**Technical Operations (TechOps)**

OS Admin

Linux Narrative Exercises

© FDM Group Ltd 2011. All Rights Reserved.

Any unauthorised reproduction or distribution in part  
or in whole will constitute an infringement of copyright.

|  |
| --- |
| Page left intentionally blank |

# Advanced Permissions Narrative

|  |
| --- |
| **L**ogging into the Linux Env |
| |  | | --- | | **L**ogging into the Linux Env | | Your instructor will explain how to connect to the linux machine you are to use for the practical exercises. Use the ssh command from the windows powershell. You will first need to know your *connection string* which you past into the poweshell window and run as a poweshell command. This you can obtain by following the link in the email sent to you from Azure.    This gives you:    The connection string lives here. (Click)    Click to copy the connection string into the buffer.  If you are accessing the Azure environment, you will be given a connection string that will look something like:  ssh -p 49196 instructor@ml-lab-80add82f-5463-4584-b9cd-cdc3616ee4b5.ukwest.cloudapp.azure.com  The part of the string that comes after the ‘@’ symbol is the hostname. In this case the hostname is ml-lab-80add82f-5463-4584-b9cd-cdc3616ee4b5.ukwest.cloudapp.azure.com  The name in front of the ‘@’ symbol is the user name. In the example you are logging is as the user instructor. If you wanted to login as ‘jill’ the connaction string would be like:  ssh -p 49196 **jill**@ml-lab-80add82f-5463-4584-b9cd-cdc3616ee4b5.ukwest.cloudapp.azure.com  The connection string should be pasted directly into a powershell window. To launch the powershell app, type ‘power’ in the search bar. (Do not launch the version with (x86) in the name.). Select the App as per below.    Once you have launched powershell you can use the right mouse click to paste the connection string into the powershell window.    If you want to log in as another user, simply use the arrow keys to move your cursor to the left of the ‘@’ symbol, delete the name instructor and replace it with the user name of your choice.  The ssh command is now built into powershell (Windows 10) and is the same as the command you might type from a Linux terminal. In case you are wondering, the -p option defines a network port number. Azure use this number to direct you to the ssh login of the correct virtual machine.  On Apple Mac machines, just bring up the terminal App. Paste the connectiob string into the terminal window.  If you login as ‘instructor’, as you will do 98% of the time, you can run the command  “sudo -i” in order to become ‘root’. For these exercises you will need to be root, unless you are directed otherwise. | |
| Login in to your unix machine and become root, using a powershell window (or alternative). Using a second separate window login as user robert. (You should have created an account called robert with a password of FDM@2020! in the previous exercise.)  Using the touch command create a file called apple. Use “ls –l apple“. Which group is associated with this file?  touch apple  ls –l apple  $ touch apple  $ ls -l  total 0  -rw-rw-r--. 1 robert robert 0 Apr 27 11:26 apple  Type “id” to confirm your primary group is ‘*robert’*. Use the command “newgrp cluster”. Type “id” to confirm your primary group is now *cluster*. Use the touch command to create a file called “mac”, followed by “ls –l mac”. With which group is mac associated?  touch mac  $ ls -l mac  -rw-r--r--. 1 robert cluster 0 Apr 27 11:30 mac  Do not log out from this session but allow it to run alongside your other window, the one where you are the ‘root’ user. |

Please use the window where you are the root user and the window in which you are logged in as robert in parallel.

As you may know, there are extra permission bits beyond those used to set read, write and execute permissions. These are

* the sticky user ID - 4
* the sticky group ID - 2
* the sticky bit - 1

These ‘advanced’ permission bits have different effects on files compared to folders.

#### On Files

**The sticky User ID**, SUID, causes an executable file to run with the user ID of the owner of the file as opposed to running with the ID of the user who ran the command.

**The sticky Group ID**, SGID, is the same as SUID but for groups.

The so called **sticky bit** has no effect,

#### On Folders

The **SUID** bit has no effect.

**The SGID bit causes files created within the directory to have their associated group set to the one associated with the parent directory, as opposed to the primary group associated with the user creating the file.**

The **sticky bit** is set on publicly writeable directories. It prevents one user deleting another user’s work. To delete a file, you not only need write permission on the parent directory but, in addition, you must own the file.

In the ‘**root’** window run the following command.

This is just the ps executable, now known by a new name.

# cp /bin/ps /tmp/stickyUserID

In the ‘**robert’** window run

$ /tmp/stickyUserID -f

UID PID PPID C STIME TTY TIME CMD

robert 43692 43689 0 16:04 pts/2 00:00:00 -bash

robert 43803 43692 0 16:06 pts/2 00:00:00 bash

robert 44613 43803 0 16:33 pts/2 00:00:00 /tmp/stickyUserID -f

Note: The stickyUserID process runs as Robert, because ‘robert’ ran the executable.

Now in the ‘**root’** window set SUID on /tmp/stickyUserID

# chmod 4755 /tmp/stickyUserID

# ls -l /tmp/stickyUserID

-rw*s*r-xr-x. 1 root root 100112 Apr 27 16:32 /tmp/stickyUserID

Note: The ‘4755;’ specifies the permission bits. The 4 sets the SUID, ‘755’ set the normal permissions. The little ‘s’, shows the SUID bit is set.

In the ‘**robert’** window re-run the /tmp/stickyUserID executable (with the –f flag).

