CS355 Lab Notes #0 Observing the Linux Kernel

Kernel is the name for the OS core. For the purpose of this class, you can pretty much consider it another name for OS. The Linux kernel can be thought of as a collection of data structure instances (kernel variables) and functions – after all, an OS is just another C program, although a pretty large one that does a lot of things. The collective kernel variables define the kernel's perspective of the state of the entire computer system. Each externally invoked function – a system call or an IRQ (interrupt request) – provides a prescribed service and causes the system state to be changed by having the kernel code change its kernel variables. If you could inspect the kernel variables, then you could infer the state of the entire system.

Many variables make up the entire kernel state, including variables to represent each process, each open file, the memory and the CPU. Kernel variables are no different than ordinary C program variables, other than they are kept in kernel space. Because the kernel is implemented as a monolithic module, many of the kernel variables are declared as global static variables. Some of the kernel variables are instances of built-in C data types, but most are instances of an extensible data type, a C **`struct`**.

In general, **`struct`**s are defined in C header files (also called include files or .h files), in the Linux source code tree. The Linux source code is located in **`/usr/src/linux-source-*`**, where the **`*`** stands for version codes (use "**`uname -r`**" to find out which version we are running). The **`include/linux`** subdirectory contains many .h files that offer insights on the data structure declarations. Please look over some of these files on your own for a taste of things to come.

Linux provides a very useful mechanism for inspecting the kernel state, called the **`/proc`** file system. It is an OS mechanism whose interface appears as a directory in the conventional Unix file system (in the root directory). However, a file in **`/proc`** or one of its subdirectories is actually a program that reads kernel variables and reports them as ASCII strings. Some of these routines read the kernel tables only when the pseudo file is opened, whereas others read the tables each time that the file is read. Consult the **`proc`** man pages for more details (under section 5). If you don't know how to find the manual pages for **`proc`**, start with reading the manual pages for **`man`** ("**`man man`**").

Lab exercise (make sure you complete these before Monday 1/29):

Answer the following questions about the Linux machine you are on, using the **`/proc`** pseudo file system (record which files you used):

1. What is the CPU type and model?
2. What version of Linux kernel is being used?
3. How long (in days, hours and minutes) has it been since the system was last booted?
4. How much of the total CPU time has been spent executing in user mode? System mode? Idle?
5. How much memory is configured into it?

6. How much memory is currently available on it?
7. How many disk read/writes requests have been made?
8. How many context switching has the kernel performed?
9. How many processes have been created since the system was booted?