Linux File Permissions

Abraham J. Reines

February 25, 2024

Introduction

In order to verify the answers of our project on Linux File Permissions, multiple bash scripts were written. The scripts are found in the *'Script Listings'* section at the bottom of this report. The output of the script `assignment_solutions.sh` can be found at the top of sections 3.4.1, 3.4.2, 3.4.3 of this report. The scripts are intended to be submitted along side of this PDF.

3.4.1 Understanding Linux file permissions

Output of Solutions for 3.4.1:

```
3.4.1 Solutions:
--------------------------------
Question 1:
yes - Alice can read the file /cs/home/stu/bob/data.txt because the 'others'
    ↪  permissions on the file allow read access.
Question 2:
no - Alice cannot remove the file /cs/home/stu/bob/data.txt because she
    ↪ lacks write permissions.
Question 3:
no - Alice cannot read the file /cs/home/stu/bob/secret.txt because she
    ↪ lacks read permissions.
Question 4:
no - Alice cannot remove the file /cs/home/stu/bob/secret.txt because she
    ↪ lacks write permissions.
Question 5:
The full permissions for the new file /cs/home/stu/bob/mysecret2.txt are:
-rw-r-sr-- 1 bob faculty 0 Feb 22 16:48 /cs/home/stu/bob/mysecret2.txt
```

Listing 1: Solutions to 3.4.1.

1. **Can Alice read Bob's file `data.txt`? Why?**
   Yes, the file `data.txt` permissions are set as `-rw-r--r-`, which grants read access to all users including Alice.

2. **Can Alice remove Bob's file `data.txt`? Why?**
   No, Alice lacks write permission on Bob's home directory `/cs/home/stu/bob`, which is required to delete files.

3. **Can Alice read Bob's file `secret.txt`? Why?**
   No, the file `secret.txt` has permissions `-rw-r---`, which restricts read access to the owner and group 'faculty'. Since Alice is not a member of the 'faculty' group, she does not have read access.

4. **Can Alice remove Bob's file `secret.txt`? Why?**
   No, Alice cannot delete `secret.txt` due to the absence of write permission in Bob's home directory.

5. **When Bob creates a new file with the command `echo "My Super Secret is b7d5d78shes" > mysecret.txt`, what are the full permissions of this file?**
   The full permissions for the new file `/cs/home/stu/bob/mysecret2.txt` are: `-rw-r-sr-- 1 bob faculty 0 Feb 22 16:48 /cs/home/stu/bob/mysecret2.txt`

3.4.2 Setting Linux file permissions

Output of Solutions for 3.4.2:

```
--------------------------------
3.4.2 Solutions:
--------------------------------
Yes, Bob can change the permissions. The commands used are:
chgrp csmajor /cs/home/stu/bob/data.txt
chmod 640 /cs/home/stu/bob/data.txt
Bob should use the command: umask 002 to set default file permissions.
Created new file /cs/home/stu/bob/newfile.txt with default permissions.
Changed ownership of /cs/home/stu/bob/newfile.txt to bob:csmajor.
Permissions for new file /cs/home/stu/bob/newfile.txt:
-rw-rw-r-- 1 bob csmajor 0 Feb 23 10:45 /cs/home/stu/bob/newfile.txt
Final Permissions for /cs/home/stu/bob/data.txt:
-rw-r----- 1 bob csmajor 0 Feb 23 10:45 /cs/home/stu/bob/data.txt
Permissions for Bob's home directory:
drwxr-s--- 2 bob faculty 4096 Feb 23 10:42 /cs/home/stu/bob
```

Listing 2: Solutions to 3.4.2.

1. **Can Bob change the permissions so that all other students in `csmajor` can read `data.txt`, but any other users who are not in `csmajor` cannot?**
   Yes, by executing `chmod 640 /cs/home/stu/bob/data.txt`, Bob sets the file `data.txt` to be readable and writable by the owner and only readable by the group.
2. **If Bob wants to set the default permission of his new files to be readable and writable by himself and the group, and readable by others, what commands should he use? Hint: use umask.**
   Bob should issue the command `umask 002` to ensure files are created with `rw-rw-r--` permissions and directories with `drwxr-s--`.

Bob wants to set the default permissions for his new files to be:

- Non-executable by himself (user)
- Readable and writable but not executable by the group
- Readable but not writable or executable by others

Given the most permissive default permissions for files are `666` (`rw-rw-rw-`), the `umask` subtracts from these defaults to determine the actual default permissions for new files and directories.

The target permissions for files should therefore be:

- `rw` for the user: `6` in octal (readable, writable, non-executable)
- `rw-` for the group: `6` in octal (readable, writable)
- `r-` for others: `4` in octal (readable)

Given the `umask` is subtracted from default permissions, we want a `umask` to result in `664` for files. We want a `umask` to results in `664` permissions for files:

- User: `6` (`rw-`)
- Group: `6` (`rw-`)
- Others: `4` (`r-`)

Subtract these from the default `666`:

- For user: `6` (default) - `6` (desired) = `0`
- For group: `6` (default) - `6` (desired) = `0`
- For others: `6` (default) - `4` (desired) = `2`

The `umask` should therefore be `002`, ensuring files are created with `664` permission.So, the correct `umask` command for Bob's scenario would be:

```
umask 002
```

3.4.3 A more complex case

Please see `assignment_solutions.sh` (specifically, lines *114-185*) for the proper commands to accomplish Alice's desired effect. This is listed in the *Script Listings* section. The following are commands for running the scripts:

```
sudo chmod +x setup_assignment.sh
sudo ./setup_assignment.sh
sudo chmod +x assignment_solutions_v2. sh
sudo ./assignment_solutions_v2.sh > solutions_output.txt
```

Alice should verify these settings in her environment to ensure they are correct. Should she face permission-related issues, the system administrator's intervention may be required. She will most likely need to use sudo. If Alice can request sudo privileges for specific commands, she can ask the administrator to add Bob to the group or do it herself if granted those privileges.

