Everything You've Ever Wanted To Know About RACLIST and WAY, WAY More!

Ross Cooper, CISSP
rdc@us.ibm.com
IBM z/OS Security Server

November 2020
Session 4AP

GSE UK Conference 2020 Charity

- The GSE UK Region team hope that you find this presentation and others that follow useful and help to expand your knowledge of z Systems.
- Please consider showing your appreciation by kindly donating a small sum to our charity this year, NHS Charities Together. Follow the link below or scan the QR Code:

http://uk.virginmoneygiving.com/GuideShareEuropeUKRegion

NHS CHARITIES TOGETHER

Authentication Performance

How many authorization requests
are processed on your z/OS systems every day?

How can you improve the performance of those requests?

RACF Authorization Overview

Resource Managers:

- Resource managers (like DB2, CICS, IMS…) can use RACF to protect their resources.
- Resources are protected by profiles created by security administrators and organized into classes in the RACF database where users and groups can be granted access to them.
- When a user attempts to access a resource, the resource manager calls RACF to query if the user has the required level of access to perform the requested action on the resource

SAF/RACF Authorization APIs:

RACROUTE REQUEST=AUTH
RACROUTE REQUEST=FASTAUTH

- These APIs attempt to match the **Resource Name** to a covering **RACF Profile Name**.
- First RACF looks for a fully qualified discrete profile, and if not found searches for a generic profile (if allowed for the class).

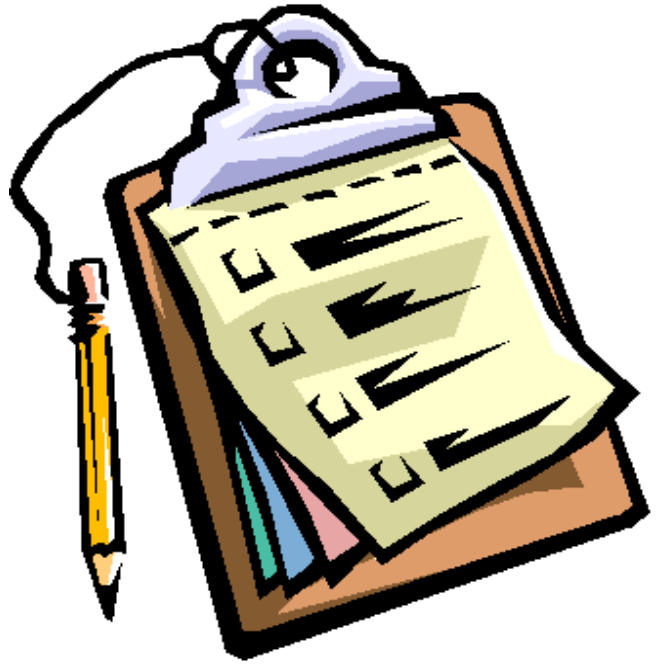
RACF Authorization Performance

- Authorization APIs can result in multiple I/Os to the RACF database which can have a significant negative impact on application performance.

- There are several features which can improve RACF Authorization performance by reducing I/O to the RACF database.

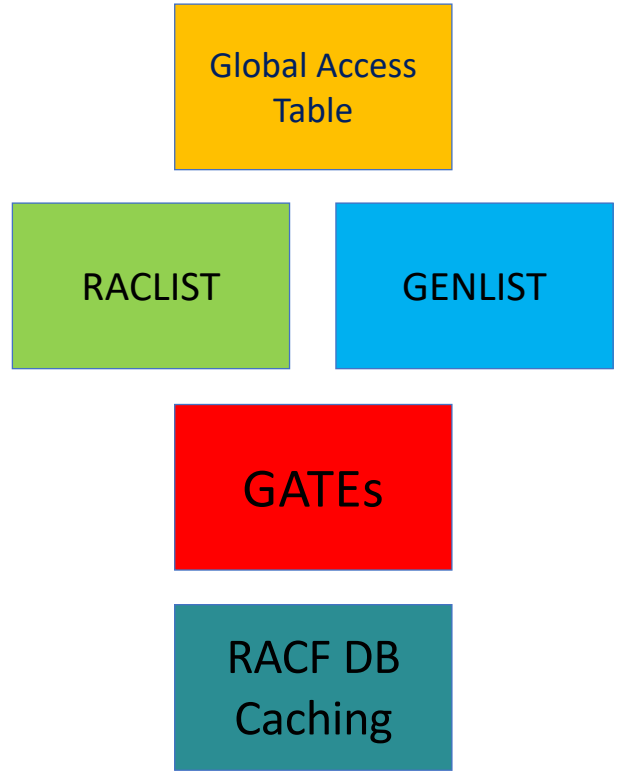
All about RACLIST (and friends)

