nvidia-automatic-builds-via-dkms

Automatic build and install of the Nvidia kernel module/driver using Dynamic Kernel Module Support (DKMS).

Red Hat based systems ship with an nvidia-compatible graphics kernel module, and user space X11 driver) called nouveau. Keeping this default configuration is recommended because nouveau is supported by Red Hat and is provided with each new kernel update.

However, in some situations the nvidia 3rd party proprietary graphics kernel module, nvidia.ko, and user space X11 driver, nvidia_drv.so, are needed instead. This 3rd party software is not supplied or supported by Red Hat. With each new kernel, a 3rd party kernel module (like the proprietary nvidia.ko) needs to be rebuilt. This process can be tedious and confusing (eg, why did my graphical login break after a reboot?). There is a framework named "DKMS" which stands for Dynamic Kernel Module Support. This enables the automatic rebuild of 3rd party kernel modules during the RPM install of a new kernel (ie, DKMS builds the nvidia driver as part of the kernel RPM install). This is done via hooks which are present in the kernel post-install RPM scriplet.

This document describes how to install the nvidia kernel module with DKMS support, so a manual rebuild of the nvidia kernel module is no longer required for every new kernel.

As of 2018-Dec, this process has only been tested on RHEL 6. I believe the process should be very similar on CentOS 7 (or RHEL 7), but I haven't tested it myself yet.

---

1. Find the nvidia graphics (video) model number.

```
$ lspci | grep -i nvidia
```

An example of what the output looks like (this is just the graphics card line from the output):

```
01:00.0 VGA compatible controller: NVIDIA Corporation GP107GL [Quadro P600] (rev a1)
```

2. A google search for 'nvidia linux' finds the nvidia linux download page:

https://www.nvidia.com/object/unix.html

3. Under the "64 bit" section at the top of that page, select "Latest Long Lived Branch Version".

It currently looks like this:

Linux x86_64/AMD/EM64T
Latest Long Lived Branch Version: 410.78

4. After selecting the latest long lived branch above, then select the "Supported products" tab, and search for the model name/number from step 1 above.

From the example in step 1, I found "Quadro P600" in the list of supported products. If you cannot find your model number, go back and look under the supported products for the Latest Legacy GPU version.

5. Click download.

The current link (NON-legacy) from above is (as of 2018-Dec-7):
http://us.download.nvidia.com/XFree86/Linux-x86_64/410.78/NVIDIA-Linux-x86_64-410.78.run

And the current LEGACY driver as of 2018-Dec-10 is (this is for older systems / older graphics cards):
http://us.download.nvidia.com/XFree86/Linux-x86_64/390.87/NVIDIA-Linux-x86_64-390.87.run

An example of how to directly download one of the above using a command line:

```
$ cd /var/tmp
$ curl -sLO 'http://us.download.nvidia.com/XFree86/Linux-x86_64/410.78/NVIDIA-Linux-x86_64-410.78.run'
```

You can view the help info (optional) with this command (use -A to view "Advanced help options). But just view the help options, don't run the installer yet.

```
$ /bin/sh ./NVIDIA-Linux-x86_64-[version].run --help
$ /bin/sh ./NVIDIA-Linux-x86_64-[version].run -A
```

6. Install the DKMS rpm (this should be automatically available from the EPEL yum repository)

```
$ sudo yum install dkms
```

Installing the dkms rpm should also prompt you to install the kernel-devel RPM (as a dependency). If you already have the kernel-devel RPM installed, you won't get prompted. The kernel-devel RPM is required for the nvidia kernel module to be built. DKMS is not supplied or supported by Red Hat. But packages available in EPEL usually work well with RHEL.

7. Stop the currently running X server by changing to run level 3.

This is required by the nvidia installer. This will kick off anyone who is logged in at the video console. So check to see if anyone is logged in at ":0" which is the video console. You can use the 'w' command for this. Here is an example of someone logged in at the video console:

```
ksa@ksa-linux01 $ w
USER    TTY    FROM    LOGIN@   IDLE     JCPU   PCPU    WHAT
ksa     tty1   :0      Tue07    27:54m   0.00s  0.04s   pam: gdm-password
```

If no one is logged in at ":0", or you get the OK to stop the graphical X server, then switch to runlevel 3:

```
$ sudo init 3
```

8. Run the installer (this installs and builds without any questions):

```
$ sudo /bin/sh ./NVIDIA-Linux-x86_64-[version].run --accept-license --dkms --run-nvidia-xconfig --no-questions
```

The installer will build any required kernel modules. This could take quite a while (several minutes to 10s of minutes).

If you prefer to have an interactive installation, so you can read and answer the questions during the install:

```
$ sudo /bin/sh ./NVIDIA-Linux-x86_64-[version].run
```

9. Next: reboot (or just switch back to run level 5).

A reboot is optional – but recommended so you can verify that the graphical login (GDM) starts after a reboot.
If you cannot reboot now, just restart GDM by switching back to run level 5.

```
$ sudo reboot
or
$ sudo init 5
```

Since there are two distinct parts to the nvidia and nouveau software – user mode (X11 driver) and kernel mode (kernel driver), there may be circumstances where a reboot is required.

