Configuring and Compiling the Linux Kernel

Objective: Configure and compile a Linux Kernel into an installable RPM.

1. With the buildroot correctly set up, it's time to modify the kernel configuration (optional).

    - **cd /home/student/rpmbuild**
    - **cd cd BUILD/kernel-3.10.0-123.el7/linux-`uname -r`** (-r stands for Kernel release)

File(s) for this lab:

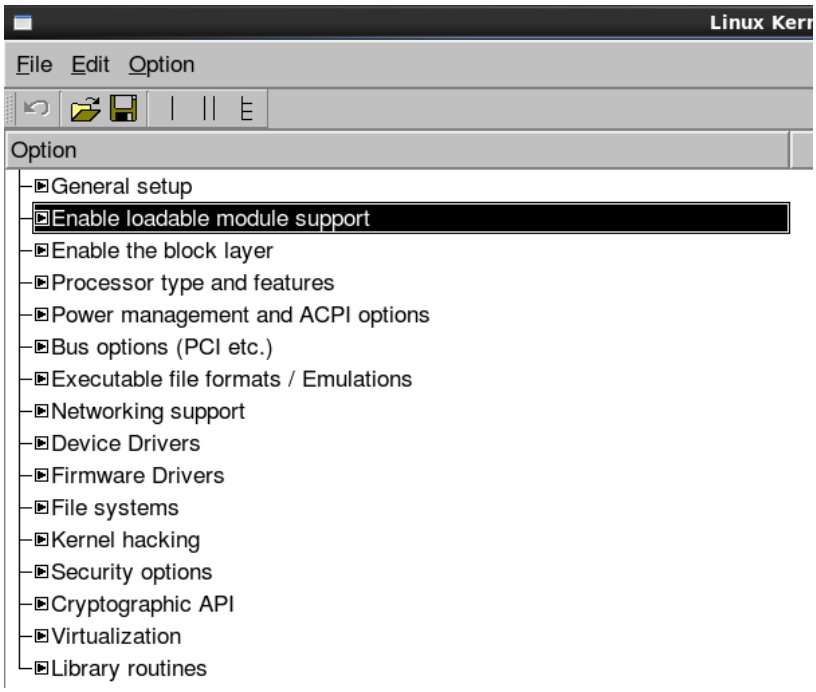
```
[user@localhost rpmbuild]$ cd BUILD/kernel-3.10.0-123.el7/linux-`uname -r`
[user@localhost linux-3.10.0-123.el7.x86_64]$
```

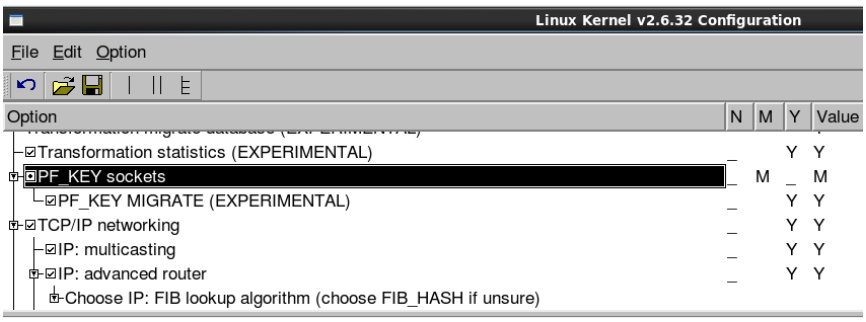
The XConfig menu system requires qt3 and qt3-devel packages:

- **sudo yum install qt3 qt3-devel libXi-devel**
- **make xconfig** (make gconfig or make menuconfig)

2. Alternatively, you can use the current running kernel's configuration file (for this walkthrough use this step).

   - **cp /boot/config-`uname -r` .config**
   - then **"make xconfig"** to load and edit that files configuration





Spend a few minutes and navigate and get familiar with the basic structure and options available for CentOS Kernel customization.

