

# Changing UIDs and GIDs

Skill Level: Intermediate

[Kurt Riley \(Kurt\\_A\\_Riley@Whirlpool.com\)](mailto:Kurt_A_Riley@Whirlpool.com)  
Whirlpool Corporation

04 Mar 2008

It's important to know what happens to file ownership in AIX® once you make a UID or GID change. If you don't understand the results of altering a UID or GID, you could cause serious issues to your server and environment.

Changing user identification numbers (UIDs) and group identification numbers (GIDs) in the IBM® AIX® operating system (AIX) isn't one of the more exciting tasks a UNIX® administrator can face. But although it's often seen as a dreadful task, it can be an essential job that an administrator must perform to keep systems in sync within the environment. Because changing UIDs and GIDs can cause serious harm to your environment, you must be careful. The most important thing is understanding what your changes do. Then, you can learn how to make the changes correctly and even automate the process with UNIX scripts.

## UID and GID: some background

File ownership in AIX is determined by the UID, and group file permissions are determined by the GID. UIDs and GIDs are integers that range from 0 to about 65,535. (This number may differ depending on the UNIX version you're using.) Each username translates to one of these assigned integers. You can view the UID and primary GID of any account in UNIX by grepping the username from the `/etc/passwd` file:

```
$grep bduda /etc/passwd
bduda:!:300:350:Ben Duda:/home/bduda:/bin/ksh
```

The third field (300) is the UID, and the fourth field (350) is the GID of the primary group you're a member of. You can gather more information about the GID by

greping it from the `/etc/group` file:

```
$grep ":350:" /etc/group
security:!:350:bduda
```

As you can see, `bduda` has a primary group membership of `security`. Because `security` is `bduda`'s primary group, this group will be assigned to any files `bduda` creates.

## Choosing a UID and a GID

There are some basic rules for UID and GID number ranges. AIX system administrators select a range at which to begin allocating UIDs. UIDs and GIDs below 100 are typically reserved for system accounts and services. About 65,000 UIDs are available in AIX, so running out isn't an issue.

## Why change a UID or GID?

Sometimes you need to change a UID or GID because you're migrating servers or applications from one server to another. Other times, you need to make a change because an administrator has made an error. Environments that use AIX High-Availability Cluster Multiprocessing (HACMP) for clustering must always have the same UIDs and GIDs across all the clustered servers; otherwise, your failover process won't work correctly.

## What happens when you change a UID or GID?

It's important to understand that whenever you change a UID or GID, you affect the permission levels of files in AIX. Changing a UID or GID causes the ownership of all the files previously owned by that user or group to change to the actual integer of the file's previous owner.

## Change the UID

You can change a UID and/or a GID two ways. You can use `smitty`, but this example uses the command line. Here is the syntax:

```
Usage: usermod [ -u uid ] login
```

Let's change user bin's UID:

```
$ grep ^bin /etc/passwd
bin:!:2:2:/:bin:
$ usermod -u 5089 bin
$ grep ^bin /etc/passwd
bin:!:5089:2:/:bin:
```

By running the `usermod` command, you change the system account bin's UID from 2 to 5089. Keep in mind that every file owned by bin will have an ownership of 2, because AIX doesn't automatically change the file ownership to the user's new UID.

Here are the user's file permissions before the UID change

```
-rw----- 1 bin bin 29 2008-01-19 12:30 tester:
```

and after the UID change:

```
-rw----- 1 5089 bin 29 2008-01-19 12:30 tester
```

The user bin no longer has permissions to the file tester; you must change the file back to the owner bin. This is why changing UIDs can be a big task for an administrator.

