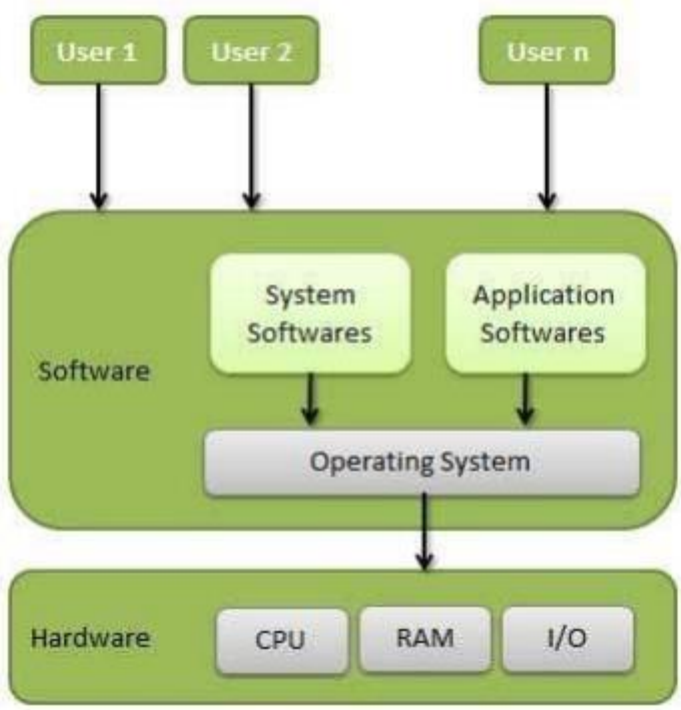
UNIT 1

INTRODUCTION TO OPERATING SYSTEM CONCEPT

Types of operating systems, operating systems concepts, operating systems services, Introduction to System call, System call types.

Operating Systems Concepts

An Operating System (OS) is an interface between computer user and computer hardware. An operating system is software which performs all the basic tasks like file management, memory management, process management, handling input and output, and controlling peripheral devices such as disk drives and printers.



Following are some of important functions of an operating System.
- Memory Management
- Processor Management
- Device Management
- File Management
- Security
- Control over system performance
- Job accounting
- Error detecting aids
- Coordination between other software and users

Memory Management:
Memory management refers to management of Primary Memory or Main Memory. Main memory is a large array of words or bytes where each word or byte has its own address. Main memory provides a fast storage that can be accessed directly by the CPU. For a program to be executed, it must in the main memory. An Operating System does the following activities for memory management –
- Keeps tracks of primary memory, i.e., what part of it are in use by whom, what part are not in use.
- In multiprogramming, the OS decides which process will get memory when and how much.

- Allocates the memory when a process requests it to do so.
- De-allocates the memory when a process no longer needs it or has been terminated.

Processor Management

In multiprogramming environment, the OS decides which process gets the processor when and for how much time. This function is called process scheduling. An Operating System does the following activities for processor management –

- Keeps tracks of processor and status of process. The program responsible for this task is known as traffic controller.
- Allocates the processor (CPU) to a process.
- De-allocates processor when a process is no longer required.

Device Management

An Operating System manages device communication via their respective drivers. It does the following activities for device management –

- Keeps tracks of all devices. Program responsible for this task is known as the I/O controller.
- Decides which process gets the device when and for how much time.
- Allocates the device in the efficient way.
- De-allocates devices.

Device Management

An Operating System manages device communication via their respective drivers. It does the following activities for device management −

- Keeps tracks of all devices. Program responsible for this task is known as the I/O controller.
- Decides which process gets the device when and for how much time.
- Allocates the device in the efficient way.
- De-allocates devices.

File Management

A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directions.

