Boot and Logon: Section 1 Transcript

Introduction 1/3

Welcome to the Boot and Logon module. We'll go over Windows' process of booting up the system, boot configuration, and logging onto the system.

Throughout this module, you'll be presented with opportunities to assess and apply what you've learned.

At the end of this module, you will be able to:

- Describe the process start sequence,
- Differentiate the boot process across Windows variants (XP, 2003, Vista, 2008, and Windows 7),
- Differentiate the logon process across Windows variants (XP, 2003, Vista, 2008, and Windows 7),
- Describe the secure logon process, and
- Utilize various tools available to examine or modify the boot configuration.

Introduction 2/3

When you press the power button on your computer, you kick off a series of processes and routines to start up or "boot" the operating system on your computer. The term "boot" comes from the word "bootstraps," which people at one time used to get their boots on. Once the computer's power is turned on, the boot process takes place.

In this module, you will learn about what goes on under the hood when booting up the Windows operating system.

Bypass Exam Introduction 3/3

If you are already familiar with the subject matter presented in this module, you can choose to take a Bypass Exam to skip this module.

The Bypass Exam option provides a single opportunity to successfully demonstrate your competence with the material presented within the module. If you pass, you'll receive credit for completing the module, unlocking the content within, and you will be free to proceed to the next module. If you do not pass, you will need to successfully complete the module, including all exercises and the Module Exam, to receive credit.

Click the Next Section button to continue.

Boot and Logon: Section 2 Transcript

Hard Disk Layout

1/4

Partitioning a disk is the process of dividing a hard drive's storage space into smaller segments. There are two types of bootable partitions - Master Boot Record (MBR) Disks, and GUID Partition Table (GPT) Disks.

The two types of firmware used by modern computers are Basic Input/Output System (BIOS) and Unified Extensible Firmware Interface (UEFI). BIOS requires MBR Partition structure while UEFI requires GPT partition structure. UEFI also requires that the version of Windows must match the PC architecture, unless there is legacy BIOS support. For example, UEFI only supports booting a 32-bit version of Windows if the PC architecture is also 32-bit. Otherwise, 32-bit legacy BIOS mode is required to boot a 32-bit version of Windows on 64-bit PC architecture.

By default, all UEFI-enabled 64-bit systems use GPT partitioning. These 64-bit versions of Windows read, write, and also boot from GPT disks. They can read and write MBR disks, but cannot boot from MBR disks. Both 32-bit and 64-bit Windows operating systems support GPT disks for data on BIOS enabled systems.

Master Boot Record

2/4

The Master Boot Record (MBR) contains the Partition Table for the disk and executable code called the master boot code. The MBR is stored at a consistent location on a physical disk, allowing the BIOS to always know where to find it. It is stored on the first sector of the hard disk, with four partitions for each disk, and has a maximum size of 2.2 TB. The MBR supports 3 partition types: primary, extended, and logical drives.

During the startup process, the computer examines the MBR to determine which partition on the installed disks is marked as active. The active partition contains the operating system startup files.

Note: A disk cannot be both a GUID Partition Table disk and an MBR disk. However, all GUID Partition Table disks contain a protective MBR that is used for legacy programs that do not understand the GUID Partition Table disk structure.

GUID Partition Table

3/4

The GUID partition table (GPT) contains an array of partition entries describing the start and end Logical Block Address (LBA) of each partition on disk.

LBA 0 contains a legacy protective MBR before the GUID Partition Table.

The protective MBR contains one primary partition. Legacy software that does not know about GPT reads only the protected MBR when it accesses a GPT disk. The protective MBR also protects the GPT disk from legacy tools that would not recognize the disk.

LBA 1 contains the Partition Table Header.

Section Completed

4/4

Boot and Logon: Section 3 Transcript

Generic Pre-Boot Process 1/4

There's a chain of handoffs beginning at the time you power up your PC. Depending on which Windows operating system you are using, the boot process may be slightly different. The generic boot process begins with the BIOS and continues through to the MBR, boot sector, and finally, the bootloader. Let's take a look at the process for both Windows XP and 2003 which is different than Windows Vista, 7, and 2008.

Components of the Boot Process - Windows XP and 2003 2/4

The pre-boot and boot process varies across Windows variants. In the pre-boot, the following steps take place:

- The power button is pressed,
- BIOS begins execution,
- BIOS reads in MBR (Master Boot Record and Partition table),
- Master Boot Code reads in PBS (Partition Boot Sector), and
- Partition Boot Sector Code reads in NT loader (NTLDR).

Click the steps in the boot process to learn more.

BIOS: (Basic Input/Output System) - Firmware that is stored in the computer's read-only memory. Its purpose is to initialize and test the hardware of the system, and locate and read into memory the Master Boot Record (MBR) code.

