CHAPTER 1

Chapter 1 introduces the general topic of operating systems and a handful of important concepts (multiprogramming, time sharing, distributed system, and so on). The purpose is to show *why* operating systems are what they are by showing *how* they developed. In operating systems, as in much of computer science, we are led to the present by the paths we took in the past, and we can better understand both the present and the future by understanding the past.

Additional work that might be considered is learning about the particular systems that the students will have access to at your institution. This is still just a general overview, as specific interfaces are considered in Chapter 3.

Exercises

1.1 In a multiprogramming and time-sharing environment, several users share the system simultaneously. This situation can result in various security problems.

a. What are two such problems?
b. Can we ensure the same degree of security in a time-shared machine as in a dedicated machine? Explain your answer.

Answer:

a. Stealing or copying one's programs or data; using system resources (CPU, memory, disk space, peripherals) without proper accounting.
b. Probably not, since any protection scheme devised by humans can inevitably be broken by a human, and the more complex the scheme, the more difficult it is to feel confident of its correct implementation.

1.2 The issue of resource utilization shows up in different forms in different types of operating systems. List what resources must be managed carefully in the following settings:

a. Mainframe or minicomputer systems
b. Workstations connected to servers
c. Handheld computers

Answer:

a. Mainframes: memory and CPU resources, storage, network bandwidth
b. Workstations: memory and CPU resources
c. Handheld computers: power consumption, memory resources

1.3 Under what circumstances would a user be better off using a timesharing system rather than a PC or a single-user workstation?

Answer:

When there are few other users, the task is large, and the hardware is fast, time-sharing makes sense. The full power of the system can be brought to bear on the user's problem. The problem can be solved faster than on a personal computer. Another case occurs when lots of other users need resources at the same time.

A personal computer is best when the job is small enough to be executed reasonably on it and when performance is sufficient to execute the program to the user's satisfaction.

1.4 Describe the differences between symmetric and asymmetric multiprocessing. What are three advantages and one disadvantage of multiprocessor systems?

Answer:

Symmetric multiprocessing treats all processors as equals, and I/O can be processed on any CPU. Asymmetric multiprocessing has one master CPU and the remainder CPUs are slaves. The master distributes tasks among the slaves, and I/O is usually done by the master only. Multiprocessors can save money by not duplicating power supplies, housings, and peripherals. They can execute programs more quickly and can have increased reliability. They are also more complex in both hardware and software than uniprocessor systems.

1.5 How do clustered systems differ from multiprocessor systems? What is required for two machines belonging to a cluster to cooperate to provide a highly available service?
Answer:
Clustered systems are typically constructed by combining multiple computers into a single system to perform a computational task distributed across the cluster. Multiprocessor systems on the other hand could be a single physical entity comprising of multiple CPUs. A clustered system is less tightly coupled than a multiprocessor system. Clustered systems communicate using messages, while processors in a multiprocessor system could communicate using shared memory.

In order for two machines to provide a highly available service, the state on the two machines should be replicated and should be consistently updated. When one of the machines fails, the other could then takeover the functionality of the failed machine.

1.6 Consider a computing cluster consisting of two nodes running a database. Describe two ways in which the cluster software can manage access to the data on the disk. Discuss the benefits and disadvantages of each.
Answer:
Consider the following two alternatives: asymmetric clustering and parallel clustering. With asymmetric clustering, one host runs the database application with the other host simply monitoring it. If the server fails, the monitoring host becomes the active server. This is appropriate for providing redundancy. However, it does not utilize the potential processing power of both hosts. With parallel clustering, the database application can run in parallel on both hosts. The difficulty in implementing parallel clusters is providing some form of distributed locking mechanism for files on the shared disk.

1.7 How are network computers different from traditional personal computers? Describe some usage scenarios in which it is advantageous to use network computers.
Answer:
A network computer relies on a centralized computer for most of its services. It can therefore have a minimal operating system to manage its resources. A personal computer on the other hand has to be capable of providing all of the required functionality in a stand-alone manner without relying on a centralized manner. Scenarios where administrative costs are high and where sharing leads to more efficient use of resources are precisely those settings where network computers are preferred.

1.8 What is the purpose of interrupts? What are the differences between a trap and an interrupt? Can traps be generated intentionally by a user program? If so, for what purpose?
Answer:
An interrupt is a hardware-generated change of flow within the system. An interrupt handler is summoned to deal with the cause of the interrupt; control is then returned to the interrupted context and instruction. A trap is a software-generated interrupt. An interrupt can be used to signal the completion of an I/O to obviate the need for device polling. A trap can be used to call operating system routines or to catch arithmetic errors.

1.9 Direct memory access is used for high-speed I/O devices in order to avoid increasing the CPU's execution load.

a. How does the CPU interface with the device to coordinate the transfer?
b. How does the CPU know when the memory operations are complete?

c. The CPU is allowed to execute other programs while the DMA controller is transferring data. Does this process interfere with the execution of the user programs? If so, describe what forms of interference are caused.