## Change the GID

You saw how easy it is to change an account's UID -- and you saw one of the biggest problems with doing this. This section looks at the syntax for changing a GID using the command line. Changing a GID can be more complex:

```
Usage: chgroup "attr=value" ... group
```

Change the group's GID:

```
$grep bduda /etc/passwd
bduda:!:300:350:Ben Duda:/home/bduda:/bin/ksh
$ grep security /etc/group
security:!:350:bduda
$ chgroup "id=7013" security
3004-719 Warning: /usr/bin/chgroup does not update /etc/passwd with the new gid.
$ grep security /etc/group
security:!:7013:bduda
```

You get a warning message because the GID number in the `/etc/passwd` file doesn't change even though you changed the group's GID. Check to make sure:

```
$ grep bduda /etc/passwd
bduda:!:300:350:Ben Duda:/home/bduda:/bin/ksh
```

The `/etc/passwd` file says that `bduda` has a primary group of 350. However, the security group has a new GID of 7013. To fix this issue, you need to run the following command:

```
$ chuser "pgrp=security" bduda
$ grep bduda /etc/passwd
bduda:!:300:7013:Ben Duda:/home/bduda:/bin/ksh
```

**Note:** In this example, you need to use the `chuser` command for each user who has security as his primary group. Remember, when you change UIDs and GIDs in AIX, the permissions on the files and directories don't change: the files must be changed manually. If you have lots of files, or you don't know what all the files are, then this situation can be difficult to fix. The next section looks at some specific examples that show files before and after a change.

## Fix file permissions

To see what happens when a user owns files and you change that user's UID and GID, first create two new files called `File1` and `File2`.

### UID

Here are the properties of two example files:

```
$ls -l File*
-rw-r----- 1 bduda security 21 May 19 00:23 File1
-rw-r----- 1 bduda security 23 May 19 00:24 File2
```

The file's owner is `bduda`, and the group membership is `security`. Change the UID for `bduda`:

```
$usermod -u 34578 bduda
$grep bduda /etc/passwd
bduda:*:34578:7:tester:/tmp/bduda:/usr/bin/ksh
```

Now, look at the file permissions, because you changed the UID:

```
$ls -l File*
-rw-r----- 1 203      security    21 May 19 00:23 File1
-rw-r----- 1 203      security    23 May 19 00:24 File2
```

The owner of your two files is now the number 203 -- the previous UID for bduda. You have to change the file permissions back to the account bduda to fix these permissions:

```
$ chown bduda File*
$ ls -l File*
-rw-r----- 1 bduda    security    21 May 19 00:23 File1
-rw-r----- 1 bduda    security    23 May 19 00:24 File2
```

Can you imagine a user who has 40,000 files whose file permissions need to be changed?

## GID

Again, here are the properties of the two example files:

```
$ls -l File*
-rw-r----- 1 bduda    security    21 May 19 00:23 File1
-rw-r----- 1 bduda    security    23 May 19 00:24 File2
```

The group ownership is security. Change the GID for security:

```
$ chgroup "id=7013" security
$grep security /etc/group
security:!:7013:root,bduda
```

Now, look at the file permissions, because you changed the GID:

```
$ls -l File*
-rw-r----- 1 bduda    7            21 May 19 00:23 File1
-rw-r----- 1 bduda    7            23 May 19 00:24 File2
```

Your two files now have the group permission 7 -- the previous GID for the group security. You must change the file permissions back to the group security to fix these

permissions:

```
$chgrp security File*
$ls -l File*
-rw-r----- 1 bduda security 21 May 19 00:23 File1
-rw-r----- 1 bduda security 23 May 19 00:24 File2
```

Assume for a moment that you don't fix the permissions on your files. If a new user or group is created with the old UID 203 or the old GID 7, then this new user or group will become the owner and group of every file on the system that the user previously owned. This is bad for the system; plus, you've created serious security issues. The next section discusses how to examine your AIX systems to find out if there are any unowned files.