MBR: (Master Boot Record) - Located at the first sector of the hard disk, contains boot code and a partition table.

PBS: (Partition Boot Sector) - Reads in the OS Loader program (NTLDR).

Ntldr: (Windows OS Loader) - The Windows OS Loader loads the boot.ini file and presents the boot select menu if necessary. In the interim NTLDR loads, but does not initialize, the kernel image, HAL images, and System registry hives into memory. (NTLDR initializes the kernel later in the boot process.)

Boot.ini - Contains information defining the paths from which files are loaded for all installed operating systems.

Ntdetect.com - Performs hardware detection. The information collected is stored in the HARDWARE hive later in the boot process.

Ntoskrnl.exe - After being initialized by NTLDR, Ntoskrnl.exe starts the process of initializing Hal.dll. This gives Hal.dll temporary control before further system initialization.

Hal.dll: (Hardware Abstraction Layer) - The HAL exists between the kernel and the hardware. It performs low-level function call translation to a form the actual hardware can interpret. It also hides any hardware dependencies from the kernel.

Smss.exe: (Session Manager) - The Session Manager process is responsible for finalizing the initialization of the operating system and prepares the computer for user logon. Launches the Windows subsystem and Winlogon processes.

Winlogon.exe: (Interactive Logon Manager) - Coordinates all user logon tasks, to include authentication, user token creation, and creating the user's session.

LSASS: (Local Security Authority Subsystem) - Performs identification and creates the user's security access token.

Services : (Service Control Manager) - Responsible for loading auto-start devices drivers and services.

Logon for Windows XP and 2003

The logon process begins when Control+Alt+Delete, the Secure Attention Sequence (SAS), is entered. Winlogon then calls the Graphical Identification and Authentication (GINA) component that obtains a username and password, creates a unique local logon SID for the user, and passes the SID to the Local Security Authority Subsystem (LSASS) which authenticates the user, verifies the user's accesses, and creates the user's access token.

User drivers and services set to autorun are started. Userinit loads the users profile and executes a shell (Explorer.exe by default). LastKnownGood is updated. Userinit exits.

Exercise Introduction

It is time for an Exercise. In order to successfully complete the Exercise, you are expected to use your notes and any available resources presented throughout the course, in addition to conducting your own Internet research.

Boot and Logon: Section 4 Transcript

Boot Configuration Data

1/8

Prior to Vista, boot order and boot options were modified by editing the boot.ini text file at the root of the system drive. To change the boot-loader behavior of pre-Vista systems, nothing more complex than Notepad and the knowledge of how to un-set boot.ini's "read only" component was needed.

Today, boot configuration information is stored in an extensible database called the Boot Configuration Database (BCD). Boot Manager (Bootmgr) loads the BCD, not boot.ini, which in turn loads an OS loader boot application, Winload.exe, which differs for each operating system. Finally, the OS loader application is responsible for initializing the kernel.

Why is this significant?

The boot process can be extended to support other applications (other operating systems or other versions on the OS). Only non-version specific components are stored in the root of the active partition. Theoretically, Windows Vista could be installed on a future version of Windows - providing it has the same boot structure (the BCD), thereby not breaking the boot process for that future version.

With legacy Windows, installing an older version causes the newer version to fail on start-up. This is due to version-specific code improvements in NTLDR. Tool-oriented boot entries can be supported to offer maintenance options on boot when a previous start-up failed.

Also, Windows supported a menu provided by a target operating system. This new structure makes tools available earlier in the boot process and means they are available even if all OS-specific boot entries are damaged. The Windows Memory Diagnostic tool is provided in the boot menu this way.

BCD Configuration Data (BCD)

2/8

The Boot Configuration Data (BCD) store contains boot configuration parameters and controls how the operating system is started in Windows Vista, 7, 2008, 2008 R2, 8, 2012, and 2012 R2 operating systems (Vista and newer systems).

These parameters were previously in the boot.ini file in BIOS-based operating systems or in the nonvolatile RAM (NVRAM) entries in Extensible Firmware Interface-based operating systems.

BCD

3/8

Now let's look at how the BCD is structured. Click each section to reveal more information.

BCD Store Location

- The BCD store is stored as %SYSTEMDRIVE%\boot\BCD on the active partition on BIOS

systems, for example c:\boot\bcd
- The BCD store is located on the EFI System Partition on UEFI systems
- This hive is loaded but hidden from view in Regedit.exe within Windows
- The store is a registry hive loaded to the following location: HKLM\BCD00000000

BCD Object: BCD object is a container for all BCD elements. Each BCD object has a GUID, which represents the object in the BCD hive by naming a subkey of the Objects key. There are 18 predefined BCD objects with a fixed GUID. Every BCD object, whether pre-defined or not, has a type, actually stored as the DWORD data for the Type value in the object's Description subkey. Broadly speaking, there are application objects, inherit objects, and device objects.

