

RACF and REXX - A Marriage Made in Heaven!

Bruce Schaefer

R&D Manager, GRC Products

Vanguard Integrity Professionals



AGENDA

- THE WAY IT WAS. Slides 3-5
- THE WAY IT IS NOW with IRRXUTIL
 - User
 - Group
 - General resources
- Return codes
- Samples
- EXTRACT vs EXTRACTN
- Live Demo of user, group and General Resource retrieval of all available data for each specific profile.

Listing a UserID

READY

lu buddy

USER=BUDDY NAME=BUDDY T KATZ OWNER=VANGUARD CREATED=09.141
DEFAULT-GROUP=VANGUARD PASSDATE=00.000 PASS-INTERVAL= 60 PHRASEDATE=N/A
ATTRIBUTES=NONE
REVOKE DATE=NONE RESUME DATE=NONE
LAST-ACCESS=11.224/14:57:39
CLASS AUTHORIZATIONS=NONE
INSTALLATION-DATA=SEC ADMIN, DEPT K5678,PURPLE
NO-MODEL-NAME
LOGON ALLOWED (DAYS) (TIME)

ANYDAY ANYTIME
GROUP=VANGUARD AUTH=USE CONNECT-OWNER=VANGUARD CONNECT-DATE=09.141
CONNECTS= 41 UACC=NONE LAST-CONNECT=11.214/17:42:30
CONNECT ATTRIBUTES=NONE
REVOKE DATE=NONE RESUME DATE=NONE
GROUP=PROD AUTH=USE CONNECT-OWNER=PROD CONNECT-DATE=09.141
CONNECTS= 00 UACC=NONE LAST-CONNECT=UNKNOWN
CONNECT ATTRIBUTES=NONE
REVOKE DATE=NONE RESUME DATE=NONE
GROUP=SALES AUTH=USE CONNECT-OWNER=SALES CONNECT-DATE=09.141
CONNECTS= 00 UACC=NONE LAST-CONNECT=UNKNOWN

Obtaining User Information

Common procedure to process user information:

```
xx = OUTTRAP('IDDATA.')
```

```
address TSO 'LISTUSER 'getid
```

```
do dix = 1 to iddata.0
```

```
  /* Get UserID's Current Name */
```

```
  if POS('NAME=',iddata.dix) <> 0 then do
```

```
    bpos = POS('NAME=',iddata.dix) + 5 /* begin position */
```

```
    username = SUBSTR(iddata.dix,bpos,20)
```

```
  end
```

```
end
```

**NOT A
RECOMMENDED
PROGRAMMING
INTERFACE**

Finding Group-Special

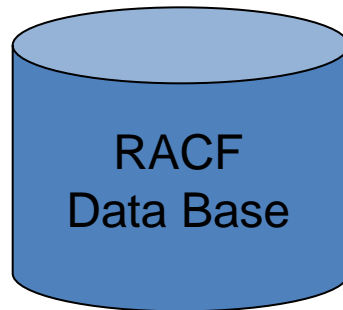
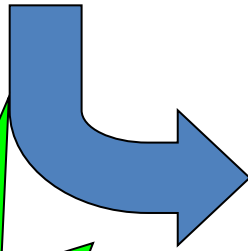
Must parse entire listing to find group connections:

```
if POS('GROUP=',IDDATA.idx) > 0 then do
  /* find group special */
  tempLen = LENGTH(WORD(iddata.idx,1)
  currentgrp = SUBSTR(word(iddata.idx,1),7,tempLen)
  if POS('ATTRIBUTES=SPECIAL',IDDATA.idx) > 0 then
    say currentgrp 'SPECIAL'
  end
```

REXX API Interface to RACF Profiles

Solution: New REXX interface (IRRXUTIL) is available to extract RACF information and store in stem variables.

/ REXX */*



Calling IRRXUTIL

Invoke IRRXUTIL using the REXX statement:

`myrc=IRRXUTIL(command,type,profile,stem,prefix,generic)`

command	EXTRACT or EXTRACTN
type	USER, GROUP, CONNECT, SETROPTS or any General Resource class
profile	the profile name to extract
stem	the REXX stem variable name
prefix	optional prefix to add to stem variable parts
generic	optional indicator for general resources to determine how to handle profile not found condition

Return Codes

IRRXUTIL returns a string containing a space separated list of return codes.

Some common ones are:

'0 0 0 0 0' means everything worked. Actually if the first word is 0, then it worked.

'8 x y 0 0' = Error in IRRXUTIL invocation

–“x” – Number of the incorrect parameter

–“y” – What’s wrong

- 1: Bad length
- 2: Bad value
- 3: Inconsistent with other parameters

'12 12 8 8 24' means you lack access to the r_admin function

'12 12 4 4 4' means profile not found.

