

# Windows booting process

April 4, 2013 by Dejan Lukan

Share:

In the previous tutorial, we've seen how one would go about booting the [Linux operating system](#) by using GRUB. We presented the basic overview of the booting process in Linux and we also mentioned that the boot loader must support the file system the root operating system is installed on.

## Learn Reverse Engineering

Build your skills around identifying malware types, characteristics and behaviors with six hands-on courses.

[START LEARNING](#)

### Booting process of Windows XP

Let's take a look at the MBR where the Windows boot loader resides. We can do this by booting off some Linux live CD on the system where Windows is installed. We should install the system to get to the live Linux environment, where we can input commands. Since the MBR was populated by Windows, we can issue the **dd** command to dump the whole MBR from the hard drive.

Below, we booted the Ubuntu live CD and copied the MBR into the `/tmp/mbr.bin` file with the **dd** command, then printed its contents with the **file** command:

```
root@ubuntu:~# dd if=/dev/sda of=/tmp/mbr.bin bs=512 count=1
1+0 records in
1+0 records out
512 bytes (512 B) copied, 0.000349974 s, 1.5 MB/s
root@ubuntu:~# file /tmp/mbr.bin
/tmp/mbr.bin: x86 boot sector, Microsoft Windows XP MBR, Serial 0xc234c234; partition 1: ID=0x7, active, starthead 1, startsector 63, 65641527 sectors, code offset 0xc0
root@ubuntu:~#
```

We can see that the boot loader is a Microsoft Windows XP MBR and there's only one active bootable partition on the whole hard drive.

The first boot loader must be able to read the NTFS file system in order to mount the right partition and load the appropriate program that will boot the rest of the operating system. There are two mechanisms in place that controls how the Windows system can be booted:

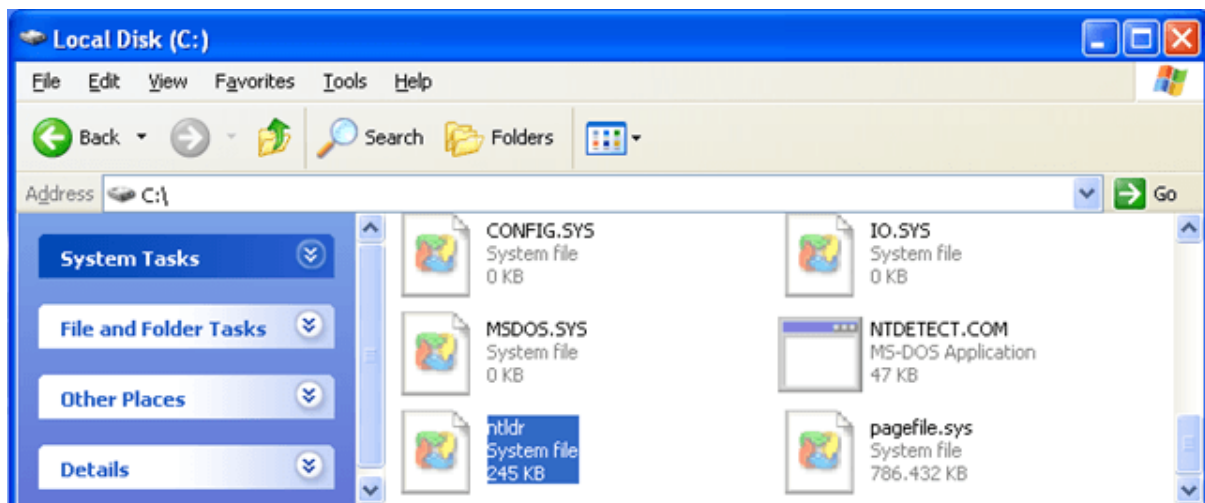
- The Old One: this approach applies to Windows 2000, Windows XP and Windows Server 2003 operating systems.

- The New One: this approach applies to Windows Vista and newer operating systems.

Before Windows can be booted, its boot loader needs to be found. The Windows boot loader is part of the Windows operating system, which is responsible for starting Windows. The old boot load manager is called **NTLDR**, while the new boot load manager is called **Bootmgr**.

