z/OS

# APARs RACF (OA35973), SAF (OA35974): Enhanced z/OS UNIX File System Security

# Preface

This information applies to APARs OA35973 (RACF) and OA35974 (SAF) which place the access control administration of zSeries File Systems within the responsibilities of the RACF security administrator.

## Overview

APARs OA35973 (RACF) and OA35974 (SAF) provide RACF support for administering access control to zSeries File Systems. This is accomplished by adding a new optionally enforced access control check that validates a user's authority to access file system objects using a RACF general resource class profile.

## Software requirements

Support for APARs OA35973 (RACF) and OA35974 (SAF) requires one of the following software releases:

- z/OS Security Server RACF Version 1 Release 12 (FMID HRF7770)
- z/OS Security Server RACF Version 1 Release 13 (FMID HRF7780)

To take advantage of this support from z/OS UNIX System Services, APAR OA35970 is required.

## Updated RACF publications

The chapters of this document supplement the V1R12 and V1R13 levels of the following RACF publications:

| Chapter | Supplements ... |
|---|---|
| Chapter 1, "Security administration updates," on page 1 | *z/OS Security Server RACF Security Administrator's Guide* |
| Chapter 2, "System Programmers Guide updates," on page 3 | *z/OS Security Server RACF System Programmer's Guide* |
| Chapter 3, "Callable services updates," on page 7 | *z/OS Security Server RACF Callable Services Reference* |
| Chapter 4, "RACF messages and code updates," on page 9 | *z/OS Security Server RACF Messages and Codes* |
| Chapter 5, "Data area updates," on page 11 | *z/OS Security Server RACF Data Areas* |
| Chapter 6, "Class descriptor table updates," on page 13 | Various publications |

# Chapter 1. Security administration updates

This information supplements *z/OS Security Server RACF Security Administrator's Guide*.

## Update to "Chapter 17. RACF and z/OS UNIX"

The following topic is added.

### Restricting access to a zFS file system

You can restrict access to a zFS file system by defining a general resource profile in the FSACCESS class. This enables you to use RACF commands to restrict z/OS UNIX access to the specified zFS file system for most users while allowing selected users and groups to remain eligible to access the file system. This method supports an improved audit posture by enabling the RACF administrator to demonstrate a single point of control for restricting access to one or more file systems that might contain sensitive or personal data.

When you define an FSACCESS profile, you restrict access to the file system, which includes all of its files and directories, at only the file system level. By contrast, the z/OS UNIX administrator can use the **setfacl** command to control access at the file system level and to control access to any of its files and directories on an individual resource basis.

When a zFS file system is protected by an FSACCESS profile with UACC(NONE), only users and groups with UPDATE access authority or higher, and users with the AUDITOR attribute, are eligible to access to the file system. Eligible users are then subject to the usual authorization checking, which includes checking for superuser authority, ownership, permission bits, access control lists (ACLs), and UNIXPRIV authorities.

When a zFS file system is protected by an FSACCESS profile and a user has insufficient access authority to it, no further authorization checking is done, and z/OS UNIX access to the protected file system, including all of its files and directories, is denied. Note that while superuser authority can be used to mount a file system protected by an FSACCESS profile, it is insufficient authority to access it. Also, note that access authority to the MVS data set that contains the file system is unaffected when you define the FSACCESS profile.

You need not authorize UPDATE access for users with the AUDITOR attribute. These users are exempt from the access restrictions enforced by the FSACCESS profile.

#### Steps for restricting access to a zFS file system

**Before you begin:** For each zFS file system, ask the z/OS UNIX administrator for the name of the MVS data set where the file system is stored.

Perform the following steps to restrict access to a zFS file system.

1. Define a profile in the FSACCESS class to protect each zFS file system. The profile name is the name of the MVS data set that contains the file system.

   **Example**:

```
RDEFINE FSACCESS OMVS.ZFS.WEBSRV.TOOLS UACC(NONE)
```

If multiple file systems are stored in data sets with similar names, you can define a generic profile name to protect multiple file systems. Before you define a generic profile in the FSACCESS class, enable generics for the class, as follows.

**Example**:

```
SETROPTS GENERIC(FSACCESS)
RDEFINE FSACCESS OMVS.ZFS.WEBSRV.** UACC(NONE)
```

2. Authorize selected users and groups with UPDATE access.

   **Example**:

   ```
   PERMIT OMVS.ZFS.WEBSRV.TOOLS CLASS(FSACCESS) ID(GROUPB USER19) ACCESS(UPDATE)
   ```

3. Activate your profile changes in the FSACCESS class, as follows.
   - If the FSACCESS class is not already active, activate and RACLIST it.

