# OS Architecture Lab 7

In this lab we will be examining how Linux and your compiler handle memory management. Spend some time using the memory commands listed in the lab. Research some of the switches that can be used with each command. Once you understand the commands, run the code at the end of the section to see if you can spot what is going on in memory.

## **Some Memory Commands**

**top**

The top command breakss down CPU and memory usage by process and also shows total memory usage.

top comes in many versions, depending on which OS and package that you have installed. You can view the version you are using with **top -v**

Memory usage is broken down into swap and mem. Sqap is the amount of swap space used on the hard disk. Mem shows RAM usage. One thing to note is that **avail mem** indicates the amount of RAM that is available before the OS has to utilize swap space on the hard disk.

The tasks section of the command shows all of the running processes on the computer. The command shows the number of total processes, the number running, sleeping, stopped and zombie.

A process can be in several different states:

* **R** – Runnable indicates that the process is running or that it is ready to run and on the run queue.
* **S-** Uninterruptable Sleep means that the process is on hold waiting for an I/O operation to complete.
* **D-** Interruptible Sleep indicates that a process is waiting for an event to be completed.
* **T-** A stopped process was stopped by a signal from the job controller or by a trace. For example, we can
* **Z-** A zombie process is one that has completed, but has not yet been removed from the process table. A zombie process is in one being terminated.

**free**

The free command gives a simple listing of basic memory statistics. It is much less complex than top.

**/proc/meminfo**

The proc file system is a set of virtual files that contain information about the computer and software. the meminfo area of proc is specific to detailed memory information including total memory, free memory, buffers, cache, swap, write information and many other data points.

We can access information using cat /proc/meminfo

**vmstat [options] [delay [count]]**

This command breaks down virtual memory statistics and is useful in finding performance problem areas. There are m any switches for vmstat. Here are some of the more useful ones.

**delay** is the delay between updates. This allows the command to run over a period of time. Example: vmstat 3 5 indicates that we want the command to update the data every 3 seconds for a total of 5 times.

* **a** Displays active and inactive memory. Example: vmstat –a
* **f** Displays the number of fork system calls made since the last time OS was booted. Example: vmstat –f
* **t** Displays a timestamp along with system data. We could combine with the delay example like this: vmstat -t 3 5
* **d**  Displays disk statistics, including I/O, reads and writes. This might be useful to monitor swap space. Example: vmstat -d

Try running the below code in combination with some of the memory commands above to see if you can find any information about what is happening with memory.

1. Execute the code below (your compiler may catch it):
2. Record what you observe.
3. Be prepared to discuss what you observed in this week's discussion forum.
4. int main( int argc, char \*argv[])

{

     char myArr[40];

     strcpy(buffer,argv[1]);

}

1. int main(int argc, char \*argv[])

{

      char \* myChar =”Testing 123”;

      char myArray[10];

      for ( int i =0;  i< 2000; i++)

        {

        myArray[i]= “$”;

        }

     char test = myChar[0];

}