CIT 371 Lab 6: Links, Permissions and Ownership

This lab can be completed with the Web Console or with SSH/PuTTY. See the Lab Access Document for more information on accessing your VMs.

Start Coivcenter and VM1 (CentOS6), login as Student, open a terminal window and cd to /home/Student.

1. We examine the difference between symbolic and hard links.
   1. Type **cp ~Student/FILES/sales.txt ~Student/FILES/sales2.txt**. This will save a file that we will delete later so that we can restore it. A link in the Linux file system is a file record that points either to a file or a file record. There are two types of pointers, hard links and soft links (also called symbolic links). The number of links pointing at an item is denoted as the number that appears after the permissions in an ls –l. Type **ls –l FILES**. These files have a 1 indicating no pointers pointing at them. Directories will always have at least a 2. Type **ls –l /home** to test this out. From ~Student, type **ln –s FILES/sales.txt sales-copy**. This creates a symbolic link in ~Student called sales-copy which points to FILES/sales.txt. Type **ls –l**. The new entry has the character ‘l’ to start its long listings (“link”). Also notice the name is now sales-copy -> FILES/sales.txt. Type **ls –l FILES**. *What number is provided for the number of links for sales.txt?* 
      1. **The number that is provided for the links for sales.txt is 1.**
   2. Delete sales-copy. Type **ls –l** of ~Student and then of FILES. The file sales-copy is gone but FILES/sales.txt still exists. Why? We deleted the symbolic link, not what the link pointed to. Repeat the ln command from 1a omitting the option **–s** to create a hard link. Type **ls –l** of both ~Student and FILES. *How many links do both files indicate now?* Notice how sales-copy is no longer indicated as a link (“l”) and no longer has the -> in its name. Delete FILES/sales.txt. Type **ls –l** in ~Student. *Is sales-copy still there? If so, how many links point at it now?* Type **mv FILES/sales2.txt FILES/sales.txt** (this replaces the file you deleted). Type **ls –l FILES**. This time you will see sales.txt has a 1 by it indicating no hard links. But sales-copy still exists, so why not? Because sales-copy pointed at a different physical file than sales2.txt and even though we renamed sales2.txt to sales.txt, sales-copy is not the same file. Type **rm sales-copy**. Type ls –l FILES and you will see the hard link count is back to 1.
      1. **The file now indicates 2 links. Sales copy is still there and it now has a value of 1.**
   3. Repeat the cp instruction from 4c to save sales.txt as sales2.txt and the ln –s command from 4c. Type **rm FILES/sales.txt** to remove the file, leaving the symbolic link. Type **ls –l**. *How does sales-copy appear now?* Type **cat sales-copy**. *What happens?* As best you can, explain the difference between hard and symbolic links. Type **mv FILES/sales2.txt FILES/sales.txt** so that we can restore the file. Type **ls –l** and you will see the symbolic link again points to the file it is supposed to. Type **rm sales-copy** to remove the symbolic link.
      1. **Sales copy is highlighted in black and red text with the FILES/sales.txt being red. Sales-copy has no such file or directory. The difference between a hard and symbolic link is that a hard link is a carbon copy that will change when the original file is changed, while a symbolic link will act as a pointer type of file.**
2. Permissions and access
   1. Type **ls –l FILES** to provide a long listing of the contents of this directory. The first 10 characters of a line are the file type and permissions. These files are regular files, indicated by -. The next 9 characters are combinations of r, w, x and -, standing for “read access”, “write access”, “execute access” and “no access”. They are in 3 triplets with the first triplet for the file owner (known as user), the second triplet for the group owner and the last triplet for the rest of the world (known as other). *For these three files, what permissions does the user have? What permissions does the group have? What permissions does others have?* The next item is the number of links pointing at this item. In this case, it is 1 for each. The next two items on each line are the owner (user) and the group owner. These should both be Student. Type **ls –n FILES**. The –n option outputs the UID and GID (user/group IDs). *What number appears for owner and group?* 
      1. **User= read, write, execute. Group= read. Other= read. The number 501 appears for both user and group.**
   2. As a user, you usually have access to your own files and you may have access to other peoples’ files as well. Let’s test this out. Type **cd /home/zappaf** (or ~zappaf). Type **ls –l**.  *Of the directories listed here, which can you cd into? Which can you view the contents of (using ls)?* If you don’t know, try them all out.
      1. **You can view all contents and cd into all but GENESIS, IQ, and VDGG**
   3. Notice that the IQ directory has permissions for group but not for other, and the group ownership is called *cool* instead of zappaf. Student is not a member of cool but foxr is. Therefore, both zappaf and foxr have access to IQ but not Student. su to foxr (password: cit371). Attempt to **ls** and **cd** (two separate instructions) into the IQ directory. *Are you successful?* As foxr, do both **ls** and **cd** to the GENESIS directory. *Why are you not successful?* *Does zappaf have ls and cd access to the GENESIS directory?* Type **exit** to exit foxr’s account and resume being Student.
      1. **I was successful in ls and cd into the IQ directory. Foxr does not have Genesis to ls or cd into, and then in zappaf we have our permission denied still.**
   4. cd to /etc. Many of these files are accessible only to root. Do an **ls –l | less**. *Which files are only readable (or readable and writable) by root.* Ignore directories when you do this. There are four files not even readable by root. *Which files are they?* (NOTE: there are two sets of two files, the second of each pair has a – at the end of its name)
      1. **Autofs\_ldap\_auth.conf, nslcd.conf, securetty, etc. The 4 files are the pair of shadow and shadow- and then the pair of gshadow, which are gshadow and gshadow-.**
   5. cd to ~Student/DUMMY-DIRECTORY. Type **ls –l**. *Which of the files and subdirectories in this directory are accessible only by Student (owner or group)? Which have write access to the world? Which are executable by anyone? Of the files/directories, which should Student change and why?*
      1. **File3.dat is only owned by Student. Aa7.bcc has access to the world to write and read. The executables by anyone is located in directory2 and files1.dat. I would change the permission of the directory2 I doubt everyone should be able to execute this privilege, it seems like a directory should have limited access.**
3. Command (program) privileges
   1. Many commands are reserved for root. This is done simply by setting permissions so that the owner and group (root) can execute it but not the world. Most of the commands that only permit root access are in /usr/sbin and /sbin. cd to /sbin. Type **ls –l | less** and scroll through the files noting commands (files) that **do not** have execute access for world. *Which ones did you find?* There are more such files in /usr/sbin.
      1. **The ones I found were audispd, auditctrl, aduitd, and autrace.**
   2. Let’s give this a try. You might notice that chkconfig has world execute access and audispd does not. Run both programs*. What is the result?*
      1. **The chkconfig operation ran correctly outputting the available services and then the audispd did not run and come up with a permission denied.**
   3. Switch directory to /usr/sbin and do **ls –l group\* user\*** (this command lists all of the files that start with either group or user). You will notice that the add, delete and mod commands do not offer world any form of access.
   4. Other system programs are located in /bin and /usr/bin. Examine these two directories. *How many of these commands do not permit execute access for world?*
      1. **The difference I found between these two directories’ was in /bin Fusermount was the only file that had non executable permission to world. Compared to /usr/bin which had multiple of these permissions such as usermod, useradd, userdel, etc.**

Shut down your VM if desired and disconnect from the VPN if you are using it. Submit your lab report.