3. Copy the entire contents of the “configs/” directory to the “/student/src/redhat/SOURCES/“ directory.

    - **sudo cp configs/* /home/student/rpmbuild/SOURCES**

4. Navigate to “**/home/student/rpmbuild/SPECS/**” and edit the file **kernel.spec** (nano, gedit, vi, ..etc.)

5. Search for “buildid” (nano CTRL+W, vi :/buildid). The definition of buildid is commented out. For custom kernels this must be uncommented and given a value to avoid a conflict with your currently installed kernel. Change the line in similar manner to the example below:

```
#
# Polite request for people who spin their own kernel rpms:
# please modify the "buildid" define in a way that identifies
# that the kernel isn't the stock distribution kernel, for example,
# by setting the define to ".local" or ".bz123456"
#
%define buildid .anrc
#
```

Since we are not building a custom kernel we can leave this commented out.

```
[user@localhost SOURCES]$ ls -alrt *.patch
-rw-r--r--. 1 user user 1 Jan 29 08:34 linux-kernel-test.patch
[user@localhost SOURCES]$ nano linux-kernel-test.patch
[user@localhost SOURCES]$
```

6. Now we are ready to build the kernel based on our configuration and patch options. Change directory to the /home/student/rpmbuild/SPECS folder and and execute the command:

    - **rpmbuild -bb --target=`uname -m` kernel.spec**

```
[user@localhost linux-3.10.0-123.el7.x86_64]$ cd ~/rpmbuild/SPECS
[user@localhost SPECS]$ gedit kernel.spec &
[1] 64193
[user@localhost SPECS]$ rpmbuild -bb --target=`uname -m` kernel.spec
Building target platforms: x86_64
Building for target x86_64
Executing(%prep): /bin/sh -e /var/tmp/rpm-tmp.DFRQ2q
+ umask 022
+ cd /home/user/rpmbuild/BUILD
+ patch_command='patch -p1 -F1 -s'
+ cd /home/user/rpmbuild/BUILD
+ rm -rf kernel-3.10.0-123.el7
+ /usr/bin/mkdir -p kernel-3.10.0-123.el7
+ cd kernel-3.10.0-123.el7
+ /usr/bin/tar -xf -
+ /usr/bin/xz -dc /home/user/rpmbuild/SOURCES/linux-3.10.0-123.el7.tar.xz
+ STATUS=0
+ '[' 0 -ne 0 ']'
+ /usr/bin/chmod -Rf a+rX,u+w,g-w,o-w .
```

7. When the build completes (this could be a couple of hours depending on your computer), your custom kernel rpm files will be found in the **"/home/student/rpmbuild/RPMS/`uname -m`/"** directory.

```
Requires(rpmlib): rpmlib(FileDigests) <= 4.6.0-1 rpmlib(PayloadFilesHavePrefix) <= 4.0-1 rpmlib
Checking for unpackaged file(s): /usr/lib/rpm/check-files /home/user/rpmbuild/BUILDROOT/kernel-
Wrote: /home/user/rpmbuild/RPMS/x86_64/kernel-3.10.0-123.el7.x86_64.rpm
Wrote: /home/user/rpmbuild/RPMS/x86_64/kernel-headers-3.10.0-123.el7.x86_64.rpm
Wrote: /home/user/rpmbuild/RPMS/x86_64/kernel-debuginfo-common-x86_64-3.10.0-123.el7.x86_64.rpm
Wrote: /home/user/rpmbuild/RPMS/x86_64/perf-3.10.0-123.el7.x86_64.rpm
Wrote: /home/user/rpmbuild/RPMS/x86_64/perf-debuginfo-3.10.0-123.el7.x86_64.rpm
Wrote: /home/user/rpmbuild/RPMS/x86_64/python-perf-3.10.0-123.el7.x86_64.rpm
Wrote: /home/user/rpmbuild/RPMS/x86_64/python-perf-debuginfo-3.10.0-123.el7.x86_64.rpm
Wrote: /home/user/rpmbuild/RPMS/x86_64/kernel-tools-3.10.0-123.el7.x86_64.rpm
Wrote: /home/user/rpmbuild/RPMS/x86_64/kernel-tools-libs-3.10.0-123.el7.x86_64