Process Management Exercise

**Purpose:** The purpose of this exercise is to familiarize students with managing processes in Windows and Linux and demonstrate changes in process states.

**Instructions:** The student should first make a copy of the ***Turn-In Sheet*** (at the end of the exercise) to a separate document. As the student works through the exercise, they should make the appropriate entries on the ***Turn-In Sheet***. Once the exercise is completed, the student will save and post the ***Turn-In Sheet*** on the assignment in Blackboard.

NOTES:

* **Users of Different OS** (e.g. Mac, or later versions of Windows OS): Please copy this document: *Update/adjust the [instructions + screen captures] with your results. Your clever/better way of doing the tasks are welcome especially* 😊.
* *“****Turn-in Sheet****”* (your data) can be seen at the bottom of this document.
* **Submission:**

Please submit to online. If that fails, email your results to: bangpanliang@gmail.com

The subject of the email should be: [Your StudentID, Assignment Name]

# Windows

This exercise will start using the command shell in Windows.

The first thing we will look at is how to find all of the active tasks or processes in Windows. To do this we can use the command:

tasklist

The result looks like this:

C:\MinGW\bin>tasklist

Image Name PID Session Name Session# Mem Usage

========================= ====== ================ ======== ============

System Idle Process 0 Console 0 28 K

System 4 Console 0 56 K

smss.exe 892 Console 0 76 K

csrss.exe 940 Console 0 2,704 K

winlogon.exe 964 Console 0 2,176 K

services.exe 1008 Console 0 1,976 K

lsass.exe 1020 Console 0 4,008 K

svchost.exe 1184 Console 0 1,764 K

svchost.exe 1324 Console 0 2,060 K

svchost.exe 1448 Console 0 32,424 K

svchost.exe 1528 Console 0 1,448 K

svchost.exe 1564 Console 0 1,784 K

ccSetMgr.exe 1716 Console 0 480 K

ccEvtMgr.exe 1848 Console 0 2,228 K

SPBBCSvc.exe 1952 Console 0 2,048 K

spoolsv.exe 2008 Console 0 1,528 K

svchost.exe 1272 Console 0 156 K

DefWatch.exe 1504 Console 0 180 K

NetworkViewerService.exe 1552 Console 0 1,072 K

jqs.exe 1828 Console 0 1,836 K

mdm.exe 272 Console 0 532 K

sqlservr.exe 300 Console 0 20,976 K

nvsvc32.exe 524 Console 0 308 K

SavRoam.exe 664 Console 0 788 K

It gives us a list of each task, with its Process ID and Memory Usage. This is useful information, but we may want more. To get this we can use the form:

tasklist /v

The results look like:

Image Name PID Session Name Session# Mem Usage Status

User Name CPU Time Window T

itle

========================= ====== ================ ======== ============ ========

======= ================================================== ============ ========

================================================================

System Idle Process 0 Console 0 28 K Running

NT AUTHORITY\SYSTEM 376:06:47 N/A

System 4 Console 0 56 K Running

NT AUTHORITY\SYSTEM 0:18:13 N/A

smss.exe 892 Console 0 76 K Running

NT AUTHORITY\SYSTEM 0:00:00 N/A

csrss.exe 940 Console 0 3,552 K Running

NT AUTHORITY\SYSTEM 0:02:33 N/A

winlogon.exe 964 Console 0 2,176 K Running

NT AUTHORITY\SYSTEM 0:00:10 N/A

services.exe 1008 Console 0 1,976 K Running

NT AUTHORITY\SYSTEM 0:52:28 N/A

explorer.exe 2112 Console 0 12,976 K Running

UNIV\fretheim 0:07:48 N/A

GrooveMonitor.exe 3900 Console 0 924 K Running

UNIV\fretheim 0:00:00 N/A

igfxsrvc.exe 1524 Console 0 80 K Running

UNIV\fretheim 0:00:01 OleMainT

hreadWndName

jusched.exe 412 Console 0 60 K Running

UNIV\fretheim 0:00:00 OleMainT

hreadWndName

OUTLOOK.EXE 3428 Console 0 66,088 K Running

UNIV\fretheim 0:11:58 Inbox -

Microsoft Outlook

jucheck.exe 3600 Console 0 180 K Running

UNIV\fretheim 0:00:00 JavaUpda

te SysTray Icon

WINWORD.EXE 5392 Console 0 61,400 K Running

UNIV\fretheim 0:12:09 Document

7 - Microsoft Word

iexplore.exe 5876 Console 0 13,144 K Running

UNIV\fretheim 0:03:00 SysFader

This may be a bit challenging to read because each entry is spaced across several lines. I have highlighted three entries which we are going to take a closer look at.

One of the important differences of this view is that we can see who owns or started the task. We note that many of the tasks have an owner of “NT AUTHORITY/SYSTEM”. These are tasks that are controlled by the operating system, or launched on start-up rather than by the user.

The first process we’ll observe is csrss.exe. This is the Client Server Runtime Server Subsystem. It is used by Windows to manage part of the Windows system, including creating threads for Windows. That makes it a pretty essential process. Because of that, the “Task Manager” window will not allow you to stop the process. Because of this, some virus writers have decided that this is a pretty good process to use for their virus. If you see more than one of these running in your process list, the chances are pretty good that one of them is a virus.

1. Start “Notepad”, then run “tasklist /v” and paste a copy of the results in the ***Turn-In Sheet***. Highlight the “notepad” process and put the PID in the space.

Once we discover the second copy of a csrss.exe or some other process we want to get rid of (perhaps it is not responding) we can do that using “taskkill”. We can either use the process number, for example if I want to get rid of the Java Update Checker, I could use the command:

C:\MinGW\bin>taskkill /PID 3600

SUCCESS: The process with PID 3600 has been terminated.

On the other hand, I may want to close my notepad session, which I can do by using the image name switch.