Keep in mind that the MBR as well as the VBR boot code executes in real mode as 16-bit instructions, so there are no protection mechanisms in place. Also the NTLDR or the Bootmgr must execute as 16-bit executables, because the switch to protected mode hasn't occurred yet and it's their job to set up the necessary data structures and switch to the protected mode. After the protected mode has been loaded, another boot manager is loaded into memory, which actually boots the Windows operating system.

Since we're in Windows XP, the old approach of booting the system is being used. This is why the **NTLDR** program is used to switch to the protected mode and execute the protected mode boot manager. It's a system file so it's hidden by default, which is why we must change IE settings to also show hidden system files. The NTLDR boot loader can be seen on the picture below:

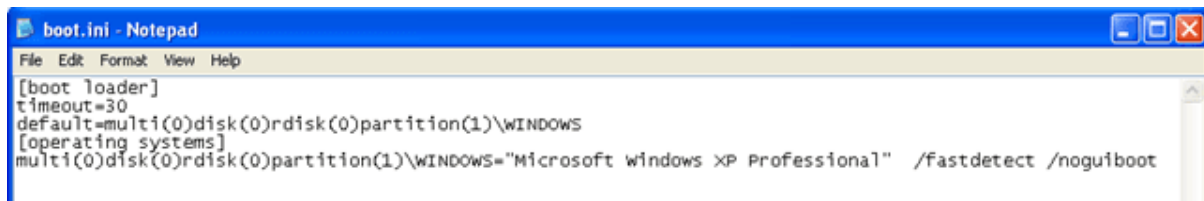


The code in the MBR or the VBR must find the NTLDR file in the root directory, copy it into memory and execute it. Once the NTLDR is executed, it must access the file system on the boot drive to access the files that it needs.

It first tries to find the hibernation file hiberfil.sys, which can be used to resume the system. If the hibernation file is not found, it reads the boot.ini file and prompts the user for the boot menu that presents the boot

options. After that, the Ntdetect.com file is executed, which identifies information about the computer's hardware. Finally, the Ntoskrnl.exe file is executed, which is the kernel of the Windows system.[1]

We've mentioned that NTLDR reads the boot.ini file. The picture below shows the contents of this file:

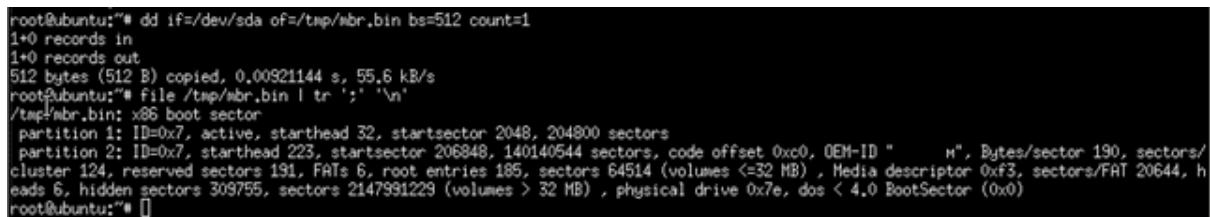


```
boot.ini - Notepad
File Edit Format View Help
[boot loader]
timeout=30
default=multi(0)disk(0)rdisk(0)partition(1)\WINDOWS
[operating systems]
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Microsoft Windows XP Professional" /fastdetect /noguiboot
```

We can take a look at all the options we can enter into the boot.ini configuration file here: [1].

