Linux File Permissions

Abraham J. Reines

February 23, 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.

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--`.

3.4.3 A more complex case

Please see lines *114-185* of `assignment_solutions.sh` for the proper commands to accomplish Alice's desired effect. 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.

Script Listings

```bash
#!/bin/bash

# File name: setup_assignment.sh
# Author: Abraham Reines
# Date: February 21, 2024

# This script complies with the requirements. Should produce output
    ↪ consistent with assignment when executed in appropriate environment.

usermod -a -G faculty bob

gpasswd -d bob bobsgroup &>/dev/null

usermod -a -G csmajor alice
usermod -a -G csmajor bob

# Set up the directory permissions
chmod 755 /cs
chmod 755 /cs/home
chmod 2755 /cs/home/stu

chown bob:faculty /cs/home/stu/bob
chmod 2750 /cs/home/stu/bob

touch /cs/home/stu/bob/data.txt
touch /cs/home/stu/bob/secret.txt
chown bob:faculty /cs/home/stu/bob/data.txt
chown bob:faculty /cs/home/stu/bob/secret.txt
chmod 644 /cs/home/stu/bob/data.txt
chmod 600 /cs/home/stu/bob/secret.txt

setfacl -m u:alice:r-- /cs/home/stu/bob/data.txt

echo "Initial Permissions:"
echo "--------------------------------"
echo "Directory permissions:"
ls -ld /cs /cs/home /cs/home/stu /cs/home/stu/bob
echo "--------------------------------"
echo "File permissions in Bob's home directory:"
ls -l /cs/home/stu/bob/data.txt /cs/home/stu/bob/secret.txt
```

```
echo "--------------------------------"
echo "ACL for data.txt:"
getfacl /cs/home/stu/bob/data.txt
echo "--------------------------------"
echo "Current users and groups:"
groups bob
groups alice
echo "--------------------------------"
echo "Setup complete. Users, groups, permissions, and ACL are all set as per
    ↪ the assignment scenario."
```

