Permissions

permissions practice for CSC420 Operating Systems Spring 2022 Lyon College

README

- This file accompanies lectures on the shell and `bash(1)`. To gain practice, you should type along in your own Org-mode file. You have to have Emacs and my `.emacs` file installed on your PC or the Pi you're working with.
- To make this easier, use the auto expansion (`<s`). This will only work if you have my `.emacs` file (from GDrive) installed.

- Add the following two lines at the top of your file, and activate each line with `C-c C-c` (this is confirmed in the echo area as `Local setup has been refreshed`)):

```
#+PROPERTY: header-args:bash :results output
```

- Remember that `C-M-\` inside a code block indents syntactically (on Windows, this may only work if you have a marked region - set the mark with `C-SPC`).
- This section is based on chapter 9 of Shotts, The Linux Command Line (2e), NoStarch Press (2019).

What is it?

- OS in the UNIX tradition are multi-tasking and multi-user systems
- If the computer is attached to a network, remote users can log in via `ssh` (secure shell) and operate the computer, including GUIs
- Multiuser capability of Linux is a deeply embedded OS feature because the first computers were not "personal"
- To make multiuser practical, users had to be protected from one another
- Related topics and commands:

COMMAND	MEANING
id	Display user identity
chmod	Change a file's mode
umask	Set the default file permissions
su	Run a shell as another user
sudo	Execute a command as another user
chgrp	Change a file's group ownership
passwd	Change a user's password

Owners, Group Members, and Everybody Else

Example: a bad experience

- `[ ]`

  Check the file type of /etc/shadow using the `file` command

```
file /etc/shadow
```

```
/etc/shadow: regular file, no read permission
```

```
/etc/shadow: regular file, no read permission
```

- Try to page the file using the `less` command. Anticipating an error, redirect the standard error to standard output.

```
less /etc/shadow 2>&1
```

```
/etc/shadow: Permission denied
```

- As regular user, you don't have permission to look at this file. Check if you can at least see who does have permission.

```
ls -la /etc/shadow 2>&1
```

```
-rw-r----- 1 root shadow 1077 Jan 30 16:36 /etc/shadow
```

The security model

- In the UNIX security model, a user may *own* files and directories.
- With ownership comes access control.
- The user can also belong to a *group* of one or more users who are given access to files and directories by their owners.
- A user may also grant access rights to everyody (aka the *world*).

- Find out who you are in this model with the command `id`.

```
id
```

```
uid=1000(pi) gid=1000(pi) groups=1000(pi),4(adm),20(dialout),24(cdrom),27(sudo),29
```

- When users are created, they are assigned a *user ID* (*uid*), which is mapped to a user name
- The user is also assigned a *group id* (*gid*) and can be part of other groups.
- The specific output is different for different Linux distros. E.g. Fedora Linux starts numbering uid at 500, Debian/Ubuntu at 1000.
- This information is stored in text files, of course: user accounts in /etc/passwd, groups in /etc/group.

- `[ ]` Take a look at /etc/passwd and /etc/group.
- /etc/shadow holds information about the user's password.
- `[ ]`

  What is the uid of the root user? Use `grep` to get the information about root from the file with the uid information

  ```
  cat /etc/passwd | grep root
  ```

  ```
  root:x:0:0:root:/root:/bin/bash
  ```

- `[ ]`

  Can you think about a way to directly get the uid for root?

  ```
  sudo id
  ```

  ```
  uid=0(root) gid=0(root) groups=0(root),117(lpadmin)
  ```

Reading, Writing, and Executing

- Access rights to files and directories are defined in terms of **read** access, **write** access, and **execution** access.
- The long listing command `ls -l` shows how this is implemented.
- `[ ]`

  Create an empty file foo.txt using file redirection, and then print a long listing of the file.

  ```
  > foo.txt
  ls -l foo.txt
  ```

  ```
  -r--r--r-- 1 pi pi 0 Mar 29 09:09 foo.txt
  ```

File attributes

- The first 10 characters of the listing are *file attributes*. Table 1 gives an overview.

ATTRIBUTE	FILE TYPE
-	regular file
d	directory
l	symbolic link
c	character special file

ATTRIBUTE	FILE TYPE
b	block special file

- `[ ]` For symbolic links, the remaining attributes are always dummy values. What do you think why that is?
- `[ ]`

  Which "character special file" did you already encounter? These files handle data as a stream of bytes.

  > Answers: 1) the /dev/null, or "bit bucket", used to redirect unwanted error messages (`2 > /dev/null`). 2) The terminal (used for shell input/output).

- A block special file handles data in blocks, e.g. a hard drive.

File modes

- The remaining nine characters are the *file mode* for the owner, the group, and the world - r=read, w=write, x=execute. Table 1 shows examples.

WHO	Owner	Group	World
WHAT	rwx	rwx	rwx
Example	pi	gpio	

- Table 1 shows the effect that the mode has on files and directories.

ATTRIBUTE	FILES	DIRECTORIES
r	can be opened	can be listed if x is set
w	can be written	files can be created, deleted, renamed if x is set
x	can be run	allows a directory to be entered, e.g. with `cd`

- Scripts(e.g. bash scripts) must also be set readable to be executed.

- Table 1 shows some examples of file attribute settings.

