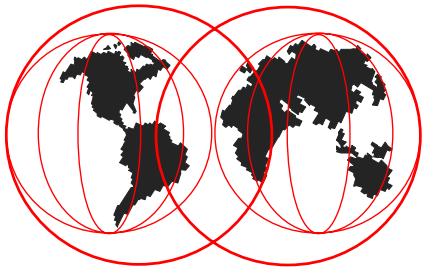
OS/390 UNIX security enhancements

Paul de Graaff ITSO Poughkeepsie S/390 Security



Mapping of the UIDs and GIDs

Agenda

- Introduction
- Assigning the UID and GID values
- Mapping to multiple user IDs and group names
- Performance considerations
- Summary

Introduction

The UNIXMAP mapping profiles are used to provide a cross-reference to RACF user and group profiles.

These profiles provide RACF with a performance-sensitive method of returning information for a given UID or GID when requested by OS/390 UNIX or application programs.

IBM Technical Support

- The mapping of the UID and GID was available already in OS/390 Version 2 Release 6.
- UNIX translates UID and GID information to RACF user IDs and group names when file information is listed.
- It is suggested that these UNIXMAP mapping profiles are used and UNIXMAP is always active when OS/390 UNIX environment is in production (see later Performance considerations).

Assigning the UID and GID values

- ADDUSER command with UID in the OMVS segment of a USER profile
  - creates a general resource profile U*uid* in the UNIXMAP class
- ADDGROUP command with GID in the OMVS segment of a GROUP profile
  - creates a general resource profile G*gid* in the UNIXMAP class
- RACF automatically maintains these profiles when changing or deleting
  - UIDs (ALTUSER, DELUSER) or GIDs (ALTGROUP, DELGROUP)
- Activate the UNIXMAP class after the mapping profiles are created

IBM Technical Support

- RACF automatically creates a general resource profile named U*uid* in the UNIXMAP class when you define UID in the OMVS segment of a USER profile. In the access list of the U*uid* profile is the RACF user ID that has been assigned this UID.
- Similarly when you define GID in the OMVS segment of a GROUP profile, RACF creates a general resource profile named G*gid* in the UNIXMAP class.
- RACF automatically maintains the mapping profiles
- UNIXMAP can be inactive when the UNIXMAP mapping profiles are added / updated.

Mapping to multiple user IDs and group names

- Assigning the same UID to multiple users
  - RACF creates / updates the general resource profile U*uid* in the UNIXMAP class with all the RACF user IDs in the access list.
  - May be necessary for some cases, such as superusers.
- The same GID to multiple RACF group names
- Default OMVS segments in user and group profiles
  - Migration aid to production
  - Test environment

IBM Technical Support

- This is possible, but not recommended, because the control at an individual user level is lost in OS/390 UNIX security checks.
- RACF maintains the general resource profile G*gid* with all the RACF group names in the access list. The RACF groups that have the same GID assignment are treated as a single group during OS/390 UNIX security checks.
- Use the Default OMVS segments in user and group profiles only as a conversion aid, not in production. If the Default OMVS segments are used, the UNIX is "open" to all users who have no OMVS segment.

Performance considerations

- When UID is translated to RACF user ID or GID to RACF group name
  - Search from VLF, if active (IRRUMAP / IRRGMAP classes)
    - If not active, must do I/O to RACF database
  - Search for the RACF profile, if UNIXMAP class is active
    - If not active, must search the entire RACF database
- Keep the VLF and UNIXMAP class active

IBM Technical Support

- VLF and the UNIXMAP class are used to map UIDs to RACF user IDs and GIDs to RACF group names. For RACF to begin using VLF for UID and GID mapping, you must define the IRRUMAP and IRRGMAP classes to VLF and VLF must be active.
- The UNIX file information listing needs to translate the owning UID and owning GID to a RACF user ID and the group name for the display.
- When the UNIXMAP class is populated, keep the class always active.
- Inactivate the VLF only when you need to make changes to it.

