Assignment 03

1. Print the full name, user-name and user-ids of the currently logged in users. Create a file MyFile1.txt. Find out the permissions for newly created file. Change the group permission to executable on newly created file. Check whether changinging accomplished or not. # Currently Logged in Users  
   w  
   # Create file MyFile1.txt  
   touch MyFile1.txt  
   # Permissions for a file  
   ls -l MyFile1.txt  
   # Changing permission for file to executable  
   chmod u+x MyFile1.txt  
   # To check whether accomplished or not  
   ls -l MyFile1.txt
2. Display the resource utilization of each running processes in the system. Check how much memory is free and also which process has maximum share in the CPU and memory.  
   # To display the resource utilization sorted according to memory use.  
   ps -eo pid,ppid,cmd,%mem,%cpu --sort=-%mem  
   # To display memory usage  
   free -m  
   # To display the process with maximum share in CPU  
   top  
   # To display the process with maximum memory usage   
   top -o %MEM
3. List how many processes running in the system. Check the owner information of each process. Change the owner to some other user.  
     
   # To display all process running in the system  
   ps aux  
   # To display the count of all running process   
   ps aux | wc -l  
   # To display processes running by particular user here “root”  
   ps -U root  
   # For more detailed output  
   ps aux -U root  
   # To change ownership of the process there is **NO COMMAND**
4. How many CPU cores does the machine have? How much memory, and what fraction of it is free? How many context switches has the system performed since bootup? How many processes has it forked since bootup?  
     
   # To display the no. of CPU cores   
   nproc  
   # To display the Total and used memory   
   free -m  
   # To display % of memory free  
   % free | grep Mem | awk '{print $4/$2 \* 100.0}'  
     
   # To display no of processes forked since bootup  
   cat /proc/stat | grep ‘processes’
5. Every process consumes some resources (CPU, memory, disk or network bandwidth, and so on). When a process runs as fast as it can, one of these resources is fully utilized, limiting the maximum rate at which the process can make progress. Such a resource is called the bottleneck resource of a process. A process can be bottlenecked by different resources at different points of time, depending on the type of work it is doing.  
   Run four processes separately, and identify what the bottleneck resource for each is.   
     
   # After running the process take it pid   
   # Command like top helps us monitor CPU, memory etc.  
   top -p $PID  
   # Command like iotop helps us monitor Disk usage  
   iotop -p $PID  
   # Iotop may need root privileges
6. Every process runs in one of two modes at any time: user mode and kernel mode. It runs in user mode when it is executing instructions / code from the user. It executes in kernel mode when running code corresponding to system calls etc. Compare the programs in terms of the amount of time each spends in the user mode and kernel mode, using information from the file system.   
     
   # Command to see execution time of a script  
   time scriptname  
   # For the ls commands  
   time ls  
   # For a script named new.sh  
   time ./new.sh
7. Recall that a running process can be interrupted for several reasons. When a process must stop running and give up the processor, it’s CPU state and registers are stored, and the state of another process is loaded. A process is said to have experienced a context switch when this happens. Context switches are of two types: voluntary and involuntary. A process can voluntarily decide to give up the CPU and wait for some event, e.g., disk I/O. A process can be made to give up its CPU forcibly, e.g., when it has run on a processor for too long, and must give a chance to other processes sharing the CPU. The former is called a voluntary context switch, and the latter is called an involuntary context switch. Compare the processes in terms of the number of voluntary and involuntary context switches.  
   # To see voluntary and non voluntary context switches of a process of pid=pid.  
   grep ctxt /proc/$pid/status