

CLOUD COMPUTING AND EDGE COMPUTING

Walkin' Through the Boot Process

Here's what should go through your brain when a user says, "My computer won't start"

Kathy Ivens | Aug 30, 2004

Every network administrator has received calls from users complaining that their "computer won't start"—a nebulous, uninformative phrase that can cause quite a bit of frustration. Typically, users report that something untoward happened during the startup process—either during the computer's Power On Self Test (POST) or during the Windows startup procedures. To diagnose and cure such maddening problems, you need to understand how the boot process works.

The phrase "boot failure" describes both machine and OS problems. But in the days of MS-DOS computers, the POST took longer than OS startup, and hardware was the source of most boot-failure problems. Computer hardware has become more reliable over the years, and—thanks to advanced BIOS features—the computer's ability to track, diagnose, and control that hardware is more robust. Therefore, you're more likely to encounter an OS problem when a system fails to boot. Let's walk through the startup process to see what happens at each step and to understand the meaning of any error messages you encounter. (For the purpose of this discussion, I assume you're using Windows 2000 or later.)

Power On

Is the user complaining that nothing's happening when he or she presses the power button? If so, first check the plug.

Here's an old administrator's trick for dealing with unplugged computers when you're working with a user over the phone. Users often don't check whether their unresponsive computer is unplugged, and when you mention this possibility, they're embarrassed if it turns out to be the cause. The user might say, "Of course, it's plugged in," but you need to know whether that's the truth. Ask the user to pull the plug and reinsert it, citing a need to "check the polarity issues." (Try not to giggle.) You'll be amazed how often users will report, "Hey, that worked."

If it's not the plug, it's probably the power supply—the most vulnerable hardware component in your system. Power supplies aren't expensive, but replacing them is a boring, labor-intensive exercise.

Hardware and BIOS Checks

If the user sees an error message during the POST, or if the computer simply hangs before the OS starts, the problem is in the hardware or the BIOS. The system reports hardware and BIOS errors to the screen, along with beeps to get your attention. Some BIOS errors appear as numbers, and at one time all BIOS manufacturers used the same numbers (the numbers that IBM used), but that changed. Today, if you see an error number, you need the documentation that came with your computer to interpret it. (You can also look it up by checking the BIOS manufacturer's Web site.) However, you're far more likely to see text rather than numbers, as in *Hard drive controller failure* or the always amusing *Keyboard error, press F1 to continue*.

You might also see an error that references memory problems. In the old days, memory components had an extra chip called a "parity chip," and part of the BIOS test was a parity test. Memory components no longer include parity checking because it's not really necessary anymore: Memory manufacturing has advanced to the point at which failure is highly unusual. However, after you add memory to a machine, you might see a memory error message at the next boot. The message

displays text such as *Mismatched memory information*. This error message is actually a confirmation that the system sees the memory you installed but finds that it doesn't match the total recorded in CMOS.

To solve this problem, try restarting the computer and entering the keystrokes required to get into the BIOS setup program. In my experience, doing so jumpstarts the solution because the correct memory count automatically appears as soon as you enter the BIOS setup screen, and all that's left to do is exit the BIOS setup program. Accessing the BIOS setup program causes the system to check the memory count and adjust it so that it matches the physical memory total.

If you add memory to a computer and encounter an error message that doesn't mention a mismatched memory count, you have a more serious problem. The system doesn't recognize the new memory. This situation is almost always caused by an error in the physical insertion of the memory, such as using the wrong slot or not inserting the teeth properly. However, I've also seen the problem when the wrong memory type is inserted (e.g., inserting DRAM in an older computer with Enhanced Data Output—EDO), when the motherboard doesn't like mixing SIMMs and DIMMs, or when the motherboard doesn't like mixed memory speeds. Some motherboards require a change in dipswitches or jumper configuration when you add memory, although those requirements are becoming less common. To avoid these problems, always check the motherboard documentation before adding memory.

If you see a hard disk error during POST, you have a serious problem. Actually, I've found that half the time the problem is the controller—not the disk—and replacing the controller lets the disk boot normally, with all data intact (whew!). If an embedded controller dies, you don't have to buy a new motherboard; instead, you can buy a controller card. Check the motherboard documentation for the tasks required to make the BIOS see the card instead of looking for the embedded chip.

If the problem is indeed the disk, you have more work to do than merely replacing a controller. In addition to replacing the disk, you have to reinstall the OS and

applications, as well as restore from the most recent backup—which is, of course, dated yesterday, right?