- **RACF** Authorization Processing and In-storage RACF Profiles
- **RACLIST:**
  - RACLIST Refresh
  - Member Grouping profiles
  - CDT - Defining a RACLIST Class
  - RACGLIST
  - Sharing the RACF DB and SYSPLEX
  - Tidbits
- **Global Access Checking**
- **GENLIST**
- **GATEs**
- **RACF DB Caching** - Resident Data Blocks / Data Sharing Mode
- **Review / Questions**
- **Reference**



RACF Profiles in Memory - Overview

- **Global Access Checking** - Sets minimum level of access to a resource for all users
  - Mainly for common resources that ALL* users can access
  - **Entries:** Discrete and Generic General resources and Dataset resources
  - **Location:** Common storage
- **RACLIST** – In memory profile lists
  - Most common method for caching general resource profiles
  - **Profiles:** Discrete and Generic General resource class profiles (not dataset)
  - **Location:** Application address space or data space
- **GENLIST** – In memory generic profiles lists
  - Mainly for VM classes
  - **Profiles:** Generic general resource profiles
  - **Location:** Common storage
- **GATEs** – (Generic Anchor Table Entries) – In memory generic profile lists
  - Generic profiles not RACLISTed or GENLISTed
  - **Profiles:** Generic general resource class profiles and generic Dataset profiles
  - **Profile Location:** User address space – 64 bit
- **RACF DB Caching:**
  - **RDB** – Resident Data Blocks – In memory blocks of the RACF DB
  - **Data Sharing Mode** – In memory (coupling facility) blocks of the RACF DB



SETROPTS LIST

SETROPTS LIST command reports on which classes are in use by the different in-storage options:
GENLIST, Global Access Table, RACLIST (SETROPTS and GLOBAL=YES)

```
SETROPTS LIST
…
GENLIST CLASSES = NONE
GLOBAL CHECKING CLASSES = VMMDISK
SETR RACLIST CLASSES = ACCTNUM DASDVOL
GLOBAL=YES RACLIST ONLY = JCICSJCT TCICSTRN
…
```

RACLIST

RACLIST – Improving Authorization Performance

RACLIST is a feature which can improve RACF Authorization performance by reducing I/O to the RACF database. When a class is RACLISTed, its profiles are loaded into memory for fast reference during authorization processing.

Different ways to initiate RACLIST:

1. **Application Initiated -** RACROUTE REQUEST=LIST
   Initiated by resource manager applications for classes for which they perform authorization checks.
   Can be local or global RACLIST.
2. **Administrator Initiated -** SETROPTS RACLIST(classname)
   Initiated by RACF Administrator. Will also cause RACLIST during IPL.
   Global RACLIST only.

Local RACLIST vs Global RACLIST:

1. **Local RACLIST -** RACROUTE REQUEST=LIST GLOBAL=NO
   Class profiles loaded into the application address space. May require large region size.
   Application must delete and re-call REQUEST=LIST to pick up profile changes.
2. **Global RACLIST –** RACROUTE REQUEST=LIST GLOBAL=YES and SETROPTS RACLIST(classname)
   Class profiles loaded into a data space where multiple applications can share them.
   Changes can be refreshed with SETROPTS RACLIST(classname) REFRESH

RACLIST Refresh

Once a class is RACLISTed, any profile changes will not take effect until the in-memory profiles are rebuilt via RACLIST Refresh.

