

The Advantages Of External Security For DB2 And The Migration Towards RACF

Kurt Struyf, infocura

Information On Demand 2010

The Premier Forum for Information & Analytics

Gain Insight. Optimize Results.



Agenda

- The need for better data security
- What does DB2 offer ?
- Why externalizing security ? Why RACF ?
- How to migrate from DB2 security to RACF security ?
 - Translate DB2 into RACF
 - Getting started with RACF
 - Fallback
 - Potential issues





The need for better data security

- **Internal reasons**

- Audit team

- separation of duties
- Controlling your administrative users
- Access to your administrative users
- Not within this scope but also important:
 - Who changed, which data and when ?
 - Did someone issue a start force command?

- Job loss





The need for better data security

- **External reasons**

- Negative publicity
- Damage to company reputation
- Share price erosion
- Loss of customer loyalty
- Loss of revenue
- Fines, penalties and bank rating
- Increased operations costs





The need for better data security

- Data theft is big business:
 - *In 2007 personal information for at least 53 million US citizens has been lost, stolen or compromised*
 - Your data is worth a lot, not just to you :
 - Credit Card Number With PIN - \$500
 - Credit Card Number with Security Code and Expiration Date - \$7-\$25
 - Drivers License - \$150
 - Birth Certificate - \$150
 - Social Security Card - \$100
 - Paypal account Log-on and Password - \$7

(Source: USA TODAY research 10/06)





Agenda

- The need for better data security
- What does DB2 offer ?
- Why externalizing security ? Why RACF ?
- How to migrate from DB2 security to RACF security ?
 - Translate DB2 into RACF
 - Getting started with RACF
 - Fallback
 - Potential issues



What does DB2 offer ?

- Multi Level Security (RACF pre-req)
- Trusted context
- Trusted role
- Secure Socket Layer
- Enterprise Identity Mapping (RACF pre-req)
- Improved auditing
 - Client userid of end-user
 - Workstation name
 - IP-address





Agenda

- The need for better data security
- What does DB2 offer ?
- Why externalizing security ?
- How to migrate from DB2 security to RACF security ?
 - Translate DB2 into RACF
 - Getting started with RACF
 - Fallback
 - Potential issues





Externalizing DB2 security in RACF

- NEW security options are possible (e.g. multi level security)
- SECURITY and DATA are separated
- Advantages of managing security in RACF
 - DB2 follows company standard
 - Generic RACF profiles
 - RACF rules can manage multiple subsystems
 - Objects don't need to exist
 - Rights are preserved when object is dropped
 - Eliminating cascading revoke (e.g. SYSADM leaves company)
 - **RACF AUDITING option is POSSIBLE**



Agenda

- The need for better data security
- What does DB2 offer ?
- Why externalizing security ? Why RACF ?
- How to migrate from DB2 security to RACF security ?
 - Translate DB2 into RACF
 - Getting started with RACF
 - Fallback
 - Potential issues



Native DB2 security

- DB2 can manage it's own security through
GRANT and REVOKE
Example: GRANT SELECT ON TAB1 TO USER1
- Information is stored SYSIBM.SYSxxxxAUTH
tables

SYSIBM.SYSCOLAUTH
SYSIBM.SYSDBAUTH
SYSIBM.SYSPACKAUTH
SYSIBM.SYSPLANAUTH
SYSIBM.SYSRESAUTH
SYSIBM.SYSROUTINEAUTH
SYSIBM.SYSSCHEMAAUTH
SYSIBM.SYSSEQUENCEAUTH
SYSIBM.SYSTABAUTH
SYSIBM.SYSUSERAUTH



Translate DB2 to RACF

- To know:
 - In DB2 one catalog/SSID to manage Security
 - One RACF to manage multiple SSID, objects must be **qualified**
 - In DB2 **catalog table** per type of object
e.g. SYSDBAUTH
 - One RACF **class** per type of object



RACF classes for DB2

Class name	Description
DSNADM	DB2 administrative authority class
MDSNBP	buffer pool privileges
MDSNCL	collection privileges
MDSNDB	database privileges
...	
MDSNSM	system privileges
MDSNTB	table, index, or view privileges
MDSNTS	tablespace privileges
...	

