Proprietary Graphics Drivers

The X.Org X11 graphical system provides many drivers, supplying at least 2D video acceleration for most video cards; however, if a system is equipped with a GPU from AMD (ATI) or nVIDIA, proprietary binary drivers can be downloaded from the web sites of both these vendors and installed.

The aim of this article is to outline the various methods of installation and configuration of these proprietary drivers in Slackware.

AMD (ATI) Catalyst Driver Installation and Configuration

AMD are dropping support for older cards (4xxx and downwards) in the current (12.6) and future releases of the proprietary drivers, so please make sure that your card is still supported. If it is not, then the open source `xf86-video-ati` driver should be sufficient, albeit a bit slower than the proprietary version.

Note: the driver will not work on 14.2 because Catalyst supports only Xorg 1.17 and 3.19 (or no?).

Download

Note that the installer will compile a driver (fglrx) and it needs the `kernel-modules` and `kernel-source` packages to be installed. The following steps are valid for both the UI and the CLI version of the installer.

Installation

The following instructions are for creating an installation package suitable for Slackware. It is generated automatically by the installer and seems to be working just fine in most cases. There is also an 'automated' installation method.

The Catalyst and driver suite for Linux can be downloaded from this link. The file is suitable for both 32 and 64 bit.

After downloading, unzip the file with:

```
$ unzip amd-driver-installer-VERSION-x86.x86_64.zip
```

Next, switch to root with:

```
$ su -
```

CD to the folder which contains the 'run' file and run the installer with:

```
# sh amd-driver-installer-VERSION-x86.x86_64.run
```

The installer will collect data about your system and will prompt for either an automatic installation or to create a distribution-specific package. Select distribution-specific package here, as it can then be managed by Slackware's packaging tools later.

When prompted to select a distribution, please Select “Detected OS: Slackware/Slackware”.

After it has finished building, The installer will generate an installation package in the same directory as the the 'run' file and you can now do:

```
# installpkg fglrx-VERSION-x86-1.tgz
```

If any errors were detected during installation you can see them in `/usr/share/ati/fglrx-install.log`.

Configuration

You will need to reboot the system for it to make use of the new drivers. Before doing so you need to edit `/etc/modprobe.d/blacklist.conf` (or create a new file: `/etc/modprobe.d/radeon_blacklist.conf`) and add the following lines to the file:

```
blacklist radeon
blacklist radeonhd
```

Next you can create a new X.Org configuration file by running:

```
# aticonfig --initial -f
```

X.Org does not usually require an `/etc/X11/xorg.conf` file, but in some cases (usually older cards) it is necessary to add the following line in the fglrx 'Device' section:

```
Option "SWCursor" "true"
```

If there were no errors and the driver was installed successfully, you should reboot now for the drivers to be used.

Testing

After rebooting, you can test the drivers by running:

```
$ fgl_glxgears
```

This should display rotating gears that run smoothly without glitches.

You can also run:

```
$ glxinfo
```

Have a look at the first few lines of output to see if DRI is enabled. If it is not, then you need to do a bit more work, because the package creation built into the ati installer is quite old (pre official 64bit Slackware) and some files need to be moved around to make it work on the 14.1 version of Slackware. A package for slackbuilds.org that fixes the problem is being worked on, but in the meantime you can try a manual solution posted here (linuxquestions post) ,or if all else fails, the automatic installation option should work.

Automatic un-install

If you chose to install the package automatically and you need to uninstall the drivers, you can run the installer with an additional `uninstall` argument, like this:

```
# sh amd-driver-installer-VERSION-x86.x86_64.run --uninstall
```

Or call the uninstall script directly with:

```
#/usr/share/ati/amd-uninstall.sh
```

No matter how it was installed, uninstalling the fglrx driver will “break” mesa as some files get moved around. It is advised that you re-install mesa. Also, if you want to revert to the kernel's own radeon/radeonhd drivers, you will have to remove the *blacklist* lines which you added earlier.

nVIDIA Driver Installation and Configuration

Installation via SlackBuilds.org

Edward Koenig maintains the libvdpau, nvidia-driver and nvidia-kernel packages at SlackBuilds.org

All three packages are required to have a functioning nVIDIA driver. The nvidia-kernel package builds the kernel module, the nvidia-driver package builds the X.Org driver and contains the OpenGL implementation as well as the COMPAT32 libraries for use on a multilib Slackware64 system. The libvdpau package is a requirement of the nvidia-driver package.

''nouveau'' Module Removal

The first step in the driver installation is to blacklist the `nouveau` driver. Failure to do so may result in a startx error of “ERROR: could not insert 'nvidia': No such device”.

Blacklisting the nouveau driver is done by installing the `xf86-video-nouveau-blacklist` package from the “`extra`” directory of the Slackware version. This can be accomplished by using the install CD or DVD, obtaining the file from a Slackware mirror, or using the `slackpkg` utility.