Summary

- The UNIXMAP mapping profiles are used to reduce the I/O to RACF database.
- RACF creates and maintains the UNIXMAP profiles automatically
- Avoid using same UID for the multiple RACF user IDs
- Keep always UNIXMAP class and VLF active to get the best performance in OS/390 UNIX

OS/390 UNIX user limits

Agenda

- Introduction
- Customer value
- Functional overview
- Using the function
- Summary

Introduction

- 2 types of OS/390 UNIX users
  - real person
  - server application
- OS/390 UNIX limits
  - set in BPXPRMxx parmlib member
  - "one size fits all"
- Superuser authority required to exceed limits

IBM Technical Support

- OS/390 UNIX has 2 types of users: a real person entering the system with TSO logon or rlogin from a UNIX workstation, and a user that is really a server. The server is supporting a large number of real users, and has different system requirements.
- Up until this release, limits specified in BPXPRMxx applied to all processes. If one user needed a particularly large region, lots of CPU time, 100's of processes or many threads, it was necessary to set the system limit high enough for the largest user.
- Only superusers can exceed these limits. But giving an application superuser authority gives them much more than just the ability to override these limits. It gives them authority to access any OS/390 UNIX resource, and use any of its functions.

Customer value

- System limits can be set to reasonable values
  - Improves system reliability
- Superuser authority can be set for those who need it
  - Improves system security
- Higher user limits can be set for those who need it
  - Improves usability

- With the system level limits currently available, the system programmer can set the system limits high enough to allow servers to run. This allows everyone to consume large amounts of system resources.
- The only alternative currently available is to set the system level limits to a reasonable level for the average user, and give the server applications superuser authority so they can exceed those limits. If the server isn't entirely trusted code, this will compromise the security of the entire system.
- Allowing limits to be set on an individual basis avoids these problems, and provides much more flexibility than is currently available

Functional Overview

- BPXPRMxx settings
  - MAXCPUTIME - cpu time
  - MAXASSIZE - address space region size
  - MAXFILEPROC - number of open files
  - MAXPROCUSER - number of processes
  - MAXTHREADS - number of threads
  - MAXMMAPAREA - amount of storage mapped by mmap
- Reset with SETOMVS or SET OMVS
- Exceeded by superusers

- BPXPRMxx contains the parameters that control OS/390 UNIX processing and the file system. User limits apply to these values:
- - the maximum amount of CPU time a process can receive, - the maximum address space region size for a process, - the maximum number of files a process can have open or active at the same time, - the maximum number of processes (address spaces) a UID can have active at the same time, - the maximum number of threads (tasks) a UID can have active at the same time, - the maximum amount of dataspace storage that can be used to map files in memory.
- These values, and many others, are set at IPL time, and can be changed by picking up a changed parmlib member (SET OMVS) or reset (SETOMVS). Most are not enforced for, or can be overridden by a superuser.

Functional Overview (continued)

- New keywords on ADDUSER allow user limits to be set in the OMVS segment of the user profile
  - CPUTIMEMAX(cpu-time)
  - ASSIZEMAX(address-space-size)
  - FILEPROCMAX(files-per-process)
  - PROCUSERMAX(processes-per-UID)
  - THREADSMAX(threads-per-process)
  - MMAPAREAMAX(memory-map-size)
- Added or deleted with ALTUSER and displayed by LISTUSER
- New keywords supported by callable service r_admin
- Limits not defined in OMVS segment are taken from BPXPRMxx.

IBM Technical Support

- New keywords have been added to the ADDUSER command to allow fields to be added to the user's OMVS segment. These correspond to the limits in the BPXPRMxx parmlib member, but apply only to this user ID.
- The format of the names has been changed slightly to allow them to be abbreviated more easily when the command is entered. The range of values allowed for each is the same as the corresponding system limit.
- ALTUSER has also been enhanced with these new keywords, and corresponding ones to remove the values. When a user limit is removed from a user profile, the system limits will again apply to that user.
- LISTUSER will display the user limits when the OMVS segment is requested.

