Installing Slackware Linux

Installing Slackware Linux (Post #1)

Slackware Linux is one of the oldest Linux distributions remaining. Over the years, it has stayed true to its roots and form.

Here's what the author, Patrick Volkerding has to say about it.

http://www.slackware.com/info/

The Slackware Philosophy

Since its first release in April of 1993, the Slackware Linux Project has aimed at producing the most "UNIX-like" Linux distribution out there. Slackware complies with the published Linux standards, such as the Linux File System Standard. We have always considered simplicity and stability paramount, and as a result Slackware has become one of the most popular, stable, and friendly distributions available.

What's this about "friendly"? You heard that Slackware was too damned hard, didn't you? If you are expecting cute graphical wizards and penguins automating every configuration step for you, that may be true. However, in essense, Slackware is one of the simplest distributions there is if you are proficient with a Linux system. If you aren't, a little perseverance with Slackware and you will be.

The reason it is easy for an experienced user is, first of all the init scripts and configuration files are easy to follow. They are generally well commented and it's easy to make changes using an ordinary text editor.

Not only that, you are getting the full, complete, standard releases of software in this distribution, installed in a sane manner. The way the developers intended. Therefore, when you go to install additional software not provided by the distribution vendor, you don't run into as many snags.

The packaging system in Slackware is quick, dirty and simple too. Slackware packages (.tgz files) are basically just tar.gz archives, that have install scripts that the packaging utilities execute. No dependency checking, which can be good or bad, depending on how you look at it. To me it's good, because I don't get annoyed by packages that won't install because of some brain dead mechanism that checks for things in specific places. The catch is, you need to be a bit careful installing system software.

Slackware also provides an excellent environment for building your own software from sources.

I could go on at length about why you should give Slackware an honest try but I'll let you follow this guide and see for yourself. We are going to be installing Slackware 9.1, which is the latest release at the time of writing.

Starting the Installation

First of all, if you intend to dual boot with Windows, take care of that first. If you're starting with a

fresh hard disk, create a partition for Windows, and leave the rest unallocated (unpartitioned). Install Windows first.

Boot with the first disk in the Slackware CD set. (or the first CD that you created from the ISO files you downloaded).

If your computer is unable to boot from the CDROM for whatever reason, it is also possible to create a floppy boot disk set for the installation. Read the file README.TXT in the bootdisks directory on the Slackware CD, as well as the rootdisks directory. In Slackware 9.1, this directory is on the first CD.

Once you boot with the installation media, this is the first screen you will see:

Most people with plain IDE systems, can just hit enter here, to load the bare.i kernel image. The README.TXT in the bootdisks directory, describes the precompiled kernel images available on the Slackware CD. If you have SCSI disks, you must read that file, because adaptec.s, scsi.s, scsi2.s and scsi3.s each contain drivers for different SCSI controllers.

So press Enter to load bare.i, or type the name of the kernel image you wish to load (e.g. scsi.s)

The kernel will boot, and then you will be instructed to log on as root.

Just type root and hit enter. You will not be prompted for a password at this time.

Now we must partition the disk. This is probably the trickiest part of Slackware Setup, for there are no point and click partitioning utilities provided. We are going to use the Linux Fdisk utility. It seems scary at first, a bit alien, but it's very easy to operate and you're unlikely to make mistakes if you follow the steps correctly, and do not write the tables to disk until you're sure. I have never had a mishap with this program, and it has never damaged any existing (Windows) partition table entries on the disk.

What I did here was, I hooked up a new Western Digital 40 Gb hard disk for this install. I booted with the Windows XP CD and during setup, created an 8 Gb partition, formatted it NTFS and blasted a quick Windows XP install on there so we can have a dual boot. I left the rest of the disk unallocated.

