Validating and Repairing RACF Database Integrity on z/VM

Prepared by the IBM z/VM Development Lab
Valid through: z/VM 6.4

Brian W. Hugenbruch, CISSP
IBM Z Virtualization Security
bwhugen@us.ibm.com

Change Summary

02 August 2017 Initial version (for z/VM 6.4 and releases under service)

Introduction

IBM's RACF® for z/VM® provides a mechanism for the control of resources associated with the z/VM environment. These external controls expand z/VM's security management and auditing capabilities in scope and granularity. RACF for z/VM stores its security policy in a database residing on disk attached to the system. Management of this database is critical for continuous operation of a secure z/VM environment.

The intent of this paper is to provide guidance for analyzing the RACF for z/VM database and repairing it, if necessary. It will outline two main operational procedures. First, steps for analyzing and repairing a RACFVM database shall be enumerated. Secondly, the steps for upgrading a RACF database shared between two or more z/VM systems shall be discussed.

Notes for Readers

1. This paper assumes that RACF for z/VM has been installed and is currently active in the environment. It also assumes that pertinent product configuration steps to enable RACF control have been correctly followed.

2. In this document, the following utilities will be referenced:
   - **RACUT200** – IRRUT200 is the "Database Verification Utility." It is most commonly used to create copies of existing databases, but it also validates the structure and format of the database. Always run RACUT200 before running RACF Convert (RACFCONV).

   - **RACUT400** – IRRUT400 is the "Database Extend Utility" program. It is most commonly used to copy the RACF database to a larger or smaller target volume, but it can also reorganize and restructure the database itself. This allows it to repair overwrite errors in the database.

   - **RACFCONV** – RACF Convert. This EXEC upgrades the template of the RACF database. (Under the covers, it invokes a program called IRRMIN00; output from this will be labeled MIN00 OUTPUT as a result.)

1. *How to Check the State of Your RACFVM Database before Beginning an Upgrade*

It is important to validate that your database is structurally sound before beginning an upgrade process. Attempting a RACF Convert on an injured database may lead to complications in the RACF database, including broken profiles or structural issues.

Reminder: it is highly recommended to keep a safe, solid, and current copy of the RACF database distinct from your primary and backup volumes. In cases where repair is required, it is often easier to swap in a valid RACF database than it is to repair the existing one. If a policy for creating back-ups is not currently in place, it is recommended that this be investigated.

To check the state of your RACFVM database, issue **RACUT200** against the database volume. **RACUT200** (as of z/VM 6.3) should be run under CST (IPL 490) and can be executed against a primary or backup volume while the system(s) is (are) running.

To verify a database while RACF is active, log onto RACMAINT or to any userid with the authority to link RACFVM's 305, 490, and database disks. In the example below, the RACF primary database (200) will be verified.

Enter the following to access the appropriate disks IF Not Already Done:

1. LINK RACFVM 490 490 RR
2. IPL 490
3. LINK RACFVM 305 305 RR
4. ACCESS 305 B
5. LINK RACFVM 200 200 RR
   (Enter read password if required)
6. LINK RACFVM 300 300 RR
   (Enter read password if required)
7. ACCESS 190 T

Then, Execute RACUT200 as follows:

1. Enter **RACUT200**
2. Reply **YES** to the 'Do you want to Verify a RACF database?' prompt.
3. If a RACVERFY FILE input file exists, you will be given the option to reuse it or overlay it. If a RACVERFY FILE does not exist, one will be created and XEDIT will be entered. Type **FILE** when editing is complete.
4. Reply **200** to the 'Enter the Input device address' prompt.
5. Press **Enter** to bypass copy.
6. Reply **YES** to the 'Do you wish to continue?' prompt.
7. Messages RPIOPN003E and IRR62003I can be ignored. You may also get messages DMSLOS013E and IRR62064I.
8. Return code from 'IRRUT200' = 0 should be issued if the command is successfully completed.
9. The IRRUT200 output report will be sent to your virtual printer.

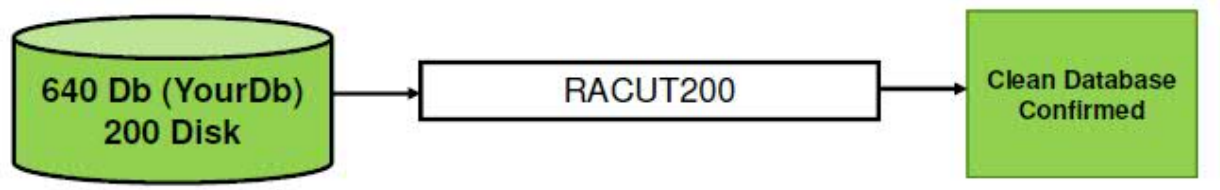

Figure 1. RACUT200 for the RACF Database

Note: It is recommended that verification is run at a time of low RACF activity, or against a database that is inactive to RACF.

