RSH CONSULTING

RACF Performance Tuning

SHARE - August 2018

RSH Consulting - Robert S. Hansel

RSH Consulting, Inc. is an IT security professional services firm established in 1992 and dedicated to helping clients strengthen their IBM z/OS mainframe access controls by fully exploiting all the capabilities and latest innovations in RACF. RSH's services include RACF security reviews and audits, initial implementation of new controls, enhancement and remediation of existing controls, and training.

- www.rshconsulting.com
- 617-969-9050

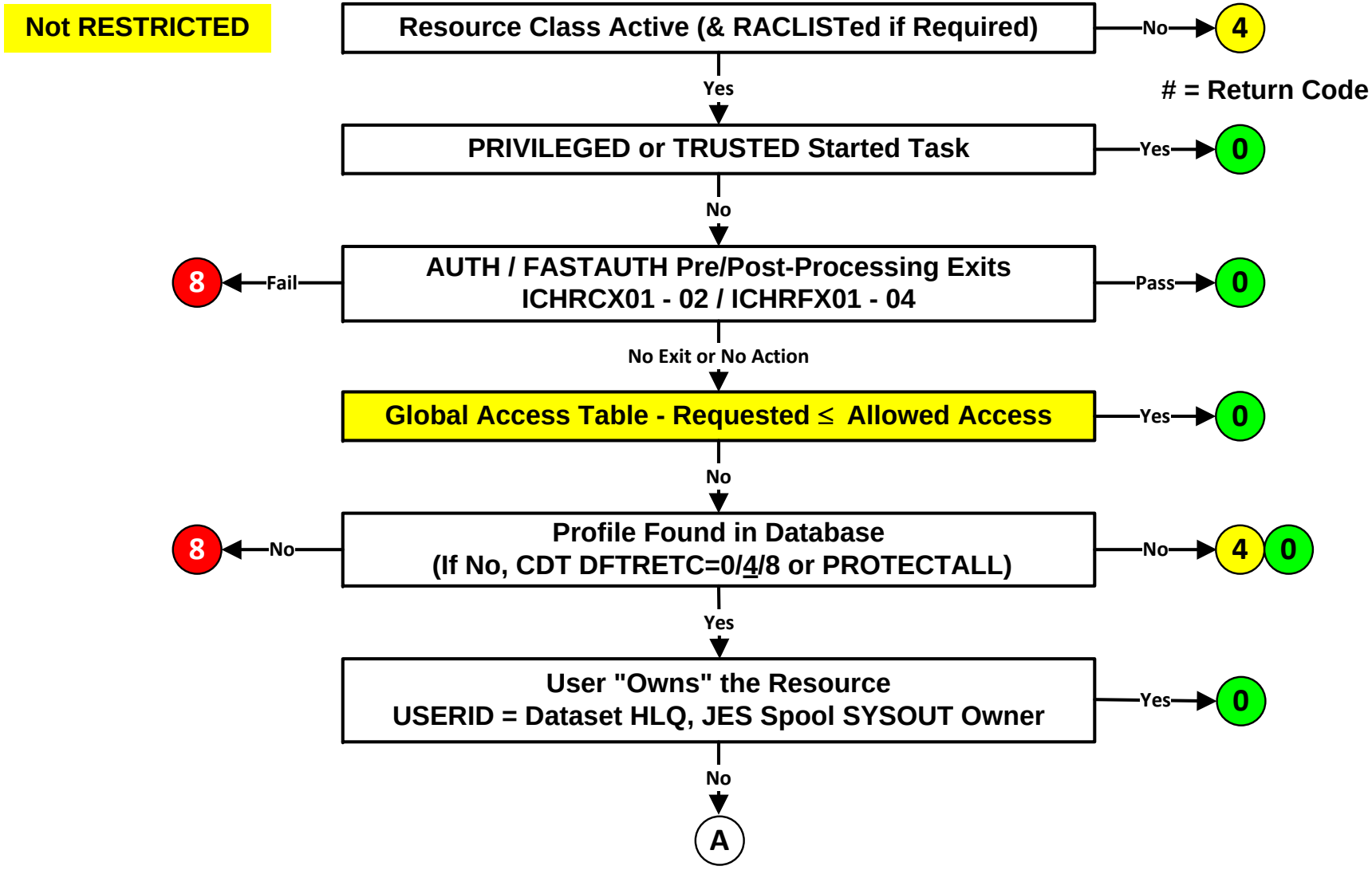
Robert S. Hansel is Lead RACF Specialist and founder of RSH Consulting, Inc. He began working with RACF in 1986 and has been a RACF administrator, manager, auditor, instructor, developer, and consultant. Mr. Hansel is especially skilled at redesigning and refining large-scale implementations of RACF using role-based access control concepts. He is a leading expert in securing z/OS Unix using RACF. Mr. Hansel has created elaborate automated tools to assist clients with RACF administration, database merging, identity management, and quality assurance.