Fdisk must be invoked with the device name of the hard disk you wish to partition. In this case, we're using the primary master hard disk, so we use the /dev/hda devicename. Here is how IDE disks are named:

/dev/hda - Primary Master
/dev/hdb - Primary Slave
/dev/hdc - Secondary Master
/dev/hdd - Secondary Slave

Note that these do not refer to partitions or filesystems, but the hard disk devices themselves. (/dev/hda1, /dev/hda2 and so on, is how partitions are addressed)

SCSI disks are named /dev/sda, /dev/sdb, /dev/sdc and so on, according to which are first enumerated on the bus.

We need to type fdisk /dev/hda

Don't worry about the informational message about the number of cylinders. Unless you're installing a very old Linux distribution, the boot loader won't have a problem.

Now what? Press m to see a list of commands.

The first thing we want to do is press p to print (display) the partition table. We do this after every step, so we can see the results. Nothing is really changed, until we press w to write the partition table to disk.

There's our 8 Gb NTFS partition, /dev/hda1. The first partition on the disk, and in Windows terms, the active partition. It's going to stay that way.

The units (for Start and End) are in cylinders of 8225280 bytes. Just remember that each unit is rougly 8 megabytes (7.84 if you do the math). It's also displayed in blocks of rougly 1 kb. Don't worry about it, we will be specifying partition sizes in megabytes.

Now, how we partition depends greatly on personal preference. All you really need to install and run Linux is a root partition, and a swap partition. However, that's a fairly large chunk of disk and we can mount parts of the Linux filesystem on separate partitions.

This is basically how I would allocate this space, for use with Slackware. It's just the way I do things, you can choose other partitioning schemes and sizes. If disk space is tight, you should create only a root partition, and save some space for a swap partition. For example, if you have 4 Gb of space to allocate, create a 3.7 Gb root partition and use the rest for swap. That would be a half decent setup.

Using multiple partitions is a bit wasteful, because we have to allow room on each partition for growth. This may result in some disk space staying unused. Err on the side of caution, and allocate plenty of space.

This is what I would do for my own use:

1 Gb root partition (primary partition)
The root filesystem, contains system software and libraries, configuration data (/etc), local state data (/var) and all other filesystems are mounted under it.

Extended partition utilizing the rest of the disk
We then create logical drives on the extended partition.

1 Gb swap partition (logical drive)
Note that you probably don't need a swap partition that large but I like the extra insurance and I have plenty of space. It allows me to work on absolutely huge files, and provides extra memory addressing in the event of some sort of race condition. 256 Mb should probably be enough swap though, if disk space is tight.

8 Gb partition for /usr (logical drive)
Most all of your software and libraries get installed in /usr. It is useful to have a large partition for this.

2 Gb partition for /opt (logical drive)
"Optional" software can be installed here. For example, KDE will be installed to /opt/kde. I install some other software to /opt as well.

18 Gb (roughly) for /home (logical drive)
We use what is leftover, for /home. This is where the user directories are, and where users will store personal files. You may also install some software to /home if desired. I do, and I keep build directories there as well.

Now, we will start creating these partitions.

To create a new partition, press n

We are prompted to choose primary, or extended. We want to create a primary partition here. (though the root partition could be a logical partition on the extended)

Press p to create a primary partition.

We then have to give it a partition number. The Windows XP partition is already partition 1, so we have to choose 2

We are then prompted for the starting cylinder. We will be just hitting enter, to accept the default value. (the first available cylinder). We will be accepting the default starting cylinder for each partition we create. We will specify the ending cylinder, by specifying the size in megabytes. For the value of "last cylinder", we type +1024M to create a partition of roughly 1 gigabyte. Partitions have to end on a cylinder boundary (or waste sectors), and partitioning software automatically adjusts that.

Now, press p to display the partition tables, and you'll see what you've done so far. At this point, if you've made a mistake, simply press d and type the partition number that you want to delete (2 in this case... just don't touch partition 1 or you'll destroy Windows). Nothing has been written yet, you can just delete the partition you've created and repeat the last step. This is why we view the partition info at every step. If satisfied, proceed with the next step. At the command prompt, you can press q at any time to quit without writing anything to disk, if you've made a serious mistake and just want to start over.

Now we are going to create an extended partition, to act as a container for our logical drives.