Appendix A (source: RACF Access Control Module Guide SC18-7433-02)



Translate DB2 to RACF

Grant **select** on **table owner.tb1** to user1;

Permit **SSID.OWNER.TB1.SELECT CLASS(MDSNTB)** id(user1) **ACCESS(READ)**

Revoke **select** on **table owner.tb1** from user1;

Permit **SSID.OWNER.TB1.SELECT CLASS(MDSNTB)** id(user1) **DELETE**

SSID.OWNER.TB1.SELECT = PROFILE



Translate DB2 to RACF

- In DB2 only grant if table exists !
- In RACF only permit if profile exists !
Define profile first

RDEF MDSNTB **SSID.OWNER.TB1.SELECT** UACC(NONE)

UACC = default access allowed on this profile

NONE = without explicit permit no access allowed

READ = this profile is open to anyone





Order of checking

For table privileges:

1. Ownership of table
2. MDSNTB privileges (select/insert/update...)
3. DSNADM privileges
 1. DBADM
 2. SYSADM
4. If no RACF profile, DB2 decides
 1. Table privileges in SYSIBM.SYSTABAUTH
 2. DBA rights in SYSIBM.SYSDBAUTH
 3. SYSADM rights in SYSIBM.SYSUSERAUTH



“granting” privileges

```
RDEF MDSNTB DB8G.CPKST.EMPLOYEE.*           UACC(NONE)
RDEF MDSNTB DB8G.CPKST.EMPLOYEE.SELECT       UACC(NONE)
RDEF MDSNTB DB8G.CPKST.EMPLOYEE.UPDATE       UACC(NONE)
RDEF MDSNTB DB8G.CPKST.EMPLOYEE.INSERT UACC(NONE)
RDEF MDSNTB DB8G.CPKST.EMPLOYEE.DELETE UACC(NONE)
```

- Only owner (CPKST) can access the table
- To “grant” access use permit command

```
PERMIT DB8G.CPKST.EMPLOYEE.SELECT CLASS(MDSNTB) ID(CPACA) ACCESS(READ)
PERMIT DB8G.CPKST.EMPLOYEE.INSERT CLASS(MDSNTB) ID(CPACA) ACCESS(READ)
PERMIT DB8G.CPKST.EMPLOYEE.UPDATE CLASS(MDSNTB) ID(CPACA) ACCESS(READ)
PERMIT DB8G.CPKST.EMPLOYEE.DELETE CLASS(MDSNTB) ID(CPACA) ACCESS(READ)
```

...

```
SETROPTS RACLIST(MDSNTB) REFRESH → to activate/refresh
```



“revoking” privileges

To “revoke” also use permit command

```
PERMIT DB8G.CPKST.EMPLOYEE.SELECT CLASS(MDSNTB) ID(CPACA) DELETE
```

Be careful

- Dynamic statement cache not flushed (access remains)
 - Run “grant select” + “revoke select” for that user **OR**
 - Run “runstats .. Update none report no” for tablespace
- Packages not invalidated



“granting” generically

```
RDEF MDSNTB DB8G.CPKST.*.*          UACC(NONE)
RDEF MDSNTB DB8G.CPKST.*.SELECT     UACC(NONE)
RDEF MDSNTB DB8G.CPKST.*.UPDATE     UACC(NONE)
RDEF MDSNTB DB8G.CPKST.*.INSERT     UACC(NONE)
RDEF MDSNTB DB8G.CPKST.*.DELETE     UACC(NONE)
```

- Protects all tables of owner CPKST
- To “grant” access to alle tables of that user

```
PERMIT DB8G.CPKST.*.SELECT CLASS(MDSNTB) ID(CPACA) ACCESS(READ)
PERMIT DB8G.CPKST.*.INSERT CLASS(MDSNTB) ID(CPACA) ACCESS(READ)
PERMIT DB8G.CPKST.*.UPDATE CLASS(MDSNTB) ID(CPACA) ACCESS(READ)
PERMIT DB8G.CPKST.*.DELETE CLASS(MDSNTB) ID(CPACA) ACCESS(READ)
```

...