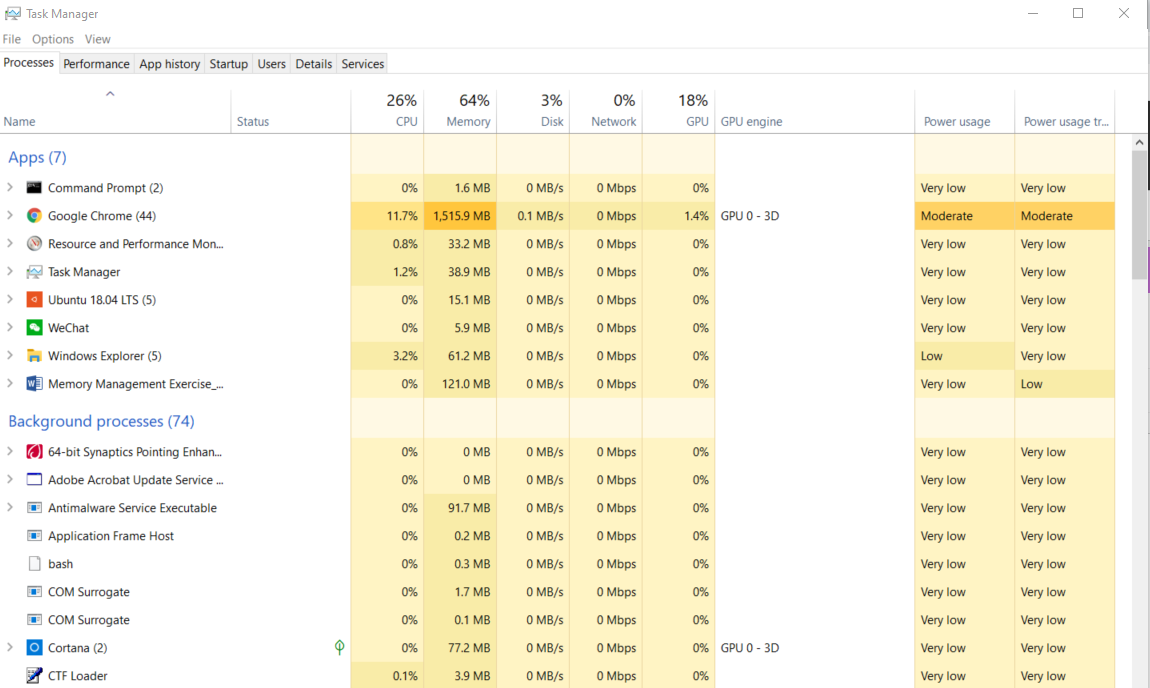
C:\MinGW\bin>taskkill /IM notepad.exe

SUCCESS: The process "notepad.exe" with PID 2624 has been terminated.

1. Use “taskkill” to terminate the notepad process and copy the command and results to the ***Turn-In Sheet***.

Going back to Windows now, we have another tool available for looking at