Linux Basics and Shell Scripting Chapter - 2

Presentation from Uplatz

Contact Us: https://training.uplatz.com/
Email: info@uplatz.com
Phone: +44 – 7836 212635

2. File Utilities

- Standard I/O, redirection and pipes
- File descriptors and its usage of meta characters (>, >>, <, <<)
- Changing file access rights (users and permissions including both Symbolic and Absolute modes) (chmod, chown, chgrp, umask)
- Soft links and hard links
- Checking file integrity (md5sum)

Standard I/O and File Descriptors

- Standard User Input – read
- Standard User Output – write
- Standard Input – 0
- Standard Output – 1
- Standard Error - 2

I/O Redirection

- Output Redirection (>)

```
$ command > file-name
$ date > cmdoutput.txt
$ echo "Output:Hello" "Error: Jump on red signal" 2>error.txt 1>&2
$ cat error.txt
```

- Output Redirection Append (> >)

```
$ command >> file-name
$ date >> cmdoutput.txt
$ cal >> cmdoutput.txt
```

- Input redirection (<)

```
$ command < file-name
$ cat < cmdoutput.txt
```

- “Here” document (<<)

```
$ command << delimeter
 $ cat << stop
>  this is test
> stop
This is test
$
```

I/O Redirection

```
cat <<End-of-message
-------------------------------------
This is line 1 of the message.
This is line 2 of the message.
This is line 3 of the message.
This is line 4 of the message.
This is the last line of the message.
-------------------------------------
End-of-message

cat << -ENDOFMESSAGE
	This is line1 of message
	This is line2 of message
	This is line3 of message
-ENDOFMESSAGE

wc -w  << EOF
	This is a test.
	Apple juice.
	100% fruit juice and no added sugar, colour or preservative.
EOF
```

Standard input

More about command input

- Lots of commands, when not given input arguments, can take their input from standard input.
  - $ sort
    - windows
    - linux
    - [Ctrl] [D]
    - linux
    - Windows

sort takes its input from the standard input: in this case, what you type in the terminal (ended by [Ctrl][D])

```
$ cat > participants.txt
suresh
naresh
jyothi
sruthi
sai
$ sort participants.txt
$ sort –n participants.txt
$ sort –nr participants.txt
$ ls | sort
$ sort < participants.txt
```

  - The standard input of sort is taken from the given file.

Standard output / redirection

More about command output

- All the commands outputting text on your terminal do it by writing to their standard output.
- Standard output can be written (redirected) to a file using the > symbol

```
$ ls ~uplatz/* > ~guest/files_list.txt
$ cat output1.txt > consolidatedlist.txt
$ echo “README: No such file or directory” > README
```

(Useful way of creating a file without a text editor)

- Standard output can be appended to an existing file using the >> symbol

```
$ cat output2.txt >> consolidatedlist.txt
```

Standard error

- Error messages are usually output (if the program is well written) to standard error instead of standard output.
- Standard error can be redirected through 2> 2>>
- Example:
  - $ cat f1 f2 nofile > newfile 2> errfile
- Note: 1 is the descriptor for standard output, so 1> is equivalent to >.
- Can redirect both standard output and standard error to the same file using &> :
  - $ cat f1 f2 nofile &> wholefile

Pipes

- Pipe operator (|) : The output of a command is sent as input for the other command.
- Unix pipes are very useful to redirect the standard output of a command to the standard input of another one.

  ```
  $ command1 | command2 | command3
  ```