Master Boot Record Takes Control

Next, the computer begins the task of loading the OS. During installation, the Windows installation program places data on the first sector of your computer's primary partition (the boot sector). That data is the Master Boot Record (MBR), and it contains executable instructions. The installation program also copies the two files that initiate the Windows boot sequence—Ntldr and Ntdetect—to the boot disk's root directory. In addition, Windows Setup copies boot.ini, the file that contains startup options, to the boot disk's root directory.

In addition to the executable instructions, the MBR has a table that defines the locations of the disk's primary partitions. (When you install Windows, you don't have to make the system partition and the boot partition the same partition, although that's the common approach.) The Windows startup files are on the system partition, and the OS files are on the boot partition. (Yes, the naming logic is backwards.)

The system partition holds the hardware-specific files that are necessary to boot Windows, including the MBR. This partition must be a primary partition, and it must be marked active. It's always drive C, because that's the drive the BIOS accesses to turn the boot process over to the MBR. The boot partition holds the OS files (the %systemroot% folder) and the OS support files (%systemroot%\System).

In the last step of the hardware boot process, the computer reads the MBR into memory and transfers control of the computer to that MBR code. The executable code searches the primary partition table for a flag on a partition that indicates that the partition is bootable. When the MBR finds the first bootable partition, it reads the first sector of the partition, which is the boot sector.

Ntldr Runs

The boot-sector code reads Ntldr into memory to start the OS boot process. Ntldr

contains read-only NTFS and FAT code. It starts running in real mode, and its first job is to switch the system to a form of protected mode. (For more information about these modes, see the sidebar, "Real Mode vs. Protected Mode.") This initial instance of protected mode can't perform the full physical-to-virtual translations that provide hardware protection—that feature becomes available when the OS has finished booting.

All the physical memory is now available to the OS, and the computer is operating as a 32-bit machine. Ntldr enables paging and creates the page tables. Next, Ntldr reads boot.ini from the root directory and, if you're dual booting or you've configured boot.ini to display a menu, the boot-selection menu appears on the monitor. If Ntldr is missing or corrupted, you'll see the error message *Ntldr is missing. Press Ctrl-Alt-Del to restart.*

Don't waste your time following the suggested sequence; you'll just recycle the system back to the same error message. You must replace Ntldr. If you created a bootable floppy for the system, you can use that disk to copy Ntldr from the floppy disk to the boot disk's root directory (usually C). If Ntldr is missing, simply copy it. If the file exists on the hard disk, it's probably corrupted. To replace it, you must first change the read-only attribute. If you don't have a bootable floppy disk, you'll have to start Setup from the Windows CD-ROM and select Repair.

Ntdetect Runs

Ntldr launches Ntdetect, which queries the system's BIOS for device and configuration information. The system sends the information that Ntdetect gathers to the registry and places it into HKEY_LOCAL_MACHINE\HARDWARE\DESCRIPTION subkeys.

If a problem occurs with Ntdetect (e.g., it's missing or corrupted), you probably won't see an error message. Instead, the boot process typically just stops. The cure for a missing or corrupted Ntdetect file is to replace it. Use a bootable floppy disk to boot the computer, then copy Ntdetect from that floppy disk to the hard disk's root directory. Alternatively, start Setup from the Windows CD-ROM and select Repair.

Ntoskrnl Runs and HAL Is Loaded

After Ntdetect finishes its hardware-checking routines, it turns the OS boot process back to Ntldr, which launches ntoskrnl.exe and loads the Hardware Abstraction Layer (HAL) .dll file. (Both files are in the `\%systemroot%\system32` directory.) Ntoskrnl is the core file for the Windows kernel and executive subsystems. It contains the Executive, the Kernel, the Cache Manager, the Memory Manager, the Scheduler, the Security Reference Monitor, and more. Ntoskrnl is the file that really gets Windows going. Ntoskrnl needs the hal.dll, which has the code that lets hardware interact with the OS.

You might see an error message that indicates a problem with Ntoskrnl, but the message is almost always spurious and appears because the directory referenced in boot.ini doesn't match the name of the directory into which the Windows system files were installed. This generally means that someone renamed the `\%systemroot%` directory or created a new directory and moved the Windows files into it. The solution is to move the files back to the location specified in boot.ini. If someone has edited boot.ini, you'll have to correct that error.