The most common type of BCD object describes a boot environment application, such as an instance of the Windows boot loader.

BCD Element: BCD element is a singular item of data like, a boot application name, or an operating system device. Each BCD element is its own subkey. The name of the subkey is a representation in hexadecimal digits of a DWORD, referred to as a datatype.

BCD Configuration Tools

4/8

The BCD Store can be modified or repaired using several tools.

Most of these tools enable you to perform simple tasks, such as:

- Setting a boot timeout,
- Setting a default operating system, and
- Configuring boot options such as /SOS and /SAFEBOOT.

Others are repair-oriented tools that provide automated mechanisms for correcting problems in the BCD, such as:

- Startup,
- Repair, and
- Bootrec.exe.

Click the tools to reveal more information.

Msconfig

- Msconfig is the preferred UI tool for managing boot settings.
- The tool supports BCD and allows the user to enumerate all BCD objects in the system store.
- It allows certain elements to be altered for each OS object, including debug settings, safe mode settings, and other popular startup options.

Bcdedit.exe

- Bcdedit.exe is a command line tool that can be used to manage BCD settings.
- Bcdedit.exe is a replacement for Bootcfg.exe.

- You can use Bcdedit.exe to modify the Windows code which runs in the pre-operating system environment by adding, deleting, editing, and appending entries in the BCD store.
- Bcdedit.exe is designed to work on previous operating systems and in recovery environments.
- Bcdedit.exe is located in the %SYSTEMROOT%\system32 directory of the boot partition.

You can view the BCD store manually in WinRE by loading the hive from the Boot folder.

- To obtain a copy of the hive on a running system, use the bcdedit /export filename command to export a hive file for offline examination.
- This command exports the BCD store to a file called filename in the current directory.

Bootrec.exe

Bootrec.exe is a tool that can be used in the Windows Recovery Environment (WinRE) to troubleshoot and repair the following items in Windows Vista or Windows 7:

- A Master Boot Record
- A Boot Sector
- A Boot Configuration Data (BCD) Store

Windows Vista, 2008, and Windows 7 Boot Process 5/8

The Windows Vista boot process differs from Windows XP in the way the operating system is located and initialized. Start-up begins with the BIOS loading the MBR on the bootable disk. The MBR in turn loads the Partition Boot Record (PBR) on the active partition. So far, Windows Vista remains relatively unchanged.

Vista, 2008, and Windows 7 are different in that the new PBR code looks for and runs Bootmgr, the new boot loader and not NTLDR.

Components of the Boot Process for Windows Vista, 2008, and Windows 7 6/8

In the pre-boot, the following steps take place:

- UEFI begins execution,
- Hardware detection is completed, and
- UEFI boot manager loads drivers and boot applications.

Now, click each item in the flow chart to learn about each step in the boot process.

UEFI Firmware: Contains boot loader code that is stored in the system's NVRAM. Performs CPU and chipset initialization, loads drivers, reads the contents of the Boot Configuration Database, and is also stored in NVRAM. Loads Bootmgfw.efi.

Hardware Detection: During the boot process, the boot loader code used EFI interfaces to detect:

- Network adapters
- Video adapters

- Keyboards
- Disk controllers
- Storage devices

UEFI Boot Manager: Loads UEFI device drivers. Presents the Boot Selection Menu and its timeout. Loads boot application (bootmgfw.efi).

Winresume.efi: Contains information defining the paths from which files are loaded for all installed operating systems.

BCD (Boot Configuration Database: Performs hardware detection. The information collected is stored in the HARDWARE hive later in the boot process.

Bootmgfw.efi: Loads Windows OS loader (Winload.efi) selected by user.

Winload.efi: Loads the Windows OS: Loads all files need by the kernel to initialize, and sets up the kernel's execution environment.

Ntoskrnl.exe: This is the kernel image. Performs executive subsystem initialization. Loads and initializes BOOT_ and SYSTEM_START device drivers. Launches Smss.exe.

Hal.dll (Hardware Abstraction Layer): The HAL exists between the kernel and the hardware. It performs low-level function call translation to a form the actual hardware can interpret. It also hides any hardware dependencies from the kernel.

Smss.exe (Session Manager): The Session Manager process is responsible for finalizing the initialization of the operating system and prepares the computer for user logon. Launches the Windows subsystem and Winlogon processes.

Win32k.sys: Kernel-mode portion of the Windows subsystem.