- Examples

  -
    ```
    $ cat *.log | grep –i error | sort
    ```

  -
    ```
    $ grep –ri error . | grep –v “ignored” | sort –u \
        > serious_errors.log
    ```

  -
    ```
    $ cat /home/*/homework.txt | grep mark | more
    ```
- This is one of the most powerful features in Unix shells!

File access rights

- Use ls –l to check file access rights

3 types of access rights

- Read access (r)
- Write access (w)
- Execute rights (x)

3 types of access levels

- User (u): for the owner of the file
- Group (g): each file also has a “group” attribute, corresponding to a given list of users
- Others (o): for all other users

File permissions

- Each file has permissions for user (u), group (g) and all others (o)
  - -rw-rw-r-- 1 student student 0 Aug 31 21:58 helloworld.txt
- The first character denotes the type of the file
- The first {rwx} set indicates the permissions for user
- The second {rwx} set indicates the permissions for group
- The third {rwx} set indicates the permissions for others
- You can use decimal equivalents of above numbers

Access right constraints

- x is sufficient to execute binaries
- Both x and r are required for shell scripts.
- Both r and x permissions needed in practice for directories: r to list the contents, x to access the contents.
- You can't rename, remove, copy files in a directory if you don't have w access to this directory.
- If you have w access to a directory, you CAN remove a file even if you don't have write access to this file (remember that a directory is just a file describing a list of files). This even lets you modify (remove + recreate) a file even without w access to it.

Access rights examples

- - r w - r- - r - -
  - Readable and writable for file owner, only readable for others
- - r w – r - - - - - -
  - Readable and writable for file owner, only readable for users belonging to the file group.
- d r w x - - - - - -
  - Directory only accessible by its owner
- - - - - - - - r – x
  - File executable by others but neither by your friends nor by yourself. Nice protections for a trap …

Chmod: changing permissions

- Chmod <permissions> <files>
- 2 formats for permissions
  - Octal format (abc):
    - a, b, c = r*4+w*2+x*1 (r, w, x: booleans)
    - Examples: chmod 644 <file>
    - (rw for u, r for g and o)