SETROPTS RACLIST(MDSNTB) REFRESH → to activate/refresh



“closing” security

RDEF MDSNTB DB8G.*.*.* UACC(NONE)

- With this profile all tables are protected.
- Only owners can access their own tables
- Users holding administrative rights



Profile order

The most specific profile decides

```
RDEF MDSNTB DB2P.AW.*.SELECT UACC(READ)
```

```
RDEF MDSNTB DB2P.AW.TAB*.SELECT UACC(NONE)
```

```
Permit DB2P.AW.TAB*.SELECT class(MDSNTB) id(user1) access(READ)
```

```
Permit DB2P.AW.TAB*.SELECT class(MDSNTB) id(user2) access(READ)
```

```
Permit DB2P.AW.TAB*.SELECT class(MDSNTB) id(user3) access(READ)
```

```
RDEF MDSNTB DB2P.AW.TABA.SELECT UACC(NONE)
```

```
Permit DB2P.AW.TABA.SELECT class(MDSNTB) id(user1) access(READ)
```



Examples of other classes

- **DSNADM (Administrator rights)**

```
RDEF DSNADM DB8G.SYSADM UACC(NONE)  
  PERMIT DB8G.SYSADM CLASS(DSNADM) ID(CPKST) ACCESS(READ)  
  ...  
  PERMIT DB8G.TESTDB.DBADM CLASS(DSNADM) ID(CPLDB) ACCESS(READ)  
  SETROPTS RACLIST(DSNADM) REFRESH
```

- **MDSNDB (Database privileges)**

```
RDEF MDSNDB DB8G.TESTDB.* UACC(NONE)  
  PERMIT DB8G.TESTDB.LOAD CLASS(MDSNDB) ID(CPACA) ACCESS(READ)  
  PERMIT DB8G.TESTDB.DISPLAY CLASS(MDSNDB) ID(CPACA) ACCESS(READ)  
  ...  
  SETROPTS RACLIST(MDSNDB) REFRESH
```

- **MDSNSM (System resources)**

```
RDEF MDSNSM SSID.* UACC(NONE)  
RDEF MDSNSM SSID.DISPLAY UACC(READ) → uacc(read) = grant to public
```



Examples of other classes

- Packages/collections can be protected through specific classes.
 - MDSNPK: packages
 - MDSNCL: collections
- Or they can be protected through an admin class.
 - DSNADM: *SSID.COLLID.PACKADM*



Agenda

- The need for better data security
- What does DB2 offer ?
- Why externalizing security ? Why RACF ?
- How to migrate from DB2 security to RACF security ?
 - Translate DB2 into RACF
 - Getting started with RACF
 - Fallback
 - Potential issues



How to implement RACF?

- Use RACFDB2 CLIST as inspiration
- Activate DSNX@XAC module in SDSNEXIT
- Use provided DB2 resource classes
- Define needed profiles



RACFDB2 CLIST

- Download RACFDB2 CLIST
www-03.ibm.com/servers/eserver/zseries/zos/racf/downloads/racfdb2.html
- Turns each grant/privilege into corresponding RACF control statement
- Use this output as an inspiration



Activate DSNX@XAC

- Module DSNX@XAC:
RACF access control module
- Find source in SDSNSAMP(DSNXRAC)
- Customize source and assemble into SDSNEXIT