ATTRIBUTE	MEANING
-rwx------	Regular file, readable, writable, executable by file's owner only. Nobody else can access.
-rw-------	Regular file, readable, writable by file's owner only. Nobody else can access.
-rw-r–r--	Regular file, readable, writable by file's owner. Members of file owner's group and world may read
-rwxr-xr-x	Regular file, readable, writable, executable by file's owner, can be read and executed by everybody else
-rw-rw----	Regular file, readable, writable by file's owner and members of file's owners group only
lrwxrwxrwx	Symbolic link with dummy permissions. Real permissions kept with file pointed to by the link.

ATTRIBUTE	MEANING
drwxrwx---	Directory. Owner and members of owner group may enter, create, rename and remove files here.
drwxr-x---	Directory. Owner may enter, create, rename, delete files here. Group members may enter but cannot.

- `[ ]`

  Check *sys/class/gpio*.

  Answer without checking directly, only based on the file attributes: Can you write to the files `export` and `unexport`?

  > Answer: YES, because user pi is a member of the group gpio, like the file's owner, root. You can however, not directly write to it.

- `[ ]`

  Check your $HOME. What are the permissions, and what is everybody (the world) allowed to do or see?

  > Answer: ./ has permissions "drwxr-xr-x". The world can enter the directory and see the file listing, but cannot create, rename or remove files.

Changing file modes

- Only file owners and superuser can change the mode of a file or directory using the command `chmod`.
- Mode changes can be specified using octal numbers or symbols, because each digit in an octal number represents three (8 = 2^3) binary digits.

Changing file modes with octal numbers

- Octal people were born with 8 fingers. Different base systems, like octal (base 8), binary (base 2) or hexadecimal (base 16) can be used to abbreviate patterns that adhere to the base.
- Pixels e.g. are composed of 3 color components: 8 bits of red, green, blue each. A medium blue in binary would be a 24-digit number, but it can be condensed to a 6-digit hexadecimal, 436FCD.

- Table 1 shows the file modes in binary and in octal notation. In octal, counting is done with the numbers 0 to 7.

OCTAL	BINARY	FILE MODE
0	000	---
1	001	–x
2	010	-w-
3	011	-wx
4	100	r--
5	101	r-x
6	110	rw-

OCTAL	BINARY	FILE MODE
7	111	rwx

- By setting 3 octal digits, we can set the file mode for the owner, group owner, and world.
- `[ ]`

  Example: run the block 1. An empty file is created and long-listed.

  ```
  > foo.txt
  ls -l foo.txt
  ```

  ```
  -r--r--r-- 1 pi pi 0 Mar 29 09:09 foo.txt
  ```

- `[ ]`

  In the block 1 below, change the permissions (file mode) to 600 with the command `chmod 600 [filename]` and list the file.

  Check with the table that this is what was supposed to happen: read and write permissions for the owner, and no access rights for anyone else.

  ```
  chmod 600 foo.txt
  ls -l foo.txt
  ```

  ```
  -rw------- 1 pi pi 0 Mar 29 09:09 foo.txt
  ```

- `[ ]`

  Change the mode of foo.txt to be readable by owner, group, and world, with no other permissions for any of these.

  ```
  chmod 444 foo.txt
  ls -l foo.txt
  ```

  ```
  -r--r--r-- 1 pi pi 0 Mar 29 09:09 foo.txt
  ```

Changing file modes with symbols

- Symbolic notation is divided into three parts:
  - Who the change will affect
  - Which operation will be performed
  - What permission will be set
- To specify who is affected, a combination of characters is used, as shown in table 1.

WHO	MEANING

WHO	MEANING
u	user = file or directory owner
g	group owner
o	others = world
a	all = combination of u,g,o

- If no character is specified, "all" (a) is assumed. Three operations are allowed, see table 1:

OPERATION	MEANING
+	permission to be added
-	permission to be removed
=	specified permissions to be applied and all others removed

- Table 1 shows some examples. Multiple specifications may be separated by commas.

NOTATION	MEANING
u+x	add execute permission for owner
u-x	remove execute permission for owner
+x	add execute permission for owner, group, world
a+x	add execute permission for owner, group, world
o-rw	Remove read, write permissions from anyone except owner, group
go=rw	Set group owner and anyone besides the owner to have read, write permissions.
	If group owner or world previously had execute permissions, they are removed.
u+x, go=rx	Add execute permissions for the owner, and set read,execute for group, others

- [ ]

  Example: run the block 1. An empty file is created and long-listed.

```
> bar.txt
ls -l bar.txt
```

```
-r--r--r-- 1 pi pi 0 Mar 29 09:03 bar.txt
```

- [ ]

  In the block 1 below, set the permissions for the owner, the group and others to read and write only. Use the command `chmod [operation] [filename]`, then list the file.

```
chmod ugo=rw bar.txt
ls -l bar.txt
```

```
-rw-rw-rw- 1 pi pi 0 Mar 29 09:03 bar.txt
```

- [ ]

  Change the mode of bar.txt to be readable by owner, group, and world, with no other permissions for any of these.

  ```
  chmod ugo-w bar.txt
  ls -l bar.txt
  ```

  ```
  -r--r--r-- 1 pi pi 0 Mar 29 09:03 bar.txt
  ```

Summary

Author: Marcus Birkenkrahe
Created: 2022-03-29 Tue 09:10
Validate