Return Codes

Description	RC1	RC2	RC3	RC4	RC5
Success	0	0	0	0	0
Warning, stem contained '.'	2	0	0	0	0
Bad number of parameters specified	4	Number of parms specified	Min number allowed	Max number allowed	0
Parameter Error	8	Index of bad parameter	1=Bad length 2=Bad value 3=Imcompatible with other parms	0	0
R_admin failure	12	12	R_admin safrc	R_admin racfrc	R_admin racfrsn
Environmental error	16	0=Rexx Error 4=R_admin error	For IBM support	For IBM support	0

Using IRRXUTIL

Rule of thumb: Stem variables returned always have the 0th element set to the number of elements.

Simple: True or False values

```
myrc=IRRXUTIL("EXTRACT", "USER", user, "RACF")
```

Ex: RACF.BASE.SPECIAL.0 = 1
 RACF.BASE.SPECIAL.1 = "TRUE"

Using IRRXUTIL

Retrieving User's Name:

```
myrc = IRRXUTIL("EXTRACT","USER",getid,"IDDATA","","FALSE")
```

```
/* Get UserID's Name */
```

```
username = IDDATA.BASE.NAME.1
```

IRRXUTIL Security Controls

- The caller needs access to use R_admin extract via the appropriate FACILITY class profile protecting the desired function.
- In addition, the caller must be allowed to retrieve the profile in question. The caller will only have fields they are allowed to view returned.
- This is all enforced by the R_admin extract function which IRRXUTIL calls.

Profile Type	Required FACILITY profile
User, Connect	IRR.RADMIN.LISTUSER
Group	IRR.RADMIN.LISTGRP
General Resource	IRR.RADMIN.RLIST
Setropts	IRR.RADMIN.SETROPTS.LIST

IRRXUTIL note

- A user-to-group connection can be extracted by specifying the CONNECT class and providing a profile name using the form *userID.group-name*.
- SETROPTS settings can be obtained by specifying "_SETROPTS" as both the class name and the profile name

EXTRACT vs. EXTRACTN

- EXTRACT is used when you have an exact profile you want.
- EXTRACTN is used when you want to loop through a bunch or all profiles.
- Start with one and use that to get the next one, hence the EXTRACT NEXT. Much like a Get Next in an IMS Query.....

Extract NEXT for General Resource Profiles

When extracting General Resources with EXTRACTN, start out with non generic profiles, by specifying 'FALSE' for the GENERIC parameter.



Every time IRRXUTIL(EXTRACTN...) is called, pass in the returned 'generic' indicator (stem.GENERIC), along with the returned profile name.

IRRXUTIL(EXTRACTN..) will automatically switch over to GENERIC profiles when it has gone through all discrete profiles.

Parallel Stem Variables

- IRRXUTIL makes liberal use of the concept of parallel stem variables or arrays.
- One must understand that stem.x for a certain field is necessarily related to another stem variable stem2.x where x is the value of the stem...

WHAT????????????????????

Parallel Stem Variables

IDDATA.BASE.CGROUPO=29 says that 29 connect entries for a user exists.

Lets look at number 23

- IDDATA.BASE.CGROUPO.23=SYS1
- IDDATA.BASE.CAUTHDA.23=04/29/09
- IDDATA.BASE.COWNER.23=SYS1
- IDDATA.BASE.CLJTIME.23=
- IDDATA.BASE.CLJDATE.23=
- IDDATA.BASE.CUACC.23=READ
- IDDATA.BASE.CINITCT.23=0
- IDDATA.BASE.CADSP.23=FALSE
- IDDATA.BASE.CSPECIAL.23=FALSE
- IDDATA.BASE.COPER.23=FALSE
- IDDATA.BASE.CREVOKFL.23=FALSE
- IDDATA.BASE.CGRPACC.23=FALSE
- IDDATA.BASE.CAUDITOR.23=FALSE
- IDDATA.BASE.CREVOKE.23=
- IDDATA.BASE.CRESUME.23=

Common Variables

Default Group - IDDATA.BASE.DFLTGRP.1

User Name - IDDATA.BASE.NAME.1

Installation Data - IDDATA.BASE.DATA.1

If data1 <> " then do d = 1 to IDDATA.BASE.DATA.0

 data = data IDDATA.BASE.DATA.d

End

Password Interval - IDDATA.BASE.PASSINT.1

Password Date - IDDATA.BASE.PASSDATE.1

User Attributes

SPECIAL - IDDATA.BASE.SPECIAL.1 = 'TRUE'

OPERATIONS - IDDATA.BASE.OPER.1 = 'TRUE'

AUDITOR - IDDATA.BASE.AUDITOR.1 = 'TRUE'

REVOKED - IDDATA.BASE.REVOKEFL.1 = 'TRUE'

PROTECTED - IDDATA.BASE.PROTECTD.1 = 'TRUE'

RESTRICTED - IDDATA.BASE.REST.1 = 'TRUE'

Group Connections

Finding Group Connections and Group Attributes:

```
do grp = 1 to IDDATA.BASE.CGROUP.0
  group = left(IDDATA.BASE.CGROUP.grp,8)
  gs = ''
  if IDDATA.BASE.CSPECIAL.grp = 'TRUE' then
    gs = gs || 'SPECIAL'
  if IDDATA.BASE.COPER.grp = 'TRUE' then
    gs = gs || 'OPERATIONS'
  if IDDATA.BASE.CAUDITOR.grp = 'TRUE' then
    gs = gs || 'AUDITOR'
  say "  "group gs
end
```

