ULI101: INTRODUCTION TO UNIX / LINUX AND THE INTERNET

WEEK 8: LESSON 1

LINKING FILES

LESSON 1 TOPICS

Linking Files

- i-nodes
- Hard Links / Demonstration
- Symbolic Links / Demonstration

Perform Week 8 Tutorial

- Investigation 1
- Review Questions (Questions 1 – 2)

LINKING FILES

inode (index) Number of a File:

The **i-node number** is like a "**finger-print**" which is **unique** for each file on the Unix / Linux file system.

The i-node is an **index** (**data structure**) that provides information about the file such as if the file is a **directory** or **regular file**, etc.

Referring to the diagram below, issuing the **ls** command using the **-i** option displays the **i-node** number for each file. You can see that each file has its own **unique** *i-node* number in the file system.

```
[ murray.saul ] ls -li
total 0
1162999961 -rw-r--r-- 1 murray.saul users 0 Jan 31 07:26 a.txt
1164541350 -rw-r--r-- 1 murray.saul users 0 Jan 31 07:26 b.txt
1165743019 -rw-r--r-- 1 murray.saul users 0 Jan 31 07:26 c.txt
2248130583 drwxr-xr-x 2 murray.saul users 6 Jan 31 07:26 mydir
```

no. of links / reference to that object

i-node number

LINKING FILES

change one file, will affect other linked files
original version doesn't exist and matter

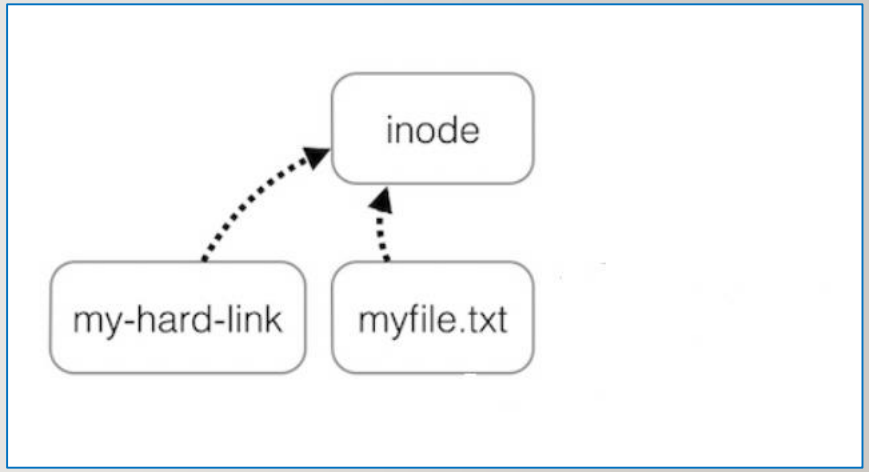
Hard Links ln

If accidently remove one file, the rest files will be linked and synced together

A **Hard link** is a **reference** to the **same index** on a file system. It does this by creating a file that **shares the same i-node number** with the other file.

An **advantage** of using hard links is that if one hard link remains (even if original file has been removed), **the data in that hard-linked file is NOT lost**. Also, any change to each file will be reflected in any hard-linked file which is useful for **backups**.

Limitations of hard links are that **they take-up extra space**, you **cannot hard link directories**. Also, you **cannot hard link files from other Unix/Linux servers** (since the i-node number may already be used by the other Unix/Linux server).