     **Example**:

     ```
     SETROPTS CLASSACT(FSACCESS) RACLIST(FSACCESS)
     ```

   - If the FSACCESS class is already active and RACLISTed, refresh it.

     **Example**:

     ```
     SETROPTS RACLIST(FSACCESS) REFRESH
     ```

You have now restricted access to a zFS file system to only the specified users and groups.

**Restricting access to all zFS file systems:** In addition to restricting access to individual zFS file systems, your installation might consider adding a *top* generic profile in the FSACCESS class to restrict access to all zFS file systems. The profile name might consist of two asterisks (**) as the MVS data set name. By defining and activating a top generic profile, you will disallow access for all users to any zFS file system that is not protected by another FSACCESS profile.

**Important:** Before implementing a top generic profile in the FSACCESS class, work with the z/OS UNIX administrator and carefully plan your profile names and access lists.

**Example**:

```
RDEFINE FSACCESS ** UACC(NONE)
```

## Update to "Appendix E. Debugging problems in the RACF database"

In the topic called "Authorizing Access to z/OS UNIX files and directories", the following step is added after Step 2.

If the user is attempting to access a zFS file system, RACF searches for an FSACCESS class profile that protects the file system when all of the following conditions are met:

- The user does not have the AUDITOR attribute.
- A file system name was specified in the CRED.
- The FSACCESS class is active and RACLISTed.

If a matching profile is found and the user does not have at least UPDATE authority, access is denied. Otherwise, access checking continues.

# Chapter 2. System Programmers Guide updates

This information supplements *z/OS Security Server RACF System Programmers Guide*.

## Changes to Exits

Changes were made to include exit processing for RACROUTE REQUEST=FASTAUTH with the FSACCESS class.

### RACROUTE REQUEST=FASTAUTH exits

RACROUTE REQUEST=FASTAUTH examines the auditing and global options in effect for the resource while determining the access authority of the caller. The FASTAUTH request returns a reason code that indicates whether the access attempt should be logged. The RACROUTE REQUEST=FASTAUTH exits allow the installation to make additional security checks or to instruct RACROUTE REQUEST=FASTAUTH to either accept or fail a request.

**Notes:**

1. The RACROUTE REQUEST=FASTAUTH exits do not get control during authorization requests for the PROGRAM class and cannot be used to affect PROGRAM processing.
2. When the FASTAUTH request is invoked for the UNIXPRIV or FSACCESS class, the FASTAUTH service is called directly from a callable service, and the SAF router exit, ICHRTX00, is not called.
3. The exits can view the values for the AUTHCHKS and CRITERIA keywords, but should not modify them.

#### Preprocessing exits (ICHRFX01 and ICHRFX03)

There are two RACROUTE REQUEST=FASTAUTH preprocessing exits. In general, ICHRFX01 is used for non-cross-memory calls and ICHRFX03 is used for cross-memory calls.

**Exceptions:** ICHRFX03, if present, is always called instead of ICHRFX01, even in non-cross-memory mode, in the following situations:

- A FASTAUTH request is invoked for the UNIXPRIV or FSACCESS class.
- The ACEEALET or ENVRIN operand is specified.
- A supervisor state or system key caller provides a nested ACEE on a FASTAUTH request. It does not matter whether the nested ACEE is processed; for example, if the client is authorized or the resource is not delegated, ICHRFX03 is still called. For information about nested ACEEs and delegated resources, see the section on delegated resources in *z/OS Security Server RACF Security Administrator's Guide*.

⋮

#### Postprocessing exits (ICHRFX02 and ICHRFX04)

There are two RACROUTE REQUEST=FASTAUTH postprocessing exits: ICHRFX02 and ICHRFX04. Figure 1 on page 4 shows the logic that RACF® uses to determine which exit to call.