Press n to create a new partition then press e to choose extended. Press 3 when prompted for the partition number and it will be designated as /dev/hda3. We will never be accessing this partition, just the logical drives we are going to create on it.

Note: How the partition numbers work is, partitions 1 to 4 are reserved for primary partitions. (the extended partition is considered a primary partition). It is an architectural limitation of PC BIOS partition tables, that only 4 primary partitions are allowed on a disk. You can have many logical drives though. Logical drives start being numbered at 5, in the Linux scheme.

Press enter when prompted for the first cylinder, to accept the default of the next available.

When prompted for the last cylinder, this time, simply press enter again. It will allocate the rest of the disk, ending at the last cylinder 4865.

Press p to display the partition tables.

Now we are going to create logical drives until we've used up the extended partition, starting with swap. I generally like to put swap in between the root partition and /usr.

You know the drill. Press n to create a new partition, but this time press l for logical. (In our case, we can't create any more primary partitions because we've already allocated the disk)

Note that we are not prompted to choose a partition number for a logical drive, as it will be assigned 5 as the first one.

Press enter to accept the default value of the first cylinder. For the last cylinder, I'll type +1024M to create a 1 Gb partition.

Press p to display the partition table, and note that our new partition is /dev/hda5. There will be no /dev/hda4, because there will be no more primary partitions on this disk.

Aside: Just so you understand how this works, let's say that when we created the extended partition, we didn't allocate the rest of the disk. We left some space unallocated. If we were to create a primary partition using that space now or some time in the future, it would become /dev/hda4.

OK, now, note the Id column in the display of the partition table. By default, when we create partitions they are of type 83, Linux Native.

We must change the partition type of the one we just created to 82, Linux Swap.

Press t to "change a partition's system id" and then press 5 when prompted for the partition number. (Following my partitioning scheme, that is. Use the correct number for your swap partition of course)

When prompted for the Hex Code (partition ID), if you were to press L, you would see a long list of possible partition types that the Linux fdisk utility is aware of.

Type 82 for Linux Swap, and hit enter. When you press p to display, you will see the change.

The rest of the partitions we'll create, will be the default type 83, Linux.

Press n to create a new partition. Choose l for logical. Press enter to accept the default first cylinder. For the last cylinder, type +8192M to create an 8 Gb partition for /usr.

Again, n for a new partition, and l for logical. Press enter for the first cylinder. For the last cylinder, type +2048M to create a 2 Gb partition for /opt.

Now, we'll allocate the last partition for /home.

When asked for the first and last cylinders, just press enter for both of those this time, as we're using up the extended partition.

If satisfied with your changes, press w to write the partition table to disk, and exit the Linux fdisk utility.

If you see a warning like that, restart the system (with the slackware CD). I am seeing that message, because I altered the partition tables on a live system (to get those screenshots easily), but I have seen similar warnings when writing the partition tables to disk if I've gone back and redone them after already writing. You should just see "Calling ioctl() to re-read partition table", and "Syncing Disks". You only need to reboot if there were warnings.

Note: I said I altered the partition tables on a live system. That means, the data on those partitions was effectively lost. The next reboot would have been oblivion. Not a problem because it was just a test install, and I planned to install the OS again (Slackware installs very quickly), but know that you can't adjust partitions on the fly, as the partitions must be formatted afterwards.

Take note of which partition devices you created to correspond with your mount points. You'll need to specify them, during setup.

Now we are ready to proceed with the Slackware installation.

Now that we have our Linux partitions created, at the root prompt we can type setup

This is the main setup menu. You can read the help if you like, but you can just skip down to ADDSWAP unless you need to remap your keyboard for some reason. Use the arrow keys to navigate, and enter to select.

It will detect your swap partition for you, format it (mkswap) and activate it (swapon)

Note: The hard disk devices in these screenshots are /dev/sda. Don't pay any attention to that, it's just because I took these screenshots from within a virtual machine. It emulates disks as scsi devices. Just know that's not the disk we partitioned in the examples above, so there's no confusion.