Warning Messages:

When a profile is updated (RALTER) or created (RDEFINE) when class is RACLISTed:

```
ICH10006I RACLISTED PROFILES FOR classname WILL NOT REFLECT THE
          ADDITION(S) UNTIL A SETROPTS REFRESH IS ISSUED.
```

When a profile is listed (RLIST) when class is RACLISTed but not yet REFRESHed:

```
ICH13007I One or more requested profiles for classname class are
          defined in the database, but are not listed. RACLIST
          REFRESH is required.
```

RACLIST Refresh – Operation

SETR RACLIST(classname) REFRESH:

1. Read the RACF DB to rebuild the RACLIST profiles in local memory (member grouping profiles merged)
2. Stores RACLISTed profiles in a new Data Space
3. Updates the in-memory CDT (Class Descriptor Table) to point to the new Data Space
4. Deletes the old Data Space

Serialization:

- RACLIST Refresh is serialized against other SETROPTS commands, but not serialized with RACROUTE REQ=AUTH / FASTAUTH.
- When Data Space is deleted while RACROUTE is referencing it:
  - RACROUTE will get a recoverable (and UNSEEN) 0E0 ABENDs, and then reread the Data Space reference from the in-memory CDT.

RACLIST Refresh – Considerations - VLF Purge

RACF can use VLF (Virtual Lookaside Facility) to store RACF security contexts (ACEEs) for improved authentication performance.

RACLIST REFRESH and **NORACLIST** of certain classes causes **ALL** ACEEs to be purged from VLF.

- APPCPORT
- APPL
- CONSOLE
- FACILITY (only when SETROPTS MLS is in effect)
- GTERMINL
- JESINPUT
- MFADEF
- SECLABEL
- SERVAUTH
- TERMINAL

Guideline:

- Minimize RACLIST REFRESH commands to avoid impact on RACF authentication performance.

RACLIST - Considerations - Grouping Class Profiles

Grouping Class Profiles:

- Similar to generic profiles - Enables protection of multiple resources with one profile
- The resources need not have similar names

Example:

The following command protects three terminals that have unlike names, M01RF267, M03RF168, and M04GG148:

```
RDEFINE GTERMINL DEPT35 UACC(NONE) ADDMEM(M01RF267 M03RF168 M04GG148)
```

RACLIST and Grouping Class Profiles:

- Have a related member class which must be RACLISTed.
- For example, the TERMINAL class must be RACLISTed, not the GTERMINL class.
- Depending on the class, RACLISTing is accomplished using the SETROPTS command or RACROUTE REQUEST=LIST.
  - For classes other than the CICS1 and IMS-related classes, you must also SETROPTS RACLIST the member class.
  - For example, for terminals, issue the following command

```
SETROPTS CLASSACT(TERMINAL) RACLIST(TERMINAL)
```

RACLIST and Installation Defined Classes

Installations can define their own RACF classes via the Static CDT or Dynamic CDT.
Installation defined classes can be defined to allow or require RACLIST.

Static CDT – ICHRRCDX - ICHERCDE:

RACLIST=ALLOWED | DISALLOWED

ALLOWED - Means it can be RACLISTed by any means.

DISALLOWED - (default) means ONLY via RACROUTE. (i.e. Programs keep total control.)

RACLREQ=YES | NO

Dynamic CDT:

CDTINFO (RACLIST(ALLOWED | DISALLOWED | REQUIRED))

RACLIST Required & Disallowed

RACLIST Required:

- Classes can be defined to be RACLIST REQUIRED
- Indicates that the application requires RACLIST for various reasons:
  - Performance boost from an AUTH (eg, OPERCMDS / CONSOLE)
  - Only using FASTAUTH without GLOBAL=YES (such as Health Checker)
  - Uses a grouping profiles (Grouping profile merging will not occur without RACLIST)

- **Message - Activate a class which is RACLIST required:**
  ICH14040I WARNING! You must RACLIST class class-name before authorization checking can occur.

RACLIST Disallowed:

- Classes can be defined to be RACLIST DISALLOWED
- SETROPTS RACLIST can not be issued for the class.

- **Message - Attempt to SETROPTS RACLIST(class-name) for RACLIST=DISALLOWED:**
  ICH14027I RACLIST OF CLASS class-name NOT ALLOWED BY THE CLASS DESCRIPTOR TABLE. OPERAND IGNORED.

Which Classes should be RACLISTed?