An Operating System does the following activities for file management −

- Keeps track of information, location, uses, status etc. The collective facilities are often known as file system.
- Decides who gets the resources.
- Allocates the resources.
- De-allocates the resources

Other Important Activities

Following are some of the important activities that an Operating System performs −

- **Security** − By means of password and similar other techniques, it prevents unauthorized access to **programs and data.**
- **Control over system performance** − Recording delays between request for a service and response from the system.
- **Job accounting** − Keeping track of time and resources used by various jobs and users.
- **Error detecting aids** − Production of dumps, traces, error messages, and other debugging and error detecting aids.
- **Coordination between other software’s and users** − Coordination and assignment of compilers, interpreters, assemblers and other software to the various users of the computer systems.

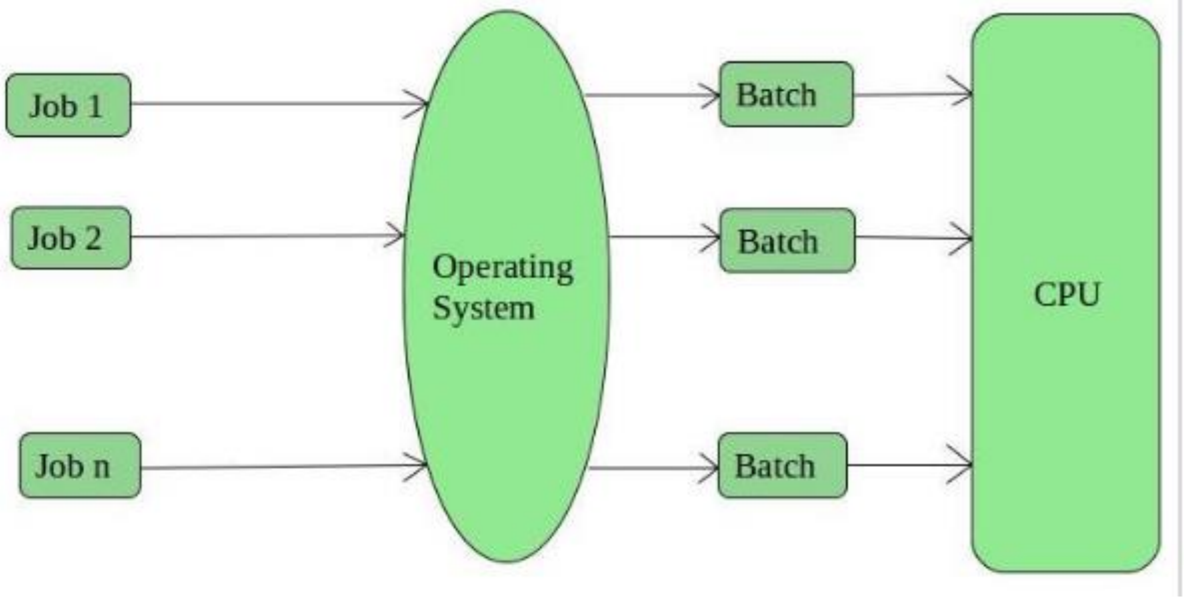
Types of Operating Systems

Operating systems are there from the very first computer generation and they keep evolving with time. In this chapter, we will discuss some of the important types of operating systems which are most commonly used. An Operating System performs all the basic tasks like managing files, processes, and memory. Thus operating system acts as the manager of all the resources, i.e. resource manager. Thus, the operating system becomes an interface between user and machine.

Some widely used Operating Systems are:

1. **Batch Operating System**

   This type of operating system does not interact with the computer directly. Each user prepares his job on an off-line device like punch cards and submits it to the computer operator. To speed up processing, jobs with similar needs are batched together and run as a group. The programmers leave their programs with the operator and the operator then sorts the programs with similar requirements into batches



   Advantages of Batch Operating System:

   • It is very difficult to guess or know the time required for any job to complete. Processors of the batch systems know how long the job would be when it is in queue

   • Multiple users can share the batch systems

   • The idle time for the batch system is very less

   • It is easy to manage large work repeatedly in batch systems

   Disadvantages of Batch Operating System:

   • The computer operators should be well known with batch systems

   • Batch systems are hard to debug

   • It is sometimes costly

   • The other jobs will have to wait for an unknown time if any job fails

   Examples of Batch based Operating System: Payroll System, Bank Statements, etc.