Functional Overview (continued)

- User limits returned to caller by initUSP and initACEE
- IRRPOUSP macro maps OUSP
  - OUSP_VRSN 1 instead of 0
  - OUSP_FLAGS if version is 1
    - OUSP_LIMITSSET if on, user limits are defined
    - OUSP_LIMITSDONOTFIT if on, there's no room for the limits
  - OUSP_NUMLIMITS number of limits in array
  - OUSP_LIMITS limit array
    - OUSP_LIMITKEY key defining the type of limit
    - OUSP_LIMITVALUE value of the limit
- The user limits are set when process is dubbed

- The initUSP and the initACEE callable services return a control block, the OUSP, to OS/390 UNIX. It contains the UID, HOME and PROGRAM values from the user's OMVS segment. The user limits will also be returned in the OUSP.
- When the OUSP version is 1, a flag byte will follow the program path name indicating if there are user limits for this user. The second flag byte indicates if there was enough room for user limits in the OUSP. The OUSP has a fixed size (2074 bytes). If both HOME and PROGRAM values are 1023 bytes, there's no room for the user limits.
- OS/390 UNIX will use any values returned in place of the system limit for this user's process.

Using the function

```
ADDUSER BRUCE DFLTGRP(OGRP11)  OWNER(USERS)
   OMVS(UID(712)  HOME(/u/BRUCE)  FILEPROCMAX(200)
   PROGRAM(/u/BRUCE/bin/myshell)  ASSIZEMAX(62914560))
ALTUSER BRUCE OMVS(CPUTIMEMAX(2400)  NOFILEPROCMAX)
LISTUSER BRUCE OMVS NORACF

USER=BRUCE
OMVS INFORMATION
UID= 0000000712
HOME= /u/BRUCE
PROGRAM= /u/BRUCE/bin/myshell
CPUTIMEMAX= 0000002400
ASSIZEMAX= 0062914560
FILEPROCMAX= NONE
PROCUSERMAX= NONE
THREADSMAX= NONE
MMAPAREAMAX= NONE
```

IBM Technical Support

- The first example adds a new user with a UID of 712. The maximum address space size for BRUCE will be 60MB. BRUCE is allowed up to 200 open files.
- ALTUSER changes an existing OMVS segment. BRUCE is allowed 2400 seconds of CPU time, and the maximum number of open files deleted from the profile. System limits will now apply.
- LISTUSER displays the OMVS information. NONE indicates the value isn't set. The system limit applies.

Summary

- Setting limits on a user basis improves system reliability and security, as well as improving usability
- New fields in the user profile are set with new command keywords or with panels
- OS/390 UNIX uses these values when a process is dubbed
- Information on OS/390 UNIX limits can be found in the OS/390 UNIX System Services Planning manual

Protected user IDs

Agenda

- Purpose and Use
- Protected Userid Overview
- Creation of Protected Userid
- RACROUTE, RACINIT and ACEE
- Downlevel Impacts
- More information

Purpose and Use

Protected user IDs are a new type of userid designed for started tasks and daemons that can not be revoked by mistake or by a malicious user.

IBM Technical Support

- Installations run with RACF userids associated with critical tasks. It is possible for any person to submit JCL or pound on a TSO keyboard and cause any userid to be revoked by repetitively entering an incorrect password. This can interrupt the critical task since its userid is revoked.

Protected Userid Overview

- Protected user IDs are a new type.
- Cannot use where a password is required.
- Maintain support:
  - Can be revoked by command.
  - Can be revoked through inactivity.
  - Can propagate JCL
  - Can be used by surrogate JCL

IBM Technical Support

- Protected userids are a new type of userid.
- Protected userids cannot be used to logon to a system where a password is required. Protected userids cannot be revoked by repetitively specifying a password via JCL or logon (for example TSO).
- Protected userids can be controlled by the system administrators. They can revoke or resume a user via AU or ALU.
- Protected userids can become revoked through inactivity.
- Protected userids can be used to submit JCL and have the userid propagated. It can also be used in SURROGATE JCL.

Creation of Protected Userid

- Issue one of the following:
  - AU userid NOPASSWORD NOOIDCARD
  - ALU userid NOPASSWORD NOOIDCARD
- r_admin callable service supports NOPASSWORD
- Use the STARTED class to associate with started task or daemon
- Restart started task or daemon (after ALTUSER)
- RACF return code 8 is set, if password specified
  - revoke count not changed

IBM Technical Support

- The protected userid is defined by giving the userid both NOPASSWORD and NOOIDCARD attributes (which was not allowed previously).
- The r_admin callable service allows value NOPASSWORD via new table entry (call parameters).
- The STARTED class will associate the userid with a started task or daemon.
- If any attempt is made to logon to a protected userid with a password, RACF return code 8 (= The password is not authorized) will be returned without updating the revoke count.