Once **RACUT200** has completed, analyze the output in your virtual printer. Do this no matter what return code RACUT200 provides. While errors will most likely be surfaced as a non-zero return code, along with associated RACF PROFILE errors or similar, there may also be structural problems in the RACF database. Compare and contrast this output with sample UT200 output from a previous (clean) execution.

If you received a non-zero return code, then PEEK or XEDIT the output file, and do a quick locate for errors. Search for "IRR62" messages and "LOCATIONS WITH POSSIBLE CONFLICTS". You may find errors in RACF PROFILEs or RBA errors. These errors will need to be corrected before running the RACFCONV utility.

If no problems are in evidence, proceed to **Section 2** (How to Upgrade a RACF Database Shared Between Two or More z/VM Systems). Otherwise, proceed to **Section 3**, for advice on repairs.

2. How to Upgrade a RACF Database Shared Between two or more z/VM systems

When upgrading your RACF-enabled z/VM system, additional steps must be taken to ensure that your database is not injured during the upgrade process. PTFs for your RACF for z/VM feature may change the layout of the database by upgrading the database template. Changes to the structure of the database may inadvertently lead to integrity or corruption problems. To that end, establishing MWV on your RACF database is not considered sufficient, even when operating in a z/VM Single System Image environment; the RACFVM operational machines must be shut down.

Please use **RACFCONV** on the Userid **RACMAINT** to update the templates for both the primary and the backup RACF database.

1. If you are sharing your RACF database, whether you are operating in a Single System Image or not, validate the database links for your RACMAINT virtual machine(s).

   ```
   LINK RACFVM 200 200 MW
   LINK RACFVM 300 300 MW
   ```

2. If operating in a Single System Image, Service RACF from MAINT6n0 virtual machine on only one SSI member.

   If sharing your database between standalone systems, apply service to only one system to start.

3. Force RACFVM from Operator on each system attached to your RACF database, whether operating in an SSI or not.

   LOGON RACMAINT on your upgraded system, using the password in your CP User Directory, and issue the following commands:

   `IPL 190`

```
RACFCONV
enter
200
yes

RACFCONV
enter
300
yes

IPL 490
RACSTART
#cp disc
```

If RACFCONV receives RC=0, then RACFCONV has successfully upgraded the RACF templates.

If RACFCONV receives RC=4, it means that your templates were already at the correct level. You can ignore RC=4. You will also receive the following response from the RACFCONV command, which means that your database has already been converted:

```
RACFCONV

An Error occurred during 'IRRMIN00' processing
Return code from 'IRRMIN00' = 4
Ready(00004);
```

4. XAUTOLOG RACMAINT on all remaining systems associated with this RACF database.

5. PUT2PROD RACF from MAINT6n0 on each system associated with your RACF database.

   Note: If Put2prod messages say to Recycle the appropriate servers. CP and RACF; then recycle of z/VM is necessary for each SSI member. Otherwise only RACF needs to be recycled on each SSI member.

6. Force RACMAINT from Operator on every system associated with your RACF database (including the first system updated).

7. XAUTOLOG RACFVM from Operator on all systems associated with your RACF database.

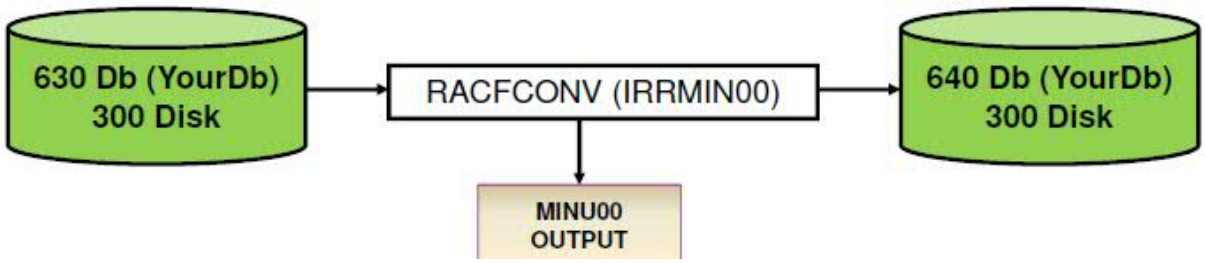

Figure 2 RACFCONV for a RACF Database

3. How to Repair a RACF Database

If problems are detected when doing validation on the RACF database via RACUT200, or if problems are later uncovered when attempting to put a problematic database into production, then some degree of repair may be required. The specifics required for repair will vary based upon the nature of the error.