**Note:** For a nested ACEE, although two authorization checks might be internally driven, ICHRFX04 is only called once, after both checks have completed. It does not matter whether the nested ACEE is processed; for example, if the client is authorized or the resource is not delegated, ICHRFX04 is still called. For information about nested ACEEs, see the section on delegated resources in *z/OS Security Server RACF Security Administrator's Guide*.

```
if cross memory mode, or ACEEALET or ENVRIN or CRITERIA is specified,
   or the class is UNIXPRIV or FSACCESS,
   or a nested ACEE is provided by a supervisor state or system key caller

  then

   only call ICHRFX04

else

   if RACLISTed by RACROUTE REQ=LIST, GLOBAL=YES or RACLISTed by SETR RACLIST
      or the class is in the dynamic class descriptor table

   then

      call ICHRFX04 first and then call ICHRFX02

   else

      only call ICHRFX02
```

*Figure 1. Logic that determines whether ICHRFX02 or ICHRFX04 is called*

The sequence of pre- and post- processing exit invocation, FASTAUTH authorization processing, and auditing (when FASTAUTH performs auditing due to LOG=ASIS or LOG=NOFAIL), is:

| Conditions | Processing sequence |
|---|---|
| Regardless of how the class is RACLISTed:<br>• Cross-memory, or<br>• The ACEEALET keyword is specified, or<br>• The ENVRIN keyword is specified, or<br>• The CRITERIA keyword is specified, or<br>• The UNIXPRIV class, or<br>• The FSACCESS class, or<br>• A nested ACEE | 1. ICHRFX03<br>2. Auth processing<br>3. ICHRFX04<br>4. Auditing |
| Non-cross-memory and the class is RACLISTed by RACROUTE REQUEST=LIST,GLOBAL=NO | 1. ICHRFX01<br>2. Auth processing<br>3. ICHRFX02<br>4. Auditing |

| Conditions | Processing sequence |
| --- | --- |
| Non-cross-memory, and the class is in the dynamic class descriptor table or the class is RACLISTed by RACROUTE REQUEST=LIST,GLOBAL=YES or by SETROPTS RACLIST<br><br>Note that RACF performs logging based on the return and reason code set by:<br>• ICHRFX01 or ICHRFX04, if they exist<br>• If ICHRFX01 and ICHRFX04 do not exist, by one of the following:<br>– The default return code defined for the class in the class descriptor table (CDT)<br>– FASTAUTH processing done before ICHRFX02 gets control<br><br>Any return and reason code set by ICHRFX02 in this case is not reflected in the auditing done by FASTAUTH, but is processed as described in "ICHRFX02" on page 327. | 1. ICHRFX01<br>2. Auth processing<br>3. ICHRFX04<br>4. Auditing<br>5. ICHRFX02 |

Default return code processing occurs prior to auditing. If the profile was not found and the postprocessing exit did not change the return code, FASTAUTH uses the default return code from the class descriptor table (CDT). The default return code, if used, is reflected in the auditing done by FASTAUTH.

⋮

# Chapter 3. Callable services updates

This information supplements *z/OS Security Server RACF Callable Services Reference*.

## Security credentials (CRED)

CRED content for audit data is updated with new information as follows:

- **Audit data:**
  - **File System Name:** the name of the file system containing the specified file name or names. If supplied by LFS for ck_access and the FSACCESS class is RACLISTed, RACF verifies that the user has access to a matching profile defined in that class.

## Updated information about ck_access (IRRSKA00): Check access

ck_access is updated with new information as follows.

### RACF authorization

If the user does not have the RACF Auditor attribute, and a file system name was specified in the CRED, and the FSACCESS class is active and RACLISTed, RACF will check for a profile in the FSACCESS class that covers the file system name. If a matching profile is found and the user does not have at least UPDATE authority, access is denied. Otherwise, authorization is determined by subsequent checks.

# Chapter 4. RACF messages and code updates

This information supplements *z/OS Security Server RACF Messages and Codes*.

## Updated information about message ICH408I

**ICH408I** **USER (***userid***) GROUP (***group-name***) NAME (***user-name***) --or-- JOB (***jobname***) STEP (***stepname***) [SUBMITTER (***userid***)] [PRIMARY USER (***userid***)] [***resource-name***] [CL(***class-name***)] --or-- [VOL(***volume-id***)] [FID(***file-identifier***)] [ID(***IPC-identifier***)] --or-- [FROM** *generic-profile-name* **(G)] [ACCESS INTENT(***intent***) ACCESS ALLOWED(***allowed***)] [EFFECTIVE UID (***nnnnnnnnn***] [EFFECTIVE GID (***nnnnnnnnn***]**