  - Or symbolic format. Easy to understand by examples:
    - chmod go+r: add read permissions to group and others.
    - chmod u-w: remove write permissions from user.
    - chmod a-x: (a:all) remove execute permissions from all.

More chmod

- chmod –R a+rX linux/
  - Makes linux and everything in it available to everyone!
  - R: apply changes recursively
  - X:x, but only for directories and files already executable
  - Very useful to open recursive access to directories without adding execution rights to all files.
- chmod a+t /tmp
  - t: (sticky). Special permission for directories, allowing only the directory and file owner to delete a file in a directory.
  - Useful for directories with write access to anyone, like /tmp
  - Displayed by ls –l with a t character.

File ownership

Particularly useful in (embedded) system development when you create files for another system.

- chown –R sco /home/linux/src (-R: recursive)
  - Makes user sco the new owner of all the files in /home/linux/src.
- chgrp –R empire /home/askywalker
  - Makes empire the new group of everything in /home/askywalker
- chown –R borg:aliens usss_enterprise/
  - chown can be used to change the owner and group at the same time.

Symbolic Links

A symbolic link is a special file which is just a reference to the name of another one (file or directory)

- Useful to reduce disk usage and complexity when 2 files have the same content.
- Example
  - sample_clinux-> sample_linuxc
- How to identify symbolic links:
  - ls -l displays -> and the linked file name
  - GNU ls displays links with a different color.
- Two types
  - Soft link (works across file systems)
  - Hard link (works only within file systems)

Creating soft links

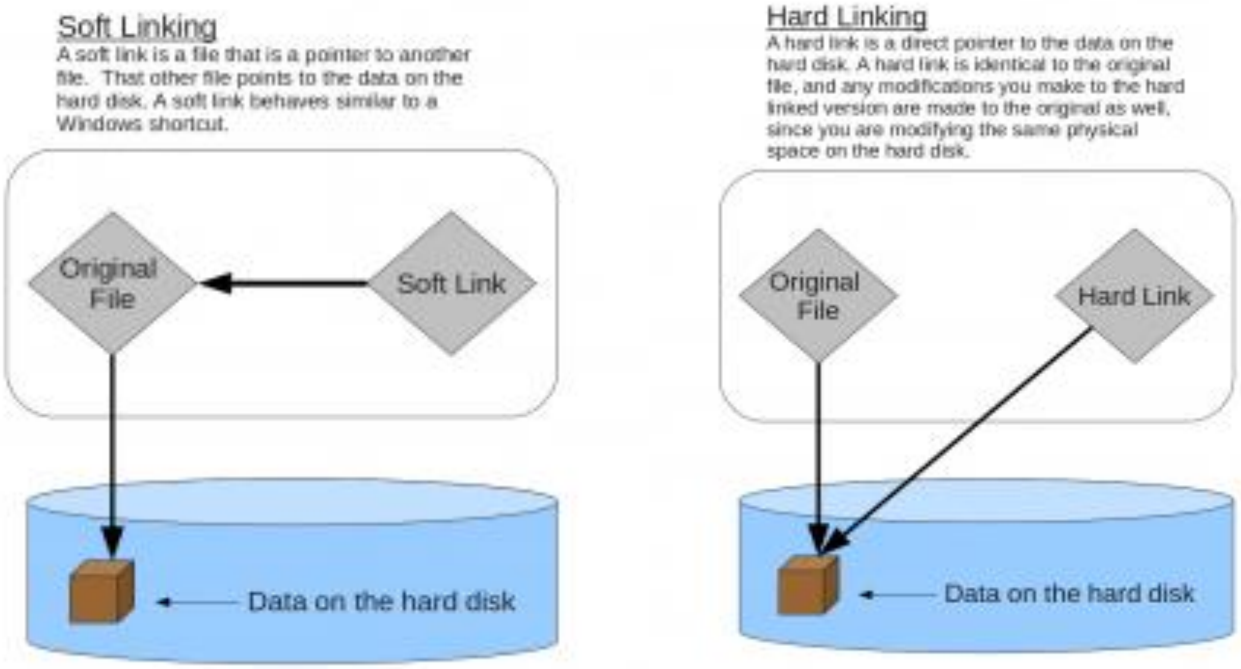
- To create a soft link (same order as in cp):
  - ln -s file_name link_name
- To create a link with to a file in another directory, with the same name:
  - ln -s .../README.txt
- To create multiple links at once in a given directory:
  - ln -s file1 file2 file3 ... dir
- To remove a link:
  - rm link_name
  - Ofcourse, this doesn’t remove the linked file!

Hard links

- The default behaviour for ln is to create hard links
  - $ ln original_file [link_name]
- A hard link to a file is a regular file with exactly the same physical contents.