When a class is defined as RACLIST REQUIRED:

- RACROUTE AUTH or FASTAUTH calls will fail if RACLIST REQUIRED class is not RACLISTed.
- A class can be defined as RACLIST DISALLOWED and REQUIRED.
  - Application must perform RACROUTE REQUEST=LIST and use FASTAUTH. Some DB2 classes are defined this way.
  - Administrator can still use RACLIST REFRESH, if RACLIST is GLOBAL=YES

When a class is defined as RACLIST ALLOWED, but not REQUIRED:

- The administrator must decide if the class should be SETROPTS RACLISTed.
- Application or RACF may document that the class is RACLIST recommended.
- Before you use RACLIST, consider how frequently the class is referenced, the number of profiles in the class, and the amount of storage that would be required to hold the profiles.

The Cost of RACLIST

The action of creating or refreshing the RACLIST dataspace can be expensive.

- I/O to RACF database (Read all profiles into storage)
- SETROPTS ENQs.

Especially for:

- Large numbers of profiles.
- Large member / grouping classes
- Profiles with large access lists

RACGLIST can improve RACLIST performance. Both during IPL and in SYSPLEX communications.

RACGLIST – Improving RACLIST Performance

RACGLIST main benefits:

1. SYSPLEX Single system image
2. RACLIST performance improvements

Creating the RACLIST dataspace can take significant computing resources. The RACGLIST class can be used to improve performance when creating the RACLIST dataspace.

When a class is defined for RACGLIST processing, the results of in-storage profiles RACLISTed to a data space are saved in RACGLIST profiles. These RACGLIST profiles can be used to rebuilt the RACLIST data space which can reduce I/O and improve performance during RACLIST processing.

Improved RACLIST performance for classes with:

- Lots of profiles
- Large access lists
- Lots of member / grouping profiles

RACGLIST – Overview

To enable RACGLIST - Activate the RACGLIST class and define a profile with the name of the CLASS:

SETR CLASSACT(RACGLIST)

RDEFINE RACGLIST *classname*

Actions that create or replace RACGLIST profiles:

- SETROPTS RACLIST
- SETROPTS RACLIST REFRESH
- RACLIST during IPL (When RACGLIST profiles do not exist)
- RACROUTE REQUEST=LIST GLOBAL=YES (When RACGLIST profiles do not exist)

RACGLIST class profiles (Size of profile is up to ~50K):

classname_00001

classname_00002

…

RACLIST performance improvements with RACGLIST:

- RACLIST during IPL
- REQUEST=LIST GLOBAL=YES (When RACLIST data space does not yet exist after IPL)
- RACLIST on peer systems in SYSPLEX communications

RACGLIST - Considerations

Considerations:

- RACGLIST uses more storage in the RACF database
- RACGLIST may not improve performance for classes without a large number of profiles
- RACLIST during IPL will build from RACGLIST. Profile changes will not become effective until a RACLIST REFRESH is issued.

When sharing the RACF database:

- All systems sharing must be in the same GRS complex.
- The major name SYSZRAC2 cannot be in the exclusion resource name list (RNL)

Certain classes are restricted from using RACGLIST:

CDT, GLOBAL, RACGLIST, USER, CONNECT, GROUP, and DATASET classes

Additionally, any class that is “not allowed to contain profiles”:

DIRAUTH, TEMPDSN, DIRACC, DIRSRCH, FSOBJ, FSSEC, IPCOBJ, PROCACT, PROCESS

RACLIST and SYSPLEX Communication

When enabled for SYSPLEX Communication RACF attempts to maintain a single system view of RACLISTed data.

SETROPTS RACLIST commands are propagated to each of the systems in the data sharing group:

- The coordinator (command origin) builds the new RACLIST data space
- If **RACGLIST** is configured for the class, new RACGLIST profiles are created
- Coordinator signals all peers to RACLIST via XCF messages
- Peers build RACLIST data space (use RACGLIST if available)
- Peers report back to coordinator when complete
- Coordinator signals peers to begin using the new data space and discard the old one

Serialization is used to prevent multiple concurrent or conflicting commands:

- If during SETROPTS RACLIST activity, ENQS develop, the best advice is to let them continue and they should clear. These mean that OTHER SETR activities are colliding. Best to leave alone.
- Try --NEVER-- to cancel a coordinator (or allow to time out) during this process. Be patient, especially for classes with large numbers of profiles.