Processes. This is the Task Manager. Generally, we start the Task Manager by using “ctrl-alt-delete” and then selecting the “Task Manager”.

On the Task Manager, select the “Processes” tab. An example is shown below:



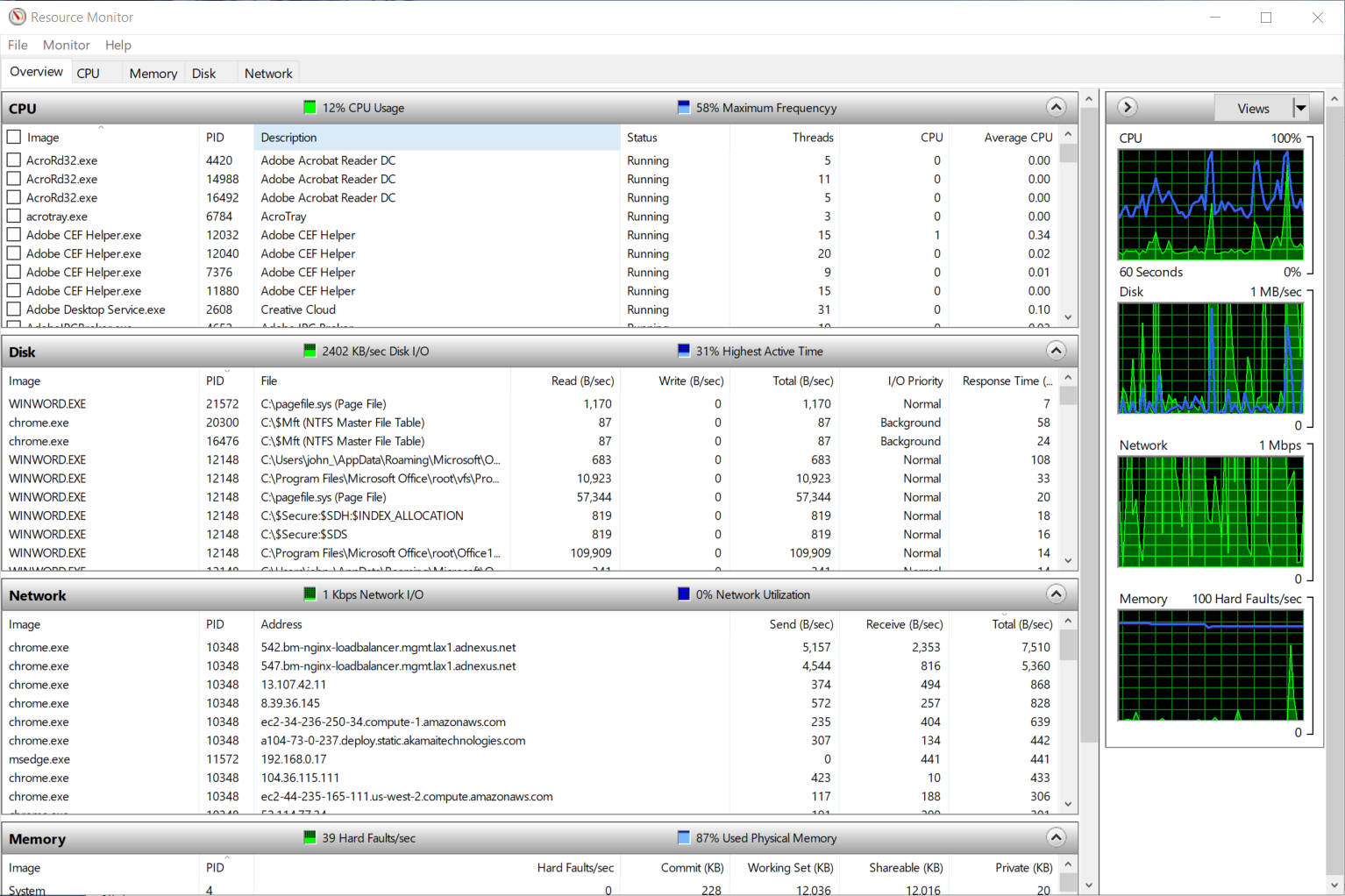
Launch a new instance of “Notepad” and then locate “notepad.exe” in the processes.

