

## OBJECTIVES

This chapter helps you to prepare for the Operating Systems Technologies module of the A+ Certification examination by covering the following objectives within the “Domain 1.0: Operating System Fundamentals” section.

### **1.2 Identify the names, locations, purposes, and contents of major system files.**

**Content may include the following:**

- **Windows 9x–specific files:**

- IO.SYS
- MSDOS.SYS
- AUTOEXEC.BAT
- COMMAND.COM
- CONFIG.SYS
- HIMEM.SYS
- EMM386.EXE
- WIN.COM
- SYSTEM.INI
- WIN.INI

- **Registry data files**

- SYSTEM.DAT
- USER.DAT

- **Windows NT–based specific files:**

- BOOT.INI
- NTLDR
- NTDETECT.COM
- NTBOOTDD.SYS
- NTUSER.DAT
- **Registry data files**



# CHAPTER 2

## Major Operating System Files

## OBJECTIVES

Computer technicians must be familiar with the functions and structure of operating systems that they may encounter in the field. Currently, the major operating system lines associated with personal computers are Windows 9x/Me and Windows NT/2000/XP.

## OUTLINE

<b>Introduction</b>	<b>93</b>
<b>Microsoft Disk Operating Systems</b>	<b>93</b>
MS-DOS Startup Process	94
The MS-DOS Core Structure	95
MS-DOS Configuration Files	97
<b>Basic Memory Management</b>	<b>98</b>
Conventional Memory	98
The Upper Memory Area	99
Extended Memory	100
Virtual Memory	100
<b>CONFIG.SYS</b>	<b>101</b>
Memory Managers	101
Files, Buffers, and Stacks	102
Device Drivers	102
<b>AUTOEXEC.BAT</b>	<b>103</b>
<b>Windows Initialization Files</b>	<b>104</b>
<b>Windows 9x Startup</b>	<b>105</b>
Windows 9x/Me Registry Data Files	108
Automated Application Startups	109
Configuration and Initialization	
File Conflicts	109

## OUTLINE

<b>Windows NT Startups</b>	<b>110</b>
Windows NT/2000/XP Registry	
Data Files	111
<b>Chapter Summary</b>	<b>113</b>
Key Terms	113
<b>Apply Your Knowledge</b>	<b>114</b>
Review Questions	114
Answers and Explanations	116
Challenge Solutions	117
<b>Suggested Readings and Resources</b>	<b>117</b>

## STUDY STRATEGIES

To prepare for the Operating System Fundamentals objective of the Operating Systems Technologies exam:

- ▶ **Use all the traditional study tools we've placed in the chapter**—Pay attention to the Objectives, Challenges, and end-of-chapter questions and use them to learn the material.
- ▶ **Use the pedagogy in this chapter to focus on the exam-specific material**—We've included lots of features geared expressly to the A+ exam. The Exam Tips scattered throughout the chapter are placed there to point to known exam-related materials. The same is true of the embedded Challenge items.
- ▶ **Key in on Exam Tips in the chapter**—While reading through the chapter, make sure to concentrate on the following test-related items:
  - Be aware of the minimum file size specification associated with the `MSDOS.SYS` file.
  - Memorize the sequence in which files are loaded in a DOS startup.
  - Memorize the filenames of the virtual memory swap files used in each operating system.
  - Know which statements and commands are normally located in the `CONFIG.SYS` file.
  - Know the purpose and function of the `HIMEM.SYS` file.
  - Remember which driver assigns logical drive letters to the system's floppy drives.
  - Know which commands are normally located in a typical `AUTOEXEC.BAT` file.
  - Be aware of how Windows versions before Windows 95 organized and monitored the system's configuration information.
  - Be aware of which file loads Windows 9x into the system.
  - Remember which two files make up the Windows 95 Registry and where they are located in the system.
  - Know where the Windows 98 Registry files are stored and what they are called.

## STUDY STRATEGIES

- Remember how to prevent the items in the Windows 9x Startup folder from running at startup.
- Know what function the VCACHE utility performs in a Windows 9x system.
- Be aware of the effect that active commands in a `CONFIG.SYS`, `AUTOEXEC.BAT`, or `.INI` file can have on the operation of an advanced Windows operating system.
- Memorize the names and functions of the files involved in the Windows NT/2000 boot process and know which order they are executed in.
- Be aware of what the `NTBOOTDD.SYS` file does in a Windows NT/2000 system.
- Know how to make a backup copy of the Registry in Windows NT 4.0.

## INTRODUCTION

All computer systems require an operating system to guide and control the operation of its hardware and to link specialized application software to the hardware. This chapter deals with basic operating system structures and operations. The initial sections of the chapter use the Microsoft MS-DOS operating systems as an example of basic operating system structures and organization because it represents one of the most fundamental operating systems available.

The second major section of the chapter deals with traditional memory organization and management strategies for Microsoft operating systems.

The third major portion of the chapter covers the names, locations, and purposes of the major Windows 9x/Me files. This is followed by a similar discussion of the Windows NT, 2000, and XP operating systems.

After completing the chapter, you should be able to describe the structures and major system files associated with the MS-DOS, Windows 9x/Me, and Windows NT/2000/XP operating systems.

## MICROSOFT DISK OPERATING SYSTEMS

Most microcomputers use a bootstrapping process to load the operating system into RAM. *Bootstrapping* is a process in which the operating system is loaded into memory by a smaller program called the *bootstrap loader*. In personal computers, the bootstrap operation is one of the functions of the ROM BIOS. The operating system can be loaded from a ROM chip, floppy disk, hard-disk drive, or another computer. Loading the more powerful operating system files from the disk increases the system's onboard intelligence considerably.

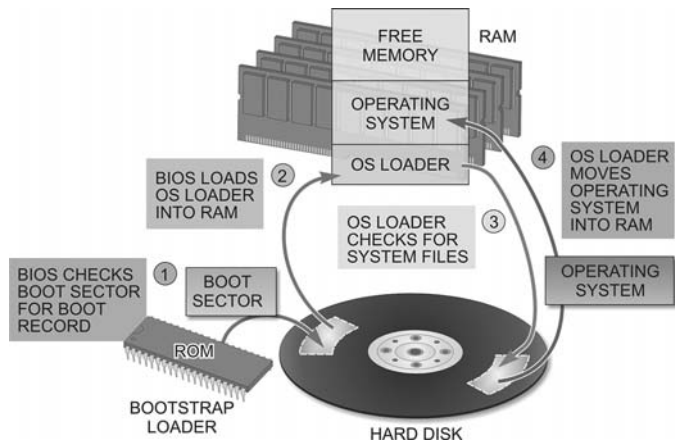
The bootstrap process is primarily used in disk-drive-based systems to load an operating system that can control such computers. MS-DOS is a disk operating system for IBM PC-compatible computers. In its day, it was easily the most popular operating system in the world. It is also the basis from which Windows 9x derives its underlying organization.