After completing a step, setup automatically takes you to the next step in sequence. Next, is to select the target partitions. Here is where we choose our root partition, and then choose mount points for the other partitions.

This is where we select our root partition (/). Following our partitioning example, that would be /dev/hda2.

Now it will prompt you to format the partition. I would choose to check for bad blocks while it's formatting.

Choose your desired filesystem. I like to use ext2 because it's a simple filesystem that's well matured, but

you may want to choose ext3 to have a journaling filesystem.

Now it prompts to choose the inode density for the filesystem. Just hit enter to go with the default of 4096 unless you know what you are doing, and specifically why you want to do it.

If you just created a root partition and swap, you are done formatting now. If you created other partitions, they must now be selected, formatted and assigned mount points.

Swap doesn't show up in this list.

We are mounting this partition as /usr.

Continuing on, we are prompted to select, assign mount points and format the rest of our partitions in the same manner.

When finished, a summary is displayed

In the next step, you will be prompted to select the source media.

Hit enter to choose a Slackware CDROM, and it should detect it automatically.

In the next step, we are prompted to select package categories.

These govern which series of packages will be installed on the system. By default, all categories are selected except KDEI (KDE i18N internationalization). If you're just going to be using English/Western charsets you don't need to install KDEI.

For your first time installing Slackware, I recommend leaving all package categories enabled. You can just choose OK here.

Next, we are prompted to choose the "prompt mode", that is, the degree of interaction for installing packages.

Full, installs all packages in the categories you've selected, without prompting. This is what I recommend for your first Slackware install. Install everything, and you can easily remove packages you don't want later after you get a feel for things. I do know what I'm doing, but this is the option I normally use. It's just easier.

Newbie prompts for each package as they are being installed. I do not recommend this, as it is quite tedious. Also, you may not know what you want/need yet.

Menu is a bit better, as it lets you choose groups of related things.

Expert. If you know what you are doing, the expert prompt mode is an excellent way to choose exactly which packages you want installed on the system, prior to package installation. This really is good, it's not terribly confusing like similar package installation modes in other distributions.

The custom/tagfile options use tagfiles to automate a custom package selection. I've never used them. This would be handy if you were wanting to roll out the same installation on several machines though.

Choose full and watch the packages install non-interactively. It won't take very long, even on a relatively slow machine.

At some point during the package installation, you will be prompted to insert the second CD.

When the package installation stage completes, you are prompted to choose a kernel.

I recommend the CDROM option, and choosing the same kernel that you chose at the initial boot prompt when you booted with the Slackware CD. It got you this far.

Because I did these screenshots in a virtual machine that uses scsi emulation for the virtual disks, I had to choose scsi.s. On an IDE system, you probably either want bare.i or bareacpi.i (warning: acpi can cause boot problems if your BIOS implementation of ACPI doesn't jibe... this is why I recommend using the same kernel you chose at the initial boot prompt)

Next, you are prompted to create an emergency boot disk.

I highly recommend taking the time to create this disk, for it can be used to start the distribution if anything ever happens to your boot loader. You will be able to easily fix it, if you can start the system using this boot floppy.

You will now be prompted to create a symbolic link for your modem device.

If you have a modem, you can do that here. Saying "no modem" doesn't mean you can't use a modem, you can create the /dev/modem symbolic link later, or just use the appropriate device (e.g. /dev/ttyS1 for COM2)

Next, you will be prompted to enable the hotplug system. If you have such devices, say Yes, otherwise No is a good idea.

As you can see, it's possible for it to cause problems on some systems. Note the information on how to get out of the trap if it happens to you.

Now we are prompted to install the LILO bootloader.

You will most likely want to choose simple here. Choosing expert, will result in lilo not behaving as you expect and you'll have to manually edit the lilo.conf file (or run liloconfig from within the OS) to get the desired functionality back (e.g. it won't even prompt you to select an operating system). If you choose to skip the installation of lilo altogether, then you will only be able to boot into your Slackware system using the boot floppy that you created in the previous steps.