- 617-969-8211
- R.Hansel@rshconsulting.com
- www.linkedin.com/in/roberthansel
- http://twitter.com/RSH_RACF

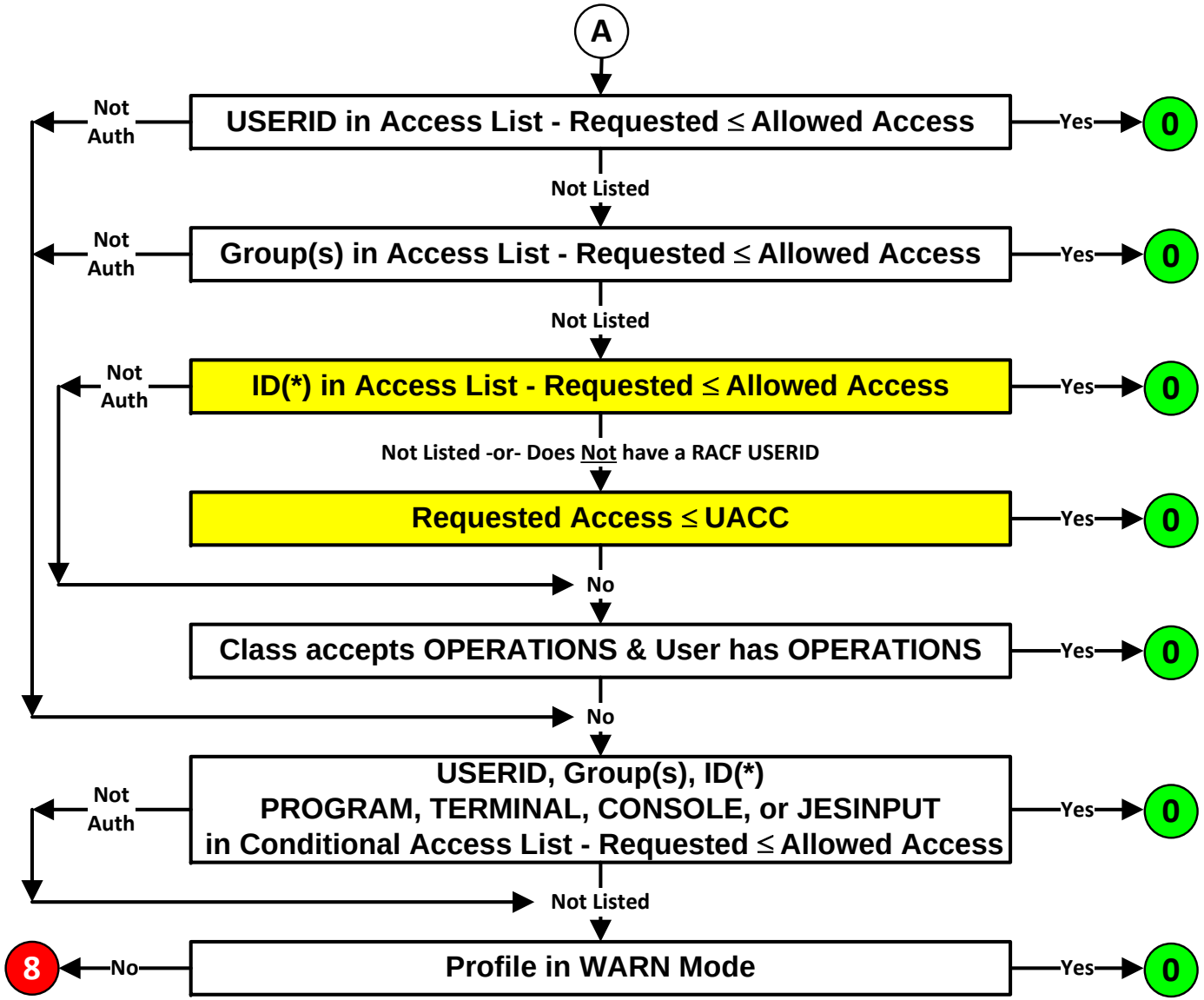
Performance Objectives

- Optimize Access Authorizations
- Expedite the Logon Process
- Minimize I/O Operations