There also may be rare circumstances where blacklisting the nouveau kernel module is required (which includes creating the appropriate configuration in /etc/modprobe.d/, rebuilding the initial ram disk (initrd) using the dracut command, and adding 'blacklist=nouveau' to the /boot/grub/grub.conf file).

Additional Information below:

The following command will tell you which kernels have the nvidia module installed:

```
$ find /lib/modules | grep nvidia.ko
```

For an explanation of the symbolic links in the weak-updates subdirectory, see the link under the "Kernel Module Weak Updates" References section below.

Verify the nvidia module is being used and the install was successful

Look for gdm in the output from the last ps command below.

```
$ lsmod | grep nvidia
$ ps axuww | grep X
$ ps axuwwf | less
```

There are also nvidia and X server log files which can be viewed.

$ less /var/log/nvidia-installer.log
$ less /var/log/Xorg.0.log

How does a kernel install trigger an nvidia rebuild using DKMS?

```
$ rpm -q --scripts kernel
  -> /sbin/new-kernel-pkg
    -> /etc/kernel/postinst.d/dkms
      -> /usr/lib/dkms/dkms_autoinstaller
        -> /usr/sbin/dkms
```

Taylor runs during the night, and taylor runs 'yum upgrade' (this is for RHEL 5 and RHEL 6 only. For CentOS 7, Chef is used instead). When there is a new kernel available, yum will install it. When it gets installed, the postinstall scripts in the RPM are run. Many things are run in the these postinstall scripts, including a script called '/sbin/new-kernel-pkg'. /sbin/new-kernel-pkg looks in the /etc/kernel/postinst.d/ directory and runs anything in there. /sbin /new-kernel-pkg finds a script called dkms inside the /etc/kernel/postinst.d/ directory. The /etc/kernel/postinst.d/dkms script runs the /usr/lib/dkms /dkms_autoinstaller script, which in turn runs the /usr/sbin/dkms script. The dkms man page describes in detail how the build happens with the /usr/sbin /dkms script.

A side note: if you want to get notified anytime a new kernel is installed via RPM on a certain host, you can write a script and put it in the /etc/kernel /postinst.d/ directory. When a new kernel is installed, the /sbin/new-kernel-pkg script will look in /etc/kernel/postinst.d/ and run anything it finds there (with 2 arguments supplied). This is a script I've installed on a machine, so I get an email whenever a new kernel is installed there:

```
cat /etc/kernel/postinst.d/notify
#!/bin/sh

cat <<EOF | mail -r unix-admin@slac.stanford.edu -s "$1 installed on `hostname`" ksa@slac.stanford.edu
New kernel was installed on `hostname` at `date`.

$1
$2
EOF
```

Log file of automatic nvidia build:

DKMS logs are saved here:

```
$ ls -l /var/lib/dkms/nvidia/
total 4
drwxr-xr-x 5 root bin 4096 Jan 17 02:22 410.78
lrwxrwxrwx 1 root bin   40 Jan 17 02:22 kernel-2.6.32-754.10.1.el6.x86_64-x86_64 -> 410.78/2.6.32-754.10.1.el6.
x86_64/x86_64
lrwxrwxrwx 1 root bin   39 Dec  7 15:45 kernel-2.6.32-754.3.5.el6.x86_64-x86_64 -> 410.78/2.6.32-754.3.5.el6.
x86_64/x86_64
lrwxrwxrwx 1 root bin   39 Dec  7 15:51 kernel-2.6.32-754.9.1.el6.x86_64-x86_64 -> 410.78/2.6.32-754.9.1.el6.
x86_64/x86_64
```

And here is the log for the automatic dkms build of the nvidia driver which was triggered by the install of the kernel-2.6.32-754.10.1.el6.x86_64 RPM:

```
$ ls -l /var/lib/dkms/nvidia/410.78/2.6.32-754.10.1.el6.x86_64/x86_64
total 8
drwxr-xr-x 2 root bin 4096 Jan 17 02:22 log
drwxr-xr-x 2 root bin 4096 Jan 17 02:22 module
```

References:

DKMS

- See the dkms manual page, using the command: 'man dkms'
- https://en.wikipedia.org/wiki/Dynamic_Kernel_Module_Support
- https://github.com/dell/dkms

Kernel Module Weak Updates

- https://trapsink.com/wiki/Kernel_Module_Weak_Updates

Nouveau

- https://nouveau.freedesktop.org/

Nvidia and linux

- Good background and explanation about nvidia and linux. Ignore the distribution specific (gentoo) instructions which are not relevant.
  https://wiki.gentoo.org/wiki/NVidia/nvidia-drivers