

Exam Report: 8.11.6 Practice Questions

Date: 4/27/2020 1:11:26 pm
Time Spent: 1:06

Candidate: Garsteck, Matthew
Login: mGarsteck

Overall Performance

Your Score: 25%

View results by: ☐ Objective Analysis ☒ Individual Responses

Individual Responses

▼ Question 1: Correct

You need to set the SUID permission on a file named *rider*. Which of the following commands will accomplish this task?

- ☐ **chmod rider 1777**
- ➡ ☒ **chmod 4777 rider**
- ☐ **chmod 1777 rider**
- ☐ **chmod 2777 rider**

Explanation

Use **chmod 4777 rider**. The SUID permission is numerically equivalent to 4000. This value is added to the other permissions, and **chmod** is used to apply it.

References

Linux Pro - 8.11 Special Permissions
[e_suid_lp5.exam.xml Q_OWNPERM_LP5_01]

▼ Question 2: Incorrect

A file named *acctg* needs to have the SGID permission set on it while keeping all other permissions at their present value.

Which of the following commands will accomplish this task?

- ☐ **chmod +g acctg**
- ☒ ~~**chmod s+g acctg**~~
- ➡ ☐ **chmod g+s acctg**
- ☐ **chmod 1777 acctg**

Explanation

Use the command **chmod g+s acctg** to add the SGID permission to the existing set of permissions. SGID has a numeric value of 2000 and would need to be added to the value of existing permissions if the numeric form of **chmod** is used.


References

Linux Pro - 8.11 Special Permissions
[e_suid_lp5.exam.xml Q_OWNPERM_LP5_02]

▼ Question 3: Incorrect

You have an application whose owner is root, but you want all users to execute the application with root user permissions.

Which of the following examples shows correct usage of the SUID flag?

-  ☐ -rwsr--r-- 3 root sys 73748 Nov 2 2005 /usr/bin/applicationx
- ☒ ~~-rwxr-sr-s 3 root sys 73748 Nov 2 2005 /usr/bin/applicationx~~
- ☐ -rwxr--r-s 3 root sys 73748 Nov 2 2005 /usr/bin/applicationx
- ☐ -rwxr-sr-- 3 root sys 73748 Nov 2 2005 /usr/bin/applicationx

Explanation

Placing the **s** in the execute bit location for the owner's permissions, as shown in **-rwsr--r--**, means that anyone or any process that executes the application will do so with root user permissions.

References


Linux Pro - 8.11 Special Permissions

[e_suid_lp5.exam.xml Q_OWNPERM_LP5_03]

▼ Question 4: Incorrect

You have an application whose group is sys, but you want all users, whether or not they are members of the *sysadmin* group, to execute the application with *sysadmin* group permissions.

Which of the following examples shows correct usage of the SGID flag?

-  ☐ -rwxr-sr-- 3 root sysadmin 73748 Nov 2 2005 /usr/bin/applicationx
- ☒ ~~-rwxr-r-s 3 root sysadmin 73748 Nov 2 2005 /usr/bin/applicationx~~
- ☐ -rwxr-sr-s 3 root sysadmin 73748 Nov 2 2005 /usr/bin/applicationx
- ☐ -rwsr--r-- 3 root sysadmin 73748 Nov 2 2005 /usr/bin/applicationx

Explanation

Placing the **s** bit in the execute location for the group's permissions, as shown in **-rwxr-sr--**, means that anyone or any process that executes the application will do so with the *sysadmin*'s group permissions.

References


Linux Pro - 8.11 Special Permissions

[e_suid_lp5.exam.xml Q_OWNPERM_LP5_04]

▼ Question 5: Correct

Setting the SUID flag is a powerful and useful feature. It can have weaknesses associated with its use.

Which of the following statements identifies a weakness?

- ☐ Setting the SUID flag is a one-time event that reverts after execution, forcing the root user to reset the flag every time.
- ☐ There are no weaknesses to using the SUID flag.
-  ☒ Setting the SUID flag for an application or process owned by the root user is a potential security hole.
- ☐ Only twelve applications or processes may be run at the same time by setting the SUID flag for root user permissions.

Explanation

Setting the SUID flag for programs owned by root is dangerous because that program executes with root user privileges. There are no limits on the use of the SUID flag, and it is not reset upon use.

