIST.GROUP')
GOTO SUITE
/* *-----------------------------------------------------------------*/
OPT6:+
IF &SYSDSN(SYS.RACF.LIST.RESOURCE) = OK THEN DO
    GOTO OPT6S
END

WRITE not here
WRITE Please wait......................

WRITE FILE >>>>>>>>>>>>>> SYS.RACF.LIST.RESOURCE
WRITE ABSENT OR BEING CREATED
WRITE PATIENCE......................
WRITE ********************************************
GOTO SUITE
OPT6S:+
ISPEXEC BROWSE DATASET ('SYS.RACF.LIST.RESOURCE')
GOTO SUITE
/* --------------------------------------------------- */
OPT7:+
IF &SYSDSN(SYS.RACF.LIST.RESOURCE.NAME) = OK THEN DO
GOTO OPT7S
END
WRITE ********************************************
WRITE FILE >>>>>>>>>>>>>> SYS.RACF.LIST.ANOMALIE
WRITE not here
WRITE Please wait......................
WRITE ********************************************
GOTO SUITE
OPT7S:+
ISPEXEC BROWSE DATASET ('SYS.RACF.LIST.RESOURCE.NAME')
GOTO SUITE
/* --------------------------------------------------- */
OPT9:+
IF &SYSDSN(SYS.RACF.LIST.ANOMALIE) = OK THEN DO
GOTO OPT9S
END
WRITE ********************************************
WRITE FILE >>>>>>>>>>>>>> SYS.RACF.LIST.ANOMALIE
WRITE not here
WRITE Please wait......................
WRITE ********************************************
GOTO SUITE
OPT9S:+
ISPEXEC BROWSE DATASET ('SYS.RACF.LIST.ANOMALIE')
GOTO SUITE
/* --------------------------------------------------- */
PRINT:+
ISPEXEC FTOPEN TEMP
ISPEXEC FTINCL RACFS3
ISPEXEC FTCLOSE
ISPEXEC VGET ZTEMPF
SUBMIT '&ZTEMPF'
IF &LASTCC = Ø THEN GOTO SUITE
WRITE ********************************************
WRITE job racfxx not submitted to jes
WRITE ********************************************
GOTO END
END:
END
%RACF DEFINE - CONTROLS - PROTECTIONS
%***
=====================================================================
%DO YOU WANT TO GENERATE THE CONTROL PROCESS (YES/NO)
%.
+ %YES==> SUBMIT JCL %RACFS1 - %RACFS2+ >>>>>>> >>>>>>> >>>>>>>>
+ SORTIE%SDSF.O3+
+ - CREATION DES FICHIERS: >>>>>%SYS.RACF.LIST.MONITOR+
+ ===> %SYS.RACF.LIST.USER+
+ >>>>>%SYS.RACF.LIST.DATA+
+ >>>>>%SYS.RACF.LIST.DATAR+
+ >>>>>%SYS.RACF.LIST.DATAR.ACCESS+
+ >>>>>%SYS.RACF.LIST.GROUP+
+ >>>>>%SYS.RACF.LIST.RESOURCE+
+ >>>>>%SYS.RACF.LIST.RESOURCE.NAME+
+ >>>>>%SYS.RACF.LIST.ANOMALIE+
%NO==> YOU WILL BROWSE OLD FILES.
+ %OPTION =>_REP+
+ INIT
+ .CURSOR = REP
+ &REP = '???'
+ PROC VER(&REP, NONBLANK, MSG=SYSTØØ1)
+ VER(&REP, LIST, OUI, NON, MSG=SYSTØØ1)
+ END
%RACF DEFINITIONS - CONTROLS - PROTECTIONS%***
=====================================================================
+ +++ RACF DATA SECURITY MONITOR (RACFDSMO).%<<<< RECAPITULATION+
+ 1 +++ LIST OF USER DEFINITIONS.
+ 2 +++ LIST OF DATASET DEFINITIONS.
+ 3 +++ LIST OF DATASET% DEFINITIONS (REDUCED EDITION).
+ 4 +++ LIST OF DATASET% DEFINITIONS (UNIVERSAL AND ID ACCESS).
+ 5 +++ LIST OF GROUP% DEFINITIONS (RESOURCES AND USERS).
+ 6 +++ LIST OF DIFFERENT RESOURCE CLASS% RECAPITULATION+
+ 7 +++ LIST OF DIFFERENT RESOURCE CLASS% WITH RESOURCE NAME+
+ 9 +++ LIST OF ANOMALIES.
**RACFUSER**