RACINIT and ACEE

- Using RACROUTE REQUEST=VERIFY or RACINIT for the PROTECTED userid and
  PASSCHK =YES or not START= keyword
  - set Return code 8
  - Do NOT set revoke count
- If protected userid, allow caching
- ACEE has a new bit value in ACEEFLG3
  - ACEENPWR - protected userid indicator

IBM Technical Support

- If a RACROUTE REQUEST=VERIFY or RACINIT for the protected userid has a password and it's PASSCHK=YES or is not START= (name of the started task), then the password is required. In this case RACINIT sets return code 8 but does not set the revoke count.
- Return code 8 = The password is not authorized
- A protected userid is allowed to be cached in VLF for performance boost.
- IHAACEE macro - ACEE mapping - has a protected userid indicator called ACEENPWR

Downlevel Impacts

- Protected userid is regular pw only userid
  - userid can be revoked by logon/JCL
- LU does NOT show PROTECTED
- ALU inadvertently "undoes" protected user IDs
  - new default in V2R8

IBM Technical Support

- If the RACF Database is shared, protected user IDs may be used to attempt logon from systems running OS/390 releases prior to OS/390 V2R8.This may result in protected user IDs being revoked.
- Prior OS/390 V2R8 do not show PROTECTED attribute
- ALU command changing PASSWORD/NOPASSWORD and OIDCARD/NOOIDCARD operands will cause "losing" the PROTECTED attribute.
- Protected user default password in V2R8 is NOT DEFAULT-group

More information

- SC28-1913 RACF System Programmer's Guide
- SC28-1915 RACF System Administrator's Guide
- SC28-1919 RACF Command Language Reference

- This presentation will describe the new support in OS/390 Version 2 Release 8 that allows an administrator to define RACF profiles to limit SuperUser authorities granted to OS/390 UNIX users.

Agenda

- Why SuperUser Granularity?
- What is SuperUser Granularity?
- How To Implement SuperUser Granularity
  - Resource names for SuperUser Granularity
- Authority Checks for SuperUser Granularity
- _POSIX_CHOWN_RESTRICTED
  - Allowing chown for a user's own files

Why SuperUser Granularity?

- Ways to grant "appropriate privilege", that is SuperUser authority
  - UID 0
  - Access to BPX.SUPERUSER profile in the FACILITY class
- Ways to grant "partial" appropriate privilege
  - None today
  - V2R8 - new RACF profiles in the UNIXPRIV class
- **GOAL**: reduce the number of users who require full UNIX SuperUser authority

IBM Technical Support

- Many functions in the OS/390 UNIX environment require SuperUser authority, which is an "all or nothing" type of authority. In order for a user to perform any function which requires SuperUser authority, the user needs to have a UID of 0, or the user must have READ access to the BPX.SUPERUSER profile in the FACILITY class, which allows them to switch to UID of 0. Once a user has a UID of 0, that user can do *all* SuperUser functions.

- In previous releases, there was no way to grant "partial" SuperUser privilege. Version 2 Release 8 of OS/390 provides a way to use RACF profiles in the new UNIXPRIV class to grant a user partial SuperUser privileges.

What is SuperUser Granularity?

- Allows an installation to allow specific users to perform specific SuperUser functions
  - without giving full (or temporary) SuperUser authority
- Some SuperUser functions which can be individually granted in OS/390 V2R8:
  - Allow a user to read or write to any HFS file
  - Allow a user to send signals to any process
  - Allow a user to view all processes
  - Allow a user to mount and unmount file systems

IBM Technical Support

- Using the new support in V2R8, you can give individual users the authority to perform specific SuperUser functions, instead of giving users the authority to perform all SuperUser functions.
- For example, you can give a user just the authority to mount and unmount file systems, without giving them the authority to do other superuser functions.

How To Implement SuperUser Granularity

- Define profiles for OS/390 UNIX privileges in UNIXPRIV class
  - RACLIST is required (puts the profiles into a data space)

```
SETROPTS CLASSACT(UNIXPRIV) RACLIST(UNIXPRIV)
```

  - Resource name format:

```
SUPERUSER.function-group.function
```

  - Example: Allow BRUCE to issue mount and unmount

```
RDEFINE UNIXPRIV SUPERUSER.FILESYS.MOUNT UACC(NONE)

PERMIT SUPERUSER.FILESYS.MOUNT CLASS(UNIXPRIV)
ID(BRUCE) ACCESS(UPDATE)

SETROPTS RACLIST(UNIXPRIV)REFRESH
```

IBM Technical Support

- The UNIXPRIV class is new in V2R8, and is specifically for defining profiles to grant superuser authorities. For performance reasons, profiles in the class must be stored in a data space with the SETROPTS RACLIST command.