### Booting process of Windows 7

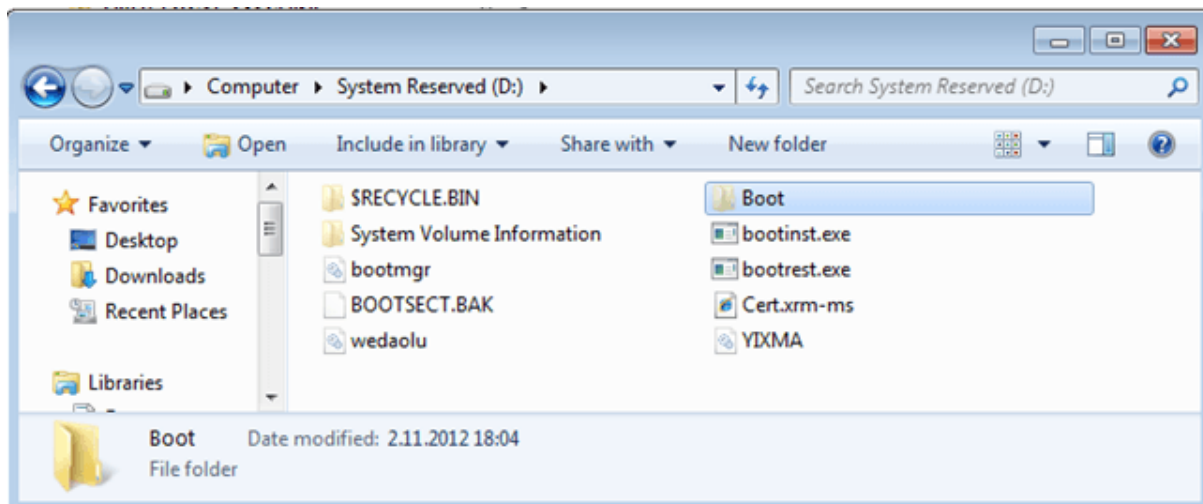
When booting the Windows 7 operating system, Bootmgr is loaded and executed, instead of NTLDR. Let's first dump the MBR record that Windows 7 has installed on the hard drive. On the picture below, we can see the contents of the MBR, dumped with dd and find Linux commands:



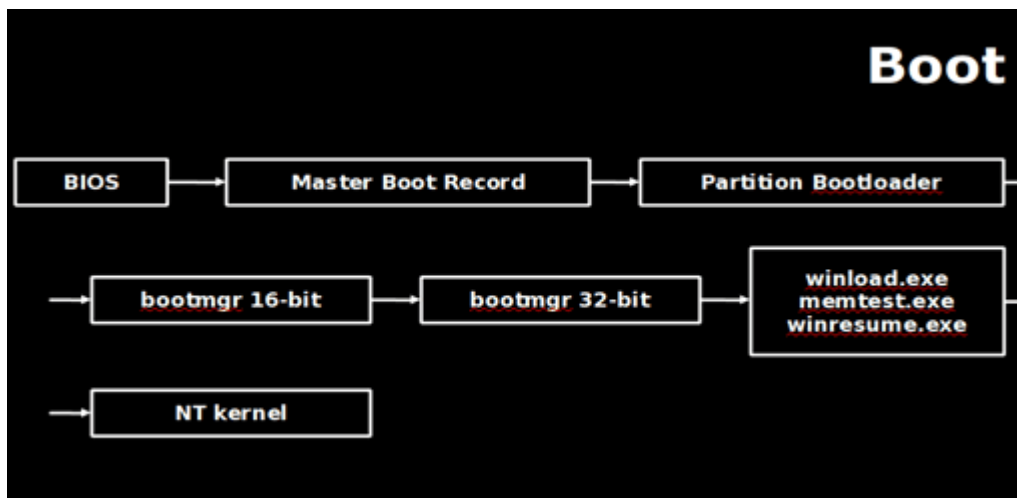
```
root@ubuntu:~# dd if=/dev/sda of=/tmp/mbr.bin bs=512 count=1
1+0 records in
1+0 records out
512 bytes (512 B) copied, 0.00921144 s, 55.6 kB/s
root@ubuntu:~# file /tmp/mbr.bin | tr ';' '\n'
/tmp/mbr.bin: x86 boot sector
partition 1: ID=0x7, active, starthead 32, startsector 2048, 204800 sectors
partition 2: ID=0x7, starthead 223, startsector 206948, 140140544 sectors, code offset 0xc0, OEM-ID "      ", Bytes/sector 190, sectors/cluster 124, reserved sectors 191, FATs 6, root entries 185, sectors 64514 (volumes <=32 MB) , Media descriptor 0xf3, sectors/FAT 20644, heads 6, hidden sectors 309755, sectors 2147991229 (volumes > 32 MB) , physical drive 0x7e, dos < 4.0 BootSector (0x0)
root@ubuntu:~#
```

Notice that there are two partitions and that the first one is active: this is the partition where the Bootmgr is located.