/* ++CLIST ++RACFUSER SHOWING THE LEVEL OF PROTECTION */
/* FOR FILES PROTECT BY ++RACF */
/* __________________________________________ */

PROC 1 NAMEUSER
CONTROL MSG NOFLUSH
ISPEXEC CONTROL DISPLAY LINE START(10)
ISPEXEC VGET ZUSER
/* _________ CONTROL OF THE USERid which uses this program ________ */
SUIT0: +
IF &ZUSER = PSY1 THEN GOTO SUIT1 /* SYSTEME */
IF &ZUSER = PSY3 THEN GOTO SUIT1 /* SYSTEME */
IF &ZUSER = PSY4 THEN GOTO SUIT1 /* SYSTEME */
IF &ZUSER = EXPR THEN GOTO SUIT1 /* ADJOINT */
IF &ZUSER = EXP1 THEN GOTO SUIT1 /* EXPLOIT */
IF &ZUSER = EXP6 THEN GOTO SUIT1 /* EXPLOIT */
WRITE ************************************************************
WRITE USER : &ZUSER not authorized to use this program
WRITE **********************************************
GOTO FINFIN
/* -------------------------------------------- */
SUIT1: +
IF &SYSDSN('SYS.RACF.LIST.DATAR') NE OK THEN DO
WRITE **** file {SYS.RACF.LIST.DATAR} doesn’t exist........
WRITE ***** use cliust : RACFC
GOTO FINFIN
END
/
SUIT2:
SET &SEPAR = &STR( >>>>
)
WRITE ********** CONTROL ACCES DATASETS **********
WRITE ********** for user or groupe : &NAMEUSER
SET &DATAR = &STR(SYS.RACF.LIST.DATAR)
SET &LISTE = &STR(SYS.RACF.CTRL.&NAMEUSER)
IF &SYSDSN(&LISTE) = OK THEN DO
  DELETE (&LISTE) PURGE
END
ALLOC FILE(LISTE) DSNAME(&LISTE) LRECL(80) BLKSIZE(11120) +
SPACE (5,1) TRACKS DIR(0) DSORG(PS) RECFM(F,B) +
VOLUME(MVSZZZ) NEW
OPENFILE LISTE OUTPUT
ALLOC FI(DATAR) DSNAME('&DATAR') SHR
SET &RC = &LASTCC
IF &RC NE Ø THEN DO
  WRITE **********************************************************
  WRITE ACCES FILE &DATAR not possible
  WRITE **********************************************************
  GOTO FINFIN
END
OPENFILE DATAR
SET &RC = Ø
ERROR DO
  SET &RC = &LASTCC
  ERROR OFF
  CLOSFILE DATAR
  IF &RC = 400 THEN DO
    FREE FI(DATAR)
    WRITE ==================================================================================
    WRITE END OF SEARCH IN RACF PROTECTIONS
    WRITE FOR USER OR GROUP : &NAMEUSER
    WRITE FILE BROWSE >>>> : SYS.RACF.CTRL.&NAMEUSER
    WRITE ==================================================================================
    GOTO SUIT5
  END
END
GETFILE DATAR
SET LISTE = &DATAR
PUTFILE LISTE
GETFILE DATAR
SET LISTE = &DATAR
PUTFILE LISTE
GETFILE DATAR
GETFILE DATAR
GETFILE DATAR
GETFILE DATAR
GETFILE DATAR
SUIT3: +
GETFILE DATAR
/* SET &INFBK = &SUBSTR(1:1, &STR(&DATAR))
/* IF &STR(&INFBK) ¬= &STR( ) THEN GOTO SUIT3
SET &INFO1 = &SUBSTR(1:24, &STR(&DATAR))
SET &INFO2 = &SUBSTR(1:7, &STR(&DATAR))
IF &STR(&INFO2) = &STR( ------) THEN GOTO SUIT3
IF &STR(&INFO2) = &STR( ......) THEN GOTO SUIT3
IF &STR(&INFO2) = &STR( ----- ) THEN GOTO SUIT3
IF &STR(&INFO1) = &STR(INFORMATION FOR DATASET ) THEN DO
SET &STOCK1 = &SUBSTR(25:50, &STR(&DATAR))
GOTO SUIT3
END
IF &STR(&INFO2) ¬= &STR(&NAMEUSER) THEN GOTO SUIT3
SET &STOCK2 = &SUBSTR(11:20, &STR(&DATAR))
SUIT4: +
SET LISTE = &STR(&NAMEUSER. &SEPAR. &STOCK1. &STOCK2)
PUTFILE LISTE
GOTO SUIT3
/* _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ end of job _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ */
SUIT5: +
CLOSFILE LISTE
FREE FI(LISTE)
ISPEXEC BROWSE DATASET(SYS.RACF.CTRL.&NAMEUSER)
FINFIN: +
END

