HOT TOPICS

User Key Common and Restricted Use of CSA

Overview of z/OS RUCSA feature - What it is, what it solves and how to migrate it.

By: Peter Fatzinger, David Hom, Steve Partlow
Published: July 9th, 2021
Read time: 5 minutes

RUCSA: Securing your user key common memory

What happened to user key common memory in z/OS v2.4?

Support for user key common memory was removed in z/OS v2.4 after years of IBM discouraging its use. Beginning in the same release, a new priced feature, RUCSA (Restricted Use Common Service Area), was introduced to protect application data without requiring changes to the programs that obtain the now forbidden user key CSA. In addition, some of the new RUCSA capabilities were backported to z/OS v2.1, v2.2, and v2.3 (with no associated customer cost) so that clients could use the tools to determine if they'd need the new feature when they upgraded to z/OS v2.4.

All three types of user key common memory are disallowed by z/OS v2.4 and may require application changes to avoid:

1. User key memory from the common service area (CSA),
2. User key SCOPE=COMMON data spaces,
3. System Queue Area (SQA) memory that was changed to user key.

Why is user key common a problem?

Storage keys are a primary way to protect memory so that programs can read or write data according to their authorization level. The use of user key common memory made it easy to communicate between different unauthorized jobs, but also put this memory at greater risk of unintended modification by any program on the system. Vulnerable, flawed, or malicious programs can disrupt users of that common memory. The integrity of the system could be compromised if authorized programs using that memory are manipulated by the rogue program to perform an action it shouldn't. Critical system control blocks may be overlayed or sensitive data exposed.

On releases prior to z/OS v2.4, an installation can disable the use of user key common with DIAGxx.

– VSM ALLOWUSERKEYCSA(NO), which prevents obtaining user key common memory, was introduced in z/OS v1.8 and became the default setting in v1.9.

– ALLOWUSERKEYCADS(NO), which prevents user key SCOPE=COMMON data spaces, was introduced in z/OS v1.11.

– NUCLABEL ENABLE(IARXLUK2), which prevents changing SQA to user key, was introduced in z/OS 2.3.

If you don't have all these DIAGxx settings restricting user key common, you may not know if you're impacted. Beginning in z/OS v2.1, health check ZOSMIGV2R3_NEXT_VSM_USERKEYCOMM can be run to determine if your system is using any user key common and would be affected by a migration to z/OS v2.4.

How can user key common be replaced?

The most secure choice for an installation is to remove all instances of user key common storage. The basic options are to change the affected software to use a system key rather than a user key or alternatively, change processing to not use storage that is common to all address spaces. There are several ways to limit the sharing of storage to the impacted address spaces:

– DSPSERV SCOPE=ALL data space

– IARVSERV SHARE for below-the-bar storage

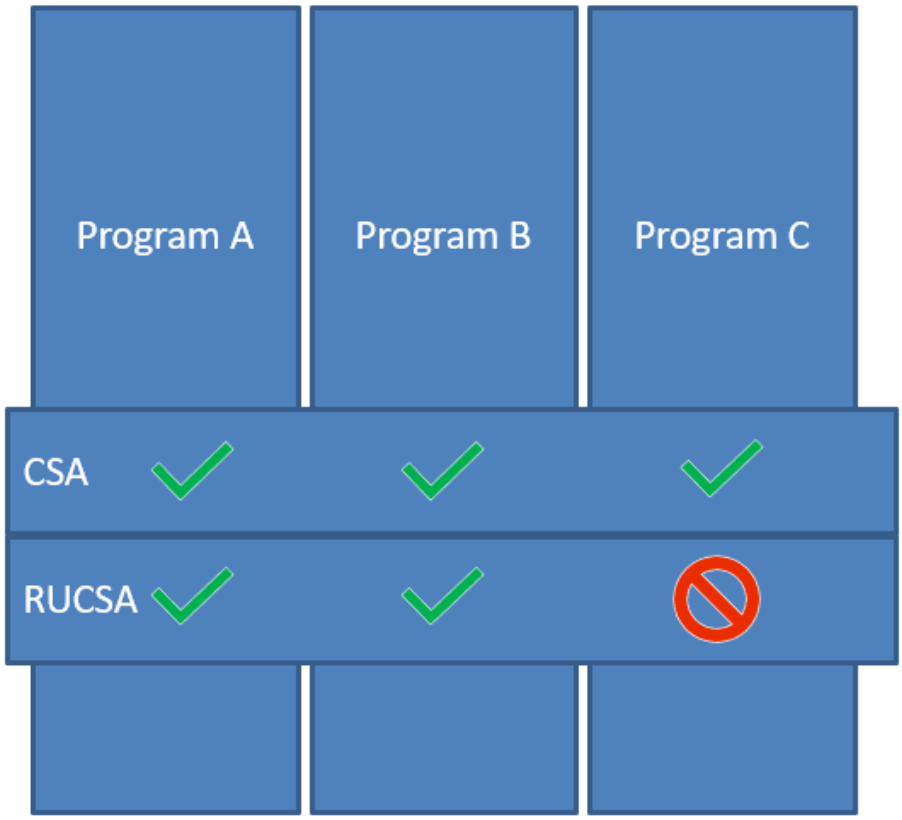
– IARV64 GETSHARED for above-the-bar storage