2. **Time-Sharing Operating Systems**

   Each task is given some time to execute so that all the tasks work smoothly. Each user gets the time of CPU as they use a single system. These systems are also known as Multitasking Systems. The task can be from a single user or different users also. The time that each task gets to execute is called quantum. After this time interval is over OS switches over to the next task.

   Advantages of Time-Sharing OS:

   • Each task gets an equal opportunity

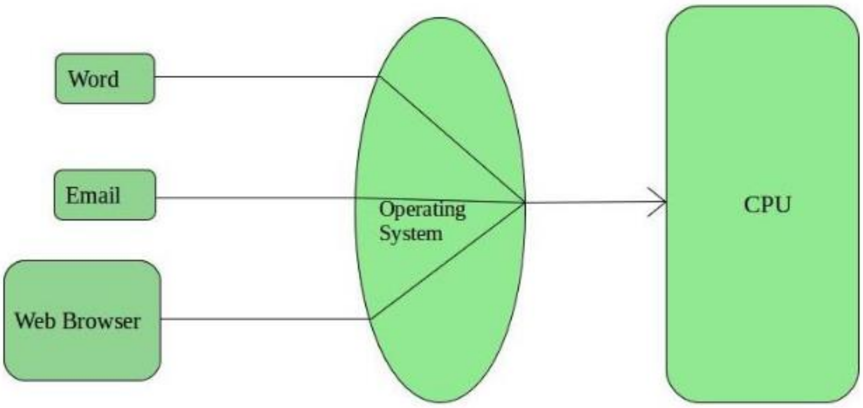
   • Fewer chances of duplication of software

   • CPU idle time can be reduced

Disadvantages of Time-Sharing OS:

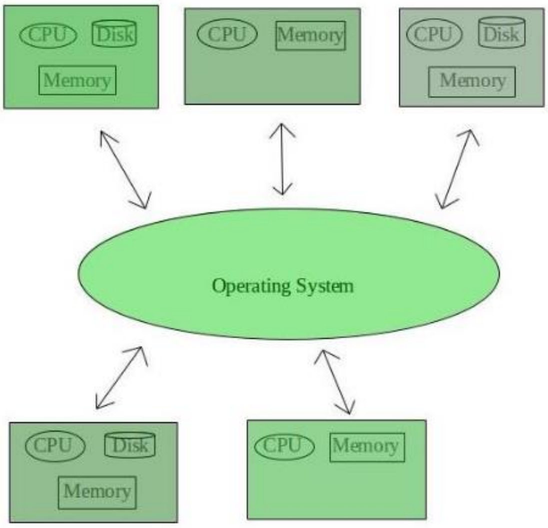
- Reliability problem
- One must have to take care of the security and integrity of user programs and data
- Data communication problem

Examples of Time-Sharing OSs are: Multics, Unix, etc.



3. **Distributed Operating System**

These types of the operating system is a recent advancement in the world of computer technology and are being widely accepted all over the world and, that too, with a great pace. Various autonomous interconnected computers communicate with each other using a shared communication network. Independent systems possess their own memory unit and CPU. These are referred to as loosely coupled systems or distributed systems. These system's processors differ in size and function. The major benefit of working with these types of the operating system is that it is always possible that one user can access the files or software which are not actually present on his system but some other system connected within this network i.e., remote access is enabled within the devices connected in that network.



Advantages of Distributed Operating System:

- Failure of one will not affect the other network communication, as all systems are independent from each other
- Electronic mail increases the data exchange speed
- Since resources are being shared, computation is highly fast and durable • Load on host computer reduces
- These systems are easily scalable as many systems can be easily added to the network
- Delay in data processing reduces

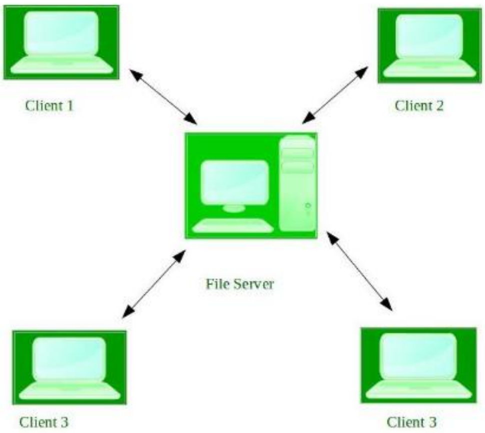
Disadvantages of Distributed Operating System:

- Failure of the main network will stop the entire communication
- To establish distributed systems the language which is used are not well defined yet
- These types of systems are not readily available as they are very expensive. Not only that the underlying software is highly complex and not understood well yet

Examples of Distributed Operating System are- LOCUS, etc.