SDSNSAMP(DSNXRAC)



```
DSNX@XAC TITLE 'RACF/DB2 External Security Module - Symbols'
          SYSSTATE ARCHLVL=1                                @L3A
*-----
* Global SET Symbols: See $SET for a description
*-----
          GBLC  &CLASSNMT,&CHAROPT,&CLASSOPT,&ERROROPT      @09C
          GBLA  &PCELLCT,&SCELLCT,&XAPLDBCK
&CLASSOPT  SETC  '2'      1 - Use Classification Model I
.*                                     (One set of classes for EACH subsys)
.*                                     2 - Use Classification Model II
.*                                     (One set of classes for ALL subsys)
&CLASSNMT  SETC  'DSN'   DB2 Subsystem Name (Up to 4 chars)
&CHAROPT   SETC  '1'     One character suffix (0-9, #, @ or $)
&PCELLCT   SETA  50      Primary Cell Count
&SCELLCT   SETA  50      Secondary Cell Count
&ERROROPT  SETC  '1'     1 - Defer to DB2 authorization if      @L3C
.*                                     exit abends, sets terminating @09A
.*                                     return code(12), or sets an      @09A
.*                                     unexpected return code.          @09A
.*                                     2 - Terminate DB2 if          @09A
.*                                     exit abends, sets terminating @09A
.*                                     return code(12), or sets an      @09A
.*                                     unexpected return code.          @09A
```

DSNX@XAC options



- **&CLASSOPT** either value '1' or '2'
 - **&CLASSOPT '2'** (recommended)
multiple subsystem scope
One set of DB2 resource classes to protect multiple ssid's
Those resource classes resemble → **MDSNxx** e.g. **MDSNTB**
Profiles are prefixed with ssid ex: **ssid.creator.table.select**
 - **&CLASSOPT '1'**
single subsystem scope
One set of DB2 resource classes per ssid
You have to define resource classes Myyyyxx e.g **MPPPTB**
→ **These classes must ALL be defined (see later)**
Profiles are not prefixed with ssid ex : **creator.table.select**



DSNX@XAC options

- **&CLASSMNT 'DSN'** (recommended)
 - Only meaning for &classopt=2
 - Represents 'middle qualifier' in class name
e.g. RACF class MDSNTB
 - Can be any 1-4 characters
e.g. &classMNT 'AAAA' → MAAAAATB
this requires you defining extra classes



