Tutorial 1

The overview of OS

Hongfan Mu

hmu026@uottawa.ca

1. What are the three main roles of the operating system?

The operating system controls the hardware and coordinates its use among the various application programs for the various users.

- Hardware abstraction: To provide an environment in which a computer user can run programs in various hardware easily and efficiently.
- Resource allocation: To allocate the various computer resources as required by the programs. The allocation should be fair and effective as much as possible.
- Control program: A control program has two main functions: (1) overseeing the execution of user programs to prevent errors and misuse of the computer; and (2) operation management and control of I/O devices.

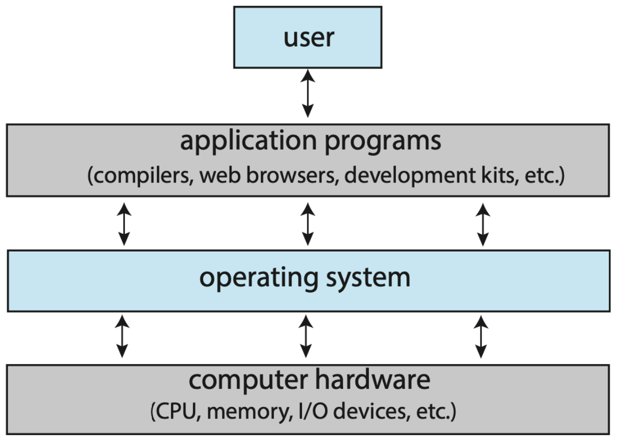

user
application programs
(compilers, web browsers, development kits, etc.)
operating system
computer hardware
(CPU, memory, I/O devices, etc.)

Figure 1.1 Abstract view of the components of a computer system.

.

1.1.2 System View

2. Consider the various definitions of the *operating system.* Consider including apps such as WEB browsers or email programs in the operating system. Debate the pros and cons of this issue with justification.

Programs installed in an OS does not mean they are part of the operating system

- Pros: Applications such as WEB browsers and email tools are playing an increasingly important role in the use of the modern PC. To meet these needs, these applications could be incorporated into the operating system. This could lead to better performance and integration with the rest of the system. In addition, these important applications could have the same look and feel of the system software of the OS.

- Cons: The fundamental role of the operating system is to manage system resources such as CPU, memory, I/O devices, etc. In addition, its role is to run software applications such as WEB browsers and email applications. By incorporating such applications into the OS, it is weighed down with these additional functions. This additional burden may result in an OS performing its system resource management task less well. In addition, the size of the OS is increased, increasing the possibility of the OS crashing and breaches of safety.

.

3. How does the distinction between kernel and user mode serve the rudimentary protection (security) of the system?

Diff: A task that is executed on behalf of the operating system and one that is executed on behalf of the user.

- The distinction between the core and the user mode offers a rudimentary form of protection in the following way:
- Some instructions can only be executed when the CPU is in kernel mode. Also, devices can only be accessed by a program when it runs in kernel mode (i.e. within a system call). The activation and deactivation control of interruptions is only done when the CPU is in kernel mode.
- Therefore, the CPU is quite limited when it runs in user mode, and thus protects critical resources while running user code.

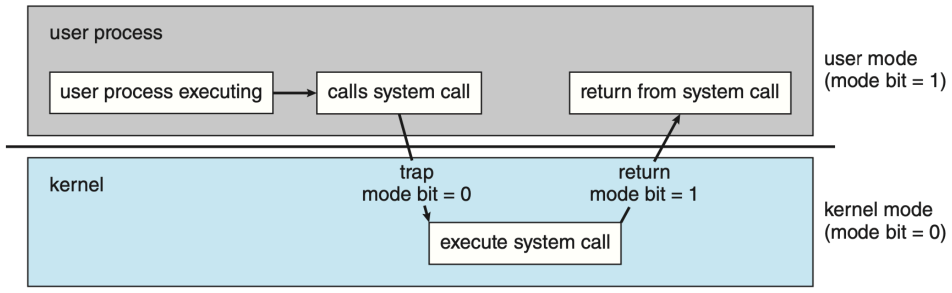


Figure 1.13 Transition from user to kernel mode.

1.4.2 Dual-Mode and Multimode Operation

4. Which instructions should require kernel privileges?

The protection by designating some of the machine instructions that may cause harm as privileged instructions.
The instruction to switch to kernel mode is an example of a privileged instruction. Some other examples include I/O control, timer management, and interrupt

a. Timer update.

b. Read the clock.

c. Clear the memory.

d. Run a software interrupt (this is a CPU instruction).

e. Turn off interruptions.

f. Change entries in a device state table.

g. Switch between user mode and kernel mode.

h. Access an I/O device.