The RACF1p job is as follows:

// RACFS1 JOB SYS, SYSTEME, CLASS=W, MSGCLASS=3, NOTIFY=&ZUSER
// *
// * CONTROLS : (RACF PROTECTIONS AND DEFINITIONS)
// * *******
// * _____________________________________________________________
// * STEPS :
// * =======
// * _ STEP DEL _________ DELETE THE FILES ON EXIT.
// * _ STEP MONRESS ______ LIST OF RESOURCES
// * _ STEP MONITOR ______ RACF DATA SECURITY MONITOR
// * _ STEP IEHLIST ______ VTOC LIST OF SYSTEM DISK M31RES
// * _ STEP STEPFI1 ______ VTOCS START UP
// * _ STEP TRINOMS ______ SORT ON FILE NAMES.
// * _ STEP STEPFI2 ______ VTOCS START UP (BY FILE).
// * _ STEP RACFLG ______ LIST OF GROUP DEFINITIONS
// * _ STEP RACUSER ______ LIST OF USER DEFINITIONS
// * _ STEP RACFDATA ______ LIST OF DATASET DEFINITIONS
// * _ STEP RACFDATR ______ REDUCED LIST OF DATASETS

//*  _ STEP RACFACCS ___________ REDUCED LIST OF DATASETS (ACCESS)
//*  _ STEP RACFCTRL ___________ LIST OF ANOMALIES
//*  *******************************
//* ****************************
//*
//********************************************************************
//DEL EXEC PGM=IDCAMS
//*==================
//SYSPRINT DD SYSOUT=* 
//SYSIN DD *
  DEL (SYS.RACF.LIST.MONITOR)
  DEL (SYS.RACF.LIST.RESOURCE)
  DEL (SYS.RACF.LIST.RESOURCE.NAME)
  DEL (SYS.RACF.VTOCM31R)
  DEL (SYS.RACF.VTOCFIC1)
  DEL (SYS.RACF.VTOCFICT)
  DEL (SYS.RACF.VTOCCTRL)
  DEL (SYS.RACF.LIST.GROUP)
  DEL (SYS.RACF.LIST.USER)
  DEL (SYS.RACF.LIST.DATA)
  DEL (SYS.RACF.LIST.DATAR)
  DEL (SYS.RACF.LIST.DATAR.ACCESS)
  DEL (SYS.RACF.LIST.ANOMALIE)
  DEL (SYS.RACF.LIST.MONITOR)
  DEL (SYS.RACF.LIST.RESOURCE)
  DEL (SYS.RACF.LIST.RESOURCE.NAME)
  DEL (SYS.RACF.VTOCM31R)
  DEL (SYS.RACF.VTOCFIC1)
  DEL (SYS.RACF.VTOCFICT)
  DEL (SYS.RACF.VTOCCTRL)
  DEL (SYS.RACF.LIST.GROUP)
  DEL (SYS.RACF.LIST.USER)
  DEL (SYS.RACF.LIST.DATA)
  DEL (SYS.RACF.LIST.DATAR)
  DEL (SYS.RACF.LIST.DATAR.ACCESS)
  DEL (SYS.RACF.LIST.ANOMALIE)