- While they still save space, hard links can't be distinguished from the original files.
- If you remove the original file, there is no impact on the hard link contents.
- The contents are removed when there are no more files (hard links) to them.

File names and inodes

Makes hard and symbolic (soft) links easier to understand!



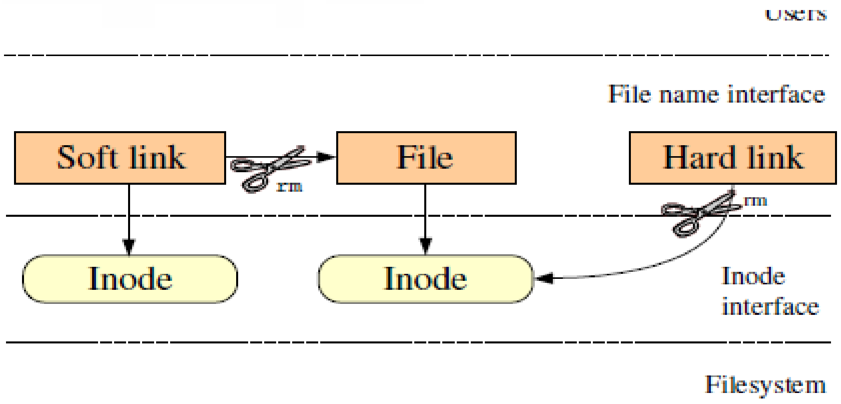
Soft Linking

A soft link is a file that is a pointer to another file. That other file points to the data on the hard disk. A soft link behaves similar to a Windows shortcut.

Hard Linking

A hard link is a direct pointer to the data on the hard disk. A hard link is identical to the original file, and any modifications you make to the hard linked version are made to the original as well, since you are modifying the same physical space on the hard disk.

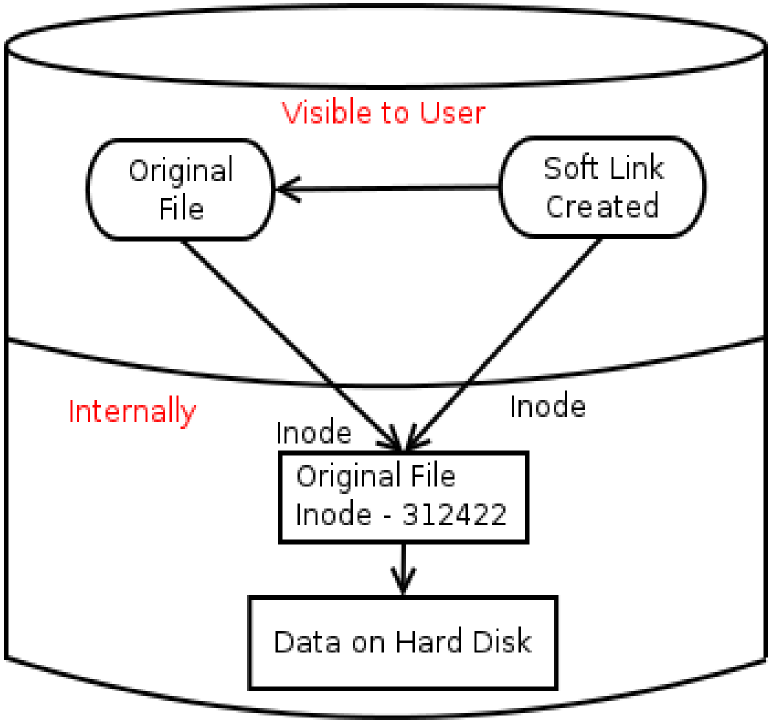
https://medium.com/@yogmazoophem/symbolic-vs-hard-link-what-93790153ec44



Hard Link and Soft Link / Symbolic Link

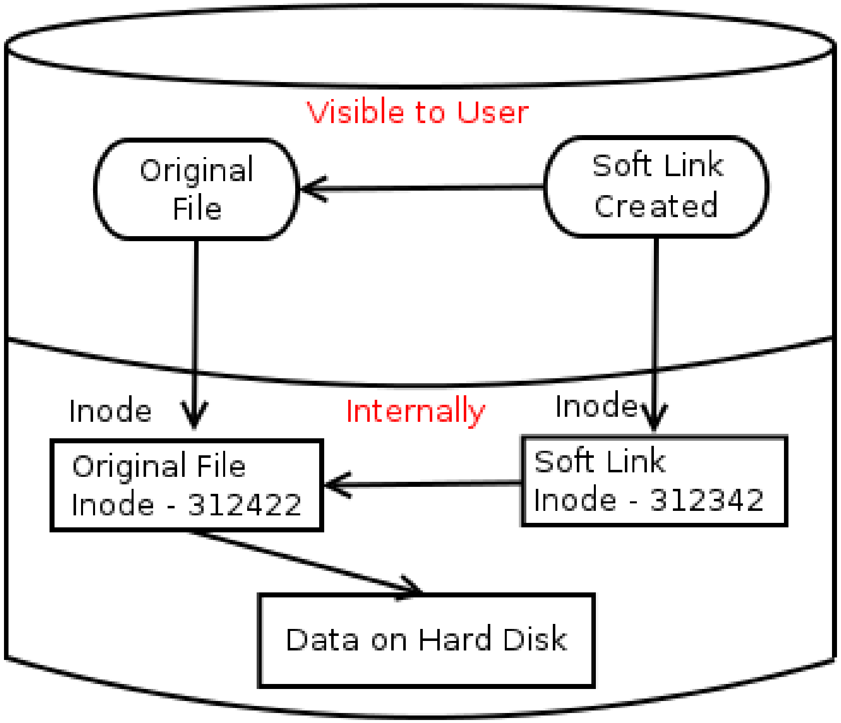
Hard Link

Hard Link is direct pointer to the original inode of the original file. If you compare the original file with hard link, there won't be any differences.



Soft Link / Symlink

A softlink is a file that have the information to point to another file/inode. That inode points to the data on the hard drive.



http://www.geekride.com/hard-link-vs-soft-link/

Checksum command and checking file integrity

Very low cost solution to check file integrity

- A checksum or hash sum is a fixed-size datum computed from an arbitrary block of digital data for purpose of detecting accidental errors that may have been introduced during its transmission or storage.
  - http://en.wikipedia.org/wiki/Checksum
- Computes a MD5 (Message Digest Algorithm 5) 128 bit checksum of the given files. Usually redirected to a file.

  -
    ```
    $ md5sum FC3-i386-disk*.iso > MD5SUM
    ```
- Example output

```
db8c7254beeb4f6b891d1ed3f689b412  FC3i386disc1.iso
2c11674cf429fe570445afd9d5ff564e  FC3i386disc2.iso
f88f6ab5947ca41f3cf31db04487279b  FC3i386disc3.iso
6331c00aa3e8c088cc365eeb7ef230ea  FC3i386disc4.iso
```

- Checks the integrity of the files in MD5SUM by comparing their actual MD5 checksum with their original one.

  -
    ```
    $md5sum –c MD5SUM
    ```

Uplatz