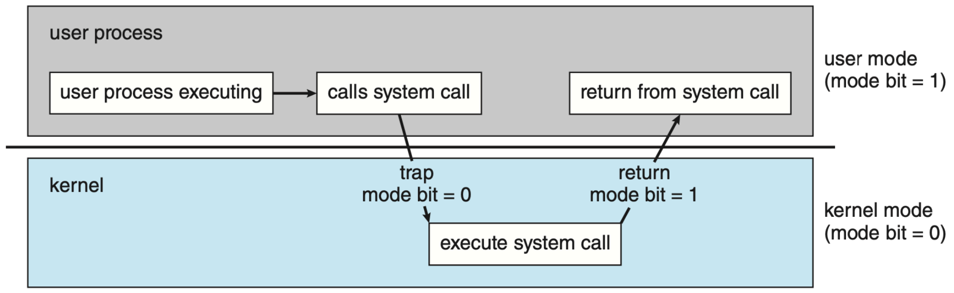


Figure 1.13 Transition from user to kernel mode.

1.4.2 Dual-Mode and Multimode Operation

5. A timer can be used to determine the current time. Give a short description of how to accomplish this task.

We use timer to ensure that the operating system maintains control over the CPU. We cannot allow a user program to get stuck in an infinite loop or to fail to call system services and never return control to the operating system.

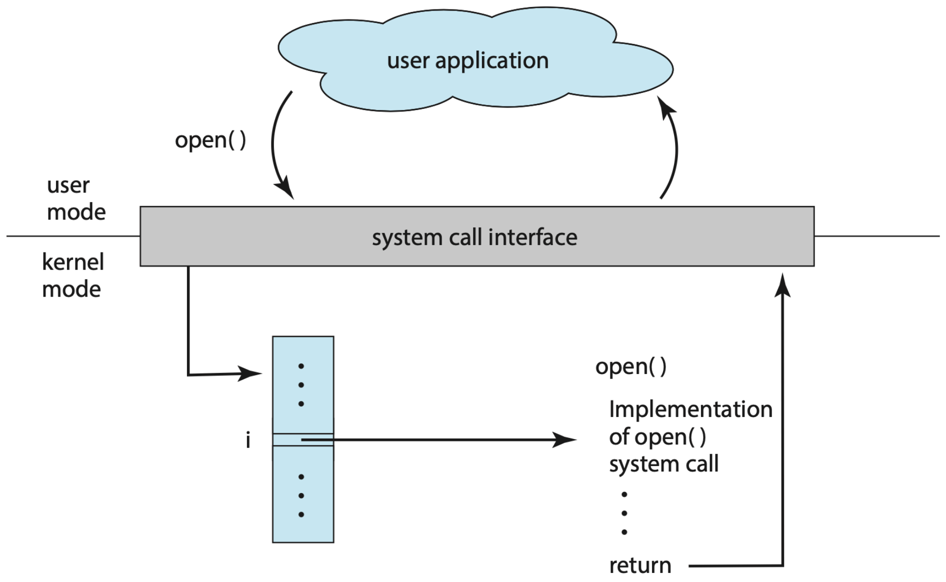
- A program can use the following approach to determine current time with timer interruptions. The program sets up the timer to generate an interruption after a specific time and then be suspended (fall asleep). When activated by the interrupt, it can update a variable whose value reflects the number of interruptions received to date. These steps are repeated continuously, i.e., set up the interruption and update the variable when the interruption is produced. The variable can thus track time count, especially if it is initialized to a significant value. UNIX OSs uses such an approach by counting the number of seconds since Jan 1, 1970, with a 32-bit counter (overflow will take place on Jan 19, 2038).

.

6. What is the purpose of system calls?

System calls provide an interface to the services made available by an operating system.

- System calls allow user processes to make requests for services to the operating system. Note that system calls are made with software interruptions. Be aware that the OS works on interruptions and that it may receive an interruption of hardware AND software (i.e. user programs) as a request to complete any action.



.

7. What are the five main activities of the operating system when managing processes?

- The creation and termination of user and system processes.
- Suspension and resumption of processes.
- The provision of mechanisms for **process synchronization**.
- The provision of mechanisms for **communication** between processes.
- The provision of mechanisms to deal with **deadlocks**.

8. What are the three main activities of the operating system when managing memory?

The CPU can load instructions only from memory, so any programs must first be loaded into memory to run.

- Consider which parts of the memory are used and by what processes.
- Decide which processes are loaded into memory when memory space becomes available.
- The allocation and release of memory space as needed.

9. What are the three main activities of the operating system when managing secondary storage memory?

The main requirement for secondary storage is that it be able to hold large quantities of data permanently.