//*_____________________________________________________________________
//*        DSMON RESOURCES
//MONRESS  EXEC PGM=ICHDSMØØ
//*==================================
//SYSUT1   DD DSN=SYS1.PARMLIB,DISP=SHR
//SYSUT2   DD DSN=SYS.RACF.LIST.RESOURCE,DISP=(,CATLG,DELETE),
  SPACE=(CYL,(3,1),RLSE),UNIT=SYSDA,
  DCB=(LRECL=133,RECFM=FB, BLKSIZE=27930), VOL=SER=MVSZZZ
//SYSPRINT DD SYSOUT=* 
//SYSIN DD *
  FUNCTION RACCDT
//*_____________________________________________________________________
//*        DSMON GLOBAL
//MONITOR  EXEC PGM=ICHDSMØØ
//*==================================
//SYSUT1   DD DSN=SYS1.PARMLIB,DISP=SHR
//SYSUT2   DD DSN=SYS.RACF.LIST.MONITOR,DISP=(,CATLG,DELETE),
  SPACE=(CYL,(3,1),RLSE),UNIT=SYSDA,
  DCB=(LRECL=133,RECFM=FB, BLKSIZE=27930), VOL=SER=MVSZZZ
//SYSPRINT DD SYSOUT=* 
//*_____________________________________________________________________
//IEHLIST EXEC PGM=IEHLIST
//*==================================
//SYSPRINT DD DSN=SYS.RACF.VTOCM31R,UNIT=SYSDA,
  SPACE=(CYL,(3,1),RLSE),
  DCB=(LRECL=121,RECFM=FB, BLKSIZE=11132),
/ * ____________________________  
//STEPFIC1 EXEC PGM=SYSGFIC1  
//*====================================================================  
//SYSPRINT DD SYSSOUT=*  
//SYSUDUMP DD SYSSOUT=W  
//SYSDBOUT DD SYSSOUT=*  
//SYSOUT DD SYSSOUT=*  
//DSKALL DD DSN=EXP.CTRL.VTOCALLS,DISP=SHR  
//FICSYSTT DD DSN=SYS.RACF.VTOCFIC1,DISP=OLD  
//FICHEDIT DD DSN=SYS.RACF.VTOCCTRL,  
//UNIT=SYSDA,DI SP=( ,CATLG,DELETE),SMALL=SYSDA,  
//SPACE=(CYL,(3,1),RLSE),VOL=SER=MVSZZZ  
//SYSIN DD *  
//SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR  
//SORTIN DD DSN=SYS.RACF.VTOCFIC1,DISP=OLD  
//SORTOUT DD DSN=SYS.RACF.VTOCFICT,UNIT=SYSDA,  
//SPACE=(CYL,(3,1),RLSE),VOL=SER=MVSZZZ  
//FICSYSTT DD DSN=SYS.RACF.VTOCFIC1,DISP=OLD  
//FICHEDIT DD DSN=SYS.RACF.VTOCCTRL,  
//UNIT=SYSDA,DI SP=( ,CATLG,DELETE),SMALL=SYSDA,  
//SPACE=(CYL,(3,1),RLSE),VOL=SER=MVSZZZ  
//DUMPHSM DD DUMMY  
*/
//* ________________________________________________________________
//*        LIST OF GROUPS
//RACFLG EXEC PGM=IKJEFTØ1, DYNAMNBR=20
//*=====================================================================
//SYSTSPRT DD DSN=SYS.RACF.LIST.GROUP,
//     UNIT=SYSDA, DI SP=(, CATLG, DELETE),
//     DCB=(LRECL=133, RECFM=FB, BLKSIZE=27930),
//     SPACE=(CYL, (3, 1), RLSE), VOL=SER=MVSZZZ
//SYSTSI N DD *
PROFILE PREFIX(&ZUSER)
SEARCH CLASS(GROUP) NOMASK CLIST('LISTGRP ' '')
PROFILE NOPREFIX
EX &ZUSER...EXEC.RACF.CLIST
/*
//* ________________________________________________________________
//*        LIST OF USERS
//RACFUSER EXEC PGM=IKJEFTØ1, DYNAMNBR=20
//*=====================================================================
//SYSTSPRT DD DSN=SYS.RACF.LIST.USER,
//     UNIT=SYSDA, DI SP=(, CATLG, DELETE),
//     DCB=(LRECL=133, RECFM=FB, BLKSIZE=27930),
//     SPACE=(CYL, (3, 1), RLSE), VOL=SER=MVSZZZ
//SYSTSI N DD *
PROFILE PREFIX(&ZUSER)
SEARCH CLASS(USER) NOMASK CLIST('LISTUSER ' '')
PROFILE NOPREFIX
EX &ZUSER...EXEC.RACF.CLIST
/*
//* ________________________________________________________________
//*        LIST OF DATA
//RACFDATA EXEC PGM=IKJEFTØ1, DYNAMNBR=20
//*=====================================================================
//SYSTSPRT DD DSN=SYS.RACF.LIST.DATA,
//     UNIT=SYSDA, DI SP=(, CATLG, DELETE),
//     DCB=(LRECL=133, RECFM=FB, BLKSIZE=27930),
//     SPACE=(CYL, (3, 1), RLSE), VOL=SER=MVSZZZ
/*
/* SEARCH CLASS(DATASET) NOMASK CLIST('LISTDSD DATASET(' ') ALL')
/*
/* SYSTSI N DD *
PROFILE PREFIX(&ZUSER)
SEARCH GENERIC NOMASK CLIST('LISTDSD DATASET(' ') ALL')
PROFILE NOPREFIX
EX &ZUSER...EXEC.RACF.CLIST
/*
RACF in focus – implementing OS/390 Unix controls

‘RACF in focus’ is a regular column focusing on a specific RACF topic. Here, we discuss the issues related to implementing OS/390 Unix controls.