– z/OS UNIX shared memory

What is Restricted Use CSA (RUCSA)?

While the ideal option is to update any program using user key common, that is not always feasible. RUCSA was developed for those cases where the elimination of user key CSA could not be accomplished. RUCSA, a new priced feature in z/OS v2.4, can protect application data by default and give access only to the programs selected by the installation. Functions of this new feature are made available to be installed and tested on all earlier supported releases free of charge to evaluate and prepare for upgrading to z/OS v2.4. Additional support in z/OS v2.4 includes a new health check, VSM_RUCSA_THRESHOLD, for monitoring usage growth of RUCSA.

RUCSA is a new area of 31-bit common, which is obtained and accessed in the same way as CSA except that programs must be permitted to it with the system authorization facility (SAF) profile IARRSM.RUCSA. Figure 1 shows three programs with their own address spaces, and CSA and RUCSA as common memory across them all. Programs A and B can access any memory in CSA or RUCSA their key allows, but program C cannot access any RUCSA regardless of its key.



How do I get started with RUCSA?

RUCSA is designed to avoid needing changes to application code, but installation changes must be made to migrate to RUCSA. The complete steps are defined in the z/OS Upgrade workflow but in brief they are the following:

- Find out if you're affected using the migration health check. If there's no current usage, congratulations, you do not need RUCSA. Restrict usage using the DIAGxx settings to make sure you won't need it in the future.

- Identify the users of user key CSA by using SMF records and other traces. If you can change the identified programs to no longer rely on user key common, do so and go back to the first step. If you are unable to change the identified programs to no longer rely on user key common, move on to the next step.

- Prepare for RUCSA by gathering usage data to determine how much will be needed. You can use RMF reports or by viewing counts in IPCS.

- Define RUCSA in IEASYSxx. At this point all user key CSA requests are instead routed to RUCSA.

- Use SMF to determine who is using RUCSA and authorize them to the SAF profile.

- Set ALLOWUSERKEYCSA(NO) in DIAGxx to enforce the SAF controls.

- Run the migration health check again to verify that you are ready for z/OS v2.4.

Is there anything else to consider?

Be aware that RUCSA is a separate 31-bit common area that can only be obtained in 1 megabyte increments so careful consideration is needed before defining it. Once it is defined, all user key CSA requests go to it and obtains don't overflow between RUCSA and CSA. Adding RUCSA will take away from available private area so consider

reducing the overall size of CSA. If possible, avoid defining RUCSA below 16M to avoid further restricting the limited private area there.

Another consideration is that authorization to RUCSA is all or nothing. Programs that don't need access to it can't use it at all. However, if a program is authorized to RUCSA, it has access to the entire RUCSA area (given its key allows that). Therefore, care must be taken when authorizing programs to RUCSA to consider what impact they could have on other programs that also access it.

The new RUCSA feature helps avoid the need for application changes while providing enhanced isolation to user key memory. By eliminating the use of unrestricted user key common in z/OS 2.4 installations are now more secure than ever.

Where can I learn more?

- z/OS MVS Initialization and Tuning: Restricted Use Common Service Area
- z/OS Upgrade Workflow BCP: Removal of support for user key common areas
- ZOSMIGV2R3_NEXT_VSM_USERKEYCOMM
- VSM_RUCSA_THRESHOLD (z/OS v2.4 only)
- V2R3 Statements and parameters for DIAGxx
- APAR OA56180 PDF

About the authors

Peter Fatzinger is a software developer for IBM Z and product owner for a team

responsible for Memory Management, Consoles and GRS.

David Hom is a software developer who has worked in the Job Management, Consoles, Memory Management and Logger components over his 30+ year career at IBM.

Steve Partlow is a software developer and lead for z/OS memory management.

Mary Fetterman contributed to the technical and editorial review of this article.