Main memory is violatile

- The management of free space.
- Secondary memory allocation.
- The disk scheduling.

10. What is the purpose of the Command Interpreter/command-line interface (CLI)?

It reads user commands or a command file is either running them directly or by launching another process to run a separate program.

.

2.2.1 Command Interpreters

11. What system calls are run by the command interpreter or shell to start a new process?

To start a new process, the shell executes a fork()() system call. Then, the selected program is loaded into memory via an exec() system call, and the program is executed.

In the UNIX system, the system calls *fork()* followed by *exec()* must be made to initiate a new process.

The call *fork() clones* the process that makes the call, while the *exec* call replaces the program in the process with a new program. What needs to be taken into account is that *fork()* is run by the original process (parent), while the *exec()* is run by the launched process (child).

2.3.3.1 Process Control
Processes are discussed in Chapter 3 with a program example using the fork() and exec() system calls.

12. What is the purpose of system programs?

System programs can be seen as system call software. It offers basic functions to users to avoid forcing them to write programs to solve common problems. Common systems programs include utilities to manipulate files and directories, shells to run programs (including utility systems), utilities to monitor the user base and network (e.g. UNIX who shows which users have an open session in the local system and remote systems), process management, etc.

.

13. What is the main advantage of designing a system with layers? What are the disadvantages of this approach?

- The main advantage of the layered approach is simplicity of construction and debugging.
- The system is easier to debug and modify since the changes only affect limited parts instead of affecting all sections of the OS. Data is maintained only where necessary and accessible from a well-defined and limited region, which means that bugs affecting data are limited to a specific module or layer.

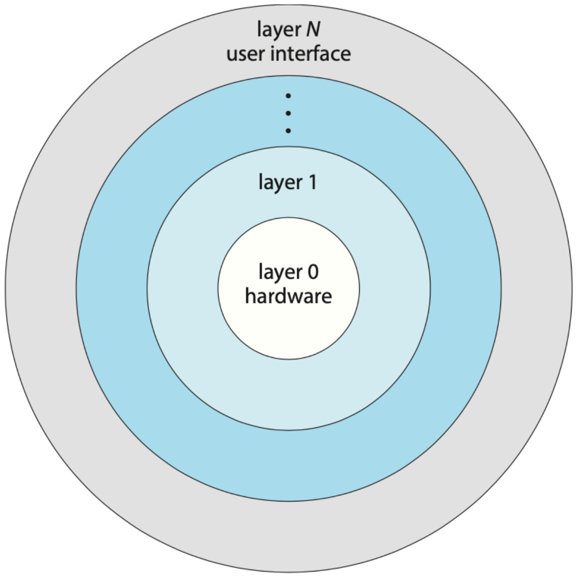


Figure 2.14 A layered operating system.

2.8.2 Layered Approach

14. For each of the following five services offered by an operating system, explain how each of them is convenient for the user. Explain when it would be impossible to offer these services with user programs.

- Program execution. The operating system loads the content (or parts of the content) of an executable file into primary memory and initiates its execution. It allocates resources to the program during its implementation (e.g. CPU). It is not possible to trust a user program to make a good allocation of resources such as CPU.
- I/O operations. Communication with discs, magnetic tapes, serial lines of communication, and other devices must be done at a very low level. The user program only specifies the device (in principle as a file) and the operations to be done with the device, and the OS converts these requests into specific instructions to the device controller. It is not possible to trust a user program to access the devices to which they are entitled, and when they are free.
- Manipulating the file system. There are many details for file creation, file deletion, file space allocation, and file name definition. The disk is organized into blocks of data that need to be managed. Protection must ensure proper access to the files. It is impossible to trust user programs to adhere to the protection rules; allocate only free blocks to the creation and updating of files; and release the blocks when deleting the file.
- Communications. The exchange of messages between systems (and processes) requires that messages be transformed into data packets that are sent to a communication controller, transmitted via a transmission medium, and reassorted into the remote system. Packets need to be ordered and corrected as well. Again, a user program might not make a good access to the communication device controller or could take packages for other processes.
- Error detection. Error detection is done at the hardware and logical level. At the hardware level, all data transfers must be inspected to ensure that the data was not corrupted during transit. All supporting data must be verified to ensure that they have not changed since registering on the support. At the logical level, the support must be checked for data deterioration; for example, ensuring that the number of blocks allocated and free blocks of secondary memory correspond to the total number of blocks of the device. Errors in these cases are often independent of specific processes (for example, data corruption on a disk), and therefore a global program (the OS) has to deal with this type of error. Also, by having the OS deal with this type of problem, it is not necessary to include in programs users of code to deal with all possible errors in the system.