Listing 4: Initial setup with setup_assignement.sh.

```
#!/bin/bash

# File name: assignment_solutions_v2.sh
# Author: Abraham Reines
# Date: February 21, 2024
# Modified: February 22, 2024

# check read permission for Alice
Can_she_read?() {
  if [ -r "$1" ]; then
    # If file is readable, check if the 'alice' user is owner or part of
      ↪ group with read permission.
    owner=$(ls -l "$1" | awk '{print $3}')
    group=$(ls -l "$1" | awk '{print $4}')
    if [ "$owner" == "alice" ] || [ "$group" == "csmajor" ] || [ "$(ls -l "
      ↪ $1" | cut -c8)" == "r" ]; then
      echo "yes - Alice can read the file $1 because the 'others'
        ↪ permissions on the file allow read access."
    else
      echo "no - Alice cannot read the file $1 because she lacks read
        ↪ permissions."
    fi
  else
    echo "no - Alice cannot read the file $1 because the file does not exist
      ↪  or is not readable."
  fi
}

# check remove permission for Alice
Can_she_remove?() {
  if sudo -u alice test -w "$1"; then
    echo "yes - Alice can remove the file $1 because she has write
      ↪ permissions."
  else
    echo "no - Alice cannot remove the file $1 because she lacks write
      ↪ permissions."
  fi
}

# simulate the creation of a new file by Bob with specific permissions
Bobs_new_file() {
  sudo -u bob touch "$1"
  sudo -u bob chmod g+s "$1"
  echo "My Super Secret is b7sd78shes" > mysecret2.txt
  echo "The full permissions for the new file $1 are:"
  ls -l "$1"
}
echo
```

```
echo "3.4.1 Solutions:"
echo "--------------------------------"
# Check permissions for Alice
echo "Question 1:"
Can_she_read? "/cs/home/stu/bob/data.txt"

echo "Question 2:"
Can_she_remove? "/cs/home/stu/bob/data.txt"

echo "Question 3:"
Can_she_read? "/cs/home/stu/bob/secret.txt"

echo "Question 4:"
Can_she_remove? "/cs/home/stu/bob/secret.txt"

# Create a new file with permissions and check
echo "Question 5:"
Bobs_new_file "/cs/home/stu/bob/mysecret2.txt"
echo
echo "--------------------------------"
echo "3.4.2 Solutions: "
echo "--------------------------------"

# Define file paths
DATA="/cs/home/stu/bob/data.txt"
bobs_home="/cs/home/stu/bob"

# set/verify the SGID bit on Bobs home directory
set_and_verify_sgid() {
  chmod g+s "$bobs_home"
  # Check SGID bit
  if [ "$(ls -ld "$bobs_home" | cut -c6)" == "s" ]; then
    echo "SGID bit is set on $bobs_home."
  else
    echo "Failed to set SGID bit on $bobs_home."
  fi
}

# check if Bob can change file permissions like in question 1
can_bob_change_permissions() {
  # Check if Bob can write to his home directory and change permissions of
     ↪ data.txt
  if [ -w "$bobs_home" ] && [ -w "$DATA" ]; then
    chgrp csmajor "$DATA"
    chmod 640 "$DATA"
    echo "Yes, Bob can change the permissions. The commands used are:"
    echo "chgrp csmajor $DATA"
    echo "chmod 640 $DATA"
  else
    echo "No, Bob cannot change the permissions as he does not have write
       ↪ access to the directory or file."
  fi
}

# determine the commands for file permissions like in question 2
default_perms() {
  umask 002
  echo "Bob should use the command: umask 002 to set default file
     ↪ permissions."

  # Create a new file
  touch "$NEW_FILE"
  echo "Created new file $NEW_FILE with default permissions."

  # Change ownership to bob:csmajor
  chown bob:csmajor "$NEW_FILE"
  echo "Changed ownership of $NEW_FILE to bob:csmajor."
}

# set_and_verify_sgid
can_bob_change_permissions
default_perms

echo "Final Permissions:"
ls -l "$DATA"

echo "Permissions for Bob's home directory:"
ls -ld "$bobs_home"
echo
echo "--------------------------------"
echo "3.4.3 Solutions: "
echo "--------------------------------"

sudo setfacl -m u:charlie:r-- /home/alice/treasure.txt

# creating Alice's home directory unless it exists
Alice_needs_a_home...() {
    local Alices_home="/home/alice"

    # Create Alice's home directory if it doesn't exist
    if [[ ! -d "$Alices_home" ]]; then
        echo "Alice's home directory does not exist. Creating the directory.
            ↪ "
        mkdir -p "$Alices_home"
        # WARNING: assumes Alice has permission to create her home directory
    else
        echo "Alice's home directory already exists."
    fi
}

# locating and removing possible duplicate copies of the file
delete_treasures() {
    local whats_my_name="treasure.txt"
    local Alices_home="/home/alice"

    # remove duplicates
    find "$Alices_home" -type f -name "$whats_my_name" ! -path "$Alices_home
        ↪ /$whats_my_name" -exec rm {} \; 2>/dev/null && \
    echo "Removed copies of $whats_my_name within the home directory."
}

# configure file with permissions and ACL
treasure_needs_permissions() {
    local file="/home/alice/treasure.txt"

    # Does the file even exist? If not, create it
    if [[ ! -f "$file" ]]; then
        echo "File $file not found. Creating the file."
        touch "$file"
    fi
    echo "File $file is ready."

    # Remove immutable attribute if necessary
    if lsattr "$file" 2>/dev/null | grep -q 'i'; then
        chattr -i "$file"
        echo "Removed immutable attribute from $file."
    fi

    chown alice:alice "$file" && echo "Changed file ownership to Alice."

    chmod 600 "$file" && echo "Changed file permissions to read/write for
      ↪ owner (Alice), no permissions for others."

    # Does the treasure_group even exist? if not, create it
    if ! getent group treasure_group &>/dev/null; then
        groupadd treasure_group && echo "Group 'treasure_group' created."
    fi

    chgrp treasure_group "$file" && chmod 660 "$file" && \
    echo "Changed the file group to 'treasure_group' and set group
      ↪ permissions to read/write."

    setfacl -m u:charlie:r-- "$file" && echo "Set ACL for 'charlie' to read
      ↪ only."

    # Show requested permissions and ACL
    echo "Final permissions for $file:"
    ls -l "$file"
    getfacl "$file" 2>/dev/null || echo "ACL not supported on this system or
      ↪  not present."
}

Alice_needs_a_home...
delete_treasures
treasure_needs_permissions

getent group treasure_group

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

Listing 5: 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."