In newer versions of Windows, NTLDR is not used anymore, only Bootmgr. Let's take a look at the contents of the system drive where the booting information are read from:

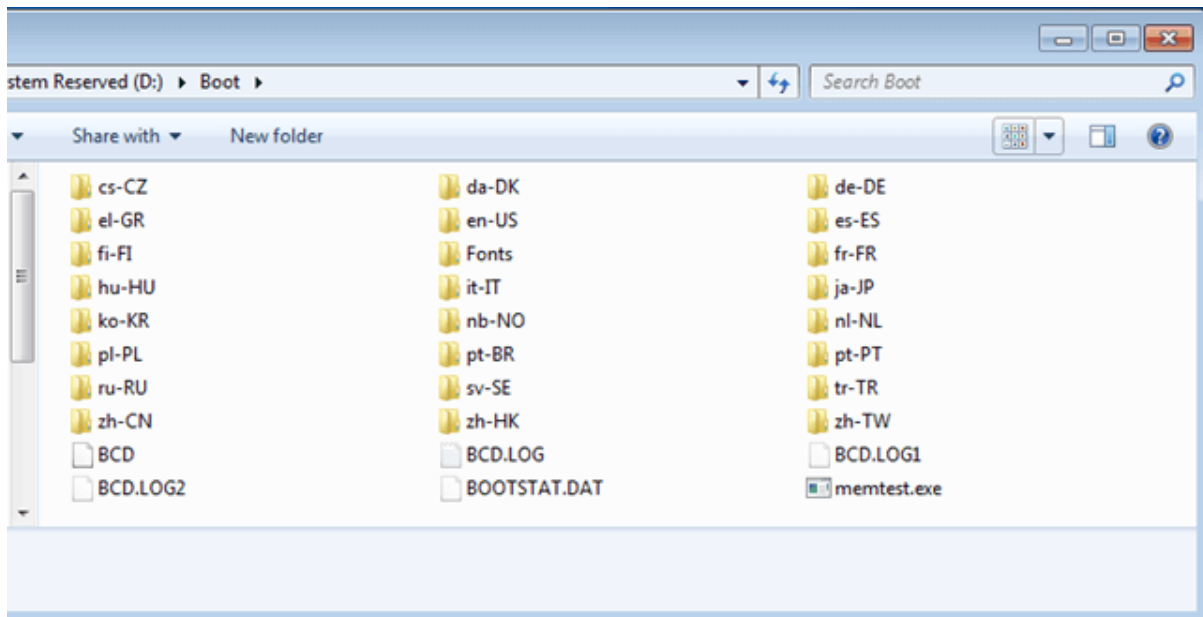


Notice that the **Bootmgr** file is present on the picture above? This is the file that gets loaded into memory and executed, which boots the Windows operating system. Keep in mind that Bootmgr is still a 16-bit program, which must be used to switch from real to protected mode. The whole booting process of newer versions of Windows operating systems can be seen on the picture below, which was taken from [2]:



Notice that at first, the BIOS must load the MBR that contains code to locate the active partition. The boot loader located on that partition then loads the 16-bit Bootmgr, which in turn loads the 32-bit Bootmgr, which then loads the winload.exe, memtest.exe and winresume.exe files. All of them are then used to load the NT kernel. The BIOS, MBR, Partition Bootloader and 16-bit Bootmgr are all executed in real mode.

The Bootmgr must also load the configuration files located in the D:\Boot directory. The picture below lists all the files in it:



In Windows, we can use the tool **bcdedit.exe** to manage the booting process. The picture below shows the Windows boot manager and Windows boot loader, which are displayed if we run the bcdedit.exe program by itself without arguments:

```

Administrator: C:\Windows\System32\cmd.exe
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Windows\system32>bcdedit.exe

Windows Boot Manager
-----
identifier                <bootmgr>
device                    partition=D:
description                Windows Boot Manager
locale                    en-US
inherit                    <globalsettings>
default                    <current>
resumeobject               <bc3d8020-2517-11e2-8603-86d440300033>
displayorder               <current>
toolsdisplayorder          <mendiag>
timeout                    30

Windows Boot Loader
-----
identifier                <current>
device                    partition=C:
path                      \Windows\system32\winload.exe
description                Windows ?
locale                    en-US
inherit                    <bootloadersettings>
recoverysequence           <bc3d8022-2517-11e2-8603-86d440300033>
recoveryenabled            Yes
osdevice                   partition=C:
systemroot                 \Windows
resumeobject               <bc3d8020-2517-11e2-8603-86d440300033>
nx                          OptIn

C:\Windows\system32>

