Operating System

Lab-2

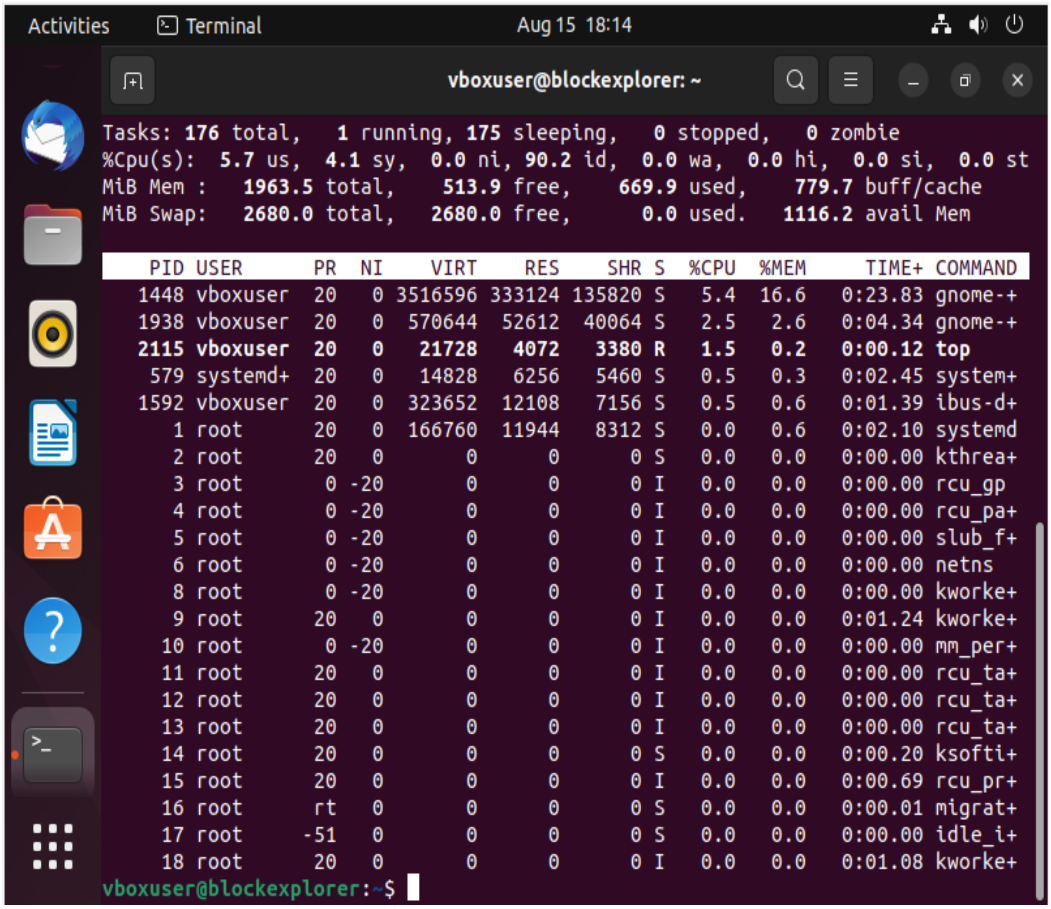
System Monitoring in Linux

Commands

1. top: The top command in Linux is a real-time system monitoring tool that provides an interactive and dynamic view of the system resource utilization. It provides information about various system metrics, including CPU usage, memory usage, load averages, running processes and more.

   Usage: top –n 5 d -2

   This command runs top command for 5 iterations with a 2 sec delay between updates.

