BRAINWARE UNIVERSITY

BNCSC202 CLASS NOTES Linux Administration-I

Module III: Permission and Advance Permission

Part III

Manage Access Control List permissions in RHEL elaborately with examples:

In Red Hat Enterprise Linux (RHEL), Access Control Lists (ACLs) provide a way to manage permissions for files and directories beyond the traditional owner, group, and other permissions. They allow you to specify permissions for specific users or groups on a file or directory.

Here is an example of how to manage ACL permissions in RHEL elaborately:

1. Install the **acl** package:

   ```
   yum install acl
   ```

2. Enable ACL support on the file system by adding the "acl" option to the mount options in the /etc/fstab file. For example, to enable ACL support on the "/data" file system:

   ```
   /dev/sda1 /data ext4 defaults,acl 1 2
   ```

3. Create a new file or directory:

   ```
   touch myfile.txt
   ```

4. Use the **setfacl** command to set ACL permissions. For example, to give the user "john" read and execute permissions on the file "myfile.txt":

   ```
   setfacl -m u:john:r-x myfile.txt
   ```

5. To set ACL permissions for a group, use the following command:

   ```
   setfacl -m g:groupname:rwx myfile.txt
   ```

6. To view the ACL permissions on a file, use the **getfacl** command:

   ```
   getfacl myfile.txt
   ```

7. You will see the output showing the owner, group, and ACL permissions for the file. For example:

   ```
   # file: myfile.txt # owner: root # group: root user::rw- user:john:r-x group::r-x mask::r-x other::r--
   ```

8. To remove ACL permissions from a file, use the **setfacl** command with the **-x** option. For example, to remove the ACL permissions for the user "john" on the file:

   ```
   setfacl -x u:john myfile.txt
   ```

9. You can also modify the default ACL for a directory, which will apply to all newly created files and subdirectories within the directory. To set the default ACL for the directory "mydir", use the following command:

   ```
   setfacl -m d:u:john:rwx mydir/
   ```

10. To view the default ACL for a directory, use the **getfacl** command and add the "-d" option:

    ```
    getfacl -d mydir/
    ```

11. To remove the default ACL for a directory, use the **setfacl** command with the **-b** option:

    ```
    setfacl -b mydir/
    ```

12. These are just some of the ways you can use ACLs to manage permissions on files and directories in RHEL. It's important to understand the security implications of using ACLs, so be sure to use them with caution and only when necessary.

Sometimes, you may want to apply such ACLs to all files in a directory. In that case, the -R switch can be used to apply changes recursively; for example, the following command allows user michael to have read and execute permissions on all files in the /home/examprep directory as well as any subdirectories that may exist:

```
# setfacl -R -m u:michael:r-x /home/examprep
```

There are two methods available to unset these options. First, you could apply the -x switch to the previous command, omitting the permission settings:

```
# setfacl -R -x u:michael /home/examprep
```

Alternatively, you could use the -b switch; however, that would erase the ACLs configured for all users on the noted directory (and with the -R switch, applicable subdirectories):

```
# setfacl -R -b /home/examprep
```

Concepts on umask

Alternatively, you could use the -b switch; however, that would erase the ACLs configured for all users on the noted directory (and with the -R switch, applicable subdirectories):

```
# setfacl -R -b /home/examprep
```

umask is a command in Unix-like operating systems that sets the default file mode creation mask. The file mode creation mask is a setting that determines the default permissions for newly created files. The **umask** value is subtracted from the maximum file permissions (typically **666** for files and **777** for directories) to determine the default permissions for newly created files.

For example, if the **umask** value is **022**, the default permissions for a newly created file would be **644** (**666** - **022**), and the default permissions for a newly created directory would be **755** (**777** - **022**).

The **umask** value can be set for an individual user in their shell profile, and it is applied each time a new file is created. The **umask** value can also be temporarily changed for a single session using the **umask** command.

The **umask** value is important because it provides a way to set default permissions for newly created files, making it easier to ensure that sensitive files are properly protected. Additionally, the **umask** value can be used to enforce a standard file permission policy across an entire system, making it easier to maintain secure and consistent file permissions.