Profile Statistics

The STATISTICS option permits an installation to record statistics on discrete profiles to see how their respective data sets and resources within specific resource classes are being used.

Statistics are not recorded:

- Generic profiles (including GENLIST, GATEs)
- Profiles that are loaded into storage by RACROUTE REQUEST=LIST or SETROPTS RACLIST
- Entries in the Global Access Table

RACLIST and ENF Signals

ENF signals:

- A system which allows interested applications to receive notification of ‘system events’.

RACLIST and ENF Signals:

- Some resource managers cache authentication decisions from RACF and use RACLIST ENF signals to stay informed of changes to RACF profiles.
- RACF can send a type 62 ENF signal when **SETROPTS RACLIST** command affects in-storage profiles.
  RACLIST, **RACLIST REFRESH**, or **NORACLIST**

RACLIST ENF Signals are sent for:

- Static CDT classes in ICHERCDE with SIGNAL=YES
- Dynamic CDT classes with SIGNAL(YES)

Notes:

- Products will document when ENF signals are required for particular classes.
- RACROUTE REQUEST=LIST,GLOBAL=YES does not cause an ENF signal to be issued.

RACLIST Class POSIT Considerations

RACF CDT POSIT Value:

- RACF Classes have a POSIT value that may be shared with other classes.

RACLIST and POSIT values:

- SETROPTS RACLIST commands process both the class name specified and all valid classes sharing the same POSIT value as the specified class.
    - SETROPTS RACLIST(*classname*)
    - SETROPTS RACLIST(*classname*) REFRESH
    - SETROPTS NORACLIST(*classname*)

Notes:

- When a classes sharing the same POSIT disallows RACLIST, those classes are skipped.
- SETROPTS GENLIST also processes all classes with the same POSIT
- RACROUTE REQEST=LIST GLOBAL=YES does not process classes with the same POSIT

Global Access Checking

Global Access Checking - Overview

- Global access to public resources that are accessed frequently.
- Checked early in RACF authorization processing which can eliminate search for a matching DB profile.
- Profiles are stored in common storage.
- **Entries:** Discrete and Generic General resources and Dataset resources

Global Access Checking - Configuration

Steps to enable Global Access Checking:

SETROPTS GLOBAL(classname)
RDEFINE GLOBAL classname
RALTER GLOBAL *classname* ADDMEM(*resource-name/access-level*)

Example Profile:

RALTER GLOBAL DATASET ADDMEM('SMITH.ABC'/READ)

Must refresh after changes (or IPL):

SETROPTS GLOBAL(*classname*) REFRESH

List entries in order they are searched by RACF:

RLIST GLOBAL classname

Can use discrete, generics and &RACUID / &RACGPID:

Allow users to have ALTER access to data sets that begin with their own user IDs:

RALTER GLOBAL DATASET ADDMEM('&RACUID.**'/ALTER)

Global Access Checking - Restrictions

- No Logging is performed
- No statistics
- Not checked by **RACROUTE REQUEST=FASTAUTH**:
  - FASTAUTH only uses RACLISTed profiles
  - Need a profile in the RACF DB that grants the same level of access if any applications use FASTAUTH
- If there is a more restrictive profile in the RACF DB, Global Access Table wins.
- **RESTRICTED User IDs:**
  - Do not get access via Global Access Table
- **RACROUTE REQUEST=VERIFY:**
  - Resources checked during authentication are not checked in the Global Access Table:
    APPL, TERMINAL, JESINPUT, CONSOLE, APPCPORT and SERVAUTH resources

GENLIST

GENLIST – Generic Profile List - Overview

GENLIST is similar to RACLIST:

- Can improve RACF Authorization performance by reducing I/O to the RACF database.

When GENLIST is active for a class:

- Generic profiles for that class are loaded into common storage (ECSA)
  - Not discrete profiles
- Only for general resource classes, not DATASETs.

RACF VM:

- Generally, when you do not share the RACF database with RACF on a VM system, RACLIST provides the best performance with the lowest usage of common storage.

GENLIST – Generic Profile List - Details

When does GENLIST occur?