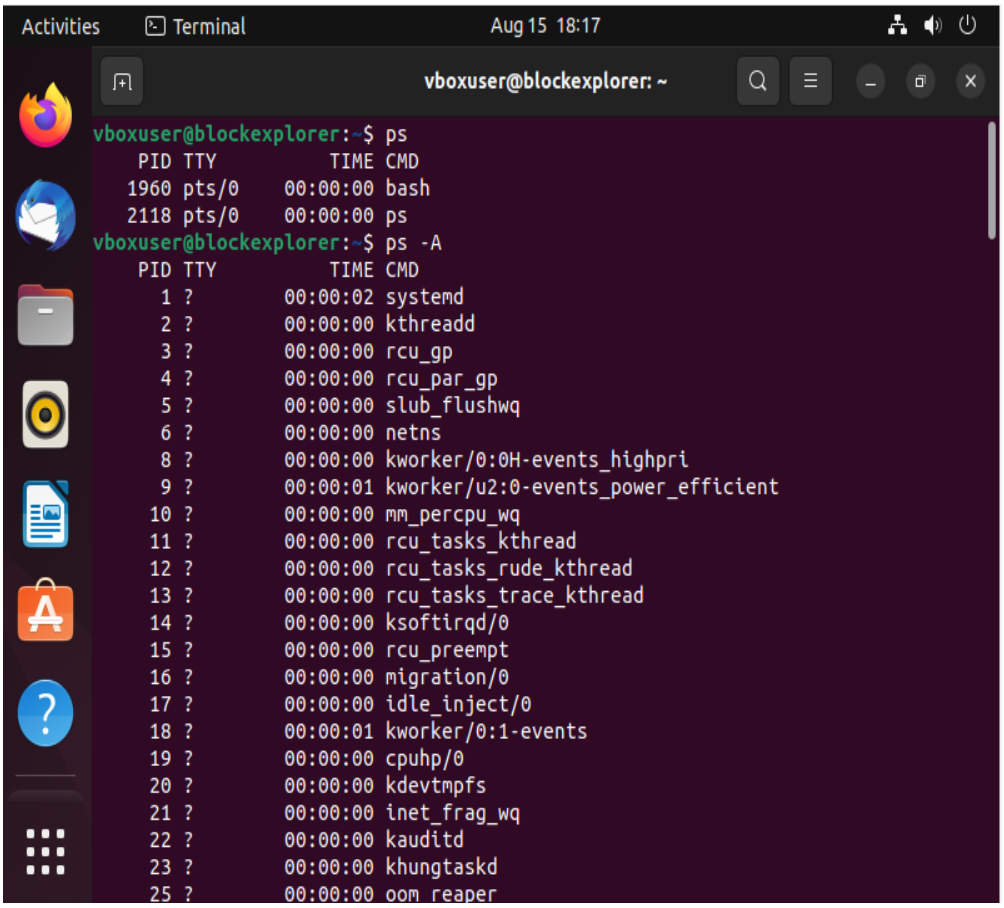


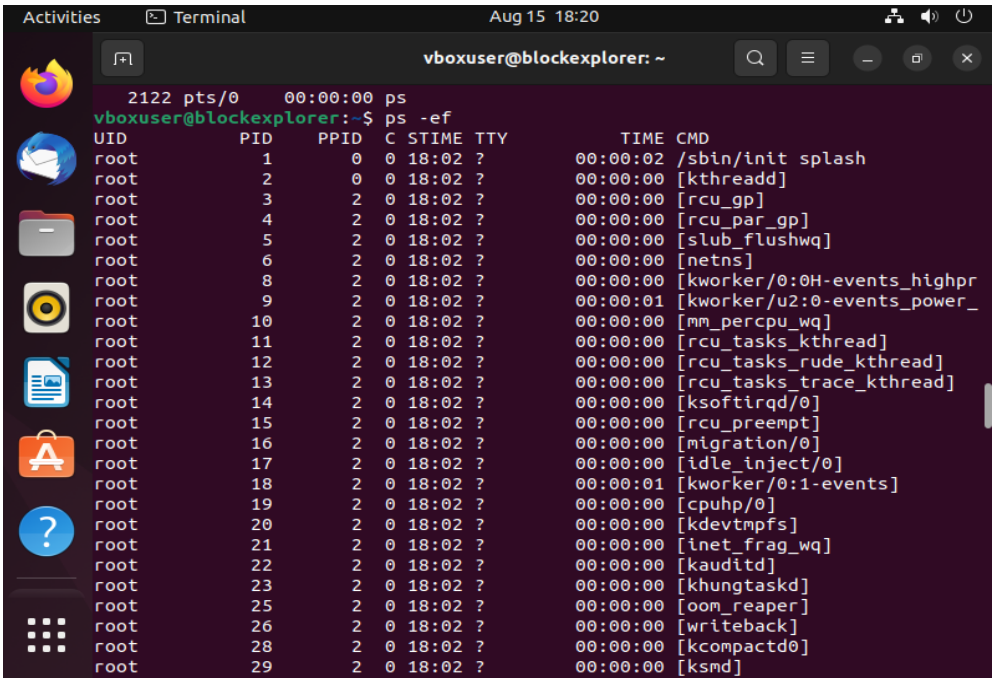
2. ps: The ps (process status) command is used to display all the running processes in the Linux system.