## MS-DOS Startup Process

The bootup process in a PC begins when the BIOS starts looking through the system for a *master boot record (MBR)*. This record can reside on drive A: or C:, or at any other location in the system. A simple single-operating system, single-disk bootup process is depicted in Figure 2.1. As you can see, this multiple-access operation uses two different bootstrap routines to locate and load two different boot records.

**FIGURE 2.1**

The bootstrap operation.



The first section on any logical disk is called the *boot sector*. This section contains information about how the disk is organized. It can also contain the small, optional master boot record that can access a larger, more powerful *bootstrap loader* program located in the root directory of the logical disk.

In most systems, the master boot record is found at sector 1, head 0, and track 0 of the first logical hard drive. Some texts might refer to the first sector as sector 0, keeping with the idea that the first one of anything in a digital system is 0. If the disk possesses a master boot record, it can boot up the hardware system to the operating system. The disk is then referred to as a *bootable disk*. If the disk does not possess an MBR, it is just a data disk that can be used for storing information.

Traditionally, BIOS programs search for the master boot record in floppy-disk drive A: first. In later models, the BIOS looked first in the floppy drive, or drives, and then in the hard-disk drive. In newer systems, the order that the BIOS uses to search for the MBR is governed by information stored in the system's CMOS configuration RAM. The order can be set to check the floppy drive first and then the hard drive, to check the hard drive first, to check the hard drive only, or most recently, to check the CD-ROM drive.

If the BIOS does not locate the boot record in one of the indicated drives, it will most likely display a `Non-System Disk or Disk Error OR ROM BASIC Interpreter Not Found` message on the screen.

When an MBR is located, the bootstrap loader moves the boot record into system RAM to be executed. This record contains the secondary bootstrap loader, also referred to as the *operating system loader*. This routine looks for an operating system boot record, typically located on the disk. When this record is found, the routine loads the bigger boot record into RAM and begins executing it. This boot record brings special operating system files into memory so that they can control the operation of the system (that is, the operating system). In the case of Microsoft DOS, the special files loaded by the OS boot record are the `IO.SYS` and `MSDOS.SYS` files.

In the MS-DOS system, the `IO.SYS` file executes the contents of the `MSDOS.SYS` file and looks for the MS-DOS command processor and moves it into system RAM along with the operating system support files. The default command processor for MS-DOS is a system file called `COMMAND.COM`. This file processor provides the basic user interface called the *command line* and interprets the input entered at the command prompt.

## The MS-DOS Core Structure

Although the concept of MS-DOS has largely disappeared from the consumer computer market, its structure and command-line interface have not gone away for technicians. There are many diagnostic and troubleshooting situations in which technicians must have a firm understanding of DOS structures and command-line operations. Therefore, we will begin with a description of the MS-DOS boot operation and structure and then use it to contrast the Windows 9x/Me and Windows NT/2000/XP operating systems.

### EXAM TIP

Know what types of error messages are produced when no valid MBR is found in a system.

The main portions of the MS-DOS operating system are the `IO.SYS`, `MSDOS.SYS`, and `COMMAND.COM` files. `IO.SYS` and `MSDOS.SYS` are special hidden system files that do not show up in a normal directory listing.

The `IO.SYS` file moves the system's basic I/O functions into memory and then implements the MS-DOS default control programs, referred to as *device drivers*, for various hardware components. They include

- The boot disk drive
- The console display and keyboard
- The system's time-of-day clock
- The parallel and serial communications port

Conversely, the `MSDOS.SYS` file provides default support features for software applications. These features include

- Memory management
- Character input and output
- Real-time clock access
- File and record management
- Execution of other programs

**EXAM TIP**

Be aware of the minimum file size specification associated with the `MSDOS.SYS` file.

There is a little known MS-DOS system requirement that the `MSDOS.SYS` file must maintain a size in excess of 1KB.

The operating system provides the system's user interface. In MS-DOS, the `COMMAND.COM` command interpreter contains the operating system's most frequently used commands. It also provides the system's primary user interface in the form of the command prompt. The command prompt for most command-line-based operating systems is some type of character on the screen (for example, `C:\>`). The command line is the space immediately following the command prompt on the screen. All commands are typed in this space. They are executed by pressing the Enter key on the keyboard.

When a command is entered at the command-line prompt, the `COMMAND.COM` program examines it to see whether it is an internal DOS command or an external DOS command. Internal commands are understood directly by `COMMAND.COM`; external commands are

stored in a directory called `\DOS`. If one of the internal commands is entered, the `COMMAND.COM` file can execute it immediately. If not, `COMMAND.COM` looks in the `\DOS` directory for the command program. Internal commands include those that can be carried out from the operating system's core, such as producing a directory listing of a disk (`DIR`) or changing directories (`CD`). External commands typically perform more complex functions, such as starting utility programs to carry out operations such as formatting disks (`FORMAT`), printing files (`PRINT`), and copying files (`XCOPY`).

When MS-DOS runs an application, `COMMAND.COM` finds the program, loads it into memory, and then gives it control of the system. When the program is shut down, it passes control back to the command interpreter.

## MS-DOS Configuration Files

In the MS-DOS system, two special configuration files, known as `CONFIG.SYS` and `AUTOEXEC.BAT`, can be included in the boot process. These programs optimize the system for operations in particular functions, or with different options. These files may also be found in systems that have been upgraded to a Windows operating system. In these cases, they supply backward compatibility with older hardware and software products.

As the system moves through the boot procedure, the BIOS checks in the root directory of the boot disk for the presence of the `CONFIG.SYS` file. Afterward, it searches for the `COMMAND.COM` interpreter, and finally looks in the root directory for the `AUTOEXEC.BAT` file. Both the `CONFIG.SYS` and `AUTOEXEC.BAT` files play key roles in optimizing the system's memory and disk-drive usage. Their involvement in the boot process can be summarized as follows:

1. The BIOS searches drives for the master boot record (MBR).
2. The primary bootstrap loader moves the master boot record into memory.
3. The system executes the secondary bootstrap loader from the master boot record.
4. The secondary bootstrap loader moves `IO.SYS` and `MSDOS.SYS` into memory.

Memorize the sequence in which files are loaded in an MS-DOS startup.