- The example shown will grant a user the authority to mount and unmount file systems, without allowing the user to perform other superuser functions.

UNIXPRIV Resource Names

HFS file and directory access

Resource Name	OS/390 UNIX Privilege	Access Required
SUPERUSER.FILESYS	Allows user to read any HFS file, and to read or search any HFS directory	READ
SUPERUSER.FILESYS	Allows user to write to any HFS file, and includes privileges of READ access	UPDATE
SUPERUSER.FILESYS	Allows user to write to any HFS directory, and includes privileges of UPDATE	CONTROL

- The UNIXPRIV resource which is used to grant a user the authority to manipulate the Hierarchical File System (HFS) is named SUPERUSER.FILESYS. There are different levels of access available, depending on the type of access the administrator wishes to grant.

UNIXPRIV Resource Names

Mount and Quiesce

Resource Name	OS/390 UNIX Privilege	Access Required
SUPERUSER.FILESYS.MOUNT	Allows user to mount and unmount a file system with the nosetuid option	READ
SUPERUSER.FILESYS.MOUNT	Allows user to mount and unmount a file system with the setuid option	UPDATE
SUPERUSER.FILESYS.QUIESCE	Allows user to quiesce and unquiesce a file system mounted with the nosetuid option	READ
SUPERUSER.FILESYS.QUIESCE	Allows user to quiesce and unquiesce a file system mounted with the setuid option	UPDATE

- A RACF administrator can use the UNIXPRIV resources shown here to authorize a user to manipulate file systems using the mount, unmount, quiesce, and unquiesce commands.

UNIXPRIV Resource Names

Other file system resources

Resource Name	OS/390 UNIX Privilege	Access Required
SUPERUSER.FILESYS.CHOWN	Allows user to change the ownership of any file using the chown service	READ
SUPERUSER.FILESYS.PFSCTL	Allows user to use the pfsctl() callable service	READ
SUPERUSER.FILESYS.VREGISTER	Allows user to issue the vreg() callable service to register as a VFS file server	READ

- Here are 3 more UNIXPRIV resources that can authorize a user to do file system operations.

- Generic profiles can be used to protect resources in the UNIXPRIV class also. For example, the resources on this page and the previous 2 pages can be protected by a single profile named:

  SUPERUSER.FILESYS.**

  if you wanted the same set of users to be authorized to perform all types of file system operations.

UNIXPRIV Resource Names

Process manipulation

Resource Name	OS/390 UNIX Privilege	Access Required
SUPERUSER.PROCESS.GETPSENT	Allows user to receive data for any process with the w_getpsent callable service	READ
SUPERUSER.PROCESS.KILL	Allows user to send signals to any process	READ
SUPERUSER.PROCESS.PTRACE	Allows user to trace any process through the dbx debugger * Also allows users to output information on all processes with the ps command	READ

* Authorization to the resource BPX.DEBUG in the FACILITY class is also required to trace processes that run with APF authority or BPX.SERVER authority.

- These 3 UNIXPRIV resources shown here can be used to grant authority to perform operations on OS/390 UNIX processes.

UNIXPRIV Resource Names

Miscellaneous resources

Resource Name	OS/390 UNIX Privilege	Access Required
SUPERUSER.IPC.RMID	Allows user to release IPC resources	READ
SUPERUSER.SETPRIORITY	Allows user to increase own priority	READ

- Here are 2 more UNIXPRIV resources.
- The first one is to authorize a user to manipulate InterProcess Communication (IPC) resources,
- The second is to authorize a user to change the priority of the user's own process.

When Do the New Profiles Get Checked?