$ /tmp/stickyUserID -f

UID PID PPID C STIME TTY TIME CMD

root 44740 43803 0 16:37 pts/2 00:00:00 /tmp/stickyUserID -f

Note: The process now runs with user ID ‘root’.

In the **root** window

# chgrp sys /tmp/stickyUserID # sys has group ID of 3

# chmod 6755 /tmp/stickyUserID

# ls -l /tmp/stickyUserID

-rwsr-sr-x. 1 root sys 100112 Apr 27 16:32 /tmp/stickyUserID

This time both SUID and SGID are set.

In the **robert** window run a special command to show the process user ID and group ID.

$ /tmp/stickyUserID -o uid,gid,pid

UID GID PID

0 3 45776

The associated group is group 3, the sys group. The UID 0 points to the root user.

In the root window:

# rm /tmp/stickyUserID

Terminate your ‘robert’ session.

#### Going on to look at the SGID bit applied to a directory.

As root, create a directory under root called ‘shared\_cluster’.

mkdir /shared\_cluster

Change the permissions on the shared\_cluster so the the owner and the associated group have read, write and execute permissions and ‘others’ have no permissions.

chmod 770 /shared\_cluster.

Check the permissions are as specified.

ls –ld /shared\_cluster

drwxrwx---. 2 root root 4096 Apr 27 12:06 /shared\_cluster

Now create a file in /shared\_cluster called test01

touch /shared\_cluster/test01

Add text into the file using the command ‘echo data for test’, redirected into test01.

echo “data for test” > /shared\_cluster/test01

Change the permissions on test01 to rw-rw---- (read,write for group and owner, nothing for ‘other’.)

chmod 660 /shared\_cluster/test01

Check the permissions on test01

ls -l /shared\_cluster/test01

**-rw-rw----. 1 root root 14 Apr 28 11:19 /shared\_cluster/test01**

**Start up a second terminal window, logging as ‘robert’.** Run the command:

cat /shared\_cluster/test01

What was the result?

Permission denied.

The command ‘chgrp’ changes the group associated with a file or directory. You have to be the owner of the file/directory or superuser to be able to do this. The option ‘-R’ makes the command recursive, so that it will work on all items in the tree from a given point downwards. Run the command ‘chgrp’ to recursively change the group associated with ‘shared\_cluster’ from root to the group ‘cluster’. You may use the man command to look up the syntax.

chgrp –R cluster /shared\_cluster

Now check the permissions on /shared\_cluster/test01, not it is now associated with the ‘cluster’ group.

ls -l /shared\_cluster/test01

-rw-rw----. 1 root cluster 14 Apr 27 12:17 /shared\_cluster/test01

Using the terminal in which you are logged in as ‘robert’: Run the command:

cat /shared\_cluster/test01

What was the result?

data for test (i.e. it worked.)

Why did it work?

Because Robert is a (secondary) member of the cluster group, which has read permission to the file.

Besides ‘root’, who has write permission to /shared\_cluster?

Anyone in the group cluster

As the user ‘robert’, create a file called ‘test02’ in /shared\_cluster and display its permissions.

$ touch /shared\_cluster/test02

$ ls -l /shared\_cluster/test02

**rw-rw-r--. 1 robert robert 0 Apr 28 11:46 /shared\_cluster/test02**

Can the user Alison read this file?

Yes.

Can the user Alison edit this file?

No.

To an extent, /shared\_cluster is acting as a folder containing shared files by allowing any member of the group cluster to create files within it. However, the files are associated with each user’s private group. It would be better if the files in this shared directory were automatically associated with the ‘cluster’ group because that would give access via the group permissions to everone in the ‘cluster’ group. That is where the SGID permissions bit comes in.

In the root window set the sticky group ID bit and show the permissions on /shared\_cluster.

# chmod g+s /shared\_cluster

# ls -ld /shared\_cluster

drwxrws---. 2 root cluster 4096 Apr 28 12:10 /shared\_cluster

As user ‘robert’, create file /shared\_cluster/test03, list the permissions.

**$ touch /shared\_cluster/test03**

**$ ls -l /shared\_cluster/test03**

**-rw-rw-r--. 1 robert cluster 0 Apr 28 12:28 /shared\_cluster/test03**

Can the user Alison read this file?

Yes.

Can the user Alison edit this file?

Yes.

Login as Alison, write the text “This text is from Alison” into the file test03 and check it worked.

$ echo “This text is from Alison” > /shared\_cluster/test03

$ cat /shared\_cluster/test03

This text is from Alison

Continuing as Alison, create a sub directory ‘alisonSub within /shared\_cluster. Look at its attributes and identify anything which differs from ‘normal’.

mkdir /shared\_cluster/alisonSub

[alison@ML-RefVm-964666 ~]$ ls -l /shared\_cluster/

total 12

drwxrwsr-x. 2 alison cluster 4096 Apr 28 14:00 alisonSub

-rw-rw----. 1 root cluster 14 Apr 28 13:44 test01

-rw-rw-r--. 1 robert robert 0 Apr 28 13:45 test02

-rw-rw-r--. 1 robert cluster 26 Apr 28 13:45 test03

The directory alisonSub has the SGID bit set automatically and is associated with the ‘cluster’ group.

Logout as Alison and exit from any sessions where you are logged in as Robert. Tidy up by running the following commands (as root).

userdel –r robert

userdel –r alison

rm –rf /shared\_cluster