Slackware Tips

About

This article contains a random collection of tips for those running *Slackware Linux*. The tips cover subjects such as recovering LILO, speeding up boot time, and keeping your applications up to date.

Quiet Console Login

Get rid of messages about mail and other things when you log in. In your home directory run run this command.

```
touch .hushlogin
```

Graphical Login

Slackware comes with the XDM and KDM graphical login managers. GDM is no longer provided, but may be installed via a third party GNOME provider such as Dropline GNOME. To switch to a graphical login manager, first make sure that X works, then edit /etc/inittab. Be very careful when changing this file. Locate the section of the file that looks similar to the following (it is usually at the top).

```
# Default runlevel. (Do not set to 0 or 6)
id:3:initdefault:
```

Change that entry so that instead it looks like the following.

```
# Default runlevel. (Do not set to 0 or 6)
id:4:initdefault:
```

You may also wish to edit /etc/rc.d/rc.4 and change the order of preferred display managers. By default, GDM is tried first (oddly enough), then KDM, and finally XDM.

Log in, Start X without a Display Manager

This is useful for people who are the only ones that use their systems, want to jump straight to the desktop when powered on, and don't want the overhead of a display manager. First, create a text file containing the following code.

```
#include <unistd.h>

int main() {
  execlp( "login", "login", "-f", "YOUR_USER_NAME", 0);
}
```

Compile this file into a program with gcc. Note that YOUR_USER_NAME should be replaced with your regular user's login name.

```
gcc -o autologin autologin.c
```

Now place autologin in /usr/local/sbin. Make sure it is owned by root with chown root:root /usr/local/sbin/autologin. There are three files that now need to be edited. First, open the /etc/inittab and locate the section looking like this.

```
# These are the standard console login getties in multiuser mode:
c1:1235:respawn:/sbin/agetty 38400 tty1 linux
c2:1235:respawn:/sbin/agetty 38400 tty2 linux
c3:1235:respawn:/sbin/agetty 38400 tty3 linux
```

Change it so that it looks like this.

```
# These are the standard console login getties in multiuser mode:
c1:1235:respawn:/sbin/agetty 38400 tty1 linux
c2:1235:respawn:/sbin/agetty 38400 tty2 linux
#c3:1235:respawn:/sbin/agetty 38400 tty3 linux
c3:235:respawn:/sbin/agetty -n -l /usr/local/sbin/autologin 38400 tty3 linux
```

Note that removing the one from the ‘c3:235’ prevents that console from activating in single user mode. Also, if you are currently using a graphical display manager (runlevel 4), switch back to console login (runlevel 3) by editing and reverting to id:3:initdefault:. Now edit /etc/login.defs and find the section looking like this.

```
# If defined, either full pathname of a file containing device names or
# a ":" delimited list of device names.  No password is required to log in
# as a non-root user on these devices.
#
#NO_PASSWORD_CONSOLE tty1:tty2:tty3:tty4:tty5:tty6
```

Change it so that it now looks like this.

```
# If defined, either full pathname of a file containing device names or
# a ":" delimited list of device names.  No password is required to log in
# as a non-root user on these devices.
#
#NO_PASSWORD_CONSOLE tty1:tty2:tty3:tty4:tty5:tty6
NO_PASSWORD_CONSOLE tty3
```

Lastly, edit the .bash_profile in the user’s home directory. You may not have one, in which case just create a new, blank one. Add this code to it.

```
if [ -z "$DISPLAY" ] && [ $(tty) == "/dev/tty3" ]; then
  startx
fi
```

You are now all set to reboot and automatically arrive in X! Console three will automatically start X, while the others remain just as they were before.

Booting faster

By default, Slackware’s boot time is quick, but can be sped up considerably with just a few configuration file edits. These edits will focus on the runlevel scripts located in /etc/rc.d. I will only touch on a few bottlenecks, but that should be enough to have you well on your way to complete boot customization. Additionally, LILO can be made a bit quicker, especially if you are only booting Linux.

Each runlevel in Slackware (single user mode – 1, multi user mode – 3, etc) has a corresponding script in the /etc/rc.d directory. In addition, there are scripts for services such as apache (/etc/rc.d/rc.httpd), mysql (/etc/rc.d/rc.mysqld), and ACPI (/etc/rc.d/rc.acpid). These scripts are run if they are executable, and skipped if they are not. This is one way of configuring services in Slackware without the help of any configuration programs.

rc.M

The /etc/rc.d/rc.M file is run when the system boots into multi user mode. There are three edits in this file that will greatly improve boot time. First, place an ‘&’ following the line that initializes the hotplug system. Find this section:

```
# Initialize the hotplugging subsystem for Cardbus, IEEE1394, PCI, and USB devices:
if [ -x /etc/rc.d/rc.hotplug -a -r /proc/modules ]; then
  # Don't run hotplug if 'nohotplug' was given at boot.
  if ! grep nohotplug /proc/cmdline 1> /dev/null 2> /dev/null ; then
    echo "Activating hardware detection:  /etc/rc.d/rc.hotplug start"
    . /etc/rc.d/rc.hotplug start
  fi
fi
```