- OS/390 UNIX currently invokes RACF callable services for authority checking
- New resource checks are done from existing RACF callable services using RACROUTE REQUEST=FASTAUTH

ck_access	ck_process_owner	R_ptrace
ck_owner_two_files	R_chown	
ck_priv	R_IPC_ctl	

- For example, the TSO MOUNT command calls RACF for a 'superuser' check via ck_priv
  - ck_priv does normal check for superuser
  - If normal superuser check fails, resource check is done for SUPERUSER.FILESYS.MOUNT profile

- The new authority checks occur within the RACF callable services. RACROUTE REQUEST=FASTAUTH calls are used to check authority to the UNIXPRIV resources.
- The publication "OS/390 Security Server (RACF) Callable Services" describes the authorization checks performed in the RACF callable services listed here.

More details on the Authority Checks

- New resource checks are done from RACF callable services using RACROUTE REQUEST=FASTAUTH
  - Global Access Table entries are ignored
  - FASTAUTH is called directly, bypassing the SAF and RACF routers.

- Exit Considerations for RACROUTE REQUEST=FASTAUTH calls for the UNIXPRIV class
  - SAF router exit (ICHRTX00) is not invoked
  - FASTAUTH pre-processing exit ICHRFX03 and post-processing exit ICHRFX04 are always called, if present, instead of ICHRFX01 and ICHRFX02.

IBM Technical Support

- For the UNIXPRIV authority check, there are some special considerations since the checks are done from within the RACF callable services. Global access table entries for the UNIXPRIV class will be bypassed. Also, the SAF and RACF routers are bypassed.

- As a result, the exits called for the RACROUTE REQUEST=FASTAUTH are a bit different than usual. ICHRFX03 and ICHRFX04 are normally called for cross-memory requests. These exits will always be called for the UNIXPRIV class, regardless of cross-memory implications. And the SAF router exit (ICHRTX00) is always bypassed.

UNIXPRIV Auditing Considerations

- Auditing Considerations for UNIXPRIV class
  - SETROPTS LOGOPTIONS settings are ignored
  - LOG=NOFAIL is specified on authorization checks - no audit records for failures, only successes.

    RALTER UNIXPRIV SUPERUSER.PROCESS.KILL AUDIT (SUCCESS(READ))
  - In some instances, you may see multiple audit records for the same event. This will only happen if your installation audits other OS/390 UNIX events for successful operations.
    - For example, a successful KILL operation could cause RACF to write an audit record for the PROCACT class and an audit record for the UNIXPRIV authority check.
- More information, see *OS/390 Security Server (RACF) Auditor's Guide* (SC28-1916)

IBM Technical Support

- There are several auditing considerations for the UNIXPRIV class. If you use SETROPTS LOGOPTIONS for the UNIXPRIV class, the settings will be ignored. This is because RACROUTE REQUEST=FASTAUTH invocations do not honor SETROPTS LOGOPTIONS settings. Also you can only audit successful accesses of UNIXPRIV resources.
- If you audit other successful OS/390 UNIX events with RACF classes such as PROCACT, FSOBJ, and IPCOBJ, you may see multiple records for the same operation if you also audit successes in the UNIXPRIV class.

Chown Enhancement

- _POSIX_CHOWN_RESTRICTED setting is in effect
  - On all releases prior to V2R8
  - Means you must have "appropriate privilege" to change ownership of a file
- In V2R8, you can change the system setting so that _POSIX_CHOWN_RESTRICTED is ***not*** in effect
  - Create the CHOWN.UNRESTRICTED profile in the UNIXPRIV class if you want to grant all users the ability to change ownership of their own files

IBM Technical Support

- There is one profile in the UNIXPRIV class which is not directly related to a superuser privilege. Instead it is used to set a system-wide option for the CHOWN function. (The CHOWN shell command and callable service are used to change the ownership of an HFS file or directory.)
- If you create the discrete profile CHOWN.UNRESTRICTED in the UNIXPRIV class, then all users can change ownership of their own HFS files. In other words, the owner of a file can change the owning UID of the file to any other UID, which essentially "gives the file away" to another user.
- In the POSIX standard, this corresponds to the setting of _POSIX_CHOWN_RESTRICTED being *not* in effect.