   In the output of ps the fields are:

   1. PID: Unique Process id
   2. TTY: terminal type of the user logged into.
   3. Time: amount of CPU time in minutes and seconds that the process has been running.
   4. CMD: name of the command that launched the process.

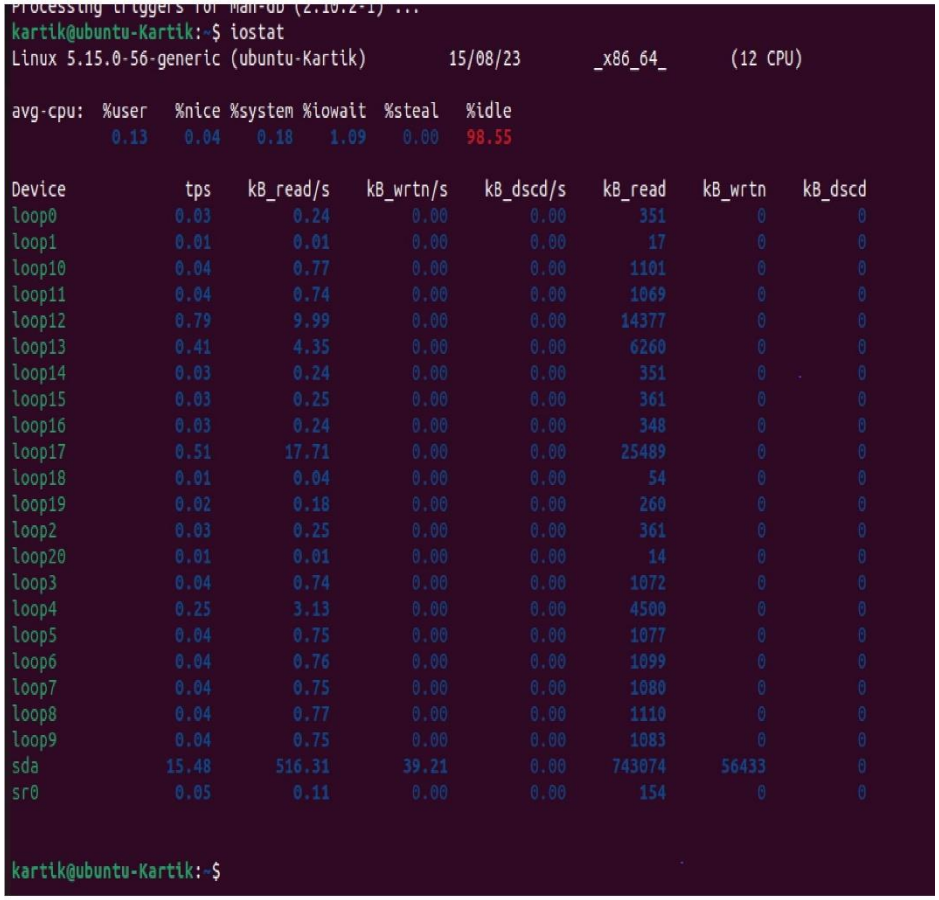
   Usage: ps –e


Activities
Terminal
Aug 15 18:17
vboxuser@blockexplorer: ~
vboxuser@blockexplorer:~$ ps
PID TTY TIME CMD
1960 pts/0 00:00:00 bash
2118 pts/0 00:00:00 ps
vboxuser@blockexplorer:~$ ps -A
PID TTY TIME CMD
1 ? 00:00:02 systemd
2 ? 00:00:00 kthreadd
3 ? 00:00:00 rcu_gp
4 ? 00:00:00 rcu_par_gp
5 ? 00:00:00 slub_flushwq
6 ? 00:00:00 netns
8 ? 00:00:00 kworker/0:0H-events_highpri
9 ? 00:00:01 kworker/u2:0-events_power_efficient
10 ? 00:00:00 mm_percpu_wq
11 ? 00:00:00 rcu_tasks_kthread
12 ? 00:00:00 rcu_tasks_rude_kthread
13 ? 00:00:00 rcu_tasks_trace_kthread
14 ? 00:00:00 ksoftirqd/0
15 ? 00:00:00 rcu_preempt
16 ? 00:00:00 migration/0
17 ? 00:00:00 idle_inject/0
18 ? 00:00:01 kworker/0:1-events
19 ? 00:00:00 cpuhp/0
20 ? 00:00:00 kdevtmpfs
21 ? 00:00:00 inet_frag_wq
22 ? 00:00:00 kauditd
23 ? 00:00:00 khungtaskd
25 ? 00:00:00 oom_reaper


Activities
Terminal
Aug 15 18:20
vboxuser@blockexplorer: ~
2122 pts/0 00:00:00 ps
vboxuser@blockexplorer:~$ ps -ef
UID PID PPID C STIME TTY TIME CMD
root 1 0 0 18:02 ? 00:00:02 /sbin/init splash
root 2 0 0 18:02 ? 00:00:00 [kthreadd]
root 3 2 0 18:02 ? 00:00:00 [rcu_gp]
root 4 2 0 18:02 ? 00:00:00 [rcu_par_gp]
root 5 2 0 18:02 ? 00:00:00 [slub_flushwq]
root 6 2 0 18:02 ? 00:00:00 [netns]
root 8 2 0 18:02 ? 00:00:00 [kworker/0:0H-events_highpr
root 9 2 0 18:02 ? 00:00:01 [kworker/u2:0-events_power_
root 10 2 0 18:02 ? 00:00:00 [mm_percpu_wq]
root 11 2 0 18:02 ? 00:00:00 [rcu_tasks_kthread]
root 12 2 0 18:02 ? 00:00:00 [rcu_tasks_rude_kthread]
root 13 2 0 18:02 ? 00:00:00 [rcu_tasks_trace_kthread]
root 14 2 0 18:02 ? 00:00:00 [ksoftirqd/0]
root 15 2 0 18:02 ? 00:00:00 [rcu_preempt]