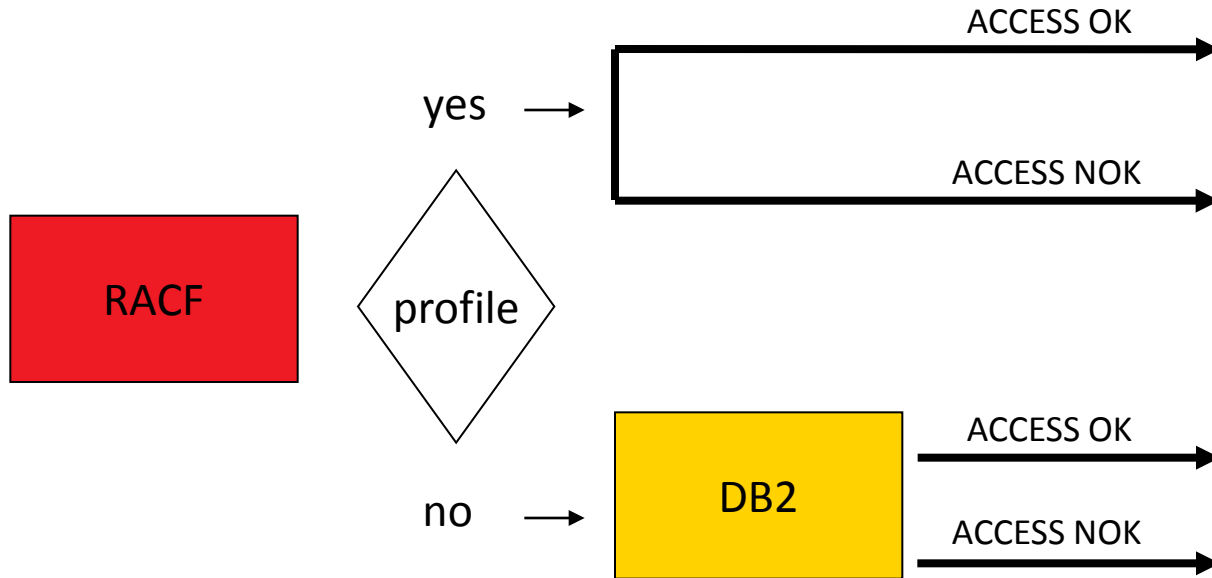
DSNX@XAC options

- **&ERROROPT values '1' or '2'**
 - &ERROROPT '1' (recommended but ...)
 - Return to DB2 security when unexpected error occurs
 - &ERROROPT '2'
 - Terminate DB2 when unexpected error occurs
- Unexpected error
 - Abend in DSNX@XAC module
 - Unexpected return code in DSNX@XAC

DSNX@XAC working



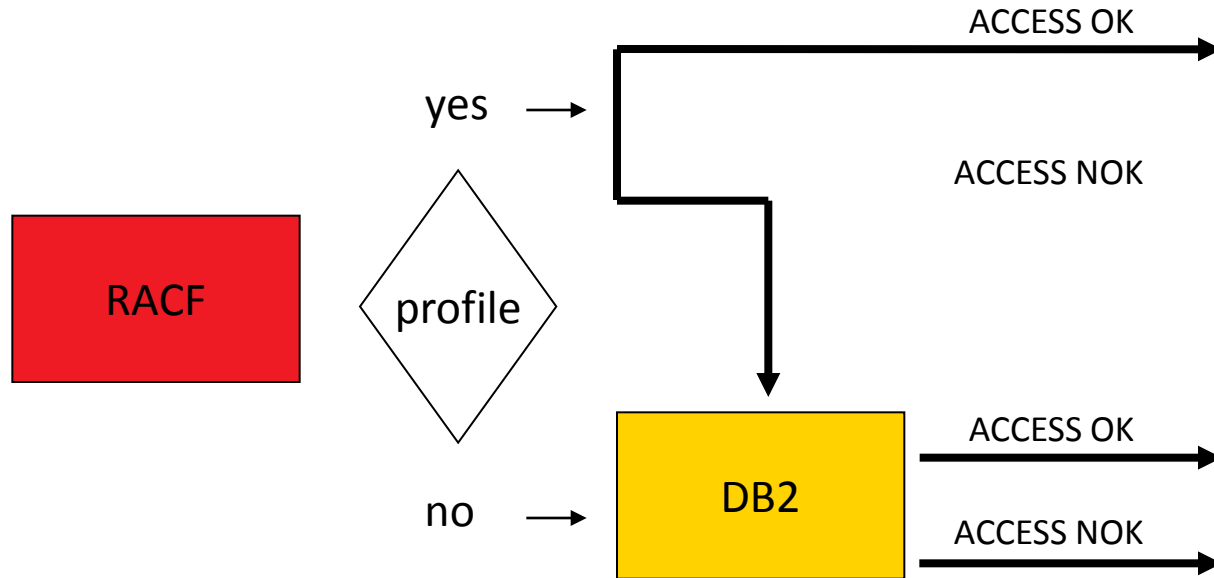
- Standard DSNX@XAC





DSNX@XAC working

- Customized DSNX@XAC



- Allows for a phased approach
- Set it up that DB2 grants access but still an ICH4081 is issued

Phases



- **Phase 0:**
CLEAN UP YOUR authorisations in DB2 first !
 - **Remove individual grants by groups**
 - **Review if access is still needed**
- **Phase I:**
Migrate all profiles to RACF
Use customized exit
Add 'forgotten' profiles when needed
New definitions only in RACF
- **Phase II:**
Switch customized exit and use standard exit
Keep catalog definitions for reference
- **Phase III:**
Clean up catalog (**optional/dangerous/not needed**)