And add the ‘&’ at the end of the . /etc/rc.d/rc.hotplug start line like this:

```
# Initialize the hotplugging subsystem for Cardbus, IEEE1394, PCI, and USB devices:
if [ -x /etc/rc.d/rc.hotplug -a -r /proc/modules ]; then
  # Don't run hotplug if 'nohotplug' was given at boot.
  if ! grep nohotplug /proc/cmdline 1> /dev/null 2> /dev/null ; then
    echo "Activating hardware detection:  /etc/rc.d/rc.hotplug start"
    . /etc/rc.d/rc.hotplug start &
  fi
fi
```

Next, find the following section and comment it out by placing a ‘#’ in front of each line.

```
# Update all the shared library links:
if [ -x /sbin/ldconfig ]; then
  echo "Updating shared library links:  /sbin/ldconfig"
  /sbin/ldconfig
fi

# Update the X font indexes:
if [ -x /usr/X11R6/bin/fc-cache ]; then
  echo "Updating X font indexes:  /usr/X11R6/bin/fc-cache"
  /usr/X11R6/bin/fc-cache
fi
```

That section should now look like the following.

```
# Update all the shared library links:
#if [ -x /sbin/ldconfig ]; then
#  echo "Updating shared library links:  /sbin/ldconfig"
#  /sbin/ldconfig
#fi

# Update the X font indexes:
#if [ -x /usr/X11R6/bin/fc-cache ]; then
#  echo "Updating X font indexes:  /usr/X11R6/bin/fc-cache"
#  /usr/X11R6/bin/fc-cache
#fi
```

These commands only need to be run on occasion. Updating shared libraries really only needs to be done when you add or remove shared libraries, and similarly X font indexes only when fonts are added and removed.

rc.inet1

Most desktop distributions are now backgrounding the request for a DHCP IP address on boot. Slackware, being a common choice for servers, is more conservative by waiting for the request to complete before continuing the boot process. In the file /etc/rc.d/rc.inet1 locate the following section (search for dhcpcd).

```
echo "/etc/rc.d/rc.inet1:  /sbin/dhcpcd -d -t 60 ${DHCP_OPTIONS} ${1}" | $LOGGER
/sbin/dhcpcd -d -t 60 ${DHCP_OPTIONS} ${1}
```

Background the DHCP request by placing a '&' at the end of the command like this.

```
echo "/etc/rc.d/rc.inet1:  /sbin/dhcpcd -d -t 60 ${DHCP_OPTIONS} ${1}" | $LOGGER
/sbin/dhcpcd -d -t 60 ${DHCP_OPTIONS} ${1} &
```

LILO

Those running slower systems are familiar with LILO's "packman trap" ("Loading Linux.."). Compiling a custom kernel can speed up boot times considerably, but also the way in which the kernel is accessed from the disk makes a difference. Open /etc/lilo.conf and add compact to the LILO global section. It will look something like this.

```
# Start LILO global section
boot = /dev/hda
compact
```

Additionally, those running only Linux (not dual booting) can benefit further. Once you have a stable kernel that you're happy with, disable all of the prompting and timeouts in the lilo.conf file. This means commenting out with '#' prompt, timeout=, and message=. Putting these tweaks together with the other booting speed ups and autologin will allow you to get 15-20 second boot times even on older machines!

Note: Remember that any changes to /etc/lilo.conf will not take effect until running the lilo command.

System and LILO Rescue (chroot)

Boot your system with the Slackware install DVD or first CD using whatever kernel options you need to see your hard disk. Once logged in and presented with the console, mount your root partition to /mnt with the following command (assuming your root partition is /dev/hda3. Use cfdisk to locate your root partition if needed.

```
mount /dev/hda3 /mnt
```

Once the root partition is mounted, mount any additional needed partitions such as /boot into the filesystem. For example, if your /boot partition is /dev/hda1, run this command.

```
mount /dev/hda1 /mnt/boot
```

Lastly, chroot into the system with this command.

```
chroot /mnt
```

Once chrooted, you may edit configuration files, rerun lilo, install or remove packages, and any number of things just as though you had booted your system (minus some hardware support since you're using the generic Slackware kernel). Once finished, simply type exit and reboot.

Packages or .....

Slackware is an extremely stable system that has a regular release approximately once per year. It is, therefore, not always as up to date as other distributions. You may find it better to run some of your most used and updated programs from source or installers rather than relying on Slackware packages. This essentially makes Slackware the base system, with the most used programs installed in a generic way.

Rather than install Firefox and Thunderbird Slackware packages, download the versions from Mozilla. Extract them to /usr/local/firefox and /usr/local/thunderbird, then create symlinks with the following commands.

```
ln -s /usr/local/firefox/firefox /usr/local/bin/firefox
ln -s /usr/local/thunderbird/thunderbird /usr/local/bin/thunderbird
```

Similarly, if XFCE is your desktop of choice, use the graphical source installer to compile and install the latest version to /usr/local.

Another option is to upgrade desired packages from Slackware-current, the development version of Slackware, or Linuxpackages, the Slackware package community.

Java JDK

Slackware comes with the Java runtime environment, but does not include the Sun Java Development Kit. To install the JDK, first run pkgtool and remove the jre package if it is installed. Download the Linux “self-extracting file” from Sun. Run the downloaded file with sh jdk-..., where jdk-... is whatever version downloaded. After scrolling through the license agreement, the package will extract to a directory. Create a permanent directory for the JDK with mkdir /usr/lib/java, then move the contents of the extracted download to this directory with mv jdk-.../* /usr/lib/java. Once complete, you may need to add /usr/lib/java/bin to your user’s PATH environment variable.