root 16 2 0 18:02 ? 00:00:00 [migration/0]
root 17 2 0 18:02 ? 00:00:00 [idle_inject/0]
root 18 2 0 18:02 ? 00:00:01 [kworker/0:1-events]
root 19 2 0 18:02 ? 00:00:00 [cpuhp/0]
root 20 2 0 18:02 ? 00:00:00 [kdevtmpfs]
root 21 2 0 18:02 ? 00:00:00 [inet_frag_wq]
root 22 2 0 18:02 ? 00:00:00 [kauditd]
root 23 2 0 18:02 ? 00:00:00 [khungtaskd]
root 25 2 0 18:02 ? 00:00:00 [oom_reaper]
root 26 2 0 18:02 ? 00:00:00 [writeback]
root 28 2 0 18:02 ? 00:00:00 [kcompactd0]
root 29 2 0 18:02 ? 00:00:00 [ksmd]

3. iostat: The iostat command in Linux is used to monitor and display the input/output statistics of block devices and partitions. It provides valuable information about the performance and utilization of various storage devices including hard drives, solid-state drives and other storage devices. This can help administrators and users to identify potential bottlenecks, monitor disk activity and optimize system performance. The iostat command provides information about various aspects of I/O performance including:

    1. Disk Utilization: Displays the percentage of time a device's disk is active and serving I/O requests.
    2. Disk I/O operation: Number of read and write operations being performed on the disk per second.
    3. Average Wait Time: Provides information about the average time an I/O request spends in queue waiting to be served.
    4. Transfer Rates: The amount of data being read or written to the disk per second.
    5. CPU Utilization: How much of CPU's process power is being used to handle I/O operation.



4. strace: The strace command in Linux is used to trace and analyze the system calls and the signals made by a process during its execution. It provides a detailed view of the interactions between a user and the operating system, showing the sequence of system calls, their arguments and their return values. This can be useful for debugging, profiling and understanding how a program interacts with the underlying system.