Next, you are prompted to choose the VGA (display) mode of your console, either standard VGA, or one of the VESA framebuffer display modes. The reason this is in the lilo configuration, is because the boot loader passes these parameters to the kernel on boot.

It is nice to have a framebuffer console for when you're not running XFree86, but if the framebuffer mode you've chosen doesn't work well with your display hardware, you could end up with an unusable display (until you fix it of course... you could boot with your boot floppy).

Consider choosing standard for now, to use standard VGA. You can change this parameter in your /etc/lilo.conf file later. If you're always going to be using the XFree86 GUI environment, it's not going to matter much anyways.

You are now prompted to enter any extra boot parameters, that lilo is to pass to the kernel.

He gives one very common example of why you might need to do this: If you have an IDE CD Writer. In the 2.4 kernel series, CD writing uses SCSI emulation and the kernel must know which drive is to use that mode, if both IDE-CD Support and IDE-SCSI support are to be loaded in the running kernel. The example of hdc, is for a secondary master. Use hdd if your writer is secondary slave.

Next, you are prompted to choose the destination for installing LILO. You will almost certainly want to choose MBR (unless you know what you are doing)

He says "possibly unsafe" because there are a few situations where writing to the master boot record is indeed unsafe. For example, if your bios doesn't support the capacity of the drive, and you have translation software installed (e.g. "MaxBlast" or "EZBios"). Another reason it could be unsafe is, if you are using another boot loader (e.g. System Commander, or Boot Magic). Also, before you ever write to the MBR (installing pretty much any OS), you must ensure that bios level MBR protection is disabled. (a.k.a boot virus protection, or "Trend ChipAway"). Installing LILO to the MBR, is the most common way that it is used and it is normally quite safe and can be used to start your Windows operating systems as well.

The "Root" option, to install LILO to the superblock of your root partition, is mainly useful if you intend to use another boot manager to invoke LILO.

Next, you are prompted to create a symbolic link for your mouse.

Even if you don't intend to use gpm, it's still useful to have a correct /dev/mouse symbolic link. This way you can just specify that device when you configure XFree86 after the OS is installed. I choose imps2 for my Logitech ps/2 wheel mouse.

I don't have much use for this (it's got nothing to do with using a mouse in the GUI), but if you wish to have mouse support at the console, you can load gpm at boot time.

At this point, you will be asked if you want to Configure your network. If you only have dial up networking, and don't even have a NIC, you can say No to that question for now, and you'll be prompted to configure your clock, timezone and set a root password. Alternatively you can proceed, and choose loopback. That is really what you should do, as then at least you'll set a hostname for the machine.

If you chose to configure your network now, the first thing you will be prompted for is a hostname. Enter something.

Now you'll be prompted to enter a domain name.

If you intend to participate as a member of a network that has a nameserver, you will want to enter your fully qualified domain name, ending in .com, .org, .edu or similar.

Otherwise just enter localdomain. In subsequent steps you can even remove that domain name. (That's what I do, for I don't really need to have one)

Next, you will be prompted to set up your computer's IP address.

If your network adapter connects to a cable modem, or a broadband router, or uses a PPPoE connection (PPP Over Ethernet... commonly used for ADSL Internet connections), then you probably want to choose DHCP to have your TCP/IP info automatically assigned.

If you choose DHCP, you will be prompted for a DHCP hostname. If you connect directly to a cable modem, you may need to specify your user ID here.

Otherwise, just leave it blank and hit enter.

Next, setup will prompt you to probe for your network adapter.

If it doesn't detect it, don't panic. It just means you'll have to figure out which kernel module your network adapter needs and configure the network later.

Ok, in this virtual machine, that's the virtual adapter it detects. It works. However, on the real Slackware installation, it doesn't automatically detect my D-Link 530TXS. Not a problem, because I know what kernel module it needs (sundance.o). That's something for later and we'll cover it then. It doesn't prevent us from configuring most of the network information though.

If you've chosen to use DHCP, a confirmation screen is what you'll see next. Your network configuration steps are completed.