4. **Network Operating System –**

These systems run on a server and provide the capability to manage data, users, groups, security, applications, and other networking functions. These types of operating systems allow shared access of files, printers, security, applications, and other networking functions over a small private network. One more important aspect of Network Operating Systems is that all the users are well aware of the underlying configuration, of all other users within the network, their individual connections, etc. and that's why these computers are popularly known as tightly coupled systems.



Advantages of Network Operating System:

- Highly stable centralized servers
- Security concerns are handled through servers
- New technologies and hardware up-gradation are easily integrated into the system
- Server access is possible remotely from different locations and types of systems

Disadvantages of Network Operating System:

- Servers are costly
- User has to depend on a central location for most operations
- Maintenance and updates are required regularly

Examples of Network Operating System are: Microsoft Windows Server 2003, Microsoft Windows Server 2008, UNIX, Linux, Mac OS X, Novell NetWare, and BSD, etc

5. **Real-Time Operating System –**

These types of OSs serve real-time systems. The time interval required to process and respond to inputs is very small. This time interval is called response time. Real-time systems are used when there are time requirements that are very strict like missile systems, air traffic control systems, robots, etc.

Advantages of RTOS:

- **Maximum Consumption:** Maximum utilization of devices and system, thus more output from all the resources
- **Task Shifting:** The time assigned for shifting tasks in these systems are very less. For example, in older systems, it takes about 10 microseconds in shifting one task to another, and in the latest systems, it takes 3 microseconds.

• **Focus on Application:** Focus on running applications and less importance to applications which are in the queue.

• **Real-time operating system in the embedded system:** Since the size of programs are small, RTOS can also be used in embedded systems like in transport and others.

• **Error Free:** These types of systems are error-free

Memory Allocation: Memory allocation is best managed in these types of systems.

Disadvantages of RTOS:

• **Limited Tasks:** Very few tasks run at the same time and their concentration is very less on few applications to avoid errors.

• **Use heavy system resources:** Sometimes the system resources are not so good and they are expensive as well.

• **Complex Algorithms:** The algorithms are very complex and difficult for the designer to write on.

• **Device driver and interrupt signals:** It needs specific device drivers and interrupts signals to respond earliest to interrupts.

• **Thread Priority:** It is not good to set thread priority as these systems are very less prone to switching tasks.

Examples of Real-Time Operating Systems are: Scientific experiments, medical imaging systems, industrial control systems, weapon systems, robots, air traffic control systems, etc.

Operating System ─ Services

An Operating System provides services to both the users and to the programs.

It provides programs an environment to execute.

It provides users the services to execute the programs in a convenient manner.

Following are a few common services provided by an operating system:

1. Program execution
2. I/O operations
3. File System manipulation
4. Communication
5. Error Handling
6. Resource Management
7. Protection

1. Program execution

Operating systems handle many kinds of activities from user programs to system programs like printer spooler, name servers, file server, etc. Each of these activities is encapsulated as a process.

A process includes the complete execution context (code to execute, data to manipulate, registers, OS resources in use). Following are the major activities of an operating system with respect to program management:

- ❖ Loads a program into memory
- ❖ Executes the program
- ❖ Handles program's execution
- ❖ Provides a mechanism for process synchronization
- ❖ Provides a mechanism for process communication
- ❖ Provides a mechanism for deadlock handling

2. I/O Operation

An I/O subsystem comprises of I/O devices and their corresponding driver software. Drivers hide the peculiarities of specific hardware devices from the users. An Operating System manages the communication between user and device drivers. I/O operation means read or write operation with any file or any specific I/O device. Operating system provides the access to the required I/O device when required.