Answer:

The CPU can initiate a DMA operation by writing values into special registers that can be independently accessed by the device. The device initiates the corresponding operation once it receives a command from the CPU. When the device is finished with its operation, it interrupts the CPU to indicate the completion of the operation.

Both the device and the CPU can be accessing memory simultaneously. The memory controller provides access to the memory bus in a fair manner to these two entities. A CPU might therefore be unable to issue memory operations at peak speeds since it has to compete with the device in order to obtain access to the memory bus.

1.10 Some computer systems do not provide a privileged mode of operation in hardware. Is it possible to construct a secure operating system for these computer systems? Give arguments both that it is and that it is not possible.

Answer:

An operating system for a machine of this type would need to remain in control (or monitor mode) at all times. This could be accomplished by two methods:

a. Software interpretation of all user programs (like some BASIC, Java, and LISP systems, for example). The software interpreter would provide, in software, what the hardware does not provide.
b. Require that all programs be written in high-level languages so that all object code is compiler-produced. The compiler would generate (either in-line or by function calls) the protection checks that the hardware is missing.

1.11 Many SMP systems have different levels of caches; one level is local to each processing core, and another level is shared among all processing cores. Why are caching systems designed this way?

Answer:

The different levels are based on access speed as well as size. In general, the closer the cache is to the CPU, the faster the access. However, faster caches are typically more costly. Therefore, smaller and faster caches are placed local to each CPU, and shared caches that are larger, yet slower, are shared among several different processors.

1.12 Consider an SMP system similar to the one shown in Figure 1.6. Illustrate with an example how data residing in memory could in fact have a different value in each of the local caches.

Answer:

Say processor 1 reads data A with value 5 from main memory into its local cache. Similarly, processor 2 reads data A into its local cache as well. Processor 1 then updates A to 10. However, since A resides in processor 1's local cache, the update only occurs there and not in the local cache for processor 2.

1.13 Discuss, with examples, how the problem of maintaining coherence of cached data manifests itself in the following processing environments:

a. Single-processor systems
b. Multiprocessor systems
c. Distributed systems

Answer:

In single-processor systems, the memory needs to be updated when a processor issues updates to cached values. These updates can be performed immediately or in a lazy manner. In a multiprocessor system, different processors might be caching the same memory location in its local caches. When updates are made,

the other cached locations need to be invalidated or updated. In distributed systems, consistency of cached memory values is not an issue. However, consistency problems might arise when a client caches file data.

1.14 Describe a mechanism for enforcing memory protection in order to prevent a program from modifying the memory associated with other programs.

Answer:

The processor could keep track of what locations are associated with each process and limit access to locations that are outside of a program's extent. Information regarding the extent of a program's memory could be maintained by using base and limits registers and by performing a check for every memory access.

1.15 Which network configuration—LAN or WAN—would best suit the following environments?

a. A campus student union
b. Several campus locations across a statewide university system
c. A neighborhood

Answer:

a. LAN
b. WAN
c. LAN or WAN

1.16 Describe some of the challenges of designing operating systems for mobile devices compared with designing operating systems for traditional PCs.

Answer:

The greatest challenges in designing mobile operating systems include:

- Less storage capacity means the operating system must manage memory carefully.
- The operating system must also manage power consumption carefully.
- Less processing power plus fewer processors mean the operating system must carefully apportion processors to applications.

1.17 What are some advantages of peer-to-peer systems over client-server systems?

Answer:

Peer-to-peer is useful because services are distributed across a collection of peers, rather than having a single, centralized server. Peer-to-peer provides fault tolerance and redundancy. Also, because peers constantly migrate, they can provide a level of security over a server that always exists at a known location on the Internet. Peer-to-peer systems can also potentially provide higher network bandwidth because you can collectively use all the bandwidth of peers, rather than the single bandwidth that is available to a single server.

1.18 Describe some distributed applications that would be appropriate for a peer-to-peer system.

Answer:

Essentially anything that provides content, in addition to existing services such as file services, distributed directory services such as domain name services, and distributed e-mail services.

1.19 Identify several advantages and several disadvantages of open-source operating systems. Include the types of people who would find each aspect to be an advantage or a disadvantage.

Answer:

Open source operating systems have the advantages of having many people working on them, many people debugging them, ease of access and distribution, and rapid update cycles. Further, for students and programmers, there is certainly an advantage to being able to view and modify the source code. Typically open source operating systems are free for some forms of use, usually just requiring payment for support services. Commercial operating system companies usually do not like the competition that open source

operating systems bring because these features are difficult to compete against. Some open source operating systems do not offer paid support programs. Some companies avoid open source projects because they need paid support, so that they have some entity to hold accountable if there is a problem or they need help fixing an issue. Finally, some complain that a lack of discipline in the coding of open source operating systems means that backward compatibility is lacking making upgrades difficult, and that the frequent release cycle exacerbates these issues by forcing users to upgrade frequently.