Access Authorization Decision Logic



Access Authorization Decision Logic



RACF Authorization Decision Logic

- Deactivate unused classes (be mindful of POSITs when deactivating)
  - Resource classes, including SECDATA and SECLABEL classes
  - Global Access Table classes
- Make access list processing efficient
  - Minimize the number of entries in access lists
    - Grant end-user access via groups instead of USERIDs
    - Remove obsolete residual entries - run IRRRID00
    - Remove redundant entries (e.g., access allowed equals UACC)
  - Minimize the number of group connects per user
- Reduce reliance on OPERATIONS authority by implementing Storage Administration authorities
- Write efficient exit code
- Implement the Global Access Table

Global Access Table

- Performance enhancement tool
  - Grants immediate access to a resource without referring to its profile and without logging
  - Used to grant access to common shared resources
- GLOBAL Class
  - Profile - Class name [RDEF GLOBAL DATASET]
  - Members - resource/access [ADDMEM('CTLG.USER'/UPDATE)]
    - Resource
      - Discrete or Generic - follows generic profile rules for General Resources
      - Need not match profile(s) protecting the resource(s)
      - For datasets, if not enclosed in quotes, appends user's USERID as the first qualifier
    - Access-levels - ALTER | CONTROL | UPDATE | READ | NONE (not EXECUTE)
- Special Variables - Used in resource names
  - &RACUID Substitute with requesting user's USERID
  - &RACGPID Substitute with requesting user's current connect group