3. **File System manipulation**

   A file represents a collection of related information. Computers can store files on the disk (secondary storage), for long-term storage purpose. Examples of storage media include magnetic tape, magnetic disk and optical disk drives like CD, DVD. Each of these media has its own properties like speed, capacity, data transfer rate and data access methods.

   A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directions. Following are the major activities of an operating system with respect to file management:

   - Program needs to read a file or write a file. The operating system gives the permission to the program for operation on file. Permission varies from read-only, read-write, denied, and so on.
   - Operating System provides an interface to the user to create/delete files.
   - Operating System provides an interface to the user to create/delete directories.
   - Operating System provides an interface to create the backup of file system.

4. **Communication**

   In case of distributed systems which are a collection of processors that do not share memory, peripheral devices, or a clock, the operating system manages communications between all the processes. Multiple processes communicate with one another through communication lines in the network.

   The OS handles routing and connection strategies, and the problems of contention and security. Following are the major activities of an operating system with respect to communication:

   Two processes often require data to be transferred between them.

   - Both the processes can be on one computer or on different computers, but are connected through a computer network.
   - Communication may be implemented by two methods, either by Shared Memory or by Message Passing

5. **Error Handling**

   Errors can occur anytime and anywhere. An error may occur in CPU, in I/O devices or in the memory hardware. Following are the major activities of an operating system with respect to error handling −

   - The OS constantly checks for possible errors.
   - The OS takes an appropriate action to ensure correct and consistent computing

6. **Resource Management**

   In case of multi-user or multi-tasking environment, resources such as main memory, CPU cycles and files storage are to be allocated to each user or job. Following are the major activities of an operating system with respect to resource management:

   - The OS manages all kinds of resources using schedulers.
   - CPU scheduling algorithms are used for better utilization of CPU.

7. **Protection**

   Considering a computer system having multiple users and concurrent execution of multiple processes, the various processes must be protected from each other's activities.

   Protection refers to a mechanism or a way to control the access of programs, processes, or users to the resources defined by a computer system.

   Following are the major activities of an operating system with respect to protection −

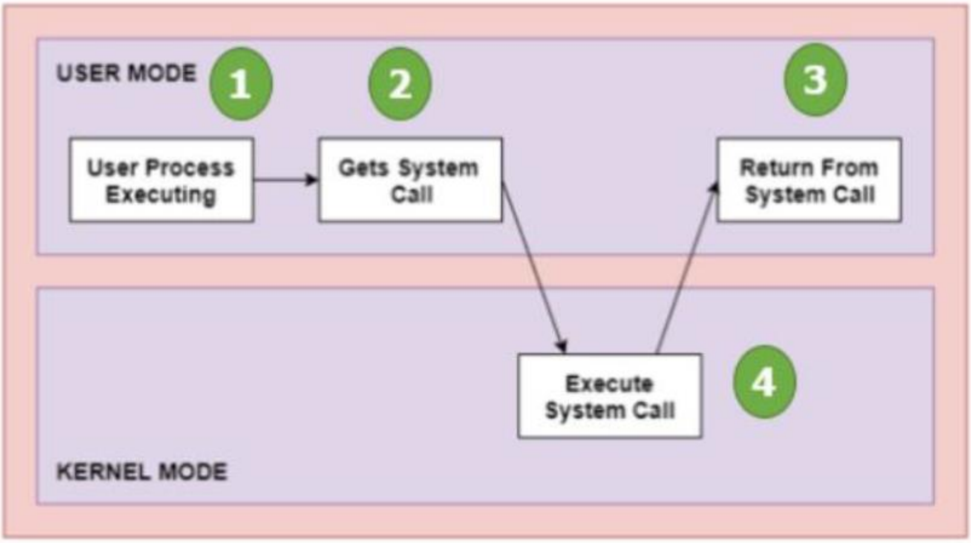
   - The OS ensures that all access to system resources is controlled.
   - The OS ensures that external I/O devices are protected from invalid access attempts.
   - The OS provides authentication features for each user by means of passwords.