5. `IO.SYS` runs the `MSDOS.SYS` file to load memory- and file-management functions.
6. `IO.SYS` checks for the `CONFIG.SYS` file in the root directory.
7. If `CONFIG.SYS` is found, `IO.SYS` uses it to reconfigure the system in three read sequences (device, install, and shell).
8. `IO.SYS` loads `COMMAND.COM`.
9. `COMMAND.COM` checks for the `AUTOEXEC.BAT` file in the root directory.
10. If the `AUTOEXEC.BAT` file is found, `COMMAND.COM` carries out the commands found in the file.
11. If no `AUTOEXEC.BAT` file is found, `COMMAND.COM` displays the Time and Date prompt on the screen.

While the system is operating, the BIOS continues to perform several important functions. It contains routines on which the operating system calls to carry out basic services.

## BASIC MEMORY MANAGEMENT

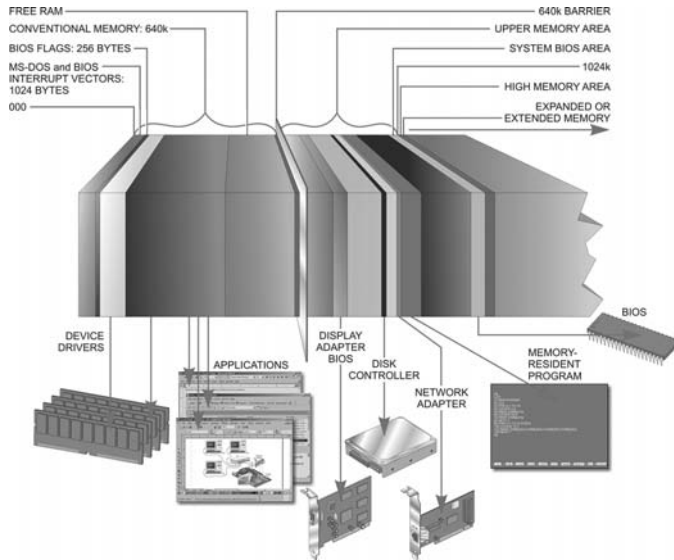
Decisions made regarding the original IBM PC design, and thereby the MS-DOS operating system that ran it, still affect the design of PCs and operating systems. Technicians who work on PC-compatible systems must understand how they allocate memory and how that memory can be manipulated to provide the best system performance.

Basically, MS-DOS could recognize the following classifications of memory: conventional memory, upper memory blocks, high memory area, expanded memory, extended memory, and virtual memory. Because of compatibility issues, these decisions have carried over into the address allocations of all MS-DOS-based PC compatibles, as depicted in Figure 2.2.

### Conventional Memory

Conventional memory (locations 00000 through 9FFFF) is the standard memory area for all PC-compatible systems. It traditionally

holds the operating system, interrupt vector tables, and relocated ROM BIOS tables. The remaining space in the conventional memory area is referred to as *DOS Program Memory*. (Programs written to operate under PC-DOS or MS-DOS use this area for program storage and execution.)



**FIGURE 2.2**  
PC memory allocations.

## The Upper Memory Area

The *upper memory area (UMA)* occupies the 384KB portion of the PC's address space from A0000 to FFFFF. This space is not normally considered to be part of the computer's total address space because programs cannot store information in this area. Instead, the area is reserved to run segments of the system's hardware. Address spaces from A0000 through BFFFF are dedicated addresses set aside for the system's video display memory. The system's ROM BIOS occupies the address space between locations FE000 and FFFFF.

Between the video memory and system BIOS areas, addresses are reserved to hold BIOS extension programs for add-on hardware adapters. Typical BIOS extensions include those for hard-drive adapters, advanced video adapters, and network adapters. After the BIOS extensions are in place, the typical UMA still has many unused memory areas that can have information mapped (copied) into them.

This space is segmented into 64KB sections called *upper memory blocks (UMBs)*. The primary use for these blocks is to hold installable device drivers and other memory-resident programs moved out of the conventional memory area. When these programs are moved out of the conventional memory area, more space is made available there for use by application programs.

PCs also use this area to incorporate a memory-usage scheme called *Shadow RAM* to improve their overall performance. With this feature, the contents of the system BIOS and/or adapter BIOS are rewritten (shadowed) into faster extended-memory RAM locations. The operating system then remaps the ROM addresses to the corresponding RAM locations through unused portions of the UMA. Shadowing enables the system to operate faster when application software uses any of the BIOS's CALL routines.

## Extended Memory

With the advent of the 80286 microprocessor and its protected operating mode, it became possible to access physical memory locations beyond the 1MB limit of the 8088. Memory above this address is generally referred to as *extended memory*.

Modern operating systems can take full advantage of extended memory through the Protected Addressing modes of the more advanced microprocessors. This capability to manage higher memory enables the system to free up more of the base memory area for use by application programs.

## Virtual Memory

As mentioned in Chapter 1, "Microsoft Windows Operating Systems," operating systems shift programs and data between RAM and special swap files on the disk drive to produce the appearance of more RAM than actually exists. This type of memory manipulation is referred to as *virtual memory*. The MS-DOS operating systems supported virtual memory operations through a special device driver named SMARTDRV.

In the Windows 9x/Me operating systems, the real-mode SMARTDRV driver was replaced by the VCACHE virtual mode dri-

ver, which was capable of operating in the virtual protected operating modes provided by the enhanced capabilities of advanced micro-processor and operating systems.

## CONFIG.SYS

The `CONFIG.SYS` program is responsible for the following:

1. Setting up any memory managers being used
2. Configuring the DOS program for use with options devices and application programs
3. Loading device-driver software into memory and installing memory-resident programs

Activities 1–3 are illustrated in the following sample `CONFIG.SYS` file. Keep in mind that an actual `CONFIG.SYS` file doesn't have the numbers (1–3), as shown in the following example. Use these numbers to match up the activities in the preceding list.

```
1. Device=C:\DOS\HIMEM.SYS
   Device=C:\DOS\EMM386.EXE 1024 RAM
2. FILES=30      BUFFERS=15
   STACKS=9,256
3. DEVICE=C:\DOS\SMARTDRV.SYS 1024
   DOS=HIGH,UMB
   DEVICEHIGH=C:\MOUSE\MOUSE.SYS
   DEVICEHIGH=C:\DOS\RAMDRIVE.SYS 4096/a
```

### EXAM TIP

Know which statements and commands are normally located in the `CONFIG.SYS` file.

## Memory Managers

In the first section of the `CONFIG.SYS` file, the system's memory-manager programs are loaded. In this case, the `HIMEM.SYS` command loads the DOS Extended Memory Manager (XMS). This driver manages the use of extended memory installed in the system so that no two applications use the same memory locations at the same time. This memory manager should normally be listed in the `CONFIG.SYS` file before any other memory managers or device drivers.