1. Make a copy of the “Task Manager” and put it on the ***Turn-in Sheet***. Highlight “notepad.exe”.

Note the number of threads that it has. Now select “File Save as..” on Notepad. Then check the number of threads associated with “Notepad”. Can you explain the difference?

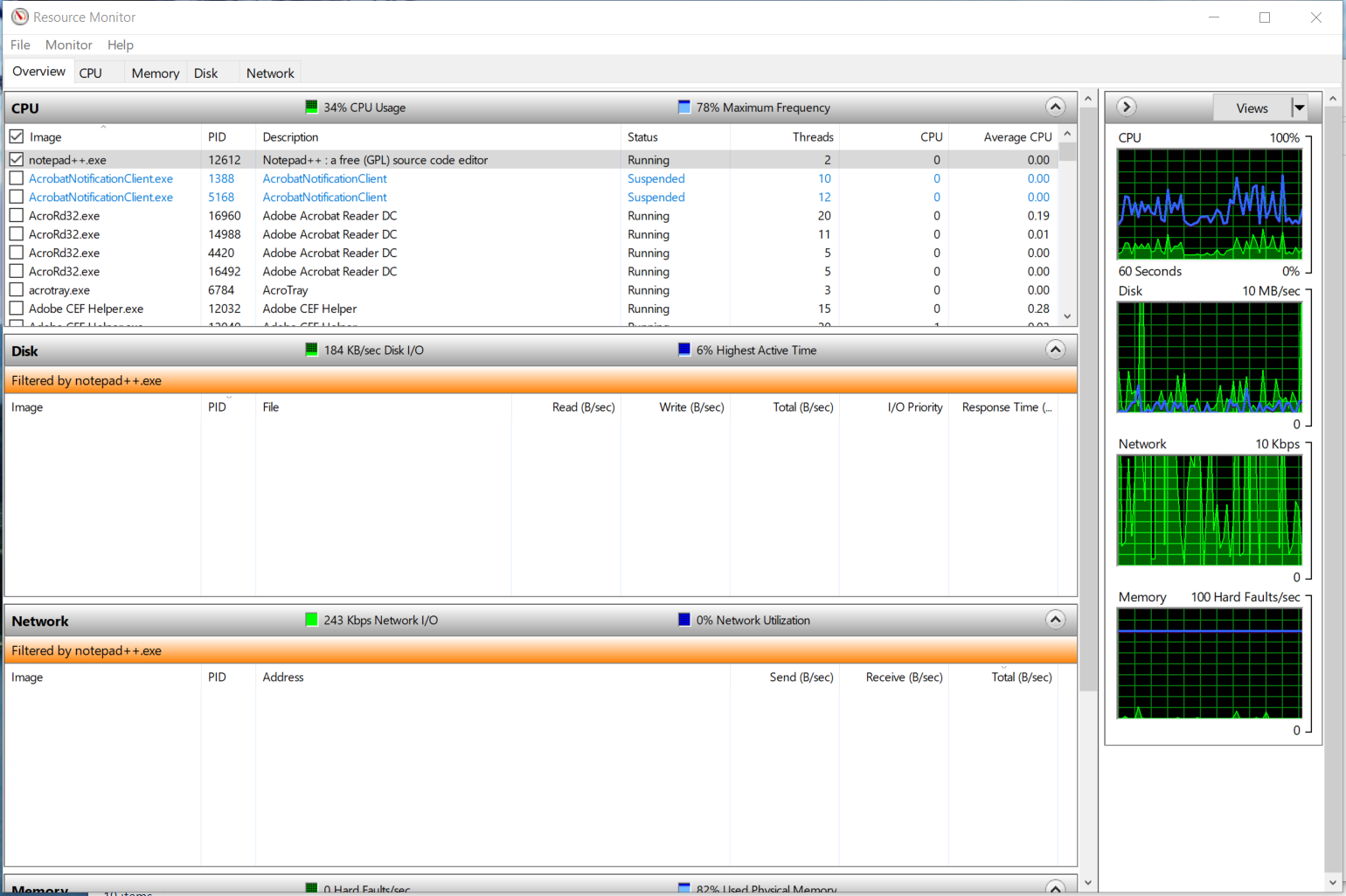
1. Place your explanation on the ***Turn-In Sheet***.