Global Access Table

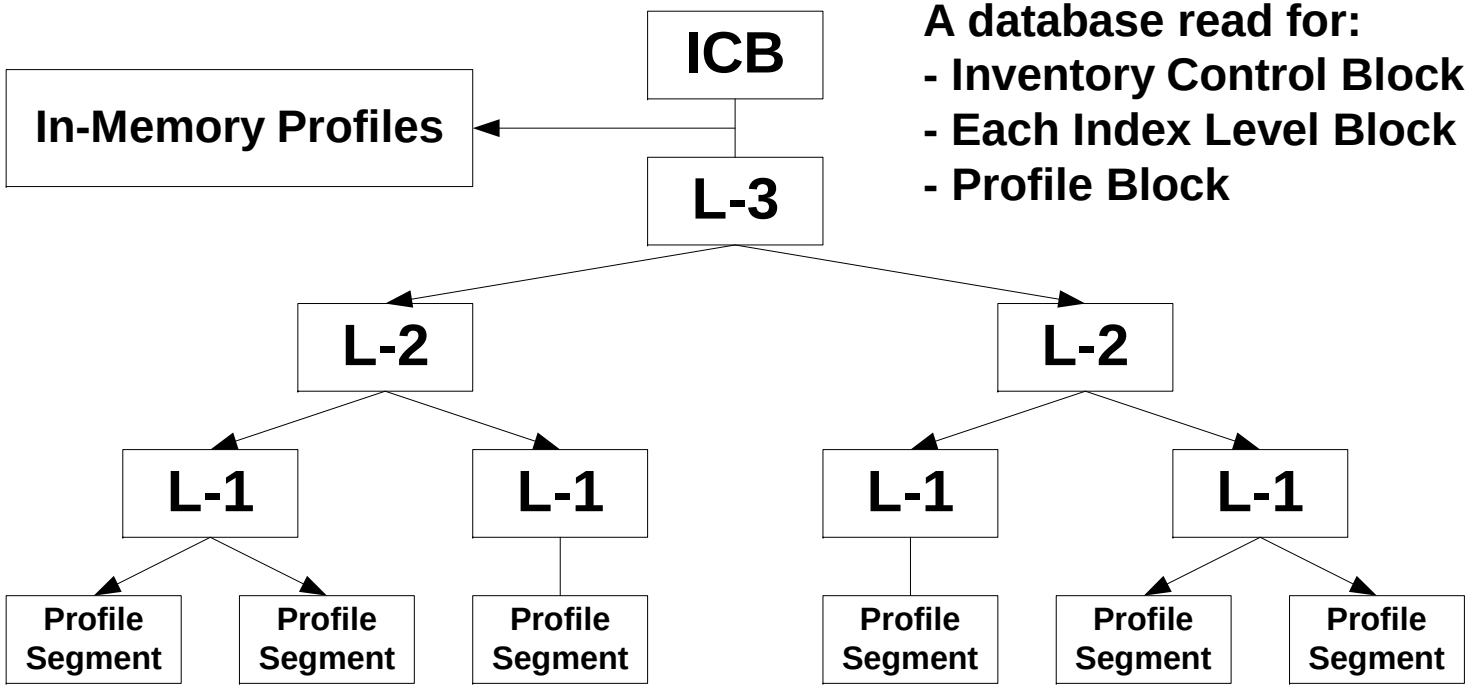
- Sample entries

DATASET	&RACUID.*.**	ALTER	
DATASET	&RACGPID.*.**	UPDATE	(avoid - unintended access)
DATASET	CATALOG.MASTER	READ	
DATASET	CATALOG.USER	UPDATE	
DATASET	ISPF.LIBRARY	READ	
DATASET	SDSF.LIBRARY	READ	
DATASET	SYS1.BRODCAST	READ	
DATASET	SYS1.HELP	READ	
DATASET	SYS1.MACLIB	READ	
DATASET	SYS1.RACF	NONE	(precludes access)
DATASET	SYS%.**	READ	(avoid - too broad)
DATASET	*.PUBLIC.**	READ	(optionally allow TSO users to share data)
DATASET	*.**.#SMSTEST	ALTER	(optional catalog/SMS testing)
FACILITY	ERBDSB.*	READ	
FACILITY	IEC.TAPERING	READ	(probably obsolete)
FACILITY	STGADMIN.ARC.ENDUSER.**	READ	
JESJOBS	SUBMIT.*.&RACUID*.&RACUID	READ	
JESJOBS	CANCEL.*.&RACUID.*	ALTER	(not needed - post RTOKEN check)
JESSPOOL	*.&RACUID.**	ALTER	
JESSPOOL	*.*.$JESNEWS.**	READ	
MQQUEUE	MQS*.ISF.USER.&RACUID.**	ALTER	(probable SDSF manual error)
OPERCMDS	MVS.CANCEL.TSU.&RACUID	UPDATE	
OPERCMDS	MVS.DISPLAY.*	READ	
OPERCMDS	MVS.MCSOPER.&RACUID	READ	
SDSF	ISFCMD.DSP.*option*.*	READ	(*option*: ACTIVE, HELD, OUTPUT)
TSOAUTH	JCL	READ	
TSOAUTH	RECOVER	READ	

Global Access Table

- Activated and managed via SETROPTS
  - SETROPTS GLOBAL(*class*) | NOGLOBAL(*class*) [REFRESH]
  - Must be refreshed if updated
- Can be used for most resource classes except …
  - Not checked in RACROUTE REQUEST=FASTAUTH processing
  - Not checked in RACROUTE REQUEST=VERIFY processing for APPL, TERMINAL, JESINPUT, CONSOLE, APPCPORT, and SERVAUTH resources
- Keep list of entries short and efficient to minimize search
- Drawbacks
  - Precludes logging (except SETROPTS AUDIT(*class*) resource defines)
  - Undermines protection if allows more access than profile UACCs

RACF Profile Retrieval



A database read for:
- Inventory Control Block (ICB)
- Each Index Level Block (L-n)
- Profile Block

- Data is written and retrieved in 4K blocks
- Individual profiles and profile segments can be greater than 4K in size and span multiple contiguous blocks, each of which requires I/O to fetch - keep profiles as small as possible

Resident Data Blocks (RDB)

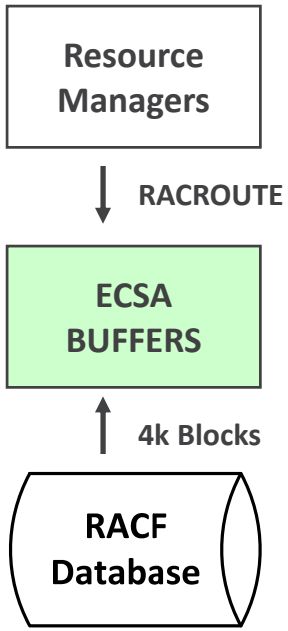
- RACF maintains buffers in Extended Common Storage Area (ECSA) to cache copies of most recently used blocks (index, BAM, and profiles)
- RACF retrieves and stores a database block in an RDB before processing it
- Frequently used blocks tend to stay in these buffers (e.g., index blocks)
- Desired number of resident blocks is specified in the Database Name Table ICHRDSNT or PARMLIB(IRRPRMxx) statements

AL1(1)	Number of databases
CL44'RACF.PRIMARY'	Primary DB name
CL44'RACF.BACKUP'	Backup DB name
AL1(100)	# of Resident Data Blocks
XL1'xx'	Flags