In the last (November 2003) issue of RACF Update, this column discussed some of the basic ideas behind OS/390 Unix security, including how to plan for Unix security, the need for unique UIDs and GIDs, issues related to superuser powers, and auditing OS/390 Unix. This time we take the ideas further, by discussing some of the implementation issues.

Because there are many considerations to contend with when implementing OS/390 Unix security under RACF, you may want to implement the controls bit by bit rather than do them all right away. This approach will allow for rational adjustments and fine-tuning as you go along, and will also avoid any adverse impact on the user community.

The reader will require a knowledge of the Unix world, and especially the OS/390 flavour of the Unix world. Note, however, that because this article focuses on RACF rather than on Unix terminology, some of the Unix concepts are not explained. You will need to get that information from one of the OS/390 Unix manuals. Remember that we’re trying to protect Unix resources, and some of the concepts mentioned here are radically different from what we may have seen thus far in OS/390 (or MVS, or z/OS). It’s essential that you work with the Unix gurus at your installation.

Most of the OS/390 Unix security implementation issues deal with creating RACF profiles in various classes and adding appropriate persons to the access lists. The essential thing is to understand what each profile does, and what are its implications.
FACILITY CLASS CONSIDERATIONS

IBM provides Facility Class profiles that greatly enhance OS/390 Unix security.

The Facility Class is a collection of assorted profiles and is used by various subsystems and applications for varied purposes. Usually, and especially for IBM-supplied profiles, the convention is that the first three characters of the profiles denote the system to which these profiles belong. This method helps group Facility Class profiles logically. For OS/390 Unix profiles in the Facility Class, IBM uses the reserved characters ‘BPX’. Hence, any profile in the Facility Class that starts with BPX is protecting an OS/390 Unix resource.

Below is a list of Facility Class profiles that apply to OS/390 Unix. In most cases, an access level of READ is sufficient to allow access.

**BPX.DEFAULT.USER**

This profile defines, via the APPLDATA field, the default Unix user and the default Unix group to be used for UID and GID purposes, when a user doesn’t have an OMVS segment defined in his profile.

It does not have anyone in the access list.

**BPX.SMIF**

Userids with READ access to this profile are allowed to write their own SMF records without the program being APF-authorized. (Normally, APF-authorization is required for this function.) This profile facilitates the objective of not having to give powerful APF authorization to a program, yet enabling it to write SMF records.

**BPX.SERVER**

Userids (usually server tasks) with READ access in this profile can change a thread’s security profile.
BPX.SUPERUSER
Userids with READ access to this profile can become superusers (by issuing the SU Unix command). Make sure you don’t have too many users in the access list of this profile – only a handful of users need to have this powerful privilege.

Read the discussion on UnixPRIV Class below, which shows how to grant just some of the superuser powers.

BPX.DAEMON
A daemon is a Unix term to denote an entity that is similar to a started task in OS/390.

Userids (daemons) with READ access to this profile can change their security identity to that of another user. A daemon may need to do this to perform a function on behalf of a user. By assuming another user’s identity, security checking is done under the user’s privileges, and not the daemon’s. The profile is there because not all daemons should be allowed to change their identities.

BPX.DEBUG
Userids with READ access to this profile can ‘dbx’ (a Unix term) on authorized programs.

BPX.WLMSERVER
Userids with READ access to this profile can use WLM (workload manager) queueing services.

BPX.FILEATTR.APF
Userids with READ access to this profile can make an HFS (Hierarchical File System) program APF-authorized – a powerful feature.

BPX.FILEATTR.PROGCTL
Userids with READ access to this profile can make an HFS file (program) ‘program controlled’.
**BPX.FILEATTR.SHARELIB**
Userids with READ access to this profile can change some attributes of the shared library region.

**BPX.STOR.SWAP**
Userids with READ access to this profile can make programs running under their authority non-swappable.

**BPX.SAFFASTPATH**
This profile is an ON/OFF switch. If it’s defined, it indicates that security access successes are not to be logged (only failures). It therefore helps reduce the number of SMF records generated.

There are no userids in the access list of this profile.

**BPX.JOBNAME**
Userids with READ access to this profile can set their own jobnames using the BPX_JOBNAME variable.

**BPX.**
This is the OS/390 Unix ‘catch-all’ profile, sometimes also known as the ‘back-stop’ profile. It’s an important profile since it’s the one used if any of the more specific BPX profiles discussed above aren’t defined at your installation. Make sure it has UACC (universal access) of NONE, and that there is no-one in the access list.