Myself, I just configure my network statically, and I don't use the DHCP server on my router. So, if you choose Static IP instead of DHCP, this is how the configuration goes.

Enter your IP Address.

Enter your Subnet Mask

Enter your Default Gateway

Enter a Nameserver

Note: I'm just entering the IP address of my router here, it acts as a DNS proxy. The Primary and Secondary DNS servers of my ISP are entered in my router's WAN configuration. You will probably want to enter your ISP's Primary DNS server in this field, and then you can add more nameservers (e.g. the secondary) to your /etc/resolv.conf file later.

Now you will be prompted to confirm your network settings.

You can edit these settings from this dialog as well. For example, I want to remove the domain name "localdomain" altogether.

This concludes the network portion of setup.

After the network configuration, you will be prompted to configure your startup services.

Many of these are network server daemons, and if you are just using your computer as a workstation, you will want to leave most all of them disabled. You may want things like the CUPS print server.

Next, you will be prompted to configure your clock and timezone.

Next, you will be prompted to choose a default window manager, for when you start XFree86.

If you are new to Linux, select KDE for now, you can try some of the others later.

Now you're prompted to enter a root password.

Say Yes. You'll be prompted to type a root password twice, for confirmation.

Slackware setup is now complete. You will be prompted to exit setup and press ctrl-alt-del to reboot your machine.

After exiting, the CDROM tray will open, so you can remove the CD. You'll be back at the root prompt after that. Press ctrl-alt-del to restart the system, and boot Slackware for the first time!

When the system cycles, you'll be at the LILO boot prompt. This is still the virtual machine, but in the real installation on the IDE disk we partitioned, liloconfig didn't add my Windows boot choice to the lilo.conf file. So what I see is exactly the same. Probably because I used the NTFS filesystem. We'll be fixing that up soon, it's not difficult.

Hit enter to start Slackware Linux, and you'll be at the logon prompt. Type root as the username, and you will be prompted for the root password you set near the end of setup.

The first thing you should probably do, is create a user for yourself. You must not use the root user account for normal operation of the system. The Slackware adduser script makes this very easy, by interactively prompting you for information instead of making you supply it with switches on the command line.

Type adduser as root, and then you will be prompted to enter a username. Use lower case for the username.

For most of the prompts you will just want to hit enter to accept the defaults unless you have a specific reason. Let it default to the next available user ID, hit enter to use /bin/bash (unless you want to use another shell of course), accept the default home directory, and accept the default of no expiry date.

You may want to enter a "full name" (I like to pick something humorous). You will then be prompted to type the user's password twice for confirmation. A user can change his own password any time, using the passwd command.

The rest of the configuration steps can really be done in any order, according to what is most important to you. You may want to get the XFree86 GUI started first, so you can use GUI based text editors and such, if you're unfamiliar with working from the command line.

The first thing I'd want to do is get my network going (if it isn't already). The netconfig utility that ran during setup, could not probe for my network adapter. However, I know that it uses the sundance module. How did I know that? Well, when I first bought those NICs, I typed D-Link 530TXS Linux (the "S" is significant in the model number) into a search engine (Google) and found the tidbit I needed in mailing list archives and the like.

Now, during setup we configured our network with the exception of the driver module for the network adapter. That means, all we have to do is load the module, and start the network. Slackware's startup scripts look for a script file named rc.netdevice in the /etc/rc.d directory. This is where the system init scripts are located on Slackware system. (It uses the BSD style init script mechanism)

It is very easy to create this file from the command line. As root, type:

echo "/sbin/modprobe sundance" > /etc/rc.d/rc.netdevice

This will redirect the output of the echo command into the specified text file that will get created. The quotes are important, because there is a space in the string we are echoing. Use the correct module name for your network adapter, of course.

Now, set the file executable:

chmod 755 /etc/rc.d/rc.netdevice

That's it, on the next reboot your network should initialize.

Alternatively, to load a network adapter module, you could uncomment the appropriate module loading line (or add one) in the /etc/rc.d/rc.modules init script.