SuperUser Granularity Summary

- Authorize selected users to do selected SuperUser functions
  - Reduce the number of users who need UID 0 and BPX.SUPERUSER access
- Example: give user BRUCE the authority to mount and unmount any file system:

```
RDEFINE UNIXPRIV SUPERUSER.FILESYS.MOUNT UACC(NONE)
PERMIT SUPERUSER.FILESYS.MOUNT CLASS(UNIXPRIV)
 ID(BRUCE) ACCESS(UPDATE)
SETROPTS CLASSACT(UNIXPRIV) RACLIST(UNIXPRIV)
```

- Create the CHOWN.UNRESTRICTED profile in the UNIXPRIV class if you want to grant all users the ability to change ownership of their own files

IBM Technical Support

- The new SuperUser Granularity support in OS/390 V2R8 allows an administrator to selectively define which users can do which SuperUser functions. By using this support, an installation may be able to reduce the number of users who need to use UID 0 or have access to the BPX.SUPERUSER profile in the FACILITY class.
- This support also provides an enhancement to the CHOWN function. An installation can now choose to allow all users to be able to change the ownership of their own files.

More Information

- OS/390 UNIX System Services Planning SC28-1890
  - List of UNIXPRIV resources and their meanings
- OS/390 Security Server (RACF) Callable Services SC28-1921
  - Each updated RACF callable service has a table describing the UNIXPRIV class authority check

- These are the IBM publications which describe this support in more detail.

OS/390 UNIX MOUNT with NOSECURITY

IBM Technical Support

- This presentation will describe the RACF changes made for the new NOSECURITY keyword on the MOUNT command in V2R7.
- Note that this support was made available in support of V2R7, but the RACF APAR OW33566 is for V2R6, since RACF had no FMID for V2R7. This apar is rolled into the base of V2R8.

Agenda

- What is MOUNT NOSECURITY?
- What is a "System CRED"?
- What happens without the RACF support?
- Summary

What is MOUNT NOSECURITY?

- NOSECURITY is new keyword on OS/390 UNIX MOUNT command in OS/390 V2R7
- When a file system is mounted with NOSECURITY:
  - **Authority checks are not enforced** for files and directories within that file system
  - SETUID, SETGID, APF, and Program Control mode bits may be turned on, but they will not be honored while mounted with NOSECURITY
  - Auditing may still occur if the installation is auditing successes

IBM Technical Support

- In V2R7 the MOUNT command has a new keyword NOSECURITY. When a file system is mounted with the NOSECURITY keyword, authority checks are not enforced for files and directories within that file system. So any person with authority to the mount point has full access to the mounted file system. If the SETUID, SETGID, APF, and program control bits are turned on, they do not take effect until the file system is mounted with the SECURITY keyword.
- Since there will be no failed authority checks in the file system mounted with NOSECURITY, the only audit records that can be produced are those for successful file system operations.

What is a "System CRED"?

- OS/390 UNIX invokes the RACF callable services for authorization checks during file manipulation
  - Some authority checks are **done on behalf of the operating system**, instead of the end user
    - End user's ACEE, ACEX, and USP are not used, because they may not exist
  - The RACF callable services recognize "system" credentials by examining the CREDUTYPE field in the CRED structure
  - RACF treats a "system CRED" as a SuperUser, so most authority checks are granted
    - Audit records indicate OS/390 UNIX "system" authority was used

IBM Technical Support

- There are some OS/390 UNIX file system operations that are performed on behalf of the operating system instead of on behalf of the end user. In those cases, the 'identity' information sent to the RACF callable services include a "system CRED". This is the CRED data structure which has the CREDUTYPE set to indicate a system function caller.
- When RACF finds a system CRED, it treats the caller like a superuser, so most operations are allowed. The audit records, if any are created, indicate that system authority was used for the operation.

What Happens without RACF APAR?

- On OS/390 V2R7, if you don't install OW33566, System CRED is rejected on some RACF callable services.
  - May get strange results when using a file system mounted with the NOSECURITY keyword.

- If for some reason a customer is running V2R7 and has not installed the RACF APAR OW33566, then some strange things happen, if someone tries to use MOUNT NOSECURITY. For example, file creation (messages for failure) or *chaudit* command (SVC abend).

MOUNT NOSECURITY Summary

- When a file system is mounted with NOSECURITY:
  - Authority checks are "bypassed"
  - Internally, the authority checks are performed using "system" credentials, which allow the authority checks to succeed
- 4 RACF Callable Services were updated to accept a System CRED and treat it like SuperUser authority

- Here is a summary of what is covered in this presentation.