CS 377 – Operating Systems

Discussion Session 2 Questions

Name: ____________________

These questions will review several key concepts covered in lecture and the textbook. Please write your answers individually without consulting the internet or your textbook. Be succinct (complete sentences not necessary).

1. **OS hardware architecture**. Memory/storage is arranged in a hierarchy in an OS. Describe why we use multiple layers as opposed to only a single layer, and give three examples of specific layers in the memory hierarchy, arranged in order.

   Solution. *Multiple layers are necessary because extremely fast memory is both small and expensive, while large memory/storage is too slow to be used exclusively. Using both allows data accessed repeatedly to be stored in fast memory, while data that is accessed less frequently may be stored in cheap and plentiful memory. Examples of specific layers include registers (fastest and most space-constrainted), L1 and L2 cache, regular RAM, disk storage, and network storage (slowest and least space-constrained).*

2. **OS design**. One popular OS design is layered, in which pieces of functionality are implemented on top of each other. Describe one advantage and one disadvantage of the layered design. Then, assuming we are using such a design, rank the following system components from lowest-level to highest-level: *(a) file system*, *(b) user interface*, *(c) system calls*, *(d) hard disk*, *(e) software libraries*, *(f) disk driver*.

   Solution. *The layered design is attractive for two primary reasons. One reason is that since each layer depends only on the layer(s) directly beneath it, the process of development and debugging is greatly simplified, since each layer does not need to depend on the other layers in the system. Another benefit is that of abstraction; since each layer is built only on the directly underlying layer, the internals of all underlying layers may be changed so long as the interface to the immediately upstream layer remains unchanged. The primary downside is that of overhead and efficiency, since layers may have to copy information several times down to the hardware and back.*

   The components would be ranked (d) hard disk, (f) disk driver, (a) file system, (c) system calls, (e) software libraries, and (b) user interface.

3. **System calls**.

   (a) Explain the purpose of system calls.

   Solution. *System calls allow user programs to use services provided by the OS. These services include operations that cannot be safely executed by user programs, such as those that could harm the state of other users or processes (e.g., directly accessing arbitrary memory).*

   (b) Give an example of a specific system call and mention what it does.

   Solution. *Examples include* `fork` *to create a copy of a process,* `open` *to open a file reading, and* `exit` *to terminate a process (along with many others described in the notes and textbook).*

   (c) Describe how user programs are prevented from executing sensitive system calls directly.

   Solution. *User programs execute in* **user mode***, in which programs are prevented from executing sensitive instructions such as* `halt`*. In contrast, the OS can execute in* **kernel mode***, which allows access to these instructions. The hardware provides for switching between modes (in a protected register) at the discretion of the OS.*

   (d) Briefly describe what is meant by a **trap** and mention how traps relate to system calls.

   Solution. *A trap is a special condition detected by the architecture to handle OS-level events, including system calls (somewhat like an exception in Java). When a trap is detected, control is passed to a specific handler (determined by the type of trap encountered), and the OS performs whatever processing is needed. For system calls, this includes verifying parameters passed by the caller, switching temporarily to kernel mode, and passing a return value back.*