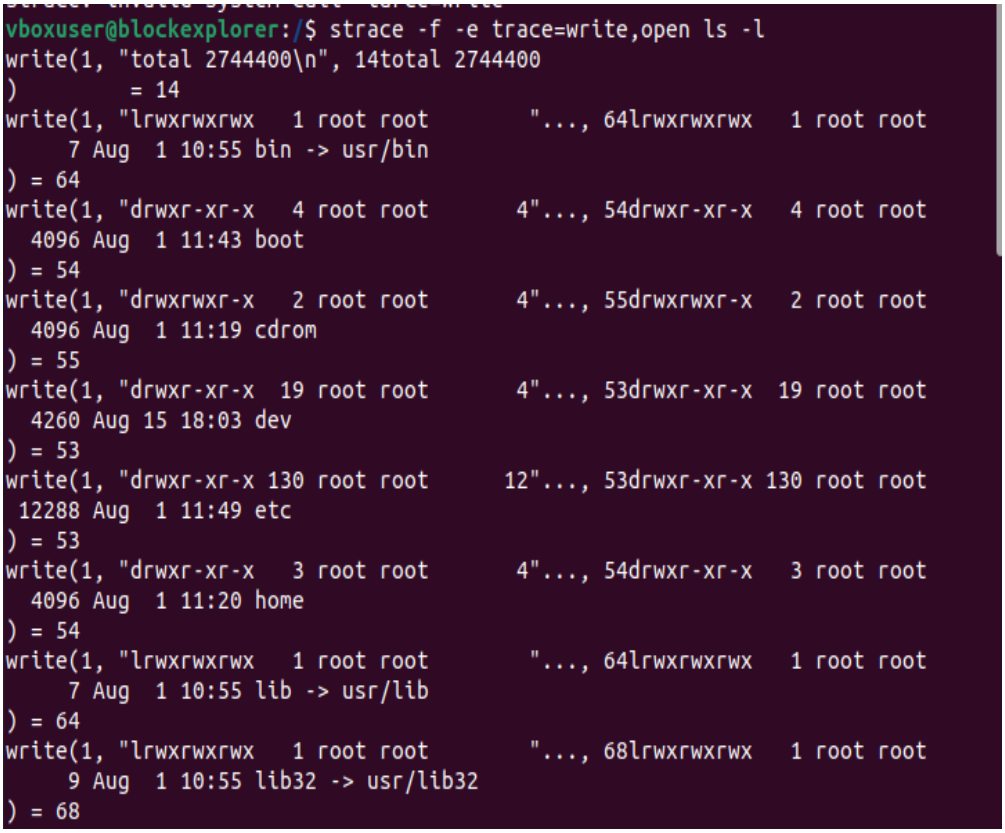
   Usage: *strace –o output.txt –f –e trace=write,open*
   ls -l

-o output.txt : specifies that the output should be saved to a file named output.txt.