Output of Solutions for 3.4.3:

```
--------------------------------
3.4.3 Solutions:
--------------------------------
Alice's home directory already exists.
Removed unauthorized copies of treasure.txt within the home directory.
File /home/alice/treasure.txt is ready.
Removed immutable attribute from /home/alice/treasure.txt.
Changed file ownership to Alice.
Changed file permissions to read/write for (Alice), no permissions for
    ↪ others.
Changed the file group to 'treasure_group' and set group permissions to read
    ↪ /write.
Set ACL for 'charlie' to read.
Final permissions for /home/alice/treasure.txt:
-rw-rw----+ 1 alice treasure_group 0 Feb 22 11:06 /home/alice/treasure.txt
# file: home/alice/treasure.txt
# owner: alice
# group: treasure_group
user::rw-
user:charlie:r--
group::rw-
mask::rw-
other::---

treasure_group:x:1006:bob
```

Listing 3: Solutions to 3.4.3.

The requirements for `treasure.txt` are specific, so let's assess the given solution for each requirement:

1. There must be only a single copy of the file in the system:
   - The provided output confirms all copies removed from the system, with a line says "Removed unauthorized copies of `treasure.txt` within the home directory." The script was designed to remove all `treasure.txt` files from the system.
2. Alice's own permissions must be readable/writable/non-executable:
   - The final permissions `rw-` for Alice suggests she has read and write access, as required.
3. Alice needs to allow Bob to read and write the file:
   - Bob is added to the `treasure_group` group, which has read/write permissions on the file (`rw-`). This satisfies the requirement.
4. Alice needs to allow Charlie to read but not write the file:

- An Access Control List (ACL) entry for Charlie with read permissions (`r-`) is set. This satisfies the requirement.

5. The execute permission must be disabled for all users:
   - The final permissions do not include execute permissions for anyone, satisfying this requirement.
6. All other users (such as Dave) in the system must not be able to read or write the file:
   - The `other` permissions are set to `--`, which means no access for users other than the owner and group, fulfilling this requirement.
7. Alice does not have sudo privilege, but she can ask the systems administrator for help if necessary:
   - Alice would most likely need sudo permissions and administrator help/intervention.

Script Listings

```
1  echo "--------------------------------"
2  echo "3.4.3 Solutions: "
3  echo "--------------------------------"
4
5  sudo setfacl -m u:charlie:r-- /home/alice/treasure.txt
6
7  # creating Alice's home directory unless it exists
8  Alice_needs_a_home...() {
9      local Alices_home="/home/alice"
10
11     # Create Alice's home directory if it doesn't exist
12     if [[ ! -d "$Alices_home" ]]; then
13         echo "Alice's home directory does not exist. Creating the directory.
               ↪ "
14         mkdir -p "$Alices_home"
15         # WARNING: assumes Alice has permission to create her home directory
16     else
17         echo "Alice's home directory already exists."
18     fi
19 }
20
21 # locating and removing possible duplicate copies of the file
22 delete_treasures() {
23     local whats_my_name="treasure.txt"
24     local Alices_home="/home/alice"
25
26     # remove duplicates
27     find "$Alices_home" -type f -name "$whats_my_name" ! -path "$Alices_home
           ↪ /$whats_my_name" -exec rm {} \; 2>/dev/null && \
28     echo "Removed copies of $whats_my_name within the home directory."
29 }
30
31 # configure file with permissions and ACL
32 treasure_needs_permissions() {
33     local file="/home/alice/treasure.txt"
34
35     # Does the file even exist? If not, create it
36     if [[ ! -f "$file" ]]; then
37         echo "File $file not found. Creating the file."
38         touch "$file"
39     fi
40     echo "File $file is ready."
41
42     # Remove immutable attribute if necessary
```

```
43      if lsattr "$file" 2>/dev/null | grep -q 'i'; then
44          chattr -i "$file"
45          echo "Removed immutable attribute from $file."
46      fi
47
48      chown alice:alice "$file" && echo "Changed file ownership to Alice."
49
50      chmod 600 "$file" && echo "Changed file permissions to read/write for
            ↪ owner (Alice), no permissions for others."
51
52      # Does the treasure_group even exist? if not, create it
53      if ! getent group treasure_group &>/dev/null; then
54          groupadd treasure_group && echo "Group 'treasure_group' created."
55      fi
56
57      chgrp treasure_group "$file" && chmod 660 "$file" && \
58      echo "Changed the file group to 'treasure_group' and set group
            ↪ permissions to read/write."
59
60      setfacl -m u:charlie:r-- "$file" && echo "Set ACL for 'charlie' to read
            ↪ only."
61
62      # Show requested permissions and ACL
63      echo "Final permissions for $file:"
64      ls -l "$file"
65      getfacl "$file" 2>/dev/null || echo "ACL not supported on this system or
            ↪  not present."
66  }
67
68  Alice_needs_a_home...
69  delete_treasures
70  treasure_needs_permissions
71
72  getent group treasure_group
```

Listing 4: assignment_solutions.sh for printing assignment solutions.

Academic Integrity Pledge

"This work complies with the JMU honor code. I did not give or receive unauthorized help on this assignment."