If this profile has UACC other than NONE, you may have security exposures. And if you don’t have security exposures now, you will introduce them later in your environment when IBM introduces more OS/390 Unix controls via a BPX profile.

If you don’t already have this profile defined, you should do so as follows:

```plaintext
RDEFINE FACILITY BPX.** UACC(NONE) OWNER(YOUR-CHOICE)
```

A word of caution: if you don’t have all the more specific profiles...
UNIXPRIV CLASS CONSIDERATIONS

The RACF UnixPRIV class is used solely for OS/390 Unix purposes. It provides granular security for the superuser authority. You’ll need to activate this class if you haven’t already done so.

The superuser authority covers a wide range of privileges, and you may want to give only portions of these powers to different people. UnixPRIV helps you do this.

If you already have a number of people with superuser authority, a good mini-project would be to remove them from these full powers and allow them access to some of the profiles in the UnixPRIV Class.

There are two basic variations in the profiles in this class. If the second qualifier is FILESYS, the profile is related to Unix file system privileges; if the second qualifier is PROCESS, the permission is related to Unix processes.

SUPERUSER.FILESYS

Userids with READ access to this profile can read or search any local file or directory. Userids with UPDATE access can, in addition, write to any local file. Userids with CONTROL access can, in addition to the above, also write to any directory.

These authorities apply only to HFS files, not to NFS (Network File System) files.

SUPERUSER.FILESYS.ACLOVERRIDE

If this profile is defined, the ACL overrides any access granted by the profile SUPERUSER.FILESYS.
SUPERUSER.FILESYS.MOUNT
Userids with READ access to this profile can issue the TSO mount command with the nosetuid option. Userids with UPDATE access can issue the mount command with the setuid option.

SUPERUSER.FILESYS.CHANGEPERMS
Userids with READ access to this profile can change file permission bits of any file.

SUPERUSER.FILESYS.QUIESCE
Userids with READ access to this profile can issue the quiesce/unquiesce commands for file systems mounted with the nosetuid option. Userids with UPDATE access can also issue these commands for file systems mounted with the setuid option.

SUPERUSER.FILESYS.VREGISTER
Userids with READ access to this profile can use the vreg callable service.

OTHER RACF CLASS CONSIDERATIONS
The Facility Class and UnixPRIV class profiles cover almost all the OS/390 Unix implementation issues. However, you also need to consider the following, if you haven’t already done so:

• You need to define Started Class profiles for started tasks OMVS (OMVS.*) and BPXOINIT (BPXOINIT.*).

• You will need to use the Program Class if you wish to control daemon authority via Program Class. You may find that there are several libraries (for example, SYS1.LINKLIB) that you’ll need to identify and control via Program Class.

• And finally, don’t forget the basics, like protecting your Unix HFS datasets!
  Create a profile for HFS.OMVS.. You may want to give UACC of READ, and ALTER access to the started task
OMVS KERN. This is especially important if the RACF protect-all option is not in effect at your installation, in which case all OS/390 Unix files would be unprotected!

**SUMMARY**

In this article, we’ve seen how to set up most of the OS/390 Unix related security. This information, combined with last issue’s column on understanding OS/390 Unix, should send you well on your way to protecting your Unix environment.

*Dinesh Dattani*

*Information Security Consultant, Toronto (Canada)*

*Dinesh123@rogers.com*

© Xephon 2004

---

**August 1995 – February 2004 index**

Items below are references to articles that have appeared in *RACF Update* since August 1995. References show the issue number followed by the page number(s). Back issues of *RACF Update* can be ordered from Xephon. See page 2 for details.

- Access authority: 35.3-11, 28.21-34, 29.37-43
- Access checking: 1.50-52, 26.3-11, 31.3-10
- Access list: 18.52-59, AUDIT: 5.41-61, 30.22-26, 30.60-61
- Access level: 30.42-58
- ACCHECK: 1.50-52
- ACEE: 15.28-34
- ACF2: 26.32-35, 28.51-58
- ADDGROUP: 1.22-27
- ADDUSER: 1.22-27
- Administration: 2.3-17, 2.50-63, 3.40-49, 22.3-27
- Aliases: 7.53-59, 9.11-42, 28.3-20
- Amending user-ids: 3.40-49
- Announcements: 26.21-22, 27.22-32, 28.59-64
- Authorized functions: 22.3-27
- Authorized libraries: 35.11-15
- Automation: 9.11-42
- Availability: 31.41-50
- Belkin: 28.59-64

