



Recovering GRUB: Dual Boot Problems and Solutions

Published by the Open Source Software Lab at Microsoft. October 2007.

Special thanks to Chris Travers, Contributing Author to the Open Source Software Lab. Most current version will be maintained at <http://port25.technet.com>.



Abstract:

Those of us who dual boot have all seen it happen. Somewhere down the line, we overwrite the bootloader (or configure it to ignore one of the operating systems) and suddenly we can only boot into one of the operating systems. This paper will outline recovery using GRUB.

Information in this document, including URL and other Internet Web site references, is subject to change without notice and is provided for informational purposes only. The entire risk of the use or results from the use of this document remains with the user, and Microsoft Corporation makes no warranties, either express or implied. Unless otherwise noted, the companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in examples herein are fictitious. No association with any real company, organization, product, domain name, e-mail address, logo, person, place, or event is intended or should be inferred. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

© 2007 Microsoft Corporation. This work is licensed under the Microsoft Public License. The Microsoft Public License is [available here](#).

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

Microsoft, Windows, Windows XP, Windows Server, and Windows Vista are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

All other trademarks are property of their respective owners.

1 Introduction

Those of us who dual boot have all seen it happen. Somewhere down the line, we overwrite the bootloader (or configure it to ignore one of the operating systems) and suddenly we can only boot into one of the operating systems. Probably the most common causes of these issues involve the use of fdisk /mbr and Windows installations overwriting GRUB¹ or LILO² (but it happens with Linux too). In this paper, I will assume that you can only boot into Windows, and that you have decided to use GRUB as your bootloader. A few of these notes are distribution-specific and those portions will be clearly marked. However, most of the process will work on any Linux distribution which conforms to accepted standards.

1.1 Overview of the Boot Process

When a PC is powered on, BIOS first runs through a series of tests called Power On Self Test (or POST), then tries to boot an operating system by looking for a "boot sector" on various boot devices (some of which, such as CDROM's and NIC's, emulate such a sector if they are bootable). On a hard disk, this sector includes two things: the partition table and the master boot record (which contains basic instructions or boot loader for the next stage of the boot process). This bootloader then takes over and provides an environment for the continued booting of the operating system.

1.1.1 Introducing GRUB

GRUB is a versatile bootloader for multiple operating systems which can be loaded via boot sectors from the hard drive, or from other boot-oriented operating environments (such as PXE). Grub supports some operating systems by directly booting their kernels, and others (such as Windows) by passing off the boot process to the native bootloaders on the first sector of specific partitions. This paper will cover GNU GRUB 0.9x.

1.2 GRUB/Linux Considerations

Many, if not most, Linux distributions use the Linux Volume Manager (LVM) for storing most of the operating environment and data. LVM is similar to the disk management capabilities of Windows. GRUB 0.9x does not support LVM directly, and so the boot files, including the kernel must be on a standard partition. Furthermore, the Linux kernel needs to know where the root environment is, and this has to be specified in the command line using LVM notation (for example, root=/dev/VolGroup00/LogVol00). If this argument is not set up properly, you will get a kernel panic during the boot process.

1.2.1 GRUB/Windows Considerations

When GRUB boots a Linux kernel, it loads the kernel directly. When GRUB loads Windows, it has to pass the boot process off to the Windows bootloader which is on the first sector of the Windows partition. This means that the configuration is fundamentally different in both these cases.

¹ GRand Unified Bootloader

² LInux LOader

<http://port25.technet.com>

1.3 The Recovery Process

The specifics of the recovery process will differ slightly depending on your Linux distribution. In general, however, the main work will need to be done from the command line. For this reason, screenshots will not be provided in this paper.

Most Linux distributions allow you to boot from the installation CD, and many of these will automatically mount the filesystems used by your existing distribution. For those that do not, you can use a distribution such as Knoppix³ or Kanotix⁴ to boot and access your system in a similar way. Once loaded, you can repair and reinstall the bootloader, and everything will be fine.

1.3.1 Accessing a Linux system from a Recovery CD

Once you have been able to boot to a Linux operating environment, you will need to access your previous Linux installation. If the CD-based Linux distribution mounted your previous root filesystem, you can find out where it is by typing (as root):

```
bash# mount
```

This will list all mounted filesystems, their types, whether they are read-only (ro) or read-write (rw), and any other options that they were mounted with. If you cannot locate your root partition here, you may need to refer to your distribution's online documentation to determine where it is likely to be created.