Activate the classes

Before you can use a class you need to activate it.

```
//S01PWD EXEC PGM=IKJEFT01,DYNAMNBR=20
```

```
//SYSUDUMP DD SYSOUT=*
```

```
//SYSPRINT DD SYSOUT=*
```

```
//SYSTSPRT DD SYSOUT=*
```

```
//SYSTSIN DD *
```

```
SETROPTS CLASSACT(DSNADM)
```

```
SETROPTS CLASSACT(MDSNTB) GENERIC(MDSNTB)
```

```
SETROPTS RACLIST(DSNADM) REFRESH
```

```
SETROPTS RACLIST(MDSNTB) REFRESH
```

Stop and Start DB2



Activate the classes

- DBM1 printout

```
IEF695I START DB8GDBM1 WITH JOBNAME DB8GDBM1 IS ASSIGNED TO USER START2 ,
$HASP373 DB8GDBM1 STARTED
IEF403I DB8GDBM1 - STARTED - TIME=11.08.49
IRR908I RACF/DB2 EXTERNAL SECURITY MODULE FOR DB2 SUBSYSTEM DB8G HAS 329
      A MODULE VERSION OF PK08127 AND A MODULE LENGTH OF 00005C00.
IRR909I RACF/DB2 EXTERNAL SECURITY MODULE FOR DB2 SUBSYSTEM DB8G 330
      IS USING OPTIONS: &CLASSOPT=2
                        &CLASSNMT=DSN
                        &CHAROPT=1
                        &ERROROPT=1
                        &PCCELLCT=50
                        &SCCELLCT=50
IRR910I RACF/DB2 EXTERNAL SECURITY MODULE FOR DB2 SUBSYSTEM DB8G 331
      INITIATED RACLIST FOR CLASSES:
      MDSNDB   MDSNPK   MDSNPN   MDSNBP   MDSNCL
      MDSNTS   MDSNSG   MDSNTB   MDSNSM   MDSNSC
      MDSNUT   MDSNUF   MDSNSP   MDSNJR   MDSNSQ
      DSNADM
IRR911I RACF/DB2 EXTERNAL SECURITY MODULE FOR DB2 SUBSYSTEM DB8G 332
      SUCCESSFULLY RACLISTED CLASSES:
      MDSNTB   DSNADM
```



ICH408i message

```
ICH408I USER(CPLDB ) GROUP(CP ) NAME(CP - DEBEUF LAURENT )  
      DB8G.CPKST.EMPLOYEE.SELECT CL(MDSNTB )  
      INSUFFICIENT ACCESS AUTHORITY  
      ACCESS INTENT(READ ) ACCESS ALLOWED(NONE )
```

- Are accompanied by SQLCODE -551
- Best keep track, scrolling SYSLOG
- In a migration these profiles, indicate potentially needed profiles.



How to start a migration?

- Two possibilities :
 - Start from your theoretic rules
 - Reinforce your rules
 - Might not be the same as before
 - Only to be used in phased approached
 - Start from reality (DB2 catalog as input)
 - Everything works as before
 - Difficult to complete
 - Difficult to manage/work generic
- **FIRST STEP :always ENSURE the RACF dataset is big enough run a test, extrapolate**



How to start a migration?

— Start from reality (DB2 catalog as input)

1. start counting per qualifier, is someone a super user ?

Example :

Owner PRD1 owns 10 tables, is there someone who has select on all 10 tables?

If so create profile SSID.PRD1.*.SELECT UACC(NONE)

What if PUBLIC has select on all 10?

2 options :

1. RDEF UACC(read)

2. RDEF UACC(none)

Permit..... ID(*)

What do you do if someone has select on 9 tables ?

Is it forgotten? Is it wanted ?



How to start a migration?

- Start from reality (DB2 catalog as input)

2. Make sure a super user has access to more specific profiles

Example :

Owner PRD1 owns 10 tables, **user1** has select on all 10 tables? But user2 only on PRD1.TB1

Profiles:

```
RDEF MDSNTB SSID.PRD1.*.SELECT UACC(NONE)
```

```
RDEF MDSNTB SSID.PRD1.TB1.SELECT UACC(NONE)
```

Permits:

```
Permit SSID.PRD1.*.SELECT cl(mdsntb) id(user1)....
```

```
Permit SSID.PRD1.TB1.SELECT cl(mdsntb) id(user2)....
```

```
Permit SSID.PRD1.TB1.SELECT cl(mdsntb) id(user1)....
```




How to start a migration?