BLKUPD 3.37-40 | Exits 32.14-17, 33.11-32
Business continuity 34.11-19 | Extracting user information 3.52-63
CANCELID 5.34-41 | Facility class 32.3-6
CAPTURE 5.11-17 | Firewall 25.44-55, 27.47-53,
Catalog alias 17.24-34, 18.17-27 | 27.54-58, 28.59-64
CATEGORY 11.3-6 | Front-end module 1.22-27
CDSPNLO1 1.3-10 | Front-end reset 9.3-11
CEMT 31.57 | FTKHPWD 8.11-14
CEXDRACF 7.53-59 | FTP 26.23-31
Checking access 4.28-37 | FTP security exit 8.11-14
Checking resources 7.53-59 | GCICS 7.16-25
CICS 3.8-11, 7.16-25, | Generation datasets 26.3-11
29.57-58 | GID 25.10-37, 30.61-62
CICSDISP 3.40-49 | Glossary 23.28-56
Class masks 7.3-15 | Group 33.3-10
Cloning 14.50-59 | Group commands 11.36-59
CMSK 7.25-52 | Groups’ accesses 5.11-17, 27.3-7
CNST 9.45-59 | GTF 16.59
CNSX 9.45-59 | Handling RJE jobs 2.17-21
COBOL transaction 23.22-27 | Hardware protection 8.3-11
Company list 23.57-63 | HASP 29.57-58
Consulting RACF 27.33-46 | Helpdesk facilities 4.40-62
Control blocks 31.29-40 | HLQ security check 34.3-10
Cryptography 11.36-59, 12.2-26 | ICHNCVOO 4.37-39
CSVAPF macro 10.20-22 | ICHPXW01 exit 6.3-11, 12.3-5,
CVH macro 1.10-21 | 20.47-53
Database 29.3-12 | ICHRXC02 14.3-8,
Database name table 23.3-21 | 29.17-19, 31.56
Dataset profile 1.22-47, 31.55 | ICHRXI02 exit 6.61-66
Dataset protection 21.49-51, 35.3-11 | ICHRRCDX 9.45-59
DB2 33.32-43 | IDCAMS 7.53-59
DDoS 25.38-43, 26.36-40 | Information 20.54-58, 21.52-58
28.3-20
Decryption of datasets 11.36-59, | Inquiry program 8.14-29
12.6-26 | INSO70 10.23-42
Deleting profiles 5.34-41 | IRRADUOO 12.28-57
Deleting user-id 3.37-40 | IRRBDUO0 5.41-61
DFHRMM 19.39-59 | IRRUUXO1 9.43-45
DFHSM 1.48-49, 19.39-59 | IRREVXO1 35.33-42
DFSMS 20.11.18 | IRRPNLOO API 5.18-33
Discrete profiles 31.55 | IRRUT100 5.11-17
Displaying system data 3.3-8 | ISFUSER 1.53-62
DSET 9.11-42 | ISP 16.7-19, 17.24-54,
DSETMAIN 9.11-42 | ISP 17.34-42, 18.17-27
DSMON 30.15-21 | ISP 10.23-42
DSN 30.42-58 | ISP table interface 1.22-47, 8.32-59,
Dynamic access 20.3-5 | 10.3-19, 11.36-59
Encryption of datasets 11.36-59,12.6-26 | JEDSPSVC 3.29-36
Enhancements 26.21-22 | JES2 31.50-54
<table>
<thead>
<tr>
<th>Contribution Area</th>
<th>Pages</th>
<th>Reference Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>SASADUOO</td>
<td>12.28-57</td>
<td>Timed Permit Facility</td>
</tr>
<tr>
<td>SAS/CPE</td>
<td>18.10-16</td>
<td>TIMEDPE</td>
</tr>
<tr>
<td>Save RACF database</td>
<td>1.48-49</td>
<td>TIMEDREM</td>
</tr>
<tr>
<td>Scrollable commands</td>
<td>4 62-63</td>
<td>TMON</td>
</tr>
<tr>
<td>SDSF</td>
<td>18.3-10</td>
<td>TSO</td>
</tr>
<tr>
<td>SDSF exit ISFUSER</td>
<td>1.53-62</td>
<td>TSO/E PROTECT/PROT</td>
</tr>
<tr>
<td>Search and display</td>
<td>4.14-28</td>
<td>TSO IDs</td>
</tr>
<tr>
<td>Search command</td>
<td>31.11-14</td>
<td>TSOPREF</td>
</tr>
<tr>
<td>SECDATA</td>
<td>11.3-6</td>
<td>UID</td>
</tr>
<tr>
<td>Security checking</td>
<td>21.6-13</td>
<td>Universal access</td>
</tr>
<tr>
<td>Security investigation</td>
<td>3.49-52</td>
<td>Unix</td>
</tr>
<tr>
<td>SETROPTS</td>
<td>13.3-18</td>
<td>Unknown categories</td>
</tr>
<tr>
<td>SETROPTS LIST</td>
<td>7.25-52</td>
<td>Updating TSO prefix</td>
</tr>
<tr>
<td>Simulator</td>
<td>10.23-42</td>
<td>User access</td>
</tr>
<tr>
<td>SKED8OF</td>
<td>11.36-59, 12.6-26</td>
<td>USERDATA</td>
</tr>
<tr>
<td>SMRM</td>
<td>3.12-24</td>
<td>User data</td>
</tr>
<tr>
<td>Software protection</td>
<td>8.3-11</td>
<td>User id</td>
</tr>
<tr>
<td>SPECIAL access</td>
<td>31.55-57</td>
<td>32.27-47, 33.3-10, 35.15-33, 35.33-42</td>
</tr>
<tr>
<td>SQL</td>
<td>27.33-46</td>
<td>User profiles</td>
</tr>
<tr>
<td>STANDBY</td>
<td>6.31-33</td>
<td>Users and datasets</td>
</tr>
<tr>
<td>STARTCLS</td>
<td>6.11-31</td>
<td>Users’ accesses</td>
</tr>
<tr>
<td>STARTED</td>
<td>6.11-31</td>
<td>USS</td>
</tr>
<tr>
<td>Started procedures</td>
<td>6.11-31</td>
<td>Validating information</td>
</tr>
<tr>
<td>Started task</td>
<td>12.27</td>
<td>Virus</td>
</tr>
<tr>
<td>STGADMIN</td>
<td>2.50-63</td>
<td>26.36-40, 26.41-44</td>
</tr>
<tr>
<td>Storage manager</td>
<td>3.12-24, 26.12-20</td>
<td>VRA</td>
</tr>
<tr>
<td>Structured display</td>
<td>10.43-59, 11.6-18</td>
<td>Watchguard SOHO</td>
</tr>
<tr>
<td>Symantec display</td>
<td>10.43-59, 11.6-18</td>
<td>Watchguard SOHO</td>
</tr>
<tr>
<td>Symantec display</td>
<td>27.47-53</td>
<td>Watchguard SOHO</td>
</tr>
<tr>
<td>Symantec Firewall/VPN 100</td>
<td>28.59-64</td>
<td>WEAKPASS</td>
</tr>
<tr>
<td>SYSGCATA</td>
<td>7.53-59</td>
<td>Web user identification</td>
</tr>
<tr>
<td>System ‘hacks’</td>
<td>14.36-38</td>
<td>Year 2000</td>
</tr>
<tr>
<td>System symbols</td>
<td>23.3-21</td>
<td>zSeries</td>
</tr>
<tr>
<td>Tape</td>
<td>26.3-11</td>
<td></td>
</tr>
</tbody>
</table>