Now let’s look one layer deeper. We’ll explore Microsoft Window 10 OS’s “Resource Monitor”. The tool can be launched simply by typing “Resource Monitor” at the MS Window Home prompt. An example is shown below.



Note: The column attributes in each panel can be sorted by just clicking onto it, as highlighted by the “Red Arrow”. This way, you can locate the needed information quickly.

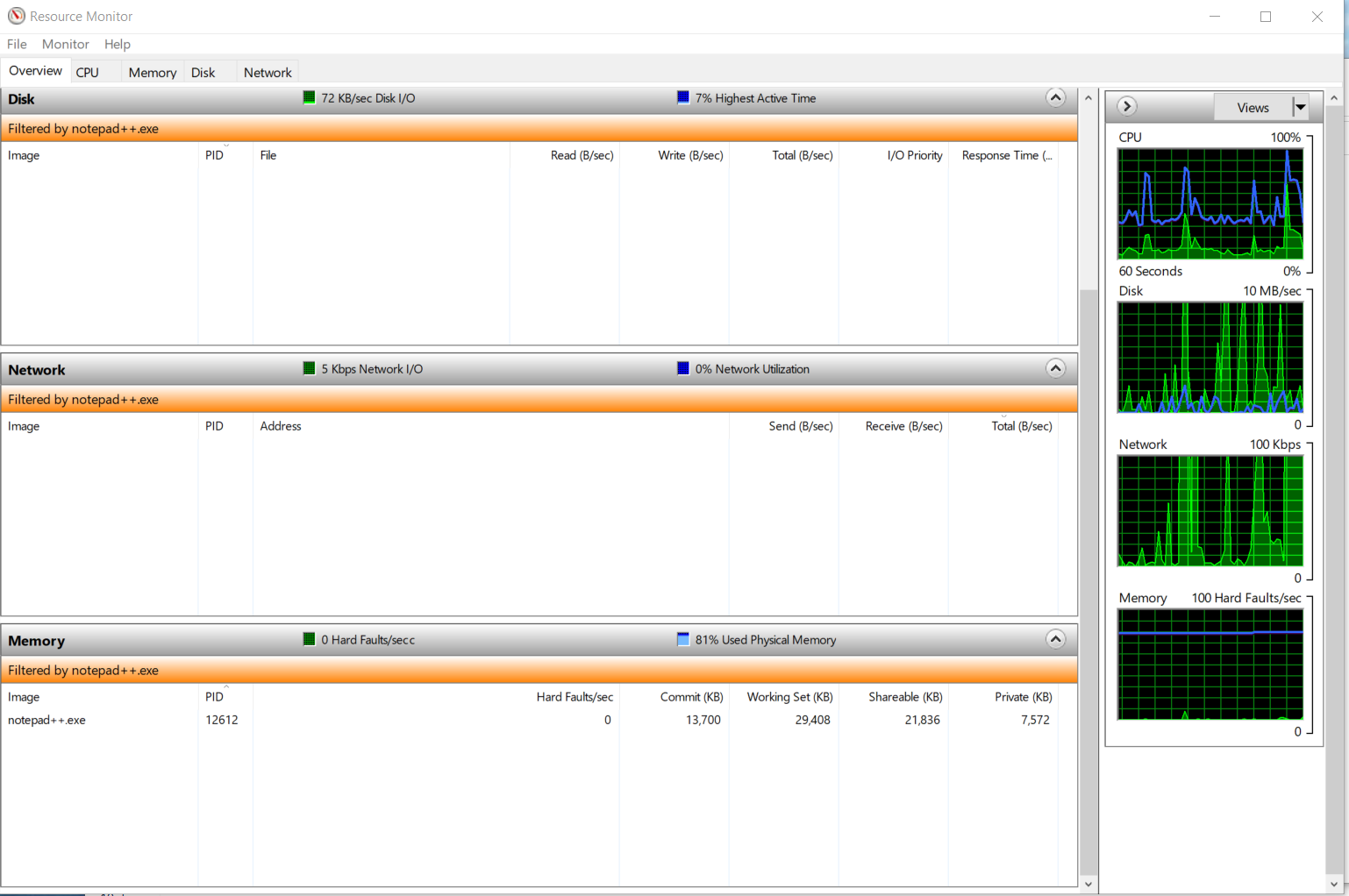
Next, we will set up a filter, so we only see items associated with “Notepad” or “Notepad++”. An example is shown below.



1. Place a copy of “Resource Monitor” showing only threads for “Notepad”, or “Notepad++” on the ***Turn-In Sheet***.

We can see on the Resource Monitor where the CPU, Memory, Disk and Network usage information for each process. When we look at the Task Manager, we can see one thread listed for “notepad.exe”.

1. Place a second copy of “Resource Monitor” on the ***Turn-In Sheet***. List the PID for “notepad” and its memory usage information. An example is shown below.



Bonus [Step 7 – 8]: Let’s download a set of utilities from Microsoft. These are called PsTools. There are a number of different utilities in the toolbox, but the one we will look at is PsList. The download is at:

<http://technet.microsoft.com/en-us/sysinternals/bb896682>

Download PsTools, extract it from the Zip file, and put it in a folder where it will be easy to get to from the Command Line. (I put mine in the same folder as Process Monitor).

Close Notepad, if it is open, and start a new one.

Open a Command Window and go to the folder with PsTools in it.

Now enter the command:

psList notepad

You will see a listing for notepad which shows the process ID, Priority, number of threads and other information about notepad.

H:\Process Monitor>pslist not

pslist v1.29 - Sysinternals PsList

Copyright (C) 2000-2009 Mark Russinovich

Sysinternals

Process information for BG3SOMEXF1:

Name Pid Pri Thd Hnd Priv CPU Time Elapsed Time

notepad 1136 8 1 41 1044 0:00:00.046 0:00:03.812

(Notice that I only had to enter enough of the process name to make it unique). Now select File->Save As.. on Notepad again, and then run another PsList. This time use the Thread Details (/d) flag:

Pslist /d note

This will give you the list of all of the Threads associated with “Notepad” and their details.

1. Enter your results on the ***Turn-In Sheet*** and explain the difference between the User Time and Kernel Time.
2. Explain, on the ***Turn-in Sheet***, how you might use these tools and techniques to make a change to a program.

