RACF Power Tools – Using IRRICE and Rexx on IRRADU00 and IRRDBU00 Part 2

NewEra Software - The z Exchange
June 17, 2015

Thomas Conley
Pinnacle Consulting Group, Inc.
59 Applewood Drive
Rochester, NY 14612-3501
P: (585)720-0012
F: (585)723-3713
pinncons@rochester.rr.com

Abstract

Do you need to audit a RACF environment? Do you need to figure out who is doing what to whom? Are you one step away from a failed PCI/HIPAA/SOX/DISA audit? If you answered yes to any of these questions, then this session is for you! Come to this session to learn about IRRADU00 and IRRDBU00, and how to use IRRICE and Rexx to extract the data you need. This session is designed for skill levels from beginners to experts.

Agenda

- Part 1 Review
- Record Layouts
- Modifying IRRICE
- Using Rexx
- Summary
- Finally…

Part 1 Review

- IRRADU00 is SMF unload of RACF event data
- IRRDBU00 is unload of RACF database
- IRRICE is set of canned reports in that can be used to audit RACF, found in SYS1.SAMPLIB(IRRICE)

IRRADU00 Record Layouts

- IRRICE reports dependent upon IRRADU00 record layouts to select and report RACF data
- Record layouts for IRRADU00 defined in RACF Macros and Interfaces manual
- Two parts to records produced by IRRADU00
  - Header section, with common information such as date/time stamp, user id, system id, and event type
  - Event type section, with specific information for event
- Dozens of event types tracked by IRRADU00

IRRADU00 Record Layouts

- IRRADU00 sample records:

```
ACCESS   INSAUTH  18:33:05 2012-04-16 SYSA YES  NO   NO   TCONLEY  PINN
DEFINE   INSAUTH  13:52:24 2012-04-16 SYSA YES  NO   NO   TCONLEY  PINN
JOBINIT  INVNPWD  10:01:56 2012-04-16 SYSA YES  NO   NO   TCONLEY  PINN
JOBINIT  INVPSWD  09:45:39 2012-04-16 SYSA YES  NO   NO   TCONLEY  PINN
JOBINIT  PWDEXPR  10:01:56 2012-04-16 SYSA YES  NO   NO   TCONLEY  PINN
```

- Event code in columns 1-8 determines record format
- Record format described in record "extension" based on event code

IRRDBU00 Record Layouts

- IRRICE reports dependent upon IRRDBU00 record layouts to select and report RACF data
- Record layouts for IRRDBU00 defined in RACF Macros and Interfaces manual
- Record types logically organized by groups, users, datasets, and resources
- Record types defined by 4-byte id number in first 4 bytes of each record

IRRDBU00 Record Layouts

- Record type id number is in format PPSF
- PP is profile type, 01 for groups, 02 for users, 04 for data sets, and 05 for general resources
- S is segment number, 0 for base segment, for all others, segment value determined by position of segment in RACF template
- F is repeat group within segment, with zero (0) indicating non-repeat groups within segment

IRRDBU00 Record Layouts

- Within each logical division, record type 00 is base record, with other record types holding more specific information, such as segments
- For groups, here are some record types:
  - 0100 - Group Basic Data
  - 0101 - Group Subgroups
  - 0102 - Group Members
  - 0120 - Group OMVS Segment

IRRDBU00 Record Layouts

- Some user record types:
  - 0200 - User Basic Data
  - 0203 - User Group Connections
  - 0220 - User TSO Segment
  - 0270 - User OMVS Segment
- Some dataset record types:
  - 0400 - Data Set Basic Data
  - 0402 - Data Set Conditional Access
  - 0404 - Data Set Access

IRRDBU00 Record Layouts

- Some general resource record types:
  - 0500 - General Resource Basic Data
  - 0503 - General Resource Members
  - 0505 - General Resource Access
  - 0507 - General Resource Conditional Access
  - 0540 - General Resource Started Task Data (STDATA)
  - 05C0 - General Resource CDTINFO Data

Modifying IRRICE

- Example of IRRADU00 report VIOL (modified to fit):

```
- 1 -        VIOL: Access Violations        17/07/08        05:37:49 pm

Date         Time       Result     User ID    Resource Name              Class      Volume   Profile
----------   ---------  ---------  ---------  -------------------------  ---------  ------   -----------
2012-04-16   18:33:05   INSAUTH    TCONLEY    MASTER.SYSACAT             DATASET    MCATV1   MASTER.*.**
```

- What's missing in this report?
- Data you'd need to remediate these events, such as access level requested and access level granted

Modifying IRRICE

- Some IRRICE reports would benefit from additional information
- In earlier examples, IRRICE reports would benefit from additional info
  - VIOL - Requested/granted access level
  - OPER - Requested/granted access level
  - URVK - Last access date/time (Is this ID obsolete?)
- To modify these reports, use record layouts in RACF Macros and Interfaces manual

Modifying IRRICE

- Let's modify VIOL report
- First step is to look at VIOLCNTL member

```
SORT    FIELDS=(32,10,CH,A,23,8,CH,A,63,8,CH,A)
INCLUDE COND=(5,8,CH,EQ,C'ACCESS',AND,
              48,3,CH,EQ,C'YES')
OPTION  VLSHRT
```

- INCLUDE COND statement indicates event type ACCESS, review of record layout shows cols 1-8 is event type and cols 44-47 is YES/NO field for "Is this a violation?"