**Contributing to RACF Update**

If you have ever experienced any difficulties with RACF, or made an interesting discovery, you could receive a cash payment simply by telling us all about it.

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

IBM’s small programming enhancement (SPE) OA03853, OA03854 enhances RACF to provide event notification using open, remote interfaces provided by the z/OS LDAP server. RACF is also enhanced to support recoverable user passwords.


* * *

Network Associates has launched a comprehensive security package for Microsoft’s new Sharepoint portal server and associated tools, called Portalshield. The package comes complete with McAfee ePolicy Orchestrator to add policy management and enforcement, rapid deployment, and distribution of the latest virus definitions.


* * *

Enterasys has announced its ‘Secure Networks’ offering, which embeds security throughout the network fabric in order to respond dynamically to threats. The system provides centralized policy management that allows companies to apply security rules based on users, applications, or organizational priorities.


* * *

BEA has released WebLogic Enterprise Security for managing application security policy.

As well as authentication, the new offering provides authorization and audit integration with single sign-on, through partnerships with Symantec and Verisign.

URL: http://dev2dev.bea.com/products/wlesecurity/index.jsp

* * *

Consul and BMC Software have announced a global alliance to offer customers total management of security administration and auditing to enable regulatory compliance.

URL: http://www.consul.com/index.php3?cid=751

* * *

Now in its ninth year, Infosecurity will be taking place at the Grand Hall at Olympia 27th–29th April 2004.

URL: www.infosec.co.uk

* * *