If you need to mount the partition manually, the command (as root) is:

```
bash# mount -t [fstype] /dev/path/to/device /path/to/mountpoint.
```

If you do not know which partition you used, the following command (if supported in your environment) may provide appropriate information:

```
bash# sfdisk -l
```

For example, if I want to mount the Logical Volume Volgroup00/LogVol00 at /mnt/sysroot, I might:

```
bash# mkdir /mnt/sysroot
```

```
bash# mount -t ext3 /dev/VolGroup00/LogVol00 /mnt/sysroot
```

In Linux, no output means no errors.

The next step is to actually access the previous installation drive as if you had booted the system before. This is done with the chroot command, which starts a new program, giving it the root directory of its first argument. If the program is different than the current shell, specify it as the second argument. For example:

```
ash# chroot /mnt/sysroot
```

```
chroot: cannot run command `/bin/ash': No such file or directory
```

```
ash# chroot /mnt/sysroot /bin/bash
```

```
bash#
```

³ <http://www.knopper.net>

⁴ <http://kanotix.com>

<http://port25.technet.com>

Specifying the shell should not be necessary, but if you get an error message like the one above, then specify `/bin/bash`. If you get the same error, you probably have mounted the wrong filesystem.

Once you have succeeded in running the `chroot` command to your mounted filesystem, you are ready to reconfigure GRUB and repair your master boot record.

1.3.2 *Repairing GRUB*

Once you get here, you will want to verify where your boot files are located. As of GRUB 0.97 (used by Red Hat Fedora Core 6), Linux LVM partitions are not supported for boot files, so you can narrow your search down to just Linux partitions if you are in doubt (again, `sfdisk -l` comes in quite useful here too).

You will want to take notes of which partition your root filesystem is on and which partition you are booting from for the next step. You will also need to mount your boot partition. For example:

```
bash# mount -t ext3 /dev/sda3 /boot
```

Next you will want to check the `/etc/grub.conf` file in your favorite text editor. If the master boot record was the only incorrect configuration, then this file will be good as-is. If you caused the problem by editing your `grub.conf`, you may want to review the manual at this time. Adding Linux installations to this file is beyond the scope of this paper, but it is possible that under certain circumstances, Windows might not be listed (particularly if you just installed that operating system). To add a Windows entry, you will add a menu item and the appropriate options. For example:

```
title Windows
    rootnoverify (hd0,1)
    chainloader +1
```

Note: `rootnoverify` should point at the hard drive Windows expects to boot from, and that all numbers start from 0. The above example tells GRUB to try to boot Windows from the **second** partition on the root drive.

Once the `grub.conf` is verified, you may want to reapply it to the hard drive. Although GRUB is designed to reread the configuration on each boot, some distributions do not properly support this. Hence, is a good idea to re-apply GRUB to the hard drive. To do this, use the `grub-install` application and supply as its only option the name of the hard drive:

```
bash# grub-install /dev/sda
```

The above command will install GRUB to the first SCSI or SATA drive found. IDE users should use instead:

```
bash# grub-install /dev/hda
```

1.4 **Sample GRUB.conf**

```
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to
this # file
# NOTICE: You have a /boot partition. This means that
```

```
#           all kernel and initrd paths are relative to /boot/,
eg.
#           root (hd0,2)
#           kernel /vmlinuz-version ro
root=/dev/VolGroup00/LogVol100
#           initrd /initrd-version.img
#boot=/dev/sda
default=0
timeout=5
splashimage=(hd0,2)/grub/splash.xpm.gz
hiddenmenu
title Fedora Core (2.6.18-1.2798.fc6)
           root (hd0,2)
           kernel /vmlinuz-2.6.18-1.2798.fc6 ro
root=/dev/VolGroup00/LogVol100 rhgb quiet
           initrd /initrd-2.6.18-1.2798.fc6.img
title Windows
           rootnoverify (hd0,1)
           chainloader +1
```

1.5 Final Thoughts

The above instructions should work (with some modifications where specified) on nearly every major and modern Linux distribution available. Of course, if you are running a fairly old distribution or if you have an advanced configuration utilizing any bootloader other than GRUB then you will need to review your documentation accordingly. Unfortunately, exact specifics are not possible in documentation such as this paper simply because there are too many factors which could affect, even to a small extent, exactly how the above steps have to be carried out. Where disk partitions are involved, I have tried to resort only to the safest tools available, but of course, a typo could cause massive data loss in even these cases. So please be careful.

1.6 About the Author

Chris Travers is the owner of Metatron Technology Consulting, a firm which specializes in helping customers utilize open source software. He has over ten years of IT experience including support of Windows and Linux.