# Linux Processes

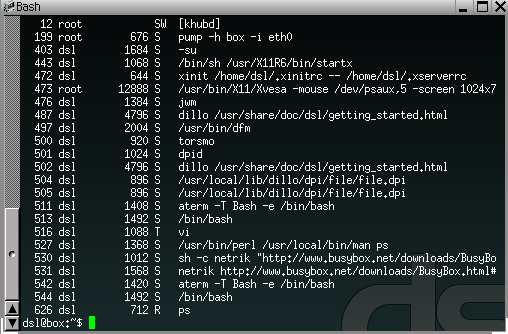
I assume you have Linux setup on your Windows 10 PC. If not, I recommend doing so. This is a useful link that get me going: <https://docs.microsoft.com/en-us/windows/wsl/install-win10>

This portion will be done in a command or terminal shell on your Linux System.

At the command prompt, start an edit session in “ping”-command.

Now that we have a process running, we can’t get to the command prompt immediately. If we wanted to we could kill the process by using the keys “ctrl c”. But instead we want to suspend the process. We do that with a “ctrl z”.

If we use the command “ps ” or “ps axx” we will get a list of processes which may be rather lengthy:



If we want, we can run the command and send the output through “grep” which is a regular expression filter:



ps axx | grep “pini”

[The ‘|’ or pipe character is one that isn’t often used. It is generally located just

above the enter key and is done with a “shift \”.

1. Place the results of your “ps | grep “ping” on the ***Turn-In Sheet***. Enter the “ping” PID.

The ‘|’ or pipe character tells the system to send the output of the first command to the input of the second. The “ping” is the argument for ‘grep’. It can be any regular expression and doesn’t need to be in quotes unless there are spaces or other blank space characters. Grep is really handy. It can also be used with a file. For example, if you want to see everyone who has the sh shell on a system, you could do:

grep “/bin” /etc/passwd

To refine that to only get the names of the users, we could pipe the output to the ‘cut’ command:

grep “/bin” /etc/passwd | cut –f1 –d”:”

Here we tell the cut command we want field 1 and that the fields use the colon as a delimiter. We’ll get the list:

daemon

bin

sys

lp

proxy

nobody

1. Copy your “grep” command and results to the Turn-In Sheet. Also use “cat /etc/passwd” to get a complete copy of the password file. If your “grep” had no results, explain why.

Back to our “ps” example. The first field is the Process ID. The code interpretation in the “STAT” fields are are:

D uninterruptible sleep (usually IO)

R runnable (on run queue)

S sleeping

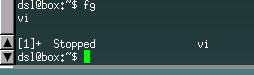
T traced or stopped

Z a defunct ("zombie") process

Based on this we can see that the “ping” process is stopped.

If we want we can also reactivate the process. If we want it to use the command line, we would put it back into the “foreground”, we do this using the “fg” command. Try it and you will see “ping” reappear.

Now use “ctrl z” to stop “ping” again.



This time we want to get rid of it entirely.

From our last “ps” command, we know that the process ID for “ping” is 516. We will use the “kill” command to terminate “ping”.

kill -9 516

Sometimes we use “kill” to send a message to a process. For example kill –HUP tells the process to hang up, or go through the process it would when it loses communications. “kill –HUP” will make some processes re-read their configuration files. “kill -9” tells processes to die immediately without going through a shut-down.

1. Put the “kill” command and results on the Turn-In Sheet. Explain two ways you can be sure the process has terminated?

Sometimes we want a process to run in the background – that is keep running, but let us have access to the command line. That is actually what the GUI is doing. In this case we could start the process, and then use “ctrl z”. Let’s try it. Type the command:

ping www.cityu.edu

1. Put the results of your “ps” command on the ***Turn-In Sheet***. Enter the PID for the “ping”.

Now type “bg”. This will put the process running in the background. If you use “ps” again you should see a different Status.

1. Explain the difference in Status on the ***Turn-In Sheet***.

If you exit the Command Window (use the command “exit”) the Terminal will still stay since it is running in the background.