What is System Call in Operating System?

A system call is a mechanism that provides the interface between a process and the operating system. It is a programmatic method in which a computer program requests a service from the kernel of the OS. System call offers the services of the operating system to the user programs via API (Application Programming Interface). System calls are the only entry points for the kernel system.

How System Call Works?

Here are steps for System Call:



As you can see in the above-given diagram.

Step 1) The processes executed in the user mode till the time a system call interrupts it.

Step 2) After that, the system call is executed in the kernel-mode on a priority basis.

Step 3) Once system call execution is over, control returns to the user mode.,

Step 4) The execution of user processes resumed in Kernel mode.

Why do you need System Calls in OS?

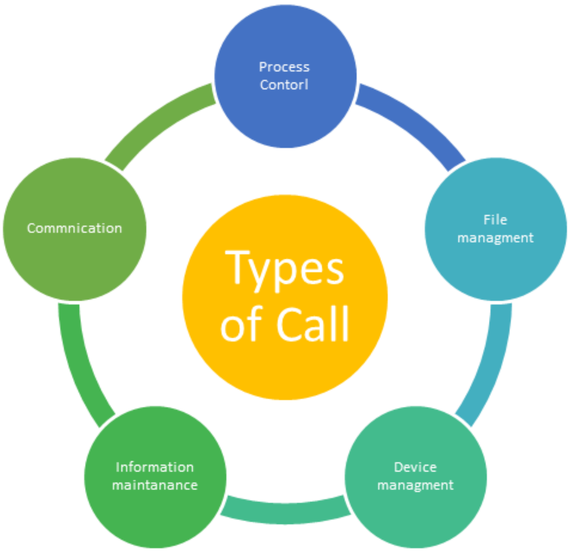
Following are situations which need system calls in OS:

- Reading and writing from files demand system calls.
- If a file system wants to create or delete files, system calls are required.
- System calls are used for the creation and management of new processes.
- Network connections need system calls for sending and receiving packets.
- Access to hardware devices like scanner, printer, need a system call.

Types of System calls

Here are the five types of system calls used in OS:

1. Process Control
2. File Management
3. Device Management
4. Information Maintenance
5. Communications



Process Control: This system calls perform the task of process creation, process termination, etc.

Functions:

- End and Abort
- Load and Execute
- Create Process and Terminate Process
- Wait and Signed Event
- Allocate and free memory

File Management: File management system calls handle file manipulation jobs like creating a file, reading, and writing, etc.

Functions:

- Create a file
- Delete file
- Open and close file
- Read, write, and reposition
- Get and set file attributes

Device Management: Device management does the job of device manipulation like reading from device buffers, writing into device buffers, etc.

Functions:

- Request and release device
- Logically attach/ detach devices
- Get and Set device attributes

Information Maintenance: It handles information and its transfer between the OS and the user program.

Functions:

- Get or set time and date
- Get process and device attributes

Communication: These types of system calls are specially used for interprocess communications.

Functions:

- Create, delete communications connections
- Send, receive message
- Help OS to transfer status information
- Attach or detach remote devices

Important System Calls Used in OS

wait()

In some systems, a process needs to wait for another process to complete its execution. This type of situation occurs when a parent process creates a child process, and the execution of the parent process remains suspended until its child process executes.

The suspension of the parent process automatically occurs with a wait() system call. When the child process ends execution, the control moves back to the parent process.

fork()

Processes use this system call to create processes that are a copy of themselves. With the help of this system Call parent process creates a child process, and the execution of the parent process will be suspended till the child process executes.