```

The Windows Boot Manager is located on the D: partition, while the Windows Boot Loader is located on the C: partition. The Windows Boot Loader is used to present different booting options, very similar to GRUB as it displays them one after the other. When booting the operating system, all of the Windows Boot Loaders are displayed to us and we can choose the one we would like to boot. If only one boot loader is present

(as on the picture above), then the screen presenting us with the options to boot from is never shown, because the only option is used.

We've seen that the Bootmgr will load the winload.exe program, which is located in the C:WINDOWS\system32 directory. A lot of interesting stuff also happens when the system is being booted, like the ntoskrnl.exe and hal.dll files being loaded into memory, other system DLLs also being loaded, services being started, etc.

Let's configure the boot log where all the actions that will happen will be logged during the booting process. To enable the boot log, we need to execute the following command:

```
C:\Windows\system32>bcdedit /set BOOTLOG TRUE
The operation completed successfully.
C:\Windows\system32>
```

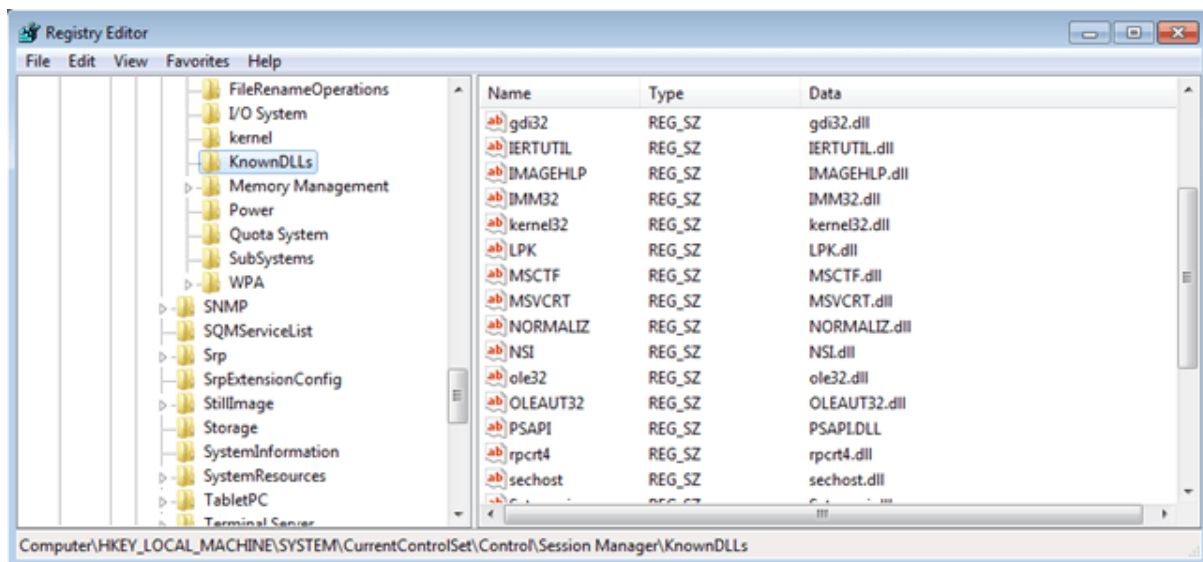
Let's restart the system and check out [C:WINDOWS\ntbtlog.txt](#), which should contain all the actions that happened during the operating system booting. Let's present the beginning of the file:

```
Microsoft (R) Windows (R) Version 6.1 (Build 7600)
 2 25 2013 00:27:45.500
Loaded driver \SystemRoot\system32\ntoskrnl.exe
Loaded driver \SystemRoot\system32\hal.dll
Loaded driver \SystemRoot\system32\kdcom.dll
Loaded driver \SystemRoot\system32\mcupdate_GenuineIntel.dll
Loaded driver \SystemRoot\system32\PSHED.dll
Loaded driver \SystemRoot\system32\CLFS.SYS
Loaded driver \SystemRoot\system32\CI.dll
Loaded driver \SystemRoot\system32\drivers\Wdf01000.sys
Loaded driver \SystemRoot\system32\drivers\WDFLDR.SYS
Loaded driver \SystemRoot\system32\DRIVERS\ACPI.sys
Loaded driver \SystemRoot\system32\DRIVERS\WMILIB.SYS
```

If we search the log file for the "Loaded driver" string, we will be able to identify all the loaded drivers and libraries.