- Default/minimum number of blocks

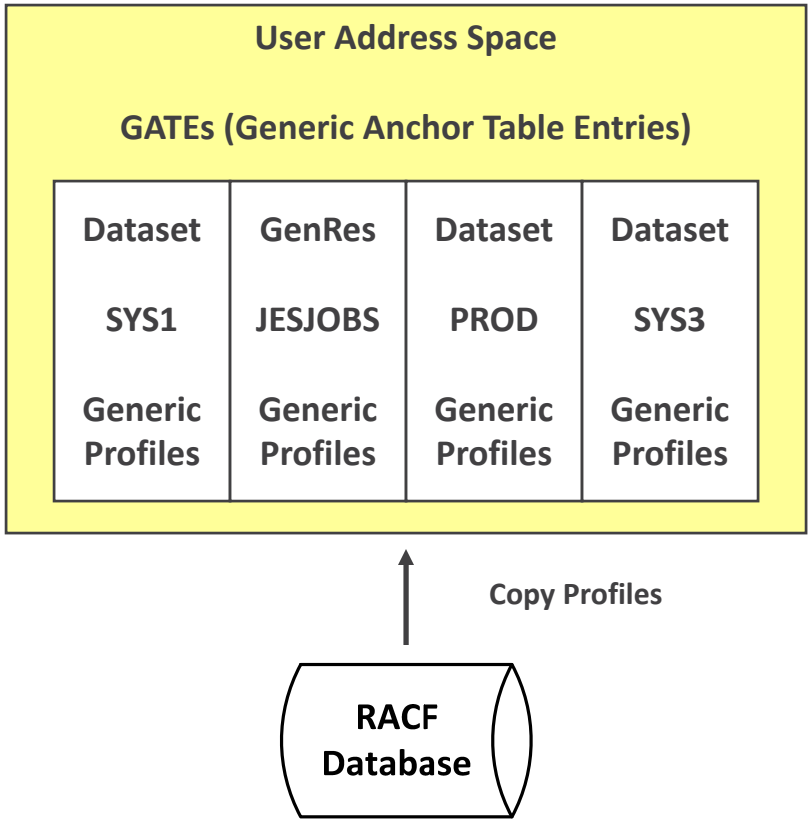
10 / 0	Non-RACF-Sysplex (none for backup database)
50 / 50	RACF-Sysplex (+ additional 20% for backup database)

- Maximum number - 255 (recommended)



Generic Profiles Stored In Memory

- Sets of generic profiles are cached in each individual user's address space memory
- Each set is comprised of generic profiles for either:
  - Dataset HLQ
  - General Resource class
- Upon first access to a resource class or HLQ, a list of all the associated generic profiles is retrieved and loaded into memory
- Individual generic profiles are retrieved as needed for authorization checking and retained in memory thereafter
- Profiles in memory are used for authorization checking - not those in the RACF database



Generic Profiles Stored In Memory

- Once all sets of generic profiles are filled, when the next new resource class or HLQ is accessed, the set with the least recently used profiles is dropped and replaced with the new one
  - Users accessing many different HLQs and/or general resources could experience thrashing (i.e. constant replacement) among the sets
- Dataset HLQs or general resources classes with many generic profiles take more I/O and CPU time to retrieve and load
- RACF can optionally keep up to 99 sets of profiles
  - Changed with the RACF operator command SET GENERICANCHOR(*option*)
  - Option can be configured for SYSTEM or JOBNAME(*jobname jobname** ...)
  - Minimum/Default is 4

Generic Profiles Stored In Memory

- Additions or changes to generic profiles requires in-memory copies to be refreshed before they become effective by one of the following methods ...
  - User must logoff and logon to renew the in-memory profiles
  - User can execute a LISTDSD GENERIC command to refresh all profiles for the HLQ

    LISTDSD DA('HLQ.anything') GENERIC
  - SETROPTS GENERIC(*class*) REFRESH - this immediately drops all in-memory profile sets for the designated class for <u>all</u> active users and requires every user to reload them upon next access

- I/O is still required for ...
  - Datasets if the RACF indicator bit is on
  - General resources to check for a discrete profile before generics are checked

- Can avoid having to retrieve and load profiles into user memory by …
  - Granting access using the Global Access Table
  - Loading profiles into memory using GENLIST and RACLIST