Before proceeding with repair:

- If this database is running under production, then it is recommended to backup the production database in error, restore it to a z/VM test system, and execute the following commands from there.
- If this database is running in a test environment, then make a back-up copy of the existing database before proceeding with the steps to repair.

It is recommended, though not required, to force off other test-system RACFVM machines sharing this reloaded test database. A production system should not be sharing a database currently under repair. **Note** also:

- The same disk linking requirements apply with RACUT400 as they do with RACUT200.
- The virtual addresses for disks (when copying via IRRUT400) must be in line with IBM conventions: 200 for your primary database, 300 for your back-up database.

If structural errors in upper-level blocks (sometimes referred to as RBA errors) are noted in your UT200 output, then it is recommended that you use **RACUT400**. These problems may manifest in the UT200 output as poor mapping (e.g., RBA=00000000B000) or as warnings under a banner “Locations With Possible Conflicts”.

> **Note** that RACUT400 is (a) not commonly used for standard copying operations, and (b) not to be used for making a diagnostic analysis. (Use **RACUT200** for those cases.)

To execute RACUT400 to perform a reparative copy, log onto RACMAINT or to any userid with the authority to link RACFVM’s 305, 490, and database disks (as done in Section 1). Once appropriate disks are linked, use the following steps:

1. Enter **RACUT400**
2. Reply **COPY** to the 'Enter SPLIT or MERGE or COPY or QUIT' prompt.
3. Reply **200 or 300** to the 'Enter the single input device address' prompt.
4. Reply **400** to the 'Enter the single output device address' prompt.
5. Reply **YES** to the 'Do you wish to continue?' prompt.
6. Reply **CONT** to enter parameters.
7. If executing against an offline or unshared database, Reply NOLOCKINPUT to the first ‘Enter Next Parameter’ prompt. Otherwise, reply **LOCKINPUT** to the first 'Enter Next Parameter' prompt.
8. Reply **END** to the second 'Enter Next Parameter' prompt.

9. Return code from 'IRRUT400' = 0 should be issued if successful.

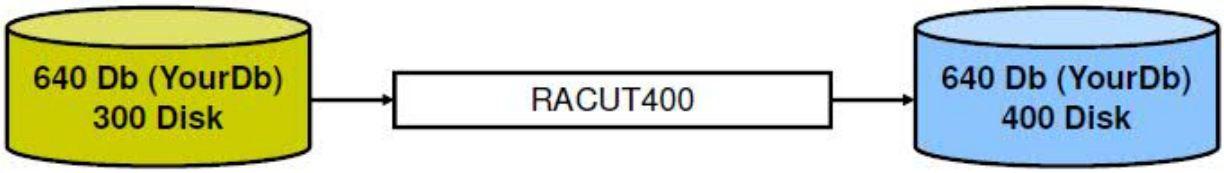

Figure 3. RACUT400 for the RACF Database

Once RACUT400 has completed, re-run **RACUT200** (as illustrated in Section 1) against the updated database and confirm that the output indicates no errors. Once the database appears to be in good working order, copy the fixed database from the 400 disk back to its original location on the 200 or 300 disk of its appropriate system, and proceed to the next section.

Note: if restoring a fixed RACF database to a production system, then either an outage will be required, or use of the RAC **RVARY** commands will be required to "juggle" backup and primary RACF databases. Consult the *RACF Command Reference* and *RACF System Administrator's Guide* for more information if you wish to use **RVARY**.

If you are seeing errors related to profiles, rather than structural issues, then a couple of steps are offered. Bear in mind that the following steps require that the RACF database be loaded and active, so it is recommended that the following steps be done from a non-production system or during periods of low security activity.

First, deletion and a re-add of offending profiles (through standard RAC commands) can be tried against the test copy of the database. The commands to use would be DELUSER/ADDUSER, DELGROUP/ADDGROUP, or RDELETE/RDEFINE (depending upon the resource profile in question). On a z/VM system, these commands are most commonly issued from a RACF-enabled user through the RAC EXEC interface.

If these profiles return error codes when attempting to delete or re-add, or yield RACUT200 errors after execution, then more drastic steps may need to be taken, as outlined below.