References

Linux Pro - 8.11 Special Permissions

[e_suid_lp5.exam.xml Q_OWNPERM_LP5_05]

▼ Question 6:

Incorrect

Which of the following special permissions prevents a file's deletion by anyone except the file's owner?

- ☐ Umask
- ☐ SGID
- ☒ ~~SUID~~
- ➡ ☐ Sticky bit

Explanation

The sticky bit marks the file (not directory) to prevent the file's deletion by anyone except the file owner.

SUID (Set User ID) allows a program to run with the permissions of the file owner, not with the permissions of the user who runs the program. SGID allows a program to run with the permissions of the group owner. If SGID is set on a directory, a newly created file will receive the same group owner as assigned to the parent directory. Umask changes (removes) the default file and directory permissions. By default, files receive rw-rw-rw- (666) permissions when they are created, and directories receive rwxrwxrwx (777) permissions when they are created.

References

Linux Pro - 8.11 Special Permissions

[e_suid_lp5.exam.xml Q_OWNPERM_LP5_06]

▼ Question 7:

Incorrect

You have just installed a new program that will be used by all users on the computer. After installation, you log in as a regular user, and you try to run the program. The program stops and displays the following error:

Error: not running as root

What should you do so that all users can run this program?

- ☐ Make all users members of the root group.
- ☒ ~~Add users to the sudoers file. Tell them to run the program using the **sudo** command.~~
- ➡ ☐ Set the SUID on the program.
- ☐ Have users run the program using the **su -c** command.

Explanation

The program must be run as the root user. The easiest method for resolving the problem is to set the SUID on the program file. When the program runs, it will run with the permissions of the root user.

Using the **sudo** command will require extra work and user training. Using the **su -c** command means that users must know the root user password. Making users members of the root group might not work and would give them extra permissions to things other than the program file.

References

Linux Pro - 8.11 Special Permissions

[e_suid_lp5.exam.xml Q_OWNPERM_LP5_07]

▼ Question 8:

Incorrect

You are the team lead for a marketing project. Only the members of the team3 group should have read/write permissions to a document called *project_data.doc*. You want to ensure that no one on the team will delete the document by accident.

Your user name is bsmith.

A long listing of the file is as follows:

```
-rwxr-xr-x 1 bsmith team3 7260 Jun 22 2004 project_data.doc
```

Which of the following commands will accomplish this task?

☐ **chown :bsmith project_data.doc**

☒ **~~chmod 744 project_data.doc~~**

☐ **chgrp bsmith project_data.doc**

➡ ☐ **chmod 1664 project_data.doc**

Explanation

chmod 1664 project_data.doc is the correct answer because you are setting the sticky bit. When the sticky bit is set on a file, that file can only be deleted by its user owner no matter what permissions have been assigned to it, even if it were assigned permissions of 777 (rwxrwxrwx).

POSIX file permissions can be displayed in a number of ways. The traditional method is rwxrwxrwx. With this method, *r* means read, *w* means write, and *x* means execute. These permissions are displayed for the user owner (first set), group owner (second set) and the world, or everyone (third set).

POSIX permissions can also be converted into binary using three bits, one for read, one for write, and one for execute. In this notation, rwxrwxrwx would be 11111111. Displaying the permissions in binary is a bit long-winded, so binary permissions are often converted into octal numbers (digits 0-7). In octal, rwxrwxrwx would be represented as 777(111 binary = 4+2+1 = 7 octal).

A fourth group of rwx (111 binary or 7 octal) can be appended to the beginning of the permissions to set SUID (100 binary or 4 octal), SGID (010 binary or 2 octal) and/or the sticky bit (001 binary or 1 octal). This is represented as 0777 octal if none of these are set. If rwx notation is used, SUID is rwsrwxrwx, SGID is rwxrwsrwx, and the sticky bit is rwxrwxrwt.

chmod 744 project_data.doc is incorrect because only the user owner would have write permissions. **chgrp bsmith project_data.doc** is incorrect because it changes the group owner to your personal group. If this were done, you would have to give the world (everyone) read/write permissions to allow team3 members to have read/write permissions. **chown :bsmith project_data.doc** would have the same effect and could only be run as root because only the root user can run chmod.

References

Linux Pro - 8.11 Special Permissions

[e_suid_lp5.exam.xml Q_OWNPERM_LP5_08]