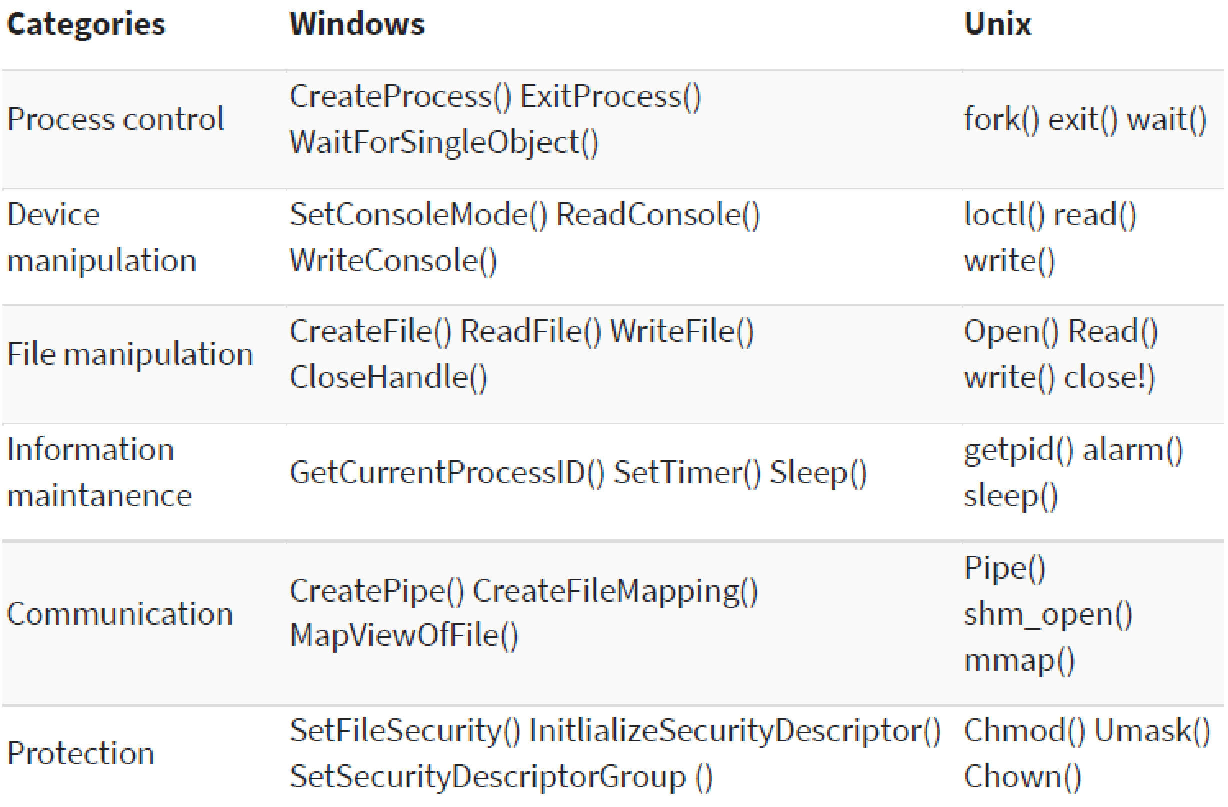
exec()

This system calls runs when an executable file in the context of an already running process that replaces the older executable file. However, the original process identifier remains as a new process is not built, but stack, data, head, data, etc. are replaced by the new process.

kill():

The kill() system call is used by OS to send a termination signal to a process that urges the process to exit. However, a kill system call does not necessarily mean killing the process and can have various meanings. exit(): The exit() system call is used to terminate program execution. Specially in the multi-threaded environment, this call defines that the thread execution is complete. The OS reclaims resources that were used by the process after the use of exit() system call.

Summary:



Categories	Windows	Unix
Process control	CreateProcess() ExitProcess() WaitForSingleObject()	fork() exit() wait()
Device manipulation	SetConsoleMode() ReadConsole() WriteConsole()	Ioctl() read() write()
File manipulation	CreateFile() ReadFile() WriteFile() CloseHandle()	Open() Read() write() close!)
Information maintanence	GetCurrentProcessID() SetTimer() Sleep()	getpid() alarm() sleep()
Communication	CreatePipe() CreateFileMapping() MapViewOfFile()	Pipe() shm_open() mmap()
Protection	SetFileSecurity() InitlializeSecurityDescriptor() SetSecurityDescriptorGroup ()	Chmod() Umask() Chown()