Let's also present some of the libraries that get loaded during the booting of the system:





At the bottom of the picture, you can see the actual entry where the DLLs are accessed: through the KnownDLLs field. Looking through the DLLs on the picture above, we can see that most of the libraries are well known and we know what they are used for, like gdi32.dll, which is used for graphical user interface, kernel32.dll, ole32.dll, etc.

We've also already mentioned that at some point of the booting process, the control needs to be passed to the `ntoskrnl.exe`, which is the Windows kernel. As you may imagine, the kernel needs to do a lot of things before the login screen can be presented to the user. While the boot loader enables the protected mode, the kernel must set up all the data structures that will be used by the system, like page tables. The kernel must also configure the interrupt table for each processor, along with a lot of other things.

One of the processes started during the booting process is also the `wininit.exe`, which is the Windows Init process. This process in turn starts three other processes:

- `services.exe`: service control manager
- `lsass.exe`: local security authority subsystem
- `lsmd.exe`: local session manager

The picture below presents a few processes, sorted from lower to higher process ID numbers:

Image Name	PID	User Name	CPU	Memory (...	UAC Virtualization	Description	Data E...
System Idle P...	0	SYSTEM	95	24 K		Percenta...	
System	4	SYSTEM	00	56 K		NT Kernel...	
svchost.exe	224	LOCAL ...	00	3.088 K	Not Allowed	Host Proc...	Enabled
smss.exe	268	SYSTEM	00	160 K	Not Allowed	Windows ...	Enabled
csrss.exe	348	SYSTEM	00	976 K	Not Allowed	Client Ser...	Enabled
wininit.exe	396	SYSTEM	00	484 K	Not Allowed	Windows ...	Enabled
csrss.exe	408	SYSTEM	00	1.000 K	Not Allowed	Client Ser...	Enabled
winlogon.exe	448	SYSTEM	00	676 K	Not Allowed	Windows ...	Enabled
svchost.exe	472	NETWO...	00	3.108 K	Not Allowed	Host Proc...	Enabled
services.exe	484	SYSTEM	00	2.484 K	Not Allowed	Services ...	Enabled
lsass.exe	496	SYSTEM	00	2.024 K	Not Allowed	Local Sec...	Enabled
lsim.exe	504	SYSTEM	00	884 K	Not Allowed	Local Ses...	Enabled
svchost.exe	612	SYSTEM	00	1.608 K	Not Allowed	Host Proc...	Enabled
VBoxService....	676	SYSTEM	00	968 K	Not Allowed	VirtualBox...	Enabled
svchost.exe	728	NETWO...	00	2.352 K	Not Allowed	Host Proc...	Enabled
svchost.exe	796	LOCAL ...	00	6.952 K	Not Allowed	Host Proc...	Enabled
svchost.exe	916	SYSTEM	00	45.688 K	Not Allowed	Host Proc...	Enabled
svchost.exe	952	SYSTEM	00	14.332 K	Not Allowed	Host Proc...	Enabled
spoolsv.exe	992	SYSTEM	00	1.248 K	Not Allowed	Spooler S...	Enabled

Notice that services.exe, lsass.exe and lsm.exe are started and have PIDs close together, which means they must have been started at approximately the same time.

## Conclusion

In this article, we've seen the basic things that happen when the Windows operating system is booted. First, we talked about the old approach still used in older Windows operating systems that use the NTLDR file to boot the system. The method used in newer versions of Windows operation system is the Bootmgr, which can be managed with the bcdedit.exe executable. We've also briefly looked at the libraries and services that are loaded into memory and started during the boot process.