- When you activate GENLIST processing for a class, a generic profile in that class is copied from the RACF database into common storage the first time an authorized user requests access to a resource protected by the profile.

Refreshing GENLIST:

- SETROPTS GENERIC(class-name) REFRESH
- Clears the GENLIST common storage. Will also clear GATEs for ALL users.

Considerations:

- Classes must be GENLIST allowed in the CDT.
- Mainly for VM classes. RACLIST on VM occupies common storage in RACF service machine not Data Spaces.
- RACF does not allow you to specify SETROPTS GENLIST and SETROPTS RACLIST for the same general resource class.

GATES

GATEs - Generic Anchor Table Entries

GATEs are Generic Profiles Stored in Memory:

- Stored in 64-bit memory in users address space anchored off the ACEE
- Multiple lists of generic profiles for resources that have been accessed by a user:
  - **Generic Dataset profiles:**
    - One list for each high-level qualifier accessed by the user
  - **Generic general resource class profiles:**
    - One list for each CDT KEYQUAL(x) qualifier accessed by the user
- Not created for classes which are **RACLIST**ed or **GENLIST**ed

GATEs - Generic Anchor Table Entries - Configuration

RACF keeps four lists per user by default:

- The SET GENERICANCHOR() RACF console command can set a larger number of lists (up to 99)
- You can set a system-wide limit, and limits for specific jobs

```
SET GENERICANCHOR(SYSTEM COUNT(8))
SET GENERICANCHOR(JOBNAME(JOBXYZ) COUNT(10))
```

- When a new generic profile list is built and limit has been reached:
  - The oldest list is deleted and replaced with the new list
- Users accessing many different Dataset HLQs or different general resources may experience thrashing as the lists are constantly changing.
  - Consider increasing the GENERICANCHOR limit

GATEs - Generic Anchor Table Entries

Notes:

- GATEs are not created for **RACLIST** classes or **GENLIST** classes or profiles from Global Access Table
- GATEs contain only contains generic profiles. Still must read RACF DB to search for a discrete profile.
- When generic profiles are added, changed or deleted, the GATEs must be refreshed before changes become effective:
  - User can log off and logon to clear their GATEs
  - **LISTDSD DA('HLQ.something') GENERIC** – Will Refresh all Dataset profiles for the HLQ
  - **SETR GENERIC(*classname*) REFRESH** – Clears GATEs (and GENLIST) for ALL users

KEYQUAL:

- Property defined in CDT for a general resource class
- KEYQUAL is a way to limit the range / size of the generic general resource profile lists
- Specifies the number of matching qualifiers that RACF uses when loading generic profile names to satisfy an authorization request if a discrete profile does not exist for the resource.
- If the value of KEYQUAL is 0, profile names for the entire class are loaded into the GATEs.

GATEs - Generic Anchor Table Entries - Pubs

Documented in System Programmer's Guide

Chapter 2. Performance considerations

Section – Using generic profiles

Using generic profiles

In each address space, RACF keeps lists of generic profiles that have been referenced. Each list comprises one DATASET high-level qualifier, or one general resource class based on the value of KEYQUAL in the class descriptor table (CDT) entry for that class (assuming the class is not RACLISTed in some way).

RACF DB Caching

RACF DB Caching - Resident Data Blocks (RDB)

- The RACF DB can be cached in memory (ECSA) in buffers called Resident Data Blocks.
- When RACF reads and writes from the RACF Database, each block is read into a 4k RDB.
- Most recently used blocks are kept in memory and older blocks are cycled out.

- **RACF Data Base Block types:**
  - ICB – Inventory Control Block – RACF DB Settings
  - Templates – RACF Data Base Field Templates
  - BAM blocks – Tracks block usage
  - Index blocks – Used to locate Profiles
  - Profiles – RACF profiles (profiles can span multiple blocks)
- **Read into storage at IPL:** ICB, Templates
- **Read into RDB:** BAM, Index and Profiles

RACF DB Caching - Resident Data Blocks (RDB)

- **Installations can configure how many RDBs are kept in memory:**
  - ICHRDSNT – RACF Data Set Name Table
    - Describes the data sets in the RACF database to RACF
  - 1-byte resident data-block count field
  - Maximum number of blocks allowed (and recommended) is 255

- **Recommendation:**
  - For best performance, specify as large a number of buffers as you can afford, preferably 255.