Package Installation

There are currently two ways to install packages from SlackBuilds.org:

- by downloading the appropriate build scripts from SBo, and following the SlackBuild installation procedure
- by using the tool sbopkg which automates downloading the sources and the sequential compilation/installation of multiple packages.

Once the packages have been installed via the instructions accompanying each SlackBuild, the X11 server can be started with full GPU support.
To configure X11 to start automatically, please see “Starting X11 with nvidia GPU Support” in the following section.

Troubleshooting

Installation via the nVIDIA Binary

Installing the nVIDIA binary driver involves the following steps:

- Downloading the appropriate `nvidia` driver installer package
- Disabling and unloading the `nouveau` driver
- Installing the `nvidia` driver
- Optionally, configuring the system to start the graphical interface automatically

The following tip lists the key steps for installing the `nvidia` driver. The full article follows.

For those who just want the appropriate commands, here is a summary of the following:

- Go to the nVIDIA Unix Drivers website and download the appropriate driver. The name of the driver package will be something like “NVIDIA-Linux-**ARCH**-**VERSION**.run” where ARCH is the computer's processor architecture and VERSION is the driver version.
- Ensure that X11 is not running; if it is, exit it and login to the text console
- Logged in as root, run the nVIDIA installer with

```
# sh /path/to/NVIDIA-Linux-ARCH-VERSION.run
```

- If prompted to blacklist nouveau, do so and restart
- Launch the installer with root permissions once more after restarting
- At a minimum, choose to accept the license and install the driver. Please consult the nVIDIA Installer Options section for automatic installer options.

If nvidia-xconfig is not run by the installer, then it can either be run upon exiting the installation and before launching the X11 server or the “`/etc/X11/xorg.conf`” file should be edited manually.

Downloading the Appropriate nVIDIA Driver Binary Installer

The first step is to download the appropriate driver from the nVIDIA Unix Drivers website . For 64-bit Slackware (including multilib systems) you should download the “**Linux x86_64/AMD64/EM64T**”

driver package, while 32-bit Slackware needs the "**Linux x86/IA32**" driver. Your safest bet for picking the correct version is to use the "*Latest Long Lived Branch version*" but in some cases (very new graphics card, or display issues which you are trying to resolve) you might want to choose "*Latest Short Lived Branch version*" instead.
Older legacy drivers are available as well for graphics cards which are no longer current. When you select a driver, a list of compatible GPUs will be displayed. Once you have determined the correct driver, the license will need to be accepted and the file saved in an accessible location. Before running the installer, there are some other actions that must be taken.

Disabling the nouveau Module

In general, an nVIDIA GPU will be detected by the Slackware system, and the "`nouveau`" OSS graphics driver will be enabled on the system. This will allow 3D video acceleration on many cards. The `nouveau` and `nvidia` modules are incompatible; thus, before proceeding with the installation of the `nvidia` module, the `nouveau` module must be removed and kept from being loaded automatically on boot by the kernel.

To accomplish this, a file must be created in the "`/etc/modprobe.d/`" directory containing the text

```
blacklist nouveau
options nouveau modeset=0
```

and named appropriately; for instance, "`disable_nouveau.conf`"

The first line will block the `nouveau` module from being loaded automatically during start-up. The module will still be able to be loaded manually by a user or by the X server. Thus, the second line is added, so that should the `nouveau` module be loaded, it will be prevented from doing a kernel modeset, allowing the module to be unloaded.[1)]

As mentioned in the modprobe.d README file, the monolithic module blacklist file has been split into smaller files and stored under /lib/modprobe.d/ However, a .conf file in the /etc/modprobe.d/ directory will override one in /lib/modprobe.d/ This is thus a more sure way to block the loading of the nouveau driver.

After displaying an error to the effect that the nouveau and nVIDIA proprietary drivers are incompatible, newer nVIDIA installers will prompt for permission to create a .conf file to blacklist nouveau in /etc/modprobe.d/ which will be named nvidia-installer-disable-nouveau.conf By default, the contents of this file will be:

```
# generated by nvidia-installer
blacklist nouveau
options nouveau modeset=0
```

The installer will then state that it has failed and request that the system be restarted to put changes into effect. Once the system has been restarted, the nouveau driver will no longer be loaded, and installation of the binary driver can continue.

Alternately, the package `xf86-video-nouveau-blacklist` from the "`/extra`" directory of the Slackware installation media should be installed. Once installed, the system should be restarted, at which point driver installation can continue.

nVIDIA Binary Driver Installation

nVIDIA binary driver installation **can not be completed** while the X11 Window System is active.