SETROPTS GENLIST and RACLIST

- Cause profiles to be stored in memory for rapid reference and to avoid I/O to the database
- Mutually exclusive SETROPTS options set for specific general resource classes
- Effects all classes with the same POSIT value
- GENLISTed and RACLISTed classes do not consume any of a user's in-memory generic profile sets

- GENLIST(*class*)
  - Retrieval of first Generic profile prompts retrieval and storage of a list of all Generic profiles for the class in ECSA
  - Generic profiles are individually retrieved on first reference and retained in ECSA for subsequent reference
  - I/O still required to check for discrete profile
  - Class must be defined in the CDT with GENLIST=ALLOWED
  - Refreshed with SETROPTS GENERIC(*class*) REFRESH
  - Recommendation - use with VM related classes

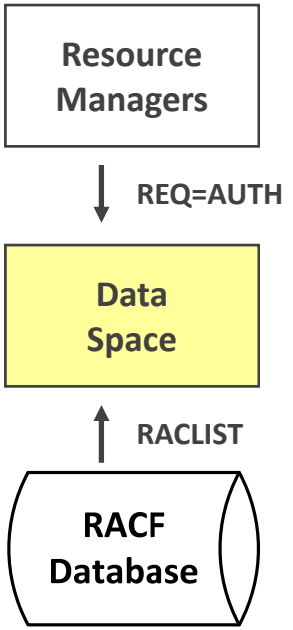
RACLIST

- All profiles for a specified class are cached in a shared dataspace
  - SETROPTS RACLIST(*class*), if RACLIST=ALLOWED in CDT
  - RACROUTE REQUEST=LIST,GLOBAL=YES by certain applications

    CICS IMS VTAM MQ DB2
  - Updated with SETROPTS RACLIST(*class*) REFRESH
  - Profile segments are not stored in memory (e.g., STDATA)
  - Required to exploit grouping classes (e.g., DASDVOL / GDASDVOL)
- CDT RACLREQ=YES - Required

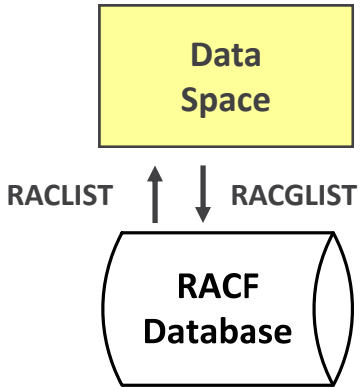
  APPCSERV APPCTP CRYPTOZ CSFKEYS CSFSERV
  DEVICES DIGTCIRT DIGTNMAP FIELD FSACCESS
  FSEXEC IDIDMAP NODES OPERCMDS PROPCNTL PSFMPL
  PTKTDATA RACFHC RACFVARS RDATALIB SDSF SECLABEL
  SERVAUTH STARTED SYSAUTO SYSMVIEW UNIXPRIV VTAMAPPL
- RACLIST recommendations:

  APPL CDT CONSOLE DASDVOL DIGT Classes DSNR
  FACILITY JES classes LDAPBIND LOGSTRM MQCMDS MQCONN
  PRINTSRV RRSFDATA TSO classes TERMINAL SURROGAT



RACGLIST Class

- Stores RACLISTed profiles in post-processed form for quick re-loading at IPL, upon initial RACROUTE REQUEST=LIST, and during REFRESH
- During RACLIST REFRESH for z/OS images sharing a database with Sysplex communications, first image fetches, merges, and stores a copy of processed member and grouping profiles for other images to simply retrieve and load
  - Systems sharing the RACF database must in the same GRS complex and GRS major name SYSZRAC2 must not be in the exclusion list
- Activated by class - profiles are class names
  - SETROPTS CLASSACT(RACGLIST)
  - RDEFINE RACGLIST *class-name*
- Especially beneficial for CICS, IMS, and DB2 classes
- Updated by SETROPTS RACLIST(*class*) REFRESH
- Ensure database has sufficient space for RACGLIST profiles
- Note: IPLs no longer cause refresh of RACGLISTed classes



RACF Database Reorganization

- Over time, administrative actions have the following effect
  - Index entry additions fill a block to overflowing requiring a block split
  - Profile and segment deletions empty all but small percentage of some blocks, wasting both database and buffer space
  - Newly added profile segments (e.g., OMVS) get stored in different blocks than the related profile requiring more I/O to fetch, especially during logon
  - Creating and deleting profiles causes fragmentation of free space making it difficult for RACF to find contiguous blocks for storing large profiles