If a current and valid copy of the RACF database is not available, then RACFDBU and RPIDIRCT can be used to build a new one. RACFDBU unloads RACF database profile information into a more easily read format. RPIDIRCT analyzes the CP User Directory and generates all the commands needed to create a new RACF database. These two utilities can be used to create a RACF database that matches the broken one. Note that some care must be taken in analysis of this output, since RPIDIRCT does not generate security policy commands (e.g. SETROPTS PASSWORD controls) and RACFDBU may be limited by the age of the database under use.

In more drastic scenarios, the **BLKUPD** utility can be used to make manual adjustments to the RACF database. Because this utility augments the data structures inside your RACF database, it should only be used when directed by the IBM support center.

Note that **BLKUPD** is not recommend for use with BAM errors (use **RACUT400** instead), should not be used in periods of high activity, and should only be used after using **RACUT200** to determine what problems are left with a RACF database.

> **Before you use the BLKUPD command, you should be very familiar with the RACF database and its configuration because improper usage of BLKUPD can result in damage to the RACF database. (See "Format of the RACF Database" on page 53 of the *z/VM RACF Diagnosis Guide*.)**
>
> **In general, use this command only when directed to by the IBM support center.**
>
> **You should read and understand the pages on the format of the database before entering the BLKUPD command. Then, before you begin to use the BLKUPD command to perform updates to your RACF database, we recommend your trying to use one of the RACF commands to alter or delete the entry in question.**

To use the **BLKUPD** command, either consult with IBM Support, or:

1. Decide which database you want to work with, and enter BLKUPD
2. Decide which block on the database you want to work with. If needed, use the LOCATE subcommand to assist you in finding the specific block. If you've reached this part of the document, IRRUT200 output should provide some guidance in this.
3. Enter the READ subcommand, specifying either UPDATE or NOUPDATE
4. Enter the subcommands of READ necessary to accomplish your task
5. Issue the END command to end the utility.

Refer to the z/VM RACF Diagnosis Guide, Chapter 5, for more detailed instructions on BLKUPD.

4. RACF for z/VM: Database Best Practices

The following practices are offered with regards to the RACF Database on z/VM:

1. **Make a policy for creating and validating copies of the RACF database**. The z/VM system requires both a valid primary and backup database during initialization. Having a procedure for validating and storing current copies will help in case of emergencies.

2. **Keep a safe, solid, and current reserve copy of the RACF database.** Keep this away from the primary and back-up volumes. It is often easier to restore an older database than it is to repair one in-use.

3. **Validate and integrity-check databases before an upgrade of any sort.** If documentation prompts for the issuance of RACFCONV, issue RACUT200 against the database first.

4. **Segment the RACF database judiciously.** While older databases may have been segmented for performance purposes, modern z/VM systems usually do not require the database to be sharded in this fashion. Similarly, one RACFVM virtual machine should suffice per z/VM LPAR.

5. **Share cautiously.** Be advised that unless the RACF Database is shared in a z/VM Single System Image, the systems sharing the database may have distinct security requirements. This can lead to semantic (rather than syntactical) confusion in terms of security policy. This is especially true in cases where the RACF database is shared between z/OS® and z/VM. While such sharing is possible, it is not always advisable.

6. **Automate extensively around RACF Startup.** While the RACFVM virtual machine is redundant in its code and database usage, consider automation products for the detection of database problems during system start-up. Automating the attach and deployment of a spare RACF database may lessen downtime and facilitate faster detection of problems.

Summary

RACF for z/VM provides enterprise-level security for your hypervisor. It acts as policy decision point and policy enforcement point for IBM Z® virtualization. Despite its importance, there is no need to panic if a problem occurs with the RACF database. There are tools available to mitigate and fix errors in those cases where they do occur. However, the most important part of such procedures is to have a plan for such emergencies.

Reference Material

- *z/VM Installation Guide* – for specific instructions on the process by which service is applied to RACF (and z/VM).

- *z/VM RACF System Programmer's Guide* – for details about the RACF Utilities listed in this paper, refer to Chapter 5 ("Utilities for the RACF Database"). It includes details regarding return codes, reports, and error messages.

- *z/VM RACF Diagnosis Guide* – Chapter 5 ("Troubleshooting your RACF Database") is required reading for anyone looking to maintain and repair a RACF database. It discusses the structure of the RACF databases, including specific addresses which may help to determine if something vital has been overwritten.

IBM Corporation
New Orchard Road
Armonk, NY 10504
U.S.A.

Produced in the United States of America,
07/2017

ZSW03366-USEN-02