Review

Review & Recommendations

There are several features which can improve RACF Authorization performance by reducing I/O to the RACF database:

- **Global Access Checking:**
  - Create GLOBAL entries for public resources that are accessed frequently.
- **RACLIST:**
  - Use for classes when applications which are RACLIST allowed.
  - Consider RACGLIST for classes with lots of profiles, many member grouping profiles and large access lists.
- **GENLIST**:
  - Mainly for VM classes or sharing DB with VM.
  - If not VM, use RACLIST if available for class.
- **GATEs:**
  - Consider increasing number of gates. Uses more user storage, but limits Database I/O.
- **Resident Data Blocks:**
  - Larger size uses more memory, but limits Database I/O.
- **Data Sharing Mode:**
  - Larger size uses more memory, but limits Database I/O.

RACF Profile Search

Resource manager calls RACF to check for authority:

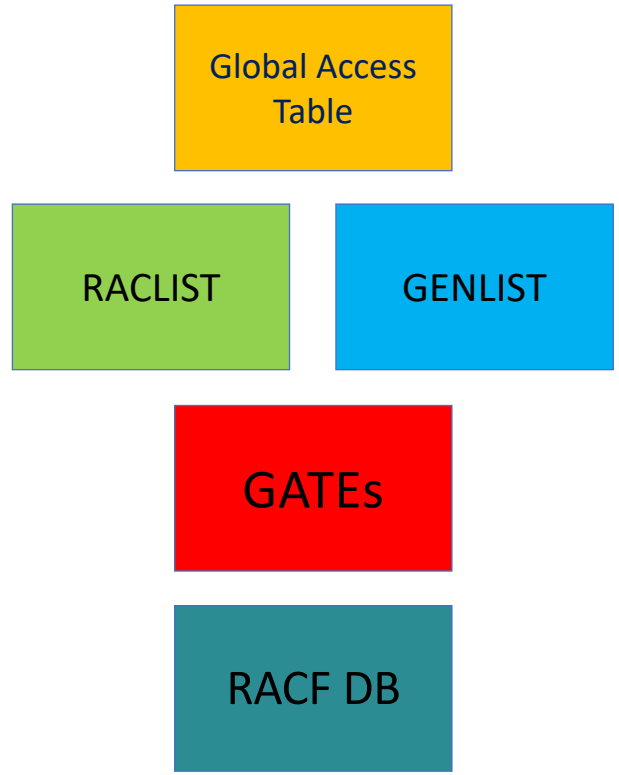
Does User ID: **JOE01** have at least **UPDATE** access to resource: **ABC.XYZ** in class: **MYCLASS**?

REQUEST=FASTAUTH searches for a matching profile:

- Search for profile in **RACLIST** storage - Done

REQUEST=AUTH searches for a matching profile:

- If class is **SETROPTS GLOBAL:**
  - Search for covering entry in **Global Access Table:**
    - If entry allows access – Done
- If class is **RACLIST**ed:
  - Search for profile in **RACLIST** storage - Done
- Search for discrete profile in **RACF DB** (RDB / CF / Disk IO)
  - If discrete profile found - Done
- If discrete profile not found, continue to search for generic profile (if allowed for class)
- If class is **GENLIST**ed:
  - If **GENLIST** class list not built for class:
    - Build **GENLIST** class list from **RACF DB**
  - Search for generic profile in **GENLIST** storage - Done
- Search for generic profile in **GATEs:**
  - If **GATE** not created for profile qualifiers:
    - Build **GATE** from **RACF DB**
  - Search for generic profile in **GATE** – Done



References and Publications

- **Global Access Checking:**
  - RACF Security Administrators Guide:
    Chapter 7. Protecting general resources - Setting up the global access checking table
  - RACF System Programmers Guide:
    Chapter 2. Performance considerations – Using global access checking
- **RACLIST & GENLIST:**
  - RACF Security Administrators Guide:
    Chapter 5. Specifying RACF options - SETROPTS options to activate in-storage profile processing
  - RACF System Programmers Guide:
    Chapter 2. Performance considerations - The SETROPTS command
  - RACF Command Language Reference:
    Chapter 5. RACF command syntax – SETROPTS (Set RACF options)
- **GATEs:**
  - RACF System Programmers Guide:
    Chapter 2. Performance considerations - Using generic profiles
- **Resident Data Blocks:**
  - RACF System Programmers Guide:
    Chapter 2. Performance considerations – Resident data blocks
- **Data Sharing Mode:**
  - RACF Diagnosis Guide:
    Chapter 4. Operating considerations – Shared database considerations

Please submit your session feedback!