Modifying IRRICE

- Event type cols 1-8, violation cols 44-47

```
SORT    FIELDS=(32,10,CH,A,23,8,CH,A,63,8,CH,A)
INCLUDE COND=(5,8,CH,EQ,C'ACCESS',AND,
              48,3,CH,EQ,C'YES')
OPTION  VLSHRT
```

- Note that SORT control statement is 4 bytes more than position listed in manual
- IRRADU00 and IRRDBU00 produce VB files, so you have to add 4 to column position for RDW

Modifying IRRICE

- Now that we know event type ACCESS, we use ACCESS record extension section of manual
- ACC_REQUEST cols. 538-545
- ACC_GRANT cols. 547-554
- Next step is to modify VIOL member (changes shown in bold italic)

Modifying IRRICE

```
**********************************************************************
* Name: VIOL                                                         *
*                                                                    *
* Find all of the resource accesses which represent a violation.     *
**********************************************************************
 SORT    FROM(ADUDATA) TO(TEMP0001) USING(RACF)
 DISPLAY FROM(TEMP0001) LIST(PRINT) -
         PAGE -
         TITLE('VIOL: Access Violations')-
         DATE(YMD/) -
         TIME(12:)  -
         BLANK -
         ON(32,10,CH)  HEADER('Date') -
         ON(23,8,CH)   HEADER('Time') -
         ON(14,8,CH)   HEADER('Result') -
         ON(63,8,CH)   HEADER('User ID') -
         ON(286,30,CH) HEADER('Resource Name') -
         ON(578,8,CH)  HEADER('Class') -
         ON(564,6,CH)  HEADER('Volume') -
         ON(605,30,CH) HEADER('Profile') -
         ON(542,8,CH)  HEADER('Acc Rqst') -
         ON(551,8,CH)  HEADER('Acc Grnt')
```

Modifying IRRICE

- Resulting report (reduced to fit):

```
- 1 -       VIOL: Access Violations       12/07/27       10:23:35 am

Date       Time     Result   User ID  Resource Name  Class    Volume Profile     Acc Rqst Acc Grnt
---------- -------- -------- -------- -------------- -------- ------ ----------- -------- --------
2012-04-16 18:33:05 INSAUTH  TCONLEY  MASTER.SYSACAT DATASET  MCATV1 MASTER.*.** UPDATE   READ
```

- As an exercise, use this example and modify OPER and URVK back at your shop

Using Rexx

- Using Rexx allows creation of much more complex and in-depth reports
- Can relate data external to RACF to internal RACF data (e.g. catalog vs. HLQ)
- Can read IRRDBU00 to determine intricate logical relationships
- Following report lists users with SPECIAL, OPERATIONS, and AUDITOR and counts for each

Using Rexx

```
/* rexx */
#specuser = 0
#operuser = 0
#audtuser = 0
specuser. = ''
operuser. = ''
audtuser. = ''
#getrec    = 50000
"EXECIO 0 DISKR IRRDBU00 (OPEN)"
eof = 0
do forever until(eof)
   drop irrdbu00.
   "EXECIO" #getrec "DISKR IRRDBU00 (STEM IRRDBU00.)"
   if rc <> 0 then
      eof = 1
```

Using Rexx

```
do i = 1 to irrdbu00.0
     parse var irrdbu00.i 1 type 5 .
     select
        when (type = '0200') then
           do
              parse var irrdbu00.i 6 userid 14 .,
                                     40 userspec 44  .,
                                     45 useroper 49  .,
                                    386 useraudt 390 .
              if userspec = 'YES' then
                 do
                    #specuser = #specuser + 1
                    specuser.#specuser = userid
                 end
```

Using Rexx

```
                  if useroper = 'YES' then
                     do
                        #operuser = #operuser + 1
                        operuser.#operuser = userid
                     end
                  if useraudt = 'YES' then
                     do
                        #audtuser = #audtuser + 1
                        audtuser.#audtuser = userid
                     end
               end
            otherwise
         end
      end
   end
```

Using Rexx

```
drop irrdbu00.
"EXECIO 0 DISKR IRRDBU00 (FINIS)"
say 'USERS WITH EXTRAORDINARY AUTHORITY'
say ' '
say 'USERS WITH SPECIAL'
say ' '
do i = 1 to #specuser
   say specuser.i
end
say ' '
say 'TOTAL NUMBER OF USERS WITH SPECIAL AUTHORITY    =' #specuser
say ' '
say 'USERS WITH OPERATIONS'
say ' '
```

Using Rexx

```
do i = 1 to #operuser
   say operuser.i
end
say ' '
say 'TOTAL NUMBER OF USERS WITH OPERATIONS AUTHORITY =' #operuser
say ' '
say 'USERS WITH AUDITOR'
say ' '
do i = 1 to #audtuser
   say audtuser.i
end
say ' '
say 'TOTAL NUMBER OF USERS WITH AUDITOR AUTHORITY    =' #audtuser
```

Using Rexx

- Output from Rexx exec:

```
USERS WITH EXTRAORDINARY AUTHORITY

USERS WITH SPECIAL

IBMUSER
TCONLEY

TOTAL NUMBER OF USERS WITH SPECIAL AUTHORITY     = 2
```

Using Rexx

- Output from Rexx exec:

```
USERS WITH OPERATIONS

IBMUSER
TCONLEY

TOTAL NUMBER OF USERS WITH OPERATIONS AUTHORITY = 2

USERS WITH AUDITOR

IBMUSER

TOTAL NUMBER OF USERS WITH AUDITOR AUTHORITY    = 1
```

Summary

- Reviewed record layouts for IRRADU00 and IRRDBU00
- Discussed modifying IRRICE reports
- Showed how to use Rexx for analysis

Finally….

- I’m always interested to hear about your experiences with RACF, IRRADU00, IRRDBU00, and IRRICE, so if you have questions or come up with a neat solution to a problem, drop me an Email pinncons@rochester.rr.com