VANGUARD
INTEGRITY PROFESSIONALS
CYBERSECURITY EXPERTS


**Brian Marshall**

VP Research and Development

Vanguard Integrity Professionals


**Session 15992**

IBM Server Proven

Business Partner IBM

# 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:

**NOT A RECOMMENDED PROGRAMMING INTERFACE**

```
Xx = OUTTRAP('IDDATA.')
ADDRESS TSO 'LISTUSER 'getid'
Do DATA = 1 to IDDATA.0
  /* Get UserID's Current Name */
  If INDEX(iddata.data,'NAME=') ¬= 0 then do
    BPOS = INDEX(iddata.data,'NAME=') + 5 /* begin position */
    username = SUBSTR(iddata.data,BPOS,20)
  End
End
```

# Finding Group-Special



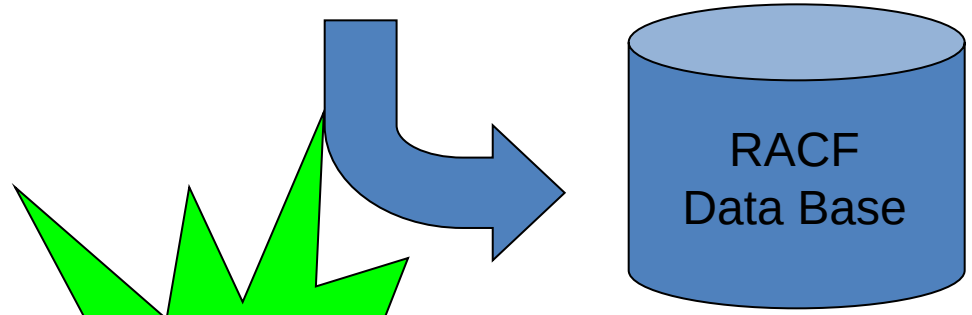
Must parse entire listing to find group connections:

```
If INDEX(IDDATA.IDDATA,'GROUP=') > 0 then
   /* find group special */
   currentgrp=SUBSTR(word(iddata.data,1),7,LENGTH(WORD(idd
   ata.data,1)
   If INDEX(IDDATA.DATA,'ATTRIBUTES=SPECIAL') > 0 then
   Say currentgrp  'SPECIAL'
```

# REXX API Interface to RACF Profiles

Solution: New REXX interface (IRRXUTIL) is added to extract RACF information and store in a stem variable

/* REXX */




**Available starting in z/OS 1.11**

# 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<br>2=Bad value<br>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<br>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



USER.BASE.CGROUP.0=29 says that 29 connect entries for a user exists.

Lets look at number 23

- BASE.CGROUP.23=SYS1
- BASE.CAUTHDA.23=04/29/09
- BASE.COWNER.23=SYS1
- BASE.CLJTIME.23=
- BASE.CLJDATE.23=
- BASE.CUACC.23=READ
- BASE.CINITCT.23=0
- BASE.CADSP.23=FALSE
- BASE.CSPECIAL.23=FALSE
- BASE.COPER.23=FALSE
- BASE.CREVOKFL.23=FALSE
- BASE.CGRPACC.23=FALSE
- BASE.CAUDITOR.23=FALSE
- BASE.CREVOKE.23=
- 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
  gs = '' ; group = ''
  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'
  group = left(IDDATA.BASE.CGROUP.grp,8)
  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 b = 1 to IDDATA.BASE.0
  Say IDDATA.BASE.b
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 "segment field "s segment " =
     "IDDATA.seg.fieldName.1
End
```

# IRRXTRCT



```
 /*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
  sc=pinf.0   /* Number of segments */
```

# IRRXTRCT Continued

```
 do sp=1 to sc
sname=pinf.sp
  interpret "fc=pinf."sname".0"
  say sname "has "fc" fields"
  do fp=1 to fc
   interpret "fname=pinf."sname".fp"
   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
trace n
  end
end
exit 0
```

# IRRXTRCT Driver



```
 /*rexx*/
SAY "************************************************"
say "**      Begin User output for TSBM00        **"
SAY "************************************************"
irrxdata=irrxtrct("USER" "TSBM00")
SAY "************************************************"
say "**    Begin group output for  SBS#ISTS      **"
SAY "************************************************"
irrxdata=irrxtrct("GROUP" "SBS#ISTS")
SAY "************************************************"
say "**      Begin general resource Facility     **"
SAY "************************************************"
irrxdata=irrxtrct("FACILITY" "BPX.SUPERUSER")
```

# Using IRRXTRT to get SETROPTS



```
 /*rexx*/
SAY "************************************************"
say "**       Begin SETROPTS                   **"
SAY "************************************************"
irrxdata=irrxtrct("_SETROPTS" "_SETROPTS")
```

# IRRXGETN



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

# IRRXGETN

```
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

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

# IRRXGETN

```
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
```

# IRRXGETNT



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 "************************************************"
irrxdata=irrgetn("USER" "NULL" "3")
SAY "************************************************"
say "**      Call with value                 **"
SAY "************************************************"
irrxdata=irrgetn("GROUP" "XXXX" "5")
SAY "************************************************"
say "**      Call with value                 **"
SAY "************************************************"
irrxdata=irrgetn("FACILITY" "XXXX" "2")
```

# Special Thanks



- Doug Behrends - PS Consultant Vanguard Integrity Professionals
  - Doug Supplied a major portion of the presentation and rexx samples.
- Mark Nelson - IBM Security Guru, all around great person and giver of candy
  - Mark supplied a major portion of the presentation (he does not know this…….. YET).
- Aubrey Shuping – Granddaughter and absolute love of my life
  - She allowed me to cobble this presentation together after she had her fill of candy (supplied by Mark Nelson and yours truly) and went to bed.

# Special Jeers



- Doug Behrends – PS Consultant Vanguard Integrity Professionals.
  - Doug was supposed to do this presentation originally… Yes, he begged out of it… But I got even…
- Roxane Rosberg – Client Care extraordinaire at Vanguard Integrity Professionals.
  - Rox was the one that replaced Doug with yours truly.
- Parker Shuping – Grandson and other absolute love of my life
  - He disallowed me to cobble this presentation together after he had his fill of candy (supplied by Mark Nelson and yours truly)… IT is MIDNIGHT as I finish this…

# Special Thanks



VANGUARD
INTEGRITY PROFESSIONALS
CYBERSECURITY EXPERTS

# Please come see us at the Expo Booth 707

Please ensure you do you session evaluations !!

**Session 15992**

IBM Server Proven

Business Partner IBM