Now, I don't feel like rebooting at the moment, so I'm going to just type a few simple commands to start the network.

I'm loading the module, then using the ifconfig utility to specify the interface, IP address of the machine, and subnet mask, and then using the route command to specify my router as the gateway.

You probably will want to attempt to start the XFree86 GUI now. By default, Slackware is set up to use the VESA Framebuffer driver for your display hardware. The /etc/X11/XF86Config file is a copy of the file XF86Config-vesa in the same directory.

So if you type startx you may have a usable GUI if the settings are compatible with your display hardware. That will do in a pinch, but you will want to properly configure XFree86 and use the accelerated driver for your video card (which hopefully exists, otherwise you've got some generic options)

I put the XFree86 configuration for Slackware in a separate tutorial, which you can read here:

Configuring XFree86 in Slackware (Opens in new window)

Next, I want to get LILO straightened around, so I can boot that Windows XP installation. At this point I have no way of starting it.

As root, open the /etc/lilo.conf file with a text editor. I drew a box around the section that I added, to the bottom of the file.

Lines that start with # are comments, and are ignored.

This is called "chainloading". What we are doing, is instructing LILO to pass control over to whatever code is in the /dev/hda1 partition's boot sector. It does not have to know anything about the filesystem or the operating system on the partition. In this case, that's the code in the boot sector that finds ntldr; Windows XP's own boot loader. Any additional Windows operating systems that the Windows XP boot loader's boot.ini file is configured to start (e.g. Win9x) will be available from the ntldr menu as well.

What you will see in the LILO boot menu, is the label windows.

While you are editing lilo.conf, you probably will want to change the timeout to a more reasonable value. It defaults to 1200, which is 2 minutes. (The value is in 10ths of a second, so a value of 300 is 30 seconds)

After you are finished editing the lilo.conf file, you must run the lilo command (or /sbin/lilo if /sbin isn't in your path) to rewrite the changes, or they will have no effect.

As root, type lilo and you should see in the output that it has added both Linux and windows to the configuration.

I rebooted the machine, and I can start both Linux and windows.

If you ever want to access that NTFS filesystem from within Linux (read-only support for NTFS), you will have to load the ntfs kernel module, and mount the filesystem.

Create a mount point (an empty directory)

mkdir /mnt/windows

Load the kernel module.

modprobe ntfs

Mount the filesystem.

mount -t ntfs /dev/hda1 /mnt/windows

You access it from /mnt/windows.

Slackware 9.1 ships with the ALSA (Advanced Linux Sound Architecture) system. I had never really used it before, beyond manually loading ALSA kernel drivers without having any of the utilities installed. It's considerably more complex than the older OSS (Open Sound System) drivers, requiring more kernel modules and module aliases to be set up in a modules.conf file. I thought I was going to really hate it, but when I saw how easy it was to configure, I had to re-evaluate that.

As root, type alsaconf and a curses based configuration utility will appear.

It probes for your sound card.

Offers to set up your modules.conf file for you.

Some nice informational messages.

That should be it, your audio should now work!

If that doesn't work for you, then it will be manual configuration. Check the Alsa Soundcard Matrix to see if your card is supported, and what module to use.

http://www.alsa-project.org/alsa-doc/

As for configuration, this is what you'll want to put in your /etc/modules.conf file. The lines should pretty much be the same for all sound cards, but what you must change is the driver module, which I have shown in bold.

quote:
Stuff for the kernel module loader
alias char-major-116 snd
alias char-major-14 soundcore
Your Driver
alias snd-card-0 snd-ens1371
alias sound-slot-0 snd-card-0
OSS Emulation
alias sound-service-0-0 snd-mixer-oss
alias sound-service-0-1 snd-seq-oss
alias sound-service-0-3 snd-pcm-oss
alias sound-service-0-8 snd-seq-oss
alias sound-service-0-12 snd-pcm-oss

Some cards may need additional modules or options. See the "details" section for your card, at the Alsa Soundcard Matrix.