- IRRUT400 utility - reorganizes the database - run periodically
  - Aligns index and associated profile blocks in sequential order
  - Fills in data blocks eliminating wasted space and fragmentation
  - Optionally places all profile segments in same block when possible
  - Compresses the index and corrects upper level index errors
  - Optionally adds free space to index blocks for subsequent growth
  - Rebuilds BAM blocks, thereby eliminating any prior errors

RACF Database Sharing

- Sharing a database in non-Sysplex Data Sharing mode (no Coupling Facility)
  - RACF uses exclusive hardware RESERVEs to serialize the database for most updates
  - System holding exclusive RESERVE locks out other systems until it has processed all its update requests
  - Lock is on entire DASD volume
- Global Resource Serialization (GRS)
  - Can convert RESERVEs to global ENQs
  - Each system given exclusive control for one update request at a time
  - Only locks the RACF database - not the entire DASD volume
  - Avoids contention and monopolization
  - PARMLIB(GRSRNLxx) conversion entry

    RNLDEF RNL(CON) TYPE(GENERIC) QNAME(SYSZRACF)
  - Restrictions
    - All z/OS systems must be part of the same GRS complex
    - Cannot be used when sharing a RACF database with a z/VM system
  - GRS required for Sysplex Data Sharing

RACF Sysplex Data Sharing

- Uses Coupling Facility as large store-through cache for the Resident Data Blocks - caches ICB, index, and profile data blocks (can improve performance for single system)

- Enabled by ICHRDSNT or PARMLIB(IRRPRMxx) option on first database entry
  - XL1'x0' No Sysplex
  - XL1'x8' RACF-Sysplex data communication without data sharing
  - XL1'xC' RACF-Sysplex data communication with data sharing

- Coupling Facility Resource Manager (CFRM) sets cache policy

- To assist in calculating the coupling facility size for RACF, go to http://www.ibm.com/systems/support/z/cfsizer/racf/

- If feasible, specify size large enough to hold all index blocks plus all data blocks for non-RACLISTed resource classes

Logging

- Use the following logging options only when necessary for essential security oversight
  - SETROPTS LOGOPTIONS(ALWAYS(*class*) | SUCCESSES(*class*))
  - SETROPTS OPERAUDIT
  - Resource AUDIT(SUCCESSES(READ))
  - Resource GLOBALAUDIT(SUCCESSES(READ))
  - User UAUDIT
    - Problematic if user makes extensive use of Unix File System objects

Statistics

- Eliminate the collection of resource access statistics
  - SETROPTS STATISTICS(*class*) | NOSTATISTICS(*class*) Option
  - Access counts kept only on Discrete profiles
  - Not incremented for GAT permitted access or RACLISTed class profiles
  - May not be accurate in a shared database environment
  - Increases CPU processing to calculate and I/O to record

- Update Statistics in the backup database as needed - ICHRDSNT or PARMLIB(IRRPRMxx) option

XL1'0x'	No updates are duplicated in the backup database (default)
XL1'8x'	Updates other than statistics are duplicated (recommended)
XL1'Cx'	Updates including statistics are duplicated (avoid)

- Limit user logon statistics updates to only once per day
  - Implemented via APPL class profiles for associated applications
  - Specify APPLDATA('RACF-INITSTATS(DAILY)') to activate

z/OS UNIX Identity Mapping

- Mapping required when corresponding identity must be determined (e.g., Unix 'ls' command - display RACF USERID and Group for Unix Owner UID and Group GID)

- Options to avoid searching all user and group OMVS segments for each look-up request
  - UNIXMAP Class
    - Contains profiles in the form U*nnn* and G*nnn*, where '*nnn*' is a UID or GID
    - Users and groups are 'permitted' access to signify UID and GID assignment
    - Profiles are automatically maintained when OMVS segments are created or altered via RACF commands
    - Class must be activated to be used for mapping
  - Application Identity Mapping (AIM)
    - Restructured database with mapping index structure
    - Implemented using IRRIRA00 utility
    - Replaces UNIXMAP profiles
    - Enables use of UID(*nnn*) and GID(*nnn*) on SEARCH command
    - Required to use newest features to replace the Unix Default User

- Additionally, cache UID and GID mappings in VLF

Virtual Lookaside Facility (VLF)

- VLF can cache RACF information for reuse
  - Accessor Environment Elements (ACEEs)
  - Group tree
  - z/OS Unix mappings of UIDs and GIDs to USERIDs and Groups
  - z/OS Unix User Security Packets (USPs)