-f: follows child process

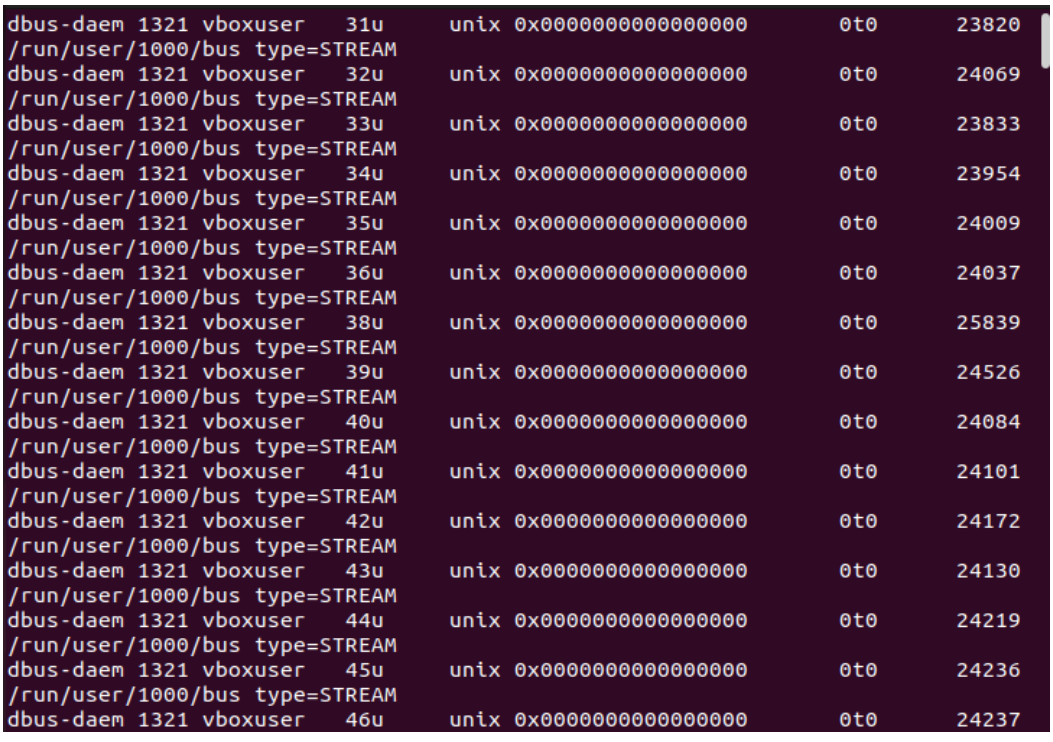
-e traces=write,open: specifies which system calls to trace. In this case it is tracing the write and open system calls.

ls –l: denotes the command you want to trace.



5. lsof: The lsof command (short for list open files) command in Linux is used to display information about the files that are currently open by various processes on the system. This command provides information to which files or new sockets are being used by the running processes, along with the details about the processes themselves. It helps in administrating, troubleshooting and identifying resource usage.

   Usage: lsof –i TCP:80 : This command lists all the processes with open TCP sockets on port80, which is commonly used for HTTP traffic. This helps to see which processes are listening on or connecting to that port.



6. lsblk: The lsblk command is used to display information about block devices, which includes storage devices such as hard drives, solid state drives, and other devices that use block I/O subsystem.

   Usage: lsblk

```
vboxuser@blockexplorer:/$ lsblk
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
loop0    7:0    0     4K  1 loop /snap/bare/5
loop1    7:1    0  63.3M  1 loop /snap/core20/1822
loop2    7:2    0  63.4M  1 loop /snap/core20/1974
loop3    7:3    0  73.9M  1 loop /snap/core22/817
loop4    7:4    0  73.9M  1 loop /snap/core22/858
loop5    7:5    0 240.6M  1 loop /snap/firefox/2356
loop6    7:6    0 237.2M  1 loop /snap/firefox/2987
loop7    7:7    0 346.3M  1 loop /snap/gnome-3-38-2004/119
loop8    7:8    0 349.7M  1 loop /snap/gnome-3-38-2004/143
loop9    7:9    0 485.5M  1 loop /snap/gnome-42-2204/120
loop10   7:10   0 485.5M  1 loop /snap/gnome-42-2204/126
loop11   7:11   0  91.7M  1 loop /snap/gtk-common-themes/1535
loop12   7:12   0  45.9M  1 loop /snap/snap-store/638
loop13   7:13   0  12.3M  1 loop /snap/snap-store/959
loop14   7:14   0  49.8M  1 loop /snap/snapd/18357
loop15   7:15   0  53.3M  1 loop /snap/snapd/19457
loop16   7:16   0   304K  1 loop /snap/snapd-desktop-integration/49
loop17   7:17   0   452K  1 loop /snap/snapd-desktop-integration/83
sda      8:0    0    25G  0 disk
├─sda1   8:1    0     1M  0 part
├─sda2   8:2    0   513M  0 part /boot/efi
└─sda3   8:3    0  24.5G  0 part /var/snap/firefox/common/host-hunspell
```

- Proc File System

A proc file system is a temporary virtual file system which is established when the system boots up and is dissolved when the system shuts down. The proc file system serves as a command and information hub for the Kernel and

provides essential information about the processes that are currently active. A channel of communication between Kernel and the user space is also offered via the proc file system.

Syntax: ls –l /proc

This command will list down all the files and directories under the '/proc' directory with detailed information like permissions, ownership, size and time of modification useful for understanding the current status of our system and diagnosing the problems.