If you don’t want to go through the start, “ctrl z”, bg set of commands to put a process in the background then simply put an ampersand “&” after the command and the operating system will start it as a background process:

ping [www.cityu.edu](http://www.cityu.edu) &

If you want to kill the “[ing” and are too dramatic to use the X icon, you can do so with the kill command.

Sometimes it is useful to know which processes are owned by other processes and what those owners are. For this there is the handy command:

pstree

This will draw a tree structure of the processes. If we use the switch ‘c’, we will see the process ID’s. The switch ‘a’ gives a longer listing which also includes the command line used to launch the process. This is very helpful if you are trying to locate a specific process from many similar, or if you want to know what switches and arguments were used when the process was launched. The format is:

pstree –ac

1. Run “pstree –ac” and put the results on the ***Turn-In Sheet***.

The final process tool we will look at in this exercise is:

top

“top” is used to provide a monitor of process activity. It will show the processes using the most resources on a single screen. “top” updates itself periodically and is useful for monitoring activity on the system. Run “top” on your system.

1. Take a snapshot of the output and place it on the **Turn-in Sheet**. Open an “ping” session or run another process and then take another snapshot of “top”. Place it on the **Turn-In Sheet**. Explain the differences between the two snapshots.

Another thing that is often useful is to find out how long a process takes and how much of that time is actually processing time. While we could do this using the other tools we’ve already looked at, pulling time off of the tools is sometimes a bit challenging, especially as the entry for each process goes away when the process terminates. But we can use the “time” command. The format is:

time command\_to\_be\_timed arguments and flags

As an example, let’s see how long it takes to start a browser, open the CityUniversity of Seattle homepage, and close the browser. The command is:

time ping -w5 www.cityu.edu

You may need to use another browser, for example, some versions of Damm Small Linux use the dillo browser. In fact we can test both to see which is faster:



The first set of times is for “dillo”.

1. Bonus: Use “time” to find how long it takes on your Linux system to open a browser, download the CityU website, and close the browser. Enter the displayed results on the ***Turn-In Sheet***. Explain the difference between the times for real, user and sys.

As you’ve seen there are a large number of tools available for managing processes in Windows and Linux. After this exercise, you should never find yourself at the mercy of the operating system again. All of these tools have many options and functions that we simply didn’t have time to explore. If you need to find out more about them, use the help switches, “/?” in Windows and “-h” or “-?” in Linux, the “Help” and “man” pages, a good book, or a web search.

1. On the **Turn-In Sheet**, write a short discussion of how you can use these tools to discover and manage a “hung” process.

Realize of course, that there are many more tools available for process management in Linux, many of which are graphically based.

Process Management Exercise - *Turn-in Sheet*

# Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Put the “tasklist” results here (give the PID for “notepad” \_\_\_\_\_\_\_\_\_\_ ):
2. Results from terminating “notepad”:
3. Copy of “Task Manager”:
4. Explain the difference in the number of threads here:
5. Copy of “Resource Monitor”:
6. Second Copy of “Resource Monitor”:

Notepad Thread ID: \_\_\_\_\_

Sub Thread ID’s: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. pslist results and Explanation of difference between User Time and Kernel Time:
2. Explain how you might use these tools and techniques to make a change to a Windows program:
3. Results of “ps | grep “vi” (“vi” PID \_\_\_\_\_\_\_\_\_ ):
4. Results from “grep” of the password file, and full copy of the file.
5. Results of the “kill” command:

Explain two ways you can be sure the process was terminated?

1. Results of the “ps” (“ping” PID \_\_\_\_\_\_ ):
2. Explanation of the difference in statuses:
3. “pstree” command and results:
4. Snapshots of “top” command:

Explain the differences:

1. Results of “time” command:

Explain the difference between the times for real, user and sys:

1. Discussion of how you can use these tools to discover and manage a “hung” process (be sure to define what a “hung” process is):