Available Base Fields

To find Base fields defined for a given user:

```
say "Valid BASE fields for this user are:"  
do i = 1 to IDDATA.BASE.0  
    say IDDATA.BASE.i  
end
```

Optional Segments

Finding Optional Segments assigned to User:

```
seg = IDDATA.u
do s = 1 to IDDATA.seg.0
  fieldName = IDDATA.seg.s
  segment = left(IDDATA.seg.s,8)
  say seg "segmentfield" segment "=" IDDATA.seg.fieldName.1
end
```

GETPINFO

```
/*rexx*/  
parse upper arg class reqname  
stemvar = "pinf."  
myrc = IRRXUTIL("EXTRACT",class,reqname,stemvar,, "FALSE")  
say " Util RC="myrc  
if word(myrc,1)<>0 then do  
    say "IRRXUTIL call failed - RC="myrc  
    exit 1  
end  
say "base Stem Variable is:"translate(stemvar)  
say "   class="pinf.CLASS  
say "  profile="pinf.PROFILE  
say "  version="pinf.VERSION  
say "  generic="pinf.GENERIC  
nSegs = pinf.0    /* Number of segments */
```

GETPINFO Continued

```
do i=1 to nSegs
  sname = pinf.i
  nFields = pinf.sname.0
  say sname "has" nFields "fields"
  do j=1 to nFields
    fname = pinf.sname.j
    rcnt = pinf.sname.fname.0
    if pinf.sname.fname.repeating = "TRUE" then do
      say sname"."fname".0="rcnt
      do rp=1 to rcnt
        say "  "sname"."fname"."rp "=" pinf.sname.fname.rp
      end
    end
  else
    say "  "sname"."fname".1="pinf.sname.fname.1
  end
end
exit 0
```


GETPINFO Driver

```
/*rexex*/  
say "*****"  
say "**      Begin User output for TSBM00      **"  
say "*****"  
irc = GETPINFO("USER" "TSBM00")  
say "*****"  
say "**      Begin group output for  SBS#ISTS      **"  
say "*****"  
irc  = GETPINFO("GROUP" "SBS#ISTS")  
say "*****"  
say "**      Begin general resource Facility      **"  
say "*****"  
irc  = GETPINFO("FACILITY" "BPX.SUPERUSER")
```

Using GETPINFO to get SETROPTS

```
/*rexx*/  
say "*****"  
say "**      Begin SETROPTS      **"  
say "*****"  
irc = GETPINFO("_SETROPTS" "_SETROPTS")
```

GETNPRFS

```
/*rexx*/  
parse upper arg class profile numberofprofiles  
if profile = "NULL" then profile = " "  
stemvar = "pinf."  
pinf.GENERIC = "FALSE"  
pinf.PROFILEX = profile  
counter = 0  
do forever  
  if counter = numberofprofiles then leave  
  counter = counter + 1  
  myrc=IRRXUTIL("EXTRACTN",class,pinf.PROFILEX,stemvar,,pinf.GENERIC)  
  say " IRRXUTIL RC="myrc  
  if word(myrc,1)<>0 then do  
    say "IRRXUTIL call failed - RC="myrc  
    exit 1  
  end
```

GETNPRFS

```
say "*****"
say "*****START OF NEW PROFILE *****"
say "*****"
say "base Stem Variable is:"translate(stemvar)
say "  class="pinf.CLASS
say "  profile="pinf.PROFILE
say "  version="pinf.VERSION
say "  generic="pinf.GENERIC
```

```
nSegs = pinf.0 /* Number of segments */
do sp=1 to nSegs
  sname = pinf.sp
  nFields = pinf."sname".0
  say sname "has" nFields "fields"
  do fp=1 to nFields
    fname = pinf.sname.fp
    rcnt = pinf.sname.fname.0
```

GETNPRFS

```
if pinf.sname.fname.repeating="TRUE" then do
  say sname"."fname".0="rcnt
  do rp=1 to rcnt
    say "  "sname"."fname"."rp"="pinf.sname.fname.rp
  end
end
else
  say "  "sname"."fname".1="pinf.sname.fname.1
end
end
end
exit 0
```

GETPRFS

PASS in NULL for PROFILE if you want it to start at the first discrete profile.

PASS in number of profiles to return.

```

/*rexx*/
say "*****"
say "**          Call with value          **"
say "*****"
irc = GETNPRFS("USER" "NULL" "3")
say "*****"
say "**          Call with value          **"
say "*****"
irc = GETNPRFS("GROUP" "XXXX" "5")
say "*****"
say "**          Call with value          **"
say "*****"
irc = GETNPRFS("FACILITY" "XXXX" "2")
exit 0
  
```

Please come see us at the Expo Booth XXX



Please ensure you do you session evaluations !!
Session 16616