**Explanation:** For attempts to use protected resources, the message shows the access attempted (`ACCESS INTENT` phrase) and the access permitted by RACF (`ACCESS ALLOWED` phrase). When the message is reporting an attempt to access a z/OS UNIX file or IPC key, the ACCESS INTENT *(intent)* is specified as "RWX", representing read, write or search/execute permission requested. More than one permission can be requested at a time. If a permission is not requested, the letter is replaced by a dash "-". ACCESS ALLOWED *(allowed)* is specified as "{OWNER/GROUP/OTHER/ACL USER/ACL GROUP/NO/RESTRICTED/FSACCESS} RWX", where OWNER indicates the owner permission bits were used, GROUP indicates the group permission bits were used, OTHER indicates the other permission bits were used, ACL USER indicates that a specific user Access Control List (ACL) entry was used, ACL GROUP indicates a specific group ACL entry (or entries) was used, NO indicates that no permission bits were used, RESTRICTED indicates the OTHER bits were not used for a RESTRICTED user, FSACCESS indicates a profile in the FSACCESS class was used, and "RWX" represents the settings of the permission bits that were checked. ACCESS ALLOWED (NO --X) occurs if a superuser attempts to execute a file that does not have OWNER, GROUP, ACL, or OTHER execute permission. ACCESS ALLOWED (RESTRICTED ——) occurs if a RESTRICTED user can have only gained file access by way of the OTHER bits, but the RESTRICTED.FILESYS.ACCESS profile in the UNIXPRIV class has forbidden this. ACCESS ALLOWED (FSACCESS ——) occurs if the user does not have access to the FSACCESS profile protecting the file system that contains the resource.

# Chapter 5. Data area updates

This information supplements *z/OS Security Server RACF Data Areas*.

## Updated information about IRRPAFC (AFC: z/OS UNIX System Services audit function codes)

New field, AFC_FSACCESS, has been added.

| Offsets | | Value | Name | Description |
|---|---|---|---|---|
| **Len** | **Type** | | | |
| 2 | DECIMAL | 123 | AFC_FSACCESS | File system access |
| 2 | DECIMAL | 124 | AFC_ENDOF_TAB | End of table |

## Updated information about IRRPCRED (CRED: z/OS UNIX System Services Credential Structure)

New fields CREDFS, CREDFSALET, CREDFSADDR have been added.

| Offsets | | | Len | Name (Dim) | Description |
|---|---|---|---|---|---|
| **Dec** | **Hex** | **Type** | | | |
| 101 | (65) | CHARACTER | 3 | * | Reserved |
| 104 | (68) | CHARACTER | 8 | CREDFS | For ck_access, file system name area |
| 104 | (68) | ADDRESS | 4 | CREDFSALET | For ck_access, ALET of the file system name |
| 108 | (6C) | ADDRESS | 4 | CREDFSADDR | For ck_access, address of a 44-byte area containing the file system name, padded with blanks |
| 112 | (70) | CHARACTER | 16 | * | Reserved |

### Cross Reference

| Name | Hex Offset | Hex Value | Level |
|---|---|---|---|
| CREDFS | 68 | | |
| CREDFSALET | 68 | | |
| CREDFSADDR | 6C | | |

# Chapter 6. Class descriptor table updates

This information supplements the topics called "Supplied class descriptor table entries" and "Supplied resource classes for z/OS systems" which affects the following publications:

- *z/OS Security Server RACF General User's Guide*
- *z/OS Security Server RACF Security Administrator's Guide*
- *z/OS Security Server RACF Command Language Reference*
- *z/OS Security Server RACF System Programmer's Guide*
- *z/OS Security Server RACF Macros and Interfaces*
- *z/OS Security Server RACROUTE MACRO Reference*

## Update to "Supplied resource classes for z/OS systems"

The following description is added for the new FSACCESS class:

| Class name | Description |
|---|---|
| FSACCESS | Controls access to z/OS UNIX file systems. |

## Update to "Supplied class descriptor table entries"

The following information is added about the attributes of the new FSACCESS class:

| | | |
|---|---|---|
| FSACCESS | POSIT=595 | OTHER=ANY |
| | RACLIST=ALLOWED | MAXLNTH=44 |
| | GENLIST=DISALLOWED | DFTRETC=4 |
| | RACLREQ=YES | DFTUACC=NONE |
| | | SLBLREQ=NO |
| | OPER=NO | KEYQUAL=0 |
| | PROFDEF=YES | ID=1 |
| | FIRST=ANY | CASE=UPPER |
| | SIGNAL=YES | GENERIC=ALLOWED |

# Trademarks

IBM®, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml.

Other company, product, and service names may be trademarks or service marks of others.