Wininit.exe: Windows initialization process used only in Session 0.

LSASS (Local Security Authority Subsystem): Performs Identification and creates the user's security access token.

Winlogon.exe (Interactive Logon Manager: Known as the Interactive Logon Manager, coordinates all user logon tasks to include authentication, user token creation, and creating the user's session.

LogonUI: Presents the Interactive logon Dialog box to collect user credentials.

Services.exe (Service Control Manager: Responsible for loading auto-start device drivers and services.

Logon for Vista, 2008, and Windows 7

The logon process begins when Control+Alt+Delete, the Secure Attention Sequence (SAS) is entered. The user enters logon credentials (username and password). The process then moves to the Local Security Authority Subsystem (LSASS) which authenticates the user, verifies the user's accesses, and creates the user's access token.

User drivers and services set to autorun are started. Userinit loads the users profile and executes a shell (Explorer.exe by default).

Last Known Good is updated. Userinit exits. The boot process is complete.

Exercise Introduction

It is time for an Exercise. In order to successfully complete the Exercise, you are expected to use your notes and any available resources presented throughout the course, in addition to conducting your own Internet research.

Boot and Logon: Section 5 Transcript

Secure Logon Process

1/6

Before anyone logs on, the visible desktop belongs to Winlogon. Winlogon/Logonui registers CTRL+ALT+DEL, the Secure Attention Sequence (SAS), as a standard hotkey sequence.

SAS takes you to the Winlogon/Logonui desktop (the Interactive desktop is locked).

The secure logon cannot be deregistered by anyone because only the thread that registers the hotkey sequence can deregister it. When Windows keyboard input processing code sees SAS it disables keyboard hooks so that no one can intercept it.

Platform and UEFI Secure Boot

2/6

Platform and UEFI Secure Boot helps defend the system against known vulnerabilities that are associated with the boot up process. Systems using the legacy BIOS boot process would assign the responsibility of loading the operating system to an unverified boot loader. This boot loader is undetected by the operating system and any anti-malware software that may be installed. Because of this, malicious software could infect the boot loader to either launch a virus immediately, or redirect the boot path in order to load a piece of malicious code before the operating system is loaded.

Platform and UEFI Boot prevents these issues by adding additional security within the boot process.

Isolated User Modes

3/6

Prior to Windows 10, the operating system would communicate directly with the LSASS, which resided in the process memory, creating a security risk.

To address those concerns, Windows 10 introduced the Isolated User Mode (IUM), also known as Virtual Secure Mode (VSM), a virtualization-based technology that uses secure kernels to separate the operating system, applications, and users' data from more sensitive information, such as user credentials and hardware device drivers. Security-related features, such as Credential Guard and Device Guard, rely on IUM to protect their data.

Windows Trusted Boot

4/6

During startup, the UEFI Secure Boot verifies that the bootloader is trusted and proceeds to start Windows.

To further secure the boot process, the Windows Trusted Boot feature protects the rest of the startup process by verifying all Windows startup components are trustworthy. If a monitored file has been tampered with, Windows Trusted Boot will halt the boot sequence. If required, the Windows Recovery Environment will recover the file in question.

Windows Hello

Microsoft passport, now known as Windows Hello, provides two-factor authentication on PCs and allows users to sign in to Windows 10 devices with just a look or a touch, providing enterprise-grade security without entering a password.

The user's credential is tied to a specific device and biometric and/or pin. Authentication can be linked to a Microsoft online account, Active Directory, Azure or any provider who supports Fast ID Online (FIDO) 2.0 authentication.

Windows Hello credentials are based on a certificate or an asymmetrical key pair that includes a private key and a public key. The credentials, as well as the token that is obtained from using the credentials, are bound to the device via Trusted Platform Module (TPM).

The private key never leaves a device. The authenticating server has a public key which is mapped to the user account during the registration process.

Exercise Introduction

It is time for an Exercise. In order to successfully complete the Exercise, you are expected to use your notes and any available resources presented throughout the course, in addition to conducting your own Internet research.

Boot and Logon: Section 6 Transcript

Summary

You have completed the Boot and Logon module.

This module included Window's process of booting up the system, boot configuration, and logging onto the system.

You should now be able to:

- Describe the process start sequence,
- Differentiate the boot process across Windows variants (XP, 2003, Vista, 2008, and Windows 7),
- Differentiate the logon process across Windows variants (XP, 2003, Vista, 2008, and Windows 7),
- Describe the secure logon process, and
- Utilize various tools available to examine or modify the boot configuration.

To receive credit and advance to the next module, you must achieve a passing score on the Module Exam. Click the Next Section button to begin the Module Exam.