`HIMEM.SYS` also creates a 64KB area of memory just above the 1MB address space called the high memory area (HMA). With this, the `DOS=HIGH` statement is used to shift portions of DOS from conventional memory into the HMA.

Know the purpose and function of the HIMEM.SYS file.

The EMM386.EXE program provides the system's microprocessor with access to the *upper memory area (UMA)* of RAM. Operating together with the HIMEM.SYS program, this program enables the system to conserve conventional memory by moving device drivers and memory-resident programs into the UMA.

## Files, Buffers, and Stacks

In the second section of the CONFIG.SYS file are the commands that define DOS for operation with optional devices and applications. The FILES command establishes the number of files that DOS can handle at any one time at 30. This number just happens to be the minimum required to load Windows for operation. The BUFFERS command sets aside 15 blocks of RAM space for storing data being transferred to and from disks. Similarly, the STACKS command establishes the number and length of some special RAM storage operations at nine memory stacks, each 256 bytes long.

## Device Drivers

Device drivers are loaded in the third part of the CONFIG.SYS file. These drivers are programs that tell DOS how to control specific devices. For example, the command DEVICEHIGH=C:\MOUSE\MOUSE.SYS loads a third-party device driver supporting the particular mouse being used with the system.

The SMARTDRV.SYS driver establishes a disk cache in an area of extended memory as a storage space for information read from the hard-disk drive. A *cache* is a special area of memory reserved to hold data and instructions recently accessed from another location. A disk cache holds information recently accessed from the hard-disk drive. Information stored in RAM is much quicker to access than if it were on the hard drive.

The RAMDRIVE.SYS driver simulates the organization of a hard-disk drive in RAM. This type of drive is called a *virtual disk*. In this case, the DEVICEHIGH= command loads the RAMDRIVE driver into the upper memory area rather than the base memory area, where a simple DEVICE= command would run it.

MS-DOS came with several other standard device driver programs. They included

- `KEYBOARD.SYS`
- `DISPLAY.SYS`
- `ANSI.SYS`
- `DRIVER.SYS`
- `PRINTER.SYS`

From the list, you can see that these basic drivers controlled the operation of the system's most basic input and output devices and ports. `KEYBOARD.SYS` was the default keyboard definition file. The `DISPLAY.SYS` driver supported code-page switching for the monitor type in use by the system. A code page represented the set of 256 characters that MS-DOS could handle at one time when displaying, printing, and manipulating text.

`ANSI.SYS` supported ANSI escape-code sequences used to modify the function of the system's display and keyboard. This file was also required to display colors on the monitor under DOS. `DRIVER.SYS` created the logical drive assignments for the system's floppy drives (that is, A: and B:). Finally, the `PRINTER.SYS` driver supported code-page switching for parallel ports. All these drivers were normally found in the `\DOS` directory.

**EXAM TIP**

Know which driver assigns logical drive letters to the system's floppy drives.

## AUTOEXEC.BAT

After completing the `CONFIG.SYS` operation and loading the `COMMAND.COM` command interpreter, the system would automatically search for the presence of the `AUTOEXEC.BAT` file. This file contains a batch of operating system commands that were automatically carried out when executed. Refer to the following sample `AUTOEXEC.BAT` file:

```
DATE
TIME
PROMPT=$P$G
SET TEMP=C:\TEMP
PATH=C:\;C:\DOS;C:\MOUSE
SMARTDRV.EXE 2048 1024
CD\
DIR
```

The first two commands cause the operating system to prompt you for the date and time (because DOS does not automatically do this when an `AUTOEXEC.BAT` file is present). The `PROMPT=$P$G` command causes the active drive and directory path to be displayed on the command line. The `SET TEMP` line sets up an area for holding data temporarily in a directory named `TEMP`.

The `PATH` command creates a specific set of paths that DOS will use to search for executable (`.EXE`, `.COM`, and `.BAT`) files. In this example, DOS will search for these files first in the root directory (`C:\`), followed by the `\DOS` directory (`C:\DOS`), and finally through the `Mouse` directory (`C:\MOUSE`). This statement effectively allows a `MOUSE.COM` or `MOUSE.EXE` driver program—normally located in the `Mouse` directory—to be executed from anywhere in the system. Upon receiving the `MOUSE` command, the operating system looks through all the directories in the path until it finds the specified filename.

The *syntax* (punctuation and organization) of the `PATH` command is very important. Each entry must be complete from the root directory and must be separated from the preceding entry by a semicolon. No spaces should appear in the `PATH` command.

**EXAM TIP**

Know which commands are normally located in a typical `AUTOEXEC.BAT` file.

## WINDOWS INITIALIZATION FILES

The original Windows 3.x operating environment was introduced between the MS-DOS and Windows 9x operating systems. Windows 3.x was a separate graphical environment that worked on top of the DOS operating system. It was built on a number of *initialization* (`.INI`) files that held the system's hardware and software configuration information. The major Windows 3.x initialization files include the following:

- `WIN.INI`—This file contains parameters that can be used to alter the Windows environment to suit the user's preferences. The text-like entries in this file govern the appearance of the windows, desktop, and colors displayed, as well as file associations and international aspects of the operating system's graphical display. `WIN.INI` also controls how Windows is installed.
- `SYSTEM.INI`—This file contains hardware setting information for the drivers and modules that Windows employs to configure itself.

- ▶ `CONTROL.INI`—This file holds settings for the applets in the Windows 3.x Control Panel interface.
- ▶ `PROGMAN.INI`—This file holds the configuration settings for the Windows 3.x Program Manager.
- ▶ `WINFILE.INI`—This file holds settings and defaults for the Windows 3.x File Manager.

Current versions of Windows 9x/Me and Windows 2000/XP continue to include the `WIN.INI` and `SYSTEM.INI` files for compatibility reasons. The `\Windows` directory may also contain several other `.INI` files. When a new Windows application is installed, it may install its own `.INI` file at that time. These files can be modified to customize, or optimize, the program's execution.

Parameters in `.INI` files are typically modified through normal Windows menus or through pop-up dialog boxes. Other parameters can be changed only by directly modifying the `.INI` files. Changes to the files are automatically updated whenever Windows is exited.

