History

Warning!

Switching back and forth between OS and programs is, in many operating systems, a relatively time-consuming operation, due to overheads that should become clear later

History

Warning!

Switching back and forth between OS and programs is, in many operating systems, a relatively time-consuming operation, due to overheads that should become clear later

For now, just think of the overheads of saving and restoring the CPU state of the running program, just as for an interrupt

History

Warning!

Switching back and forth between OS and programs is, in many operating systems, a relatively time-consuming operation, due to overheads that should become clear later

For now, just think of the overheads of saving and restoring the CPU state of the running program, just as for an interrupt

These overheads are another reason why you don't want timer interrupts too often

History

The result of all this messing with modes is certain operations like loading programs, or accessing hardware like a printer, are only available to the OS

History

The result of all this messing with modes is certain operations like loading programs, or accessing hardware like a printer, are only available to the OS

If an unprivileged program tries to access the printer directly, that again trips an interrupt and the OS takes over anyway

History

The result of all this messing with modes is certain operations like loading programs, or accessing hardware like a printer, are only available to the OS

If an unprivileged program tries to access the printer directly, that again trips an interrupt and the OS takes over anyway

Forcing access to hardware via the OS also provides protection and management for other system resources, like access to files or the network

History

The result of all this messing with modes is certain operations like loading programs, or accessing hardware like a printer, are only available to the OS

If an unprivileged program tries to access the printer directly, that again trips an interrupt and the OS takes over anyway

Forcing access to hardware via the OS also provides protection and management for other system resources, like access to files or the network

In kernel mode, everything is possible

History

The result of all this messing with modes is certain operations like loading programs, or accessing hardware like a printer, are only available to the OS

If an unprivileged program tries to access the printer directly, that again trips an interrupt and the OS takes over anyway

Forcing access to hardware via the OS also provides protection and management for other system resources, like access to files or the network

In kernel mode, everything is possible

In user mode, only “safe” things are possible

History

Preemption and protection appeared in OSs for large mainframe computers and Unix for minicomputers in the late 1960s

History

Preemption and protection appeared in OSs for large mainframe computers and Unix for minicomputers in the late 1960s

When microcomputers (IBM PC) arrived in the early 1980s much of OS knowledge was thrown away and DOS (Disk Operating System) was non-preemptive, single process and no protection

History

Preemption and protection appeared in OSs for large mainframe computers and Unix for minicomputers in the late 1960s

When microcomputers (IBM PC) arrived in the early 1980s much of OS knowledge was thrown away and DOS (Disk Operating System) was non-preemptive, single process and no protection

This was because the earliest PC hardware did not support such things (no rings)

History

Support was rapidly added in later PC hardware, but DOS and, later, Windows 3.1 took no advantage of it: the lack of protection meaning a single bad program could mess up the OS and crash the entire computer

History

Support was rapidly added in later PC hardware, but DOS and, later, Windows 3.1 took no advantage of it: the lack of protection meaning a single bad program could mess up the OS and crash the entire computer

Windows NT was the first true OS from Microsoft (mid 1990s) for PCs, possibly as much as a decade after other OSs (such as Unix derivatives) were providing preemption and protection on the same hardware

History

Support was rapidly added in later PC hardware, but DOS and, later, Windows 3.1 took no advantage of it: the lack of protection meaning a single bad program could mess up the OS and crash the entire computer

Windows NT was the first true OS from Microsoft (mid 1990s) for PCs, possibly as much as a decade after other OSs (such as Unix derivatives) were providing preemption and protection on the same hardware

Incidentally, Microsoft's need for backwards compatability with these early systems is a major reason why they have so many problems with security