## Prepare to make the change: Scan your system for unowned files

It's a good idea to scan your systems for unowned files, and on AIX you can do so using some simple commands. To scan for files that have no user, run the following command from the command line:

```
find / \( -fstype jfs -o -fstype jfs2 \) -nouser -print
```

Depending on how your file systems are set up, this command checks the entire system for unowned files while skipping Network File System (NFS) mounts. You can do the same for groups:

```
find / \( -fstype jfs -o -fstype jfs2 \) -nogroup -print
```

Finding unowned files and groups is only half the battle: you also have to decide who should own them and what group permissions they should have. As a system administrator, this isn't the easiest thing to do. A good rule of thumb is to look at the directory owner or group. Then, set the permissions to match the directory. If you're still unsure what to do, then changing the values to the root user and root group isn't a bad idea, either.

## Keep UIDs and GIDs consistent in your environment

Most companies that run UNIX have more than one server. If you perform the same processes on multiple servers, it's a best practice to make sure the UIDs and GIDs

are the same across the enterprise. If you use some type of centralized user administration, then using IBM HACMP becomes much easier because your UIDs and GIDS are in sync.

Let's say you have an application that uses IBM DB2® Universal Database™. This is a mission-critical application running on two AIX HA pairs. These servers have file systems that are passed back and forth, depending on which server is primary. The account on your primary server that runs your DB2 database has a UID of 300.

Suppose that during a standard production cycle, your primary server crashes, and your secondary server picks up the workload. The account on your secondary server that runs your DB2 database has a UID of 400. This is a serious problem. The files on the primary file system were created with the DB2 account with UID 300. Because the file systems have failed over to the secondary server, the ownership is incorrect. The DB2 files aren't owned by the DB2 user with UID 400 -- they're owned by UID 300. The database doesn't own these files, so it won't function correctly, if at all.

Sometimes too many changes are required for you to make them manually. This is when scripting can aid you in your efforts.

## Scripting

No one wants to change 40,000 file owners, one at a time. It's a good thing you can write a script to find all the files for you -- and you can have it fix the permissions, as well.

Searching your entire file system can be very time consuming. Perl is a great tool that allows you to search an entire file system quickly. Perl comes with a command called `find2perl`, which lets you turn the regular AIX `find` command into Perl code. This code searches the file system faster than the regular UNIX `find` command:

```
$find2perl / \( -fstype jfs -o -fstype jfs2 \) -nouser -print > find_owner_script.pl
$find2perl / \( -fstype jfs -o -fstype jfs2 \) -nogroup -print > find_group_script.pl
```

You can use the regular `find` command if you don't have Perl on your system:

```
$find / \( -fstype jfs -o -fstype jfs2 \) -nouser -print > find_owner_script.txt
$find / \( -fstype jfs -o -fstype jfs2 \) -nogroup -print > find_group_script.txt
```

You now have a script automatically written in Perl. This script quickly finds all your unowned files and prints the output to the screen. If you wish, you can modify the script to write the output to a file.

Now, you need to determine what the file permissions should be and then change them, as shown in the following sections.

## Owner script

You can write the following on the command line or put the code into a script:

```
$ for file in $(cat output_from_find_owner_script.pl)
do
    print "Old permissions: $(ls -l $file)" >> /tmp/UID_LOG
    chown $new_owner $file
    print "New permissions: $(ls -l $file)" >> /tmp/UID_LOG
done
```

Here's the output:

```
Old permissions: -rw----- 1 485 bin      29 2008-01-19 12:30 tester
New permissions: -rw----- 1 bin  bin      29 2008-01-19 12:30 tester
Old permissions: -rw----- 1 987 bin      4098 2008-01-26 12:30 host
New permissions: -rw----- 1 bin  bin      4089 2008-01-26 12:30 host
```

## Group script

Now you can do something similar for the group:

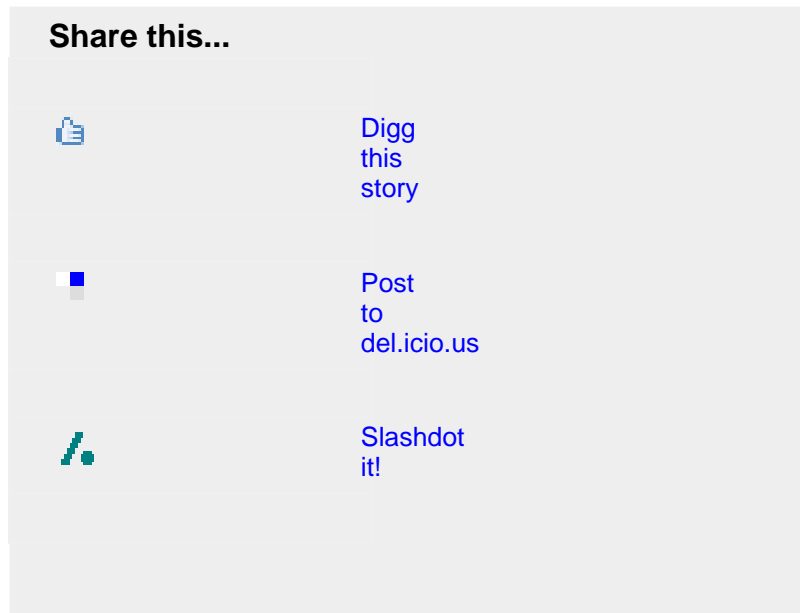
```
$for file in $(cat output_from_find_group_script.pl)
do
    print "Old permissions: $(ls -l $file)" >> /tmp/GID_LOG
    chgrp $new_group $file
    print "New permissions: $(ls -l $file)" >> /tmp/GID_LOG
done
```

Here's the output:

```
Old permissions: -rw----- 1 765 bin      29 2008-01-19 12:30 passwd
New permissions: -rw----- 1 root  bin      29 2008-01-19 12:30 passwd
Old permissions: -rw----- 1 983 bin      4098 2008-01-26 12:30 group
New permissions: -rw----- 1 root  bin      4089 2008-01-26 12:30 group
```

These examples create log files that record the file permissions before and after the change. These logs also provide you with proof that your script worked properly.

## Conclusion



Understanding how UIDs and GIDs work in UNIX can be confusing .. If you ever need to change these settings, you should fully understand how they work so you don't cause serious harm to your system. And with a little scripting, you can solve your UID and GID problems more quickly.

# Resources

## Learn

- When it comes to UNIX security, [Practical Unix & Internet Security](#) is really the only book you need. You can almost become an expert from just reading this book.
- The book [Security Warrior](#) complements *Practical Unix & Internet Security*. It provides hardcore concepts that will turn you into a subject-matter expert overnight.
- Browse the [technology bookstore](#) for books on these and other technical topics.
- [New to AIX and UNIX](#): Visit the New to AIX and UNIX page to learn more about AIX and UNIX.
- The [developerWorks AIX and UNIX zone](#) hosts hundreds of informative articles and introductory, intermediate, and advanced tutorials.
- [AIX Wiki](#): A collaborative environment for technical information related to AIX.

## Get products and technologies

- Download [IBM product evaluation versions](#) and get your hands on application development tools and middleware products from DB2®, Lotus®, Rational®, Tivoli, and WebSphere®.

## Discuss

- Participate in the AIX and UNIX forums:
  - [AIX 5L -- technical forum](#)
  - [AIX for Developers Forum](#)
  - [Cluster Systems Management](#)
  - [IBM Support Assistant](#)
  - [Performance Tools -- technical](#)
  - [Virtualization -- technical](#)
  - [More AIX and UNIX forums](#)
- Check out [developerWorks blogs](#) and get involved in the [developerWorks community](#).

## About the author

## Kurt Riley

Kurt Riley has been working with UNIX and Linux® systems since 2000. He's provided consulting services to a number of Fortune 500 companies. Kurt holds a bachelor's degree in Information Technology and currently works at Whirlpool Corporation supporting Tivoli Security products.