## WINDOWS 9X STARTUP

Windows 9x takes over the complete bootup function as a normal part of its operation. This seamless bootup may be convenient but can offer some interesting problems when the system does not boot; there's no stable command-line level to fall back to for troubleshooting purposes. Basically, the Windows 9x/Me boot sequence occurs in five phases:

- ▶ *Phase 1: Bootstrap with the BIOS*—During the bootstrap process, the BIOS is in control of the system. It performs the POST and Initialization processes and then collects information about the system's installed devices through the Plug-and-Play process. This information is passed to the operating system's Configuration Manager during the OS startup operation.
- ▶ *Phase 2: Loading DOS drivers and Terminate-and-Stay-Resident (TSR) files*—During this phase, the system is running in real mode and functions as described in the "MS-DOS Startup Process" section earlier in this chapter. `IO.SYS` checks the system's hardware profile to determine its actual configura-

### EXAM TIP

Be aware of how Windows versions before Windows 95 organized and monitored the system's configuration information.

tion. This profile is a function of the BIOS PnP detection process during the initialization phase. `IO.SYS` begins loading default drivers that were previously taken care of by the `CONFIG.SYS` file.

Windows 9x and Me possess system bootup files that replace the MS-DOS files described earlier in this chapter. The Windows 9x version of `IO.SYS` is a real-mode operating system that replaces the MS-DOS version. It also takes over many of the functions associated with the `CONFIG.SYS` file. An `MSDOS.SYS` file is created to retain compatibility with older applications; however, the Windows 9x/Me VMM32 and *virtual device driver (VxD)* files take over control of the system from the `IO.SYS` file during the startup process. Windows 9x supplies its own version of `COMMAND.COM` as well.

No `CONFIG.SYS` or `AUTOEXEC.BAT` files are created when Windows 9x or Windows Me is installed in a new system. These files are also not required by Windows 9x/Me to start up or to run. Even so, both files are retained from the previous operating system in upgraded systems to maintain compatibility with older applications; however, entries in the `CONFIG.SYS` file override the values in the Windows 9x/Me `IO.SYS` file.

The Windows 9x `IO.SYS` file also handles some of the `AUTOEXEC.BAT` commands. In both cases, the system uses `REM` statements to deactivate those `CONFIG.SYS` and `AUTOEXEC.BAT` functions implemented in the `IO.SYS` file. Similarly, the functions of the `SYSTEM.INI` and `WIN.INI` files have been moved to the Windows 9x Registry.

If no MS-DOS–based drivers or applications are in the system, the `CONFIG.SYS` and `AUTOEXEC.BAT` files are not necessary; however, the Windows 9x version of the `IO.SYS` file automatically loads the Windows 9x version of the `HIMEM.SYS` file during bootup. This file must be present for Windows 9x to boot up. Usually, multiple versions of the `HIMEM.SYS` file exist in a Windows 9x system (it could have three or more versions).

- *Phase 3: Real mode initialization of static virtual device drivers (VxDs)*—The contents of the Windows 9x Registry are located in two files under the `\windows` directory: `USER.DAT` and `SYSTEM.DAT`. The `USER.DAT` file contains user-specific information, whereas the `SYSTEM.DAT` file holds hardware- and computer-specific profiles and setting information.

In this phase of the startup process, the system checks the `SYSTEM.DAT` file for the first part of the Registry file and processes it. `SYSTEM.DAT` is a hidden file that contains all the system's hardware configuration information, including the PnP and application settings. This file is always located under the `\Windows` directory.

Afterward, `IO.SYS` loads the `WIN.COM` file, along with the `vmm32.vxd` virtual memory management driver and the contents of the `SYSTEM.INI` file. These files are used to control the loading and testing of the Windows 9x core components.

- *Phase 4: Protected-mode switch-over*—After loading all the static VxDs, the system shifts the microprocessor into protected-mode operation and begins loading the protected-mode components of the operating system. The Configuration Manager is initialized and receives the PnP information from the BIOS during this phase.
- *Phase 5: Loading remaining components*—Following the initialization process, the final Windows 9x components are loaded into the system. During this period, the following events occur:
  - The `KERNEL32.DLL` and `KERNEL386.EXE` files are executed.
  - The `GDI.EXE` and `GDI32.EXE` files are executed.
  - The `USER.EXE` and `USER32.EXE` files are executed.
  - All fonts and other associated resources are loaded.
  - The `WIN.INI` file values are checked.
  - The Windows 9x shell and machine policies are loaded.
  - The Windows desktop components are loaded.

The Windows 9x shell program is normally the Windows 9x desktop. Because Windows 9x/Me operating systems are designed so that they can be customized to different users, Windows 9x and Windows Me search the Registry's `HKEY_LOCAL_MACHINE` key and the user's home directory for user profile information that it can use to load personalized system settings.

Be aware of which file loads Windows 9x into the system.

## Windows 9x/Me Registry Data Files

When applications were removed from the system in earlier Windows versions, the configuration information distributed between the various `.INI` files remained, unless the user, or a special Windows *Uninstall* program, looked them up and removed them individually. In Windows 9x and Windows Me, the system's configuration information is held in a large hierarchical database called the *Registry*.

The Registry structure is primarily used to hold information about system hardware that has been identified by the enumeration or detection processes of the Plug-and-Play system. When a device is installed in the system, Windows 9x detects it, either directly or through the system's bus managers, and searches the Registry and installed media sources for an appropriate driver. When the driver is found, it is recorded in the Registry along with its selected settings. However, when a device or program is removed from a Windows system, its headings and the associated configuration information are all removed from the Registry in a single, clean operation, unlike the old `.INI` method of tracking this information.

**EXAM TIP**

Know what type of database Windows uses for its Registry.

The Registry also holds information that enables the system to serve and track multiple users. It does this by retaining user- and configuration-specific information that can be used to customize the system to different users or to different configuration situations. This includes the local hardware configuration, the network environment, file associations, and user configurations. In Windows 9x systems, the Registry information is stored in two files under the `\Windows` directory: `USER.DAT` and `SYSTEM.DAT`. The `USER.DAT` file contains user-specific information, and the `SYSTEM.DAT` file holds hardware and computer-specific profiles and settings information.

Windows 9x and Me create a folder for each user who logs on to the system. This profile is held in the `\Windows\Profiles` subdirectory. Each profile contains a `USER.DAT` file (the second half of the Registry) that holds the Registry information for that user. It also contains a number of other files that customize the desktop just for that user.

While the operating system pulls system and user configuration information from the Registry during startup, any information that remains in `WIN.INI`, `SYSTEM.INI`, and `WINFILE.INI` files that existed in earlier versions of Windows will still be processed as part of the configuration process. If these files exist, they remain in the `\Windows`

directory to maintain compatibility functions with older software. These files are retained for use with older 16-bit applications and are not necessary for the operation of Windows 9x applications. These files must be checked if the Windows 9x system has conflicts with older 16-bit applications.

## Automated Application Startups

Windows 9x provides a mechanism for automatically starting programs whenever the operating system starts by adding them to the system's Startup folder. You can accomplish this by accessing the Programs entry from the Start menu and then selecting the Add option. From here, you can simply browse until the desired program is found and then double-click on it. Finish the addition by clicking on the Next button and then double-clicking on the Startup folder. You can bypass these programs for troubleshooting purposes by pressing the Left-Shift key during startup.