Drivers and Services Load

Ntldr now loads the low-level system services and device drivers, but the services aren't initialized—that occurs later. This is the end of the boot sequence, and the process that begins now is the *load sequence*, or the *kernel phase*.

Ntldr has a pecking order for loading system services and device drivers. When you install Windows, drivers and system services are copied to your computer, and information about them is written to the registry. The registry data is a hexadecimal entry that ends with a number in parentheses. That number gives Ntldr its pecking order for loading drivers and system services. For an example, open the registry and go to `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services`. You'll see a long list of services and device drivers. Select any subkey and look at the `REG_DWORD` data item named Start.

- The data value (0) means the service is loaded during the kernel load phase.

- The data value (1) means the service is loaded during the kernel initialization phase (the next phase).
- The data value (2) means the service is loaded during the services load phase.
- The data value (3) means the service is enabled but not initialized (the service requires a manual startup, which you perform in the Microsoft Management Console (MMC) Services snap-in).
- The data value (4) means the service isn't enabled.

The OS Loads

Ntoskrnl begins to load the OS. The Windows kernel is initialized, and subsystems are loaded and initialized. These actions provide the basic systems that are necessary to complete the task of loading the OS. The boot drivers that Ntldr loaded earlier are now initialized, followed by initialization of the rest of the drivers and services. When the first-level drivers are initialized, you might encounter a problem, typically in the form of a STOP error or a Blue Screen of Death. This problem almost always occurs during the first boot after you update a driver. When Ntoskrnl initializes the driver, the OS balks because it doesn't like it.

To solve this problem, restart the computer, press F8 to display the Advanced Options menu, and load the Last Known Good Configuration to roll back to the previous driver. Then, either obtain a better driver from the manufacturer or stick with your rollback to the previous driver.

The Windows kernel and executive systems are now operational. The Session Manager Subsystem (smss.exe) configures the user environment. The system checks information in the registry so that it can begin loading the remaining drivers and software that need to be added. The kernel also loads kernel32.dll, gdi32.dll, and user32.dll, which provide the Win32 API services that software programs require.

The Computer Logs on to the Domain

While the kernel is still loading and initializing drivers, the computer logs on to the domain. Using its machine account (a unique name, with its own password), the

computer opens a secure channel (sometimes called a clear channel) to a domain controller (DC). All this occurs before the user logon features are available.

Machine accounts are used between client computers (including member servers) and DCs. Within each domain, the same process occurs among multiple DCs. Therefore, the order in which you restart computers after a shutdown is important. Computers use the secure channel to exchange the information necessary for authentication and authorization functions. Machine accounts enhance network security, making sure that a computer attempting to send sensitive information is really a member of the domain.

As an additional security feature, computers (like users in a security-conscious network configuration) must change their passwords periodically. By default, the password change interval is 30 days. When it's time to change the password, the computer generates a new password and sends it through the secure channel (which it accesses by using the previous password) to the nearest DC. Thereafter, the computer must use the new password to access a secure channel.

The DC updates its database and immediately replicates the computer password change to the other DCs in the domain. Computer account passwords are flagged as *Announce Immediately* events, so they don't wait for the next scheduled DC replication. Sometimes, these events can cause serious performance hits. If many (or all) of the computers in your domain have passwords that expire on the same day, the work that the DCs have to do can immediately slow down other important DC tasks, such as authenticating users or running scheduled replications. The situation is even worse if you have a DC that's providing other services, such as acting as a DNS server. You can change the way machine passwords are managed for the domain, for an OU, or for an individual computer—although attempting to improve performance by configuring one computer at a time isn't efficient. In a future article, I'll discuss the methods available for changing the way computers log on to a domain.

User Logon Services Load

The Win32 subsystem launches winlogon.exe, which sends the logon dialog box to the screen and loads the Local Security Authority (lsass.exe). The logon process begins, and the user must enter a username and password in the Log On To Windows dialog box. Assuming the user knows the username and password, the system completes the logon process, and the user can begin working. At this point, the Windows startup is complete, and the current startup configuration settings become the newest Last Known Good. Notice that a successful logon is necessary to make startup a Last Known Good.

No More Nerves

Startup failures make everyone nervous—both users and Help desk personnel. Understanding the startup process makes problems less intimidating and helps you resolve those problems quickly and easily.

Source URL:<https://www.itprotoday.com/cloud-computing/walkin-through-boot-process>