- Do it online at http://conferences.gse.org.uk/2020/feedback/nn
- This session is 4AP

1. What is your conference registration number?

This is the three digit number on the bottom of your delegate badge

2. Was the length of this presention correct?

1 to 4 = "Too Short" 5 = "OK" 6-9 = "Too Long"

1 2 3 4 5 6 7 8 9

3. Did this presention meet your requirements?

1 to 4 = "No" 5 = "OK" 6-9 = "Yes"

1 2 3 4 5 6 7 8 9

4. Was the session content what you expected?

1 to 4 = "No" 5 = "OK" 6-9 = "Yes"

1 2 3 4 5 6 7 8 9

Reminder - GSE UK Conference 2020 Charity

- The GSE UK Region team hope that you find this presentation and others that follow useful and help to expand your knowledge of z Systems.
- Please consider showing your appreciation by kindly donating a small sum to our charity this year, NHS Charities Together. Follow the link below or scan the QR Code:

http://uk.virginmoneygiving.com/GuideShareEuropeUKRegion

NHS CHARITIES TOGETHER

Backup

RACROUTE and RACLIST

RACROUTE considerations when using SETROPTS RACLIST:

Authorized programs:

Can use FASTAUTH for profiles that are SETROPTS RACLISTed.

Unauthorized programs:

Can use FASTAUTH only for profiles brought into storage by a RACROUTE REQUEST=LIST

If your application uses RACROUTE REQUEST=LIST,GLOBAL=NO for a class, RACF uses locally RACLISTed profiles for authorization checking. You should not issue a SETROPTS RACLIST for the same class.

When an application RACLISTs a class using RACROUTE REQUEST=LIST,GLOBAL=YES, the RACLISTed profiles are stored in a data space. The data space can be shared by many applications. Applications that issue a subsequent RACROUTE REQUEST=LIST,GLOBAL=YES for the same class simply access the data space built by the first application.

When all applications have relinquished their access to the data space by issuing a RACROUTE REQUEST=LIST,ENVIR=DELETE request, the data space can be deleted by issuing a SETROPTS NORACLIST(*classname*) command.

RACLIST and sharing the RACF Database

RACF with SYSPLEX Communication:

Using the coupling facility and XCF messaging

Data Sharing Mode:

Sharing RACF database in the coupling facility

RACLIST and SYSPLEX Comm:

RACLIST when enabled for SYSPLEX Communication:

SETROPTS RACLIST commands are propagated to each of the systems in the data sharing group via XCF messages

RACLIST when not enabled for SYSPLEX Communication:

Must issue the SETROPTS RACLIST commands on all systems to have the results effective on all systems

Failures during RACLIST:

Data sharing mode:

If the command fails on any of the peer systems RACF stops processing the command and backs it out of all the member systems, including the system on which it was issued.

Non-data sharing mode:

The command can fail on a peer system without backing out of the other systems.

Data Sharing Mode

- In data sharing mode, the coupling facility is exploited to provide a large buffer for records from the RACF database, allowing a decrease in the I/O rate to the RACF database.

- RACF uses the coupling facility as a large sysplex-wide store-through cache for the RACF database that reduces contention and I/O to DASD. Serialization is done using global resource serialization instead of RESERVE/RELEASE while in data sharing mode. RACF sysplex data sharing also provides improved efficiency of invalidating resident data blocks.

- Even with a single-system sysplex, you can use the coupling facility to reduce RACF database I/O.

- This is in addition to RDBs. The existing RDBs are fed from the CF. When an RDB is read in Data Sharing Mode, the CF is also read to check if another system has updated that block. If so, the RDB will be invalidated and re-read from the CF.

End