- MAXVIRT parameter - VLF Maximum Virtual Storage
  - Optionally specified in PARMLIB(COFVLFxx) for each VLF CLASS
  - MAXVIRT(*nnnnnn*) - 4K block increments
    - Default: 4096
    - Range: 256 - 524288
  - Monitor VLF use - SMF record type 41, subtype 3
  - Default normally sufficient

Virtual Lookaside Facility (VLF)

- Accessor Environment Elements (ACEEs)
  - ACEE is created during logon process - contains user's attributes, lists of groups, and logon characteristics (e.g., Point-of-Entry (POE), application)
  - Caching avoids repeated retrieval of the user profile to build ACEEs for subsequent logons
  - PARMLIB(COFVLFxx) entry

    CLASS NAME(IRRACEE)

    EMAJ(ACEE)
  - Most changes to a user profile causes purge of some or all cached ACEEs for that user
    - Systems sharing a RACF database <u>without</u> Sysplex Communications purge all ACEEs for a user change
  - Refresh of certain classes causes purge of <u>all</u> cached ACEEs
    - APPCPORT APPL CONSOLE JESINPUT SERVAUTH TERMINAL
- Group tree
  - Used to determine scope-of-groups for Group-level authorities

    SPECIAL OPERATIONS AUDITOR
  - Caching avoids repeated retrieval of group profiles and tree reconstruction
  - Implement if group authority is used extensively
  - PARMLIB(COFVLFxx) entry

    CLASS NAME(IRRGTS)

    EMAJ(GTS)

Virtual Lookaside Facility (VLF)

- z/OS Unix mappings of UIDs and GIDs to USERIDs and Groups
  - Mappings give the associated USERID or Group for a UID or GID (e.g., 'ls' command)
  - Caching avoids repeated retrieval of mapping information
    - Recommended even with AIM restructured database
  - PARMLIB(COFVLFxx) entry

```
CLASS NAME(IRRGMAP)
    EMAJ(GMAP)
CLASS NAME(IRRUMAP)
    EMAJ(UMAP)
```

- z/OS Unix User Security Packets (USPs)
  - USP is created when user dubs (invokes z/OS Unix function)
  - Caching avoids repeated rebuilding of USPs during subsequent dubbing
    - Especially helpful for applications using thread level security
  - PARMLIB(COFVLFxx) entry

```
CLASS NAME(IRRSMAP)
    EMAJ(SMAP)
```

Enqueue Residency - ERV

- Contention issue - low priority TSO user or batch job gets swapped out while still holding an enqueue on SYSZRACF or a hardware RESERVE on the RACF database volume, and thereby holds up other address spaces and systems waiting on RACF

- Solution - grant more CPU Service Units to address spaces enqueued on system resources or holding hardware RESERVEs enabling them to complete work before being swapped out

- PARMLIB(IEAOPTxx) - ERV parameter
  - Range: 0 - 999999
  - Default: 500
  - Recommended: 40000 - 50000

RACF Commands and Utilities

- Avoid use of commands and utilities that are I/O or processing intensive during peak system activity periods (especially morning logon)

  LU * LG * RL class *

  LD with ID(), PREFIX(), or DSNS

  SR with NOMASK and AGE, USER, or WARNING

  SETROPTS GENERIC(*class*) REFRESH - especially DATASET

  SETROPTS RACLIST(*class*) REFRESH - especially classes with many profiles

  Large batches of commands - especially CONNECTs and REMOVEs

  IRRUT100 IRRUT200 IRRUT400 BLKUPD

  ICHDSM00 with FUNCTION RACGRP or RACUSR

  IRRDBU00 using live RACF database (use off-line IRRUT200 backup instead)

  RACF admin product extracts using live RACF database (use backup instead)

- Specify parameter NOYOURACC (or NOY) on RLIST commands to avoid retrieval and RACLIST processing of all grouping class profiles simply to determine your access

Miscellaneous

- Keep the RACF database clean of unnecessary permissions and obsolete USERIDs, groups, and resource profiles
  - Avoids wasted space in data blocks and cache
  - Reduces processing for IRRDBU00 unload and RACF admin product extracts
- RACF Database placement
  - Place each database dataset on a separate volume
  - Isolate the database datasets from other files or place them with infrequently accessed files
- Split RACF Database into multiple datasets
  - Requires implementation of RACF Range Table ICHRRNG or PARMLIB equivalent
  - Advantages - spreads workload across multiple DASD devices; each dataset gets its own I/O queue and set of RDBs
  - Disadvantages - complex; more datasets to manage, backup, etc.; requires all-way IPL to change
  - Avoid implementing - only required for very large databases (> 2GB)