- Start from reality (DB2 catalog as input)

3. Do you have special RACF needs ?

talk to your SECADM

Example :

Are there restricted users ?

restricted user only has access to if authority is explicitly given !

User3 is a restricted user & TB1 has select to public

Profiles:

RDEF MDSNTB SSID.PRD1.TB1.SELECT UACC(READ)

Permits:

Permit SSID.PRD1.TB1.SELECT cl(mdsntb) id(**user3**)....



How to start a migration?

DO you need to follow your special users ?

need to activate

RDEF *class profile uacc* AUDIT(ALL)

Example :

RDEF DSNADM DB2P.SYSADM UACC(NONE) AUDIT(ALL)



Agenda

- The need for better data security
- What does DB2 offer ?
- Why externalizing security ? Why RACF ?
- How to migrate from DB2 security to RACF security ?
 - Translate DB2 into RACF
 - Getting started with RACF
 - Fallback
 - Potential issues



Fallback 3 methods

1. Proactive approach

- When generating your RDEF statements, create a second file containing all corresponding RDEL commands

2. Reactive approach

- If RACF information is loaded into DB2 tables, use the information to generate RDEL statements
- If RACF information is NOT loaded into DB2 tables, use the RACF dataset to unload (next slides)

3. Very bad reactive approach

- Deactivate RACF exit → requires stop start of DB2



Fallback (from RACF dataset)

- Step 1: Unload “SYS1.RACF.DS01”

```
//UNLOAD EXEC PGM=IRRDBU00,PARM=NOLOCKINPUT
//SYSPRINT DD SYSOUT=*
//INDD1 DD DISP=SHR,DSN=SYS1.RACF.DS01
//OUTDD DD DSN=myfile
// RECFM=VB,LRECL=3100
```

- Step2

REXX to screen output unload and generate

RDEL for all profiles defined on certain date, class, ssid,...

And execute all RDEL

- Step 3

Correct errors in SQL script and begin all over



Agenda

- The need for better data security
- What does DB2 offer ?
- Why externalizing security ? Why RACF ?
- How to migrate from DB2 security to RACF security ?
 - Translate DB2 into RACF
 - Getting started with RACF
 - Fallback
 - Potential issues



Some potential issues

**RACF Access Control Module Guide (SC18-7433-02)
Chapter 10 : special considerations**

READ IT !!



Some examples

- **CREATETMTAB privilege**

DB2: DBMAINT, DBCTRL, and DBADM are sufficient

RACF: *CREATETMTAB, CREATETAB, SYSCTRL, SYSADM*

- **PUBLIC* not supported**

Public at all locations is not supported by RACF

- **REVOKE**

doesn't invalidate plans, drop views etc

- ***Bindagent (see next slides)***





Bindagent problem

- In DB2

- Bindagent can bind for another owner
- Authorization owner is used for bind
 - Select
 - Insert/update/delete

- In RACF

- Bindagent needed to bind for another owner
- Authorization BINDER used for bind
- Binder must have all sel/ins/upd/del needed



Bindagent solution

Stored procedure **ADMIN_COMMAND_DSN**

- Version 9 feature
- Version 8 retro fitted (UK32059)
- Can execute **BIND/REBIND/FREE**
- Returns output to caller
- To solve bindagent use **SECURITY DEFINER**
- Definer technical user with all needed authorizations e.g. DB2BNDR
- Grant execute stored procedure to developers



Some quotes :

- The way to be safe, is never to feel secure
(Czechoslovakian proverb)
- The user's going to pick dancing pigs over security every time
(Bruce Schneier)
- Precaution is better than cure
(Edward Coke)
- Better to be safe than sorry
(my mom)

Thank You

Kurt.Struyf@infocura.be

www.infocura.be