The default behavior of Slackware is to boot into a text-only terminal. In this case, the nVIDIA installer can simply be run. However, if the graphical environment was already launched, it must be exited. If lunched with the "`startx`" command, closing the graphical environment can be accomplished by logging out or, and only if necessary, pressing Ctrl+Atl+Bksp.

If using a graphical login manager, such as KDM, simply exiting via the menu or pressing Ctrl+Alt+Bksp **will not work** because the graphical login manager will be restarted automatically. The best way to shut down X gracefully is by logging off, switching to a conole terminal by pressing Alt+F2, logging into the console as root and running:

```
telinit 3
```

Another method to return to the console, crude and thus not recommended but effective if necessary, is to stop/kill the DM and then exit the X Server by pressing Ctrl+Atl+Bksp. For example, KDM can be killed by issuing the command

```
killall -9 kdm
```

Again, this should only be used if other methods fail.

Once X11 is no longer running, the nVIDIA installer can be launched as root. Either log in as root or issue the "`su -`" command, then run the installer. Sometimes, it is advantageous to run the installer using some of the available command-line options. For more information, consult "nVIDIA Installer Options" at the end of this section.

```
$ su -
Password:
# sh /path/to/NVIDIA-Linux-ARCH-VERSION.run
```

When the installer is launched without any command line options and excepting any errors, the installation will proceed as follows:

- The installer will extract itself and start an ncurses interface. The first step is to accept the license terms.
- When installing the module for a system that has no previous version, the installer will begin to build the module. If, however, a previous nVIDIA driver was found, the installer will prompt for permission to remove the previous driver as part of the new installation. To proceed with the installation, permission should be granted to remove the previous installation.
- After building the module on x86_64 systems, the installer will prompt about installing nVIDIA's 32-bit compatibility OpenGL libraries. You will only need the 32-bit compatibility libraries if your Slackware is 64-bit *multilib*.
- Next, in the case of a previous version, the installer will uninstall the module.
- Once no conflicting X and (should this option be chosen) OpenGL files are found, the module will be installed.
- The program will then offer to use the nvidia-xconfig utility to modify the xorg.conf file to reflect

the change in video drivers. This utility often works, but could possibly change other values in xorg.conf. It does, however, back-up the original file to "`/etc/X11/xorg.conf.nvidia-xconfig-original`"
- A prompt confirms that the driver installation is complete and was a success, at which point the program exits to the console.

The `nvidia` module should now be installed for the currently running kernel.

You must create an X.Org configuration file which loads the binary Nvidia driver if you decided *not* to let the `nvidia-xconfig` utility modify your computer's `xorg.conf` file. The X.Org of Slackware supports individial "*.conf" files in a directory `/etc/X11/xorg.conf.d`. Any file with a `.conf` extension will be included together with the main `/etc/X11/xorg.conf` file.
You could create for instance a file named `/etc/X11/xorg.conf.d/10-nvidia.conf` with the following content:

```
Section "Device"
    Identifier  "Device0"
    Driver  "nvidia"
    VendorName "Nvidia Corporation"
    BoardName ""
EndSection
```

Without this definition, you will not get accelerated Nvidia GPU support! X.Org would fall-back to VESA mode because the nouveau driver has been blacklisted. *The kernel will not auto-detect the binary driver as opposed to the nouveau driver*.

Starting X11 with nvidia GPU Support

All that remains is to start the X.org server. This can be accomplished by configuring Slackware to start in *runlevel 4*, which will start a graphical login manager, such as KDM or XDM, on boot. If you want this, then you need to edit the file "/etc/inittab" and change the line

```
id:3:initdefault:
```

to

```
id:4:initdefault:
```

Otherwise you can login to a user account and issue the "`startx`" command to start an X session.

By default, startx will start the window manager which was chosen during installation. To change this behavior, the file "`.xinitrc`" in the user's home directory can be edited to start a different WM. Alternatively, the default WM can be altered on a per-user basis by using the command "`xwmconfig`" and selecting one of the available WM's.

nVIDIA Installer Options

The nVIDIA Installer has many options available to the system maintainer which can be accessed by running the installer with the -A option:

```
# sh ./NVIDIA-Linux-ARCH-VERSION.run -A
```

Some common options are

- `-a, --accept-license` : bypasses the nVIDIA license screen. By doing so, the license is accepted.
- `--update` : check for an updated driver on the nVIDIA website; if it exists, automatically download and install the new version.
- `--uninstall` : removes the nVIDIA driver and other files installed previously.
- `-q, --no-questions` : assumes default answers to all questions. Note: does not automatically accept the license.
- `-s, --silent` : runs the binary installer without a curses UI, automatically accepts the license and uses the default answers for all questions.

Troubleshooting

Fix Me!

Sources

- nVidia section originally written by rinias

howtos, software, nvidia, author rinias

1)

ftp://download.nvidia.com/XFree86/Linux-x86_64/256.44/README/commonproblems.html