## Configuration and Initialization File Conflicts

If a Windows 9x system has a `CONFIG.SYS`, `AUTOEXEC.BAT`, or `.INI` file that has been held over from a previous operating system, you should be aware that any unneeded commands in these files have the potential to reduce system performance. In particular, the `SMARTDRV` function from older operating systems inhibits dynamic `VCACHE` operation and slows down the system. The `VCACHE` driver establishes and controls a disk cache in an area of RAM as a storage space for information read from the hard-disk drive, CD-ROM, and other drives and file operations.

If the system runs slowly, check the `CONFIG.SYS` and `AUTOEXEC.BAT` files for `SMARTDRV` and any other disk cache software settings. Remove these commands from both files to improve performance. Also, remove any Share commands from the `AUTOEXEC.BAT` file. The `SYSTEM.INI`, `WIN.INI`, `PROTOCOL.INI`, `CONFIG.SYS`, and `AUTOEXEC.BAT` files can be modified through the *System Editor (SysEdit)* in Windows 9x. You can type the `SysEdit` command in the Start, Run dialog box to access this utility.

### EXAM TIP

Remember how to prevent the items in the Windows 9x Startup folder from running at startup.

Know what function the `VCACHE` utility performs in a Windows 9x system.

Be aware of the effect that active commands in a `CONFIG.SYS`, `AUTOEXEC.BAT`, or `.INI` file can have on the operation of an advanced Windows operating system.

## WINDOWS NT STARTUPS

The sequence of events in the Windows NT/2000/XP startup process is similar to that presented for MS-DOS and Windows 9x systems. The main differences are found in the terminology and filenames that Windows NT employs.

Like any other PC system, the Windows NT-based PC starts up by running the POST test, performing an initialization of its intelligent system devices, and performing a system boot process. It is in the boot process that the two operating systems diverge.

When the BIOS executes the master boot record on the hard drive, the MBR examines the disk's partition table to locate the active partition. The boot process then moves to the boot sector of that partition (referred to as the *partition boot sector*) located in the first sector of the active partition. Here, it finds the code to begin loading the secondary bootstrap loader from the root directory of the boot drive.

In the case of a Windows NT partition, the bootstrap loader is the NT loader file named `NTLDR`. This file is the Windows NT equivalent of the DOS `IO.SYS` file and is responsible for loading the NT operating system into memory. Afterwards, `NTLDR` passes control of the system over to the Windows NT operating system.

Next, a temporary miniature file system that can read both FAT and NTFS file structures is loaded to aid `NTLDR` in reading the rest of the system. Recall that Windows NT has the capability to work in either FAT or proprietary NTFS partitions. At this stage of the boot process, however, the operating system is still uncertain as to which system it will be using.

With the mini file system in place, the `NTLDR` can locate and read a special hidden boot loader menu file named `BOOT.INI`. `NTLDR` uses this text file to generate the Boot Loader Menu that is displayed on the screen. If no selection is made after a given period of time, the default value is selected.

If Windows NT is the designated operating system to be used, the `NTLDR` program executes a hardware detection file called `NTDETECT.COM`. This file is responsible for collecting information about the system's installed hardware devices and passing it to the `NTLDR` program. This information is later used to upgrade the Windows NT Registry files.

If a different operating system is to be loaded, as directed by the Boot Loader Menu entry, the `NTLDR` program loads a file called `BOOTSECT.DOS` from the root directory of the system partition and passes control to it. From this point, the `BOOTSECT` file is responsible for loading the desired operating system.

Finally, the `NTLDR` program examines the partition for a pair of files named `NTOSKRNL.EXE` and `HAL.DLL`. `NTOSKRNL.EXE` is the Windows NT kernel file that contains the Windows NT core and loads its device drivers. `HAL.DLL` is the Hardware Abstraction Layer driver that holds the information specific to the CPU that the system is being used with.

Even though `NTLDR` reads the `NTOSKRNL` and `HAL` files at this time, it does not load or execute them. Instead, it uses a file named `NTDETECT.COM` to gather information about the hardware devices present and pass it to the `NTLDR`.

After the information is passed back to `NTLDR`, it opens the System hive portion of the Registry to find the Current Control Set. At this point, the system displays the Starting Windows NT logo on the screen along with a sliding progress bar that shows the degree of progress being made in loading the drivers. After the drivers are loaded, the `NTLDR` program passes control to the `NTOSKRNL` file to complete the boot sequence.

When `NTOSKRNL` gains control of the system, it initializes the `HAL.DLL` file (along with the `BOOTVID.DLL` file in Windows 2000) and shifts the video display to graphics mode. It then initializes the drivers prepared by `NTLDR` and uses the `NTDETECT` information to create a temporary Hardware hive in memory. Finally, `NTOSKRNL` executes a Session Manager file titled `SMSS.EXE` to carry out pre-start functions such as running a boot-time version of `CHKDSK` called `AUTOCHK`. It also establishes parameters concerning the Windows NT paging file (`PAGEFILE.SYS`) to hold RAM swap pages.

## Windows NT/2000/XP Registry Data Files

Microsoft's use of Registries actually began with earlier versions of Windows NT. Windows 2000 and XP rely on the same Registry structure used in previous Windows NT versions. For this reason,

### NOTE

This process should not be confused with the PnP enumeration process that occurs later in the Windows 2000 boot process. The information gathered by `NTDETECT` is stored to be used later for updating the Hardware Registry hive.

If the Windows NT system employs a SCSI disk drive, a driver file named `NTBOOTDD.SYS` needs to be present in the root directory of the system partition. This condition must also be noted in the `BOOT.INI` file by placing a mark in its `SCSI(x)` or `Multi(x)` locations. The `NTLDR` program can also load driver files that have been renamed as `NTBOOTDD.SYS` to enable Windows NT 4.0 and Windows 2000 to use drives greater than 8GB in size (even EIDE drives).

### EXAM TIP

Memorize the names and functions of the files involved in the Windows NT/2000 boot process and know which order they are executed in.

Be aware of what the `NTBOOTDD.SYS` file does in a Windows NT/2000 system.

their Registries are not compatible with Windows 9x/Me Registries described earlier. However, their functions are the same: During the startup process, the Windows NT/2000/XP Registries are accessed to establish the correct configuration settings for the system's installed hardware, applications, and logged-on user.

As with Windows NT 4.0, the user settings portion of the Registry is stored in the `NTUSER.DAT` file located in the

`\Documents_and_Settings\username` folder. The System portions of the Registry are stored in special groupings called hives. The Windows NT groups include the SOFTWARE, SYSTEM, SECURITY, and SAM hives. These files are stored in the `\Winnt\System32\Config` folder along with a backup copy and log file for each hive.

Configuration information about every user who has logged in to the system is maintained in a named subfolder under the

`\Winnt\Profiles` directory. The actual user configuration file is named `NTUSER.DAT` (for example, `\Winnt\Profiles\Charles\Ntuser.dat`). In Windows 2000 and XP, the `NTUSER.DAT` file is stored in `\Documents_and_Settings\username`.

**EXAM TIP**

Know how to make a backup copy of the Registry in Windows NT 4.0.

As mentioned in Chapter 1, it is a good practice to back up the contents of the Registry before installing new hardware or software, or modifying the Registry directly. In Windows 4.0, the `RDISK.EXE` utility, located in the `\Winnt\System32` folder, can be used to create a backup copy of the Windows NT 4.0 Registry in the `\Winnt\Repair` folder. In Windows 2000 and XP, the Registry is backed up as part of the System State data using the Backup/Restore features.

---

**CHALLENGE #1**

You have been sent to a customer's site to upgrade one of his Windows NT 4.0 workstations to Windows 2000 Professional. Because the equipment is getting a little old, you want to protect the system in case you need to reinstall it. You are not too familiar with Windows NT, but you know there is a utility for creating a backup of the system's Registry. What is this utility named, and where can you access it?

Refer to the "Challenge Solutions" section at the end of this chapter for the resolution to the challenge.

---

## CHAPTER SUMMARY

This chapter presented the detailed steps of the basic bootup sequence associated with the MS-DOS operating system as well as various Windows versions. The MS-DOS information is included to provide a simplified example of common operating system functions and to introduce elements that still reside in PCs to provide compatibility with older systems.

The chapter also introduced the Windows methodology for storing information about the system's hardware, software, and users. This is accomplished through a special database structure called the Registry. The Registry information in this chapter focused on the files that make up the Registry in the various operating system versions. This information built on the introductory material about the various Windows Registries described in Chapter 1.

At this point, review the objectives listed at the beginning of the chapter to be certain that you understand each point and can perform each task listed there. Afterward, answer the review questions that follow to verify your knowledge of the information.

### KEY TERMS

- IO.SYS
- MSDOS.SYS
- AUTOEXEC.BAT
- COMMAND.COM
- CONFIG.SYS
- HIMEM.SYS
- EMM386.EXE
- WIN.COM
- SYSTEM.INI
- WIN.INI
- SYSTEM.DAT
- USER.DAT
- BOOT.INI
- NTLDR
- NTDETECT.COM
- NTBOOTDD.SYS
- NTUSER.DAT

## APPLY YOUR KNOWLEDGE

### Review Questions

- What is the minimum file size specification associated with the `MSDOS.SYS` file?
  - 16 bits
  - 1KB
  - 2KB
  - 4KB
- What is the order of execution for the files involved in the MS-DOS boot process?
  - `IO.SYS`, `MSDOS.SYS`, `COMMAND.COM`, `CONFIG.SYS`, `AUTOEXEC.BAT`
  - `IO.SYS`, `MSDOS.SYS`, `CONFIG.SYS`, `AUTOEXEC.BAT`, `COMMAND.COM`
  - `IO.SYS`, `MSDOS.SYS`, `CONFIG.SYS`, `COMMAND.COM`, `AUTOEXEC.BAT`
  - `MSDOS.SYS`, `IO.SYS`, `CONFIG.SYS`, `COMMAND.COM`, `AUTOEXEC.BAT`
- What is the swap file in Windows NT/2000 called?
  - `SWAP.SYS`
  - `PAGEFILE.SYS`
  - `WIN386.SWP`
  - `WINSWAP.SWP`
- What virtual memory swap file is used in Windows 9x?
  - `TEMP.SWP`
  - `WIN.SWP`
  - `WIN386.SWP`
  - `PAGEFILE.SYS`
- Which of these statements are normally located in the `CONFIG.SYS` file? (Select all that apply.)
  - `PATH=C:\,C:\DOS,C:\MOUSE,C:\PROGRAMS`
  - `DEVICE=C:\DOS\EMM386.EXE`
  - `DOS=HIGH,UMB`
  - `FILES=30`
- What is the purpose of the `HIMEM.SYS` file?
  - expanded memory management
  - extended memory management
  - OS configuration management
  - creating the UMA
- Which driver assigns logical drive letters to the system's floppy drives?
  - `DRIVER.SYS`
  - `DISPLAY.SYS`
  - `ANSI.SYS`
  - `KEYBOARD.SYS`
- Which of these commands are normally located in a typical `AUTOEXEC.BAT` file? (Select all that apply.)
  - `PROMPT=$P$G`
  - `SET TEMP=C:\TEMP`
  - `DOS=HIGH,UMB`
  - `SMARTDRV`
- How did the Windows versions before Windows 95 organize and monitor the system's configuration information?
  - `.LOG` files
  - `.CFG` files

**APPLY YOUR KNOWLEDGE**

- C. .DAT files  
D. .INI files
10. Which file loads the Windows 9x operating system into the system?  
A. START.EXE  
B. IO.SYS  
C. WIN.COM  
D. AUTOEXEC.BAT
11. Where are the files that make up the Windows 9x Registry located in the system?  
A. C:\  
B. C:\Windows  
C. C:\Windows\Registry  
D. C:\Windows\Sysbckup
12. Which files make up the Windows 9x Registry? (Select two answers.)  
A. REG.DAT  
B. SYSTEM.DAT  
C. HIVE.DAT  
D. USER.DAT
13. How are the items in the Windows 9x Startup folder prevented from running at startup?  
A. Press Left-Ctrl during startup.  
B. Press Left-Alt during startup.  
C. Press Left-Shift during startup.  
D. Press Tab during startup.
14. What function does the VCACHE utility perform in a Windows 9x system?  
A. It creates and manages the HMA.  
B. It creates and manages a disk cache in RAM.  
C. It creates and manages a RAM cache.  
D. It creates and manages the UMB.
15. What effect can active commands in a CONFIG.SYS, an AUTOEXEC.BAT, or an .INI file have on the operation of an advanced Windows operating system?  
A. It runs faster.  
B. It runs better with older devices.  
C. It runs more slowly.  
D. It runs better with older programs.
16. What are the names of the files involved in the Windows NT boot process, and what order are they executed in?  
A. BOOT.INI, NTLDR, NTOSKRNL.EXE, NTDETECT.COM  
B. NTLDR, BOOT.INI, NTOSKRNL.EXE, NTDETECT.COM  
C. BOOT.INI, NTLDR, NTDETECT.COM, NTOSKRNL.EXE  
D. NTLDR, BOOT.INI, NTDETECT.COM, NTOSKRNL.EXE, HAL.DLL
17. In a Windows NT/2000 system, the NTBOOTDD.SYS file is used to \_\_\_\_\_.  
A. boot the system  
B. enable SCSI disk drives  
C. detect bootup errors  
D. enable double-density booting
18. What utility is used to create a backup copy of the Registry in Windows NT 4.0?

## APPLY YOUR KNOWLEDGE

- A. RDISK
- B. BACKUP
- C. SYSBACK
- D. REGBACK

## Answers and Explanations

1. **B.** There is a little-known MS-DOS system requirement that the `MS-DOS.SYS` file must maintain a size in excess of 1KB.
2. **C.** The files required to boot an MS-DOS system and their execution order are `IO.SYS`, `MSDOS.SYS`, `CONFIG.SYS`, `COMMAND.COM`, `AUTOEXEC.BAT`.
3. **B.** The Windows NT paging file (`PAGEFILE.SYS`) is used to hold RAM swap pages.
4. **C.** The size of the Windows 9x swap file (`WIN386.SWP`) is variable and is dynamically assigned.
5. **B, C, D.** A sample `CONFIG.SYS` file might consist of command lines such as
 

```
Device=C:\DOS\HIMEM.SYS
Device=C:\DOS\EMM386.EXE 1024 RAM
FILES=30
BUFFERS=15
STACKS=9,256
DEVICE=C:\DOS\SMARTDRV.SYS 1024
DOS=HIGH,UMB
DEVICEHIGH=C:\MOUSE\MOUSE.SYS
DEVICEHIGH=C:\DOS\RAMDRIVE.SYS 4096/a
```
6. **B.** `HIMEM.SYS` loads the DOS Extended Memory Manager (XMS). This driver manages the use of extended memory installed in the system so that no two applications use the same memory locations at the same time. This memory manager should normally be listed in the `CONFIG.SYS` file before any other memory managers or device drivers. `HIMEM.SYS` also creates a 64KB area of memory just above the 1MB address space called the high memory area (HMA). With this, the `DOS=HIGH` statement is used to shift portions of DOS from conventional memory into the HMA.
7. **A.** `DRIVER.SYS` creates the logical drive assignments for the system's floppy drives (that is, A: and B:).
8. **A, B, D.** A sample `AUTOEXEC.BAT` file might consist of command lines such as
 

```
DATE
TIME
PROMPT=$P$G
SET TEMP=C:\TEMP
PATH=C:\;C:\DOS;C:\MOUSE
SMARTDRV.EXE 2048 1024
CD\
DIR
```
9. **D.** Windows 3.x was built on a number of initialization (`.INI`) files that held the system's hardware and software configuration information.
10. **C.** `IO.SYS` loads the `WIN.COM` file into RAM, which controls the loading and testing of the Windows 9x core components.
11. **B.** The contents of the Windows 9x Registry are located in two files under the `\Windows` directory.
12. **B, D.** The system checks the `SYSTEM.DAT` file for the first part of the Registry file and processes it. `SYSTEM.DAT` is a hidden file that contains all the system's hardware configuration information, including the PnP and application settings. It is always located under the `\Windows` directory. Windows 9x searches the `Hkey_Local_Machine` key and the user's home directory for user profile information. Windows 9x creates a folder for each user who logs on to the system. This profile is held in the `\Windows\Profiles` subdirectory. Each profile contains a `USER.DAT` file (the second half of the Registry) that holds the Registry information for that user. It also contains a number of other files that customize the desktop just for that user.

## APPLY YOUR KNOWLEDGE

- As with the `SYSTEM.DAT` file, the `USER.DAT` file is backed up as `USER.DA0` each time the Windows 95 system is rebooted. Under Windows 98, the `USER.DAT` backup is part of the `RBOXX.CAB` files.
13. **C.** You can bypass the programs in the Startup folder for troubleshooting purposes by pressing the Left-Shift key during startup.
  14. **B.** The `VCACHE` driver establishes and controls a disk cache in an area of RAM as a storage space for information read from the hard-disk drive, CD-ROM, and other drives and file operations.
  15. **C.** If the system runs slowly, check the `CONFIG.SYS` and `AUTOEXEC.BAT` files for `SMARTDRV` and any other disk cache software settings. Remove these commands from both files to improve performance. Also, remove any Share commands from the `AUTOEXEC.BAT` file. The `SYSTEM.INI`, `WIN.INI`, `PROTOCOL.INI`, `CONFIG.SYS`, and `AUTOEXEC.BAT` files can be modified through the System Editor (SysEdit) in Windows 9x.
  16. **D.** The Windows NT/2000/XP boot sequence includes `NTLDR`, `BOOT.INI`, `NTDETECT.COM`, `NTOSKRNL.EXE`, and `HAL.DLL`.
  17. **B.** If the Windows NT system employs a SCSI disk drive, a driver file named `NTBOOTDD.SYS` must be present in the root directory of the system partition. This condition must also be noted in the `BOOT.INI` file by placing a mark in its `SCSI(x)` or `Multi(x)` locations. The `NTLDR` program can also load driver files that have been renamed as `NTBOOTDD.SYS` to enable Windows NT 4.0 and Windows 2000 to use drives greater than 8GB in size (even EIDE drives).
  18. **A.** The `RDISK.EXE` utility, located in the `\Winnt\System32` folder, can be used to create a backup copy of the Windows NT Registry in the `\Winnt\Repair` folder.

## Challenge Solutions

1. It is the `RDISK.EXE` utility, located in the `\Winnt\System32` folder, and it can be used to create a backup copy of the Windows NT Registry in the `\Winnt\Repair` folder.

## Suggested Readings and Resources

### 1. DOS Structure

<http://www.patersonstech.com/Dos/Byte/InsideDos.htm>

### 2. `CONFIG.SYS` and `AUTOEXEC.BAT`

[http://www.butterwick0.freemove.co.uk/tutor/part\\_5.html](http://www.butterwick0.freemove.co.uk/tutor/part_5.html)

### 3. `.INI` Files

[http://www.webopedia.com/TERM/\\_/\\_INI\\_file.html](http://www.webopedia.com/TERM/_/_INI_file.html)

### 4. DOS Memory Management

<http://support.microsoft.com/default.aspx?scid=kb;EN-US;q95555>

### 5. Virtual Memory

<http://www.howstuffworks.com/virtual-memory1.html>

### 6. `DLL`

<http://webopedia.com/TERM/D/DLL.html>