hard link files only
within one server

LINKING FILES

Hard Links

Examples:

```
touch myfile.txt
ln myfile.txt myfile1.hard.lnk
ln myfile.txt myfile2.hard.lnk
ln myfile.txt ~/backups/myfile.hard.lnk
ls -li myfile*
```

```
[ murray.saul ] pwd
/home/murray.saul/link-demo1
[ murray.saul ] touch myfile.txt
[ murray.saul ] ln myfile.txt myfile1.hard.lnk
[ murray.saul ] ln myfile.txt myfile2.hard.lnk
[ murray.saul ] ln myfile.txt ~/myfile3.hard.lnk
[ murray.saul ]
[ murray.saul ] ls -li . ~/myfile3.hard.lnk
3261599590 -rw-r--r-- 4 murray.saul users  0 Feb  3 08:39 /home/murray.saul/myfile3.hard.lnk

.:
total 0
3261599590 -rw-r--r-- 4 murray.saul users 0 Feb  3 08:39 myfile.txt
3261599590 -rw-r--r-- 4 murray.saul users 0 Feb  3 08:39 myfile1.hard.lnk
3261599590 -rw-r--r-- 4 murray.saul users 0 Feb  3 08:39 myfile2.hard.lnk
```

LINKING FILES

Instructor Demonstration

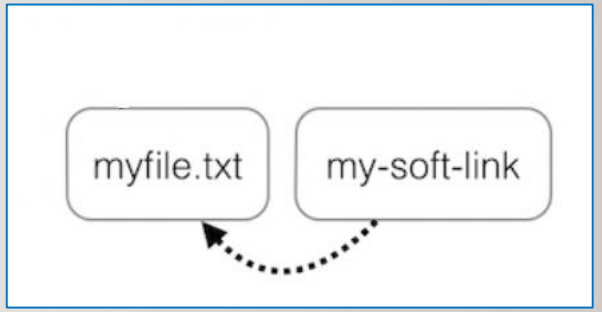
Your instructor will now demonstrate how to create **Hard Links**.

```
echo hello there > hello.txt
ln hello2.txt
ls -li
chmod 644 hello2.txt (will change hello.txt as well)
ln hello2.txt hello3.txt

rm hello.txt ( hello2, hello3 still exist)

** No original file, all are the equivalent
```

LINKING FILES

Symbolic Links ln -s

A **Symbolic Link** is an **indirect pointer** to a file and are also known as a **soft link** or **symlink**. The symbolic link file contains the **pathname** to the original file.



An **advantage** of using symbolic links is they act as **shortcuts** to other files (in fact, the symbolic linked file only contains the pathname to the original file). Also, you can create symbolic links on **different** Unix/Linux servers, and that you can create symbolic links for **directories**.

A **limitation** of using symbolic links is that they are **NOT good for backup purposes** since a symbolic link can point to a **nonexistent** file (referred to as a **"broken link"**).

LINKING FILES

Symbolic Links

Examples:

```
touch otherfile.txt
ln -s otherfile.txt otherfile1.sym.lnk
ln -s otherfile.txt otherfile2.sym.lnk
ln -s otherfile.txt ~/backups/otherfile.sym.lnk
ls -li otherfile*
```

```
[ murray.saul ] pwd
/home/murray.saul/link-demo2
[ murray.saul ] touch otherfile.txt
[ murray.saul ] ln -s otherfile.txt otherfile1.sym.lnk
[ murray.saul ] ln -s otherfile.txt otherfile2.sym.lnk
[ murray.saul ] ln -s ~murray.saul murray
[ murray.saul ] ls -li
total 0
3267712746 lrwxrwxrwx 1 murray.saul users 17 Feb  3 09:08 murray -> /home/murray.saul
3267712744 -rw-r--r-- 1 murray.saul users  0 Feb  3 09:08 otherfile.txt
3267712742 lrwxrwxrwx 1 murray.saul users 13 Feb  3 09:08 otherfile1.sym.lnk -> otherfile.txt
3267712745 lrwxrwxrwx 1 murray.saul users 13 Feb  3 09:08 otherfile2.sym.lnk -> otherfile.txt
```

LINKING FILES

Instructor Demonstration

Your instructor will now demonstrate how to create **Symbolic (Soft) links**.

HOMEWORK

Getting Practice

Perform **Week 8 Tutorial:**

(Due: Friday Week 9 @ midnight for a 2% grade):

- INVESTIGATION 1: LINKING FILES
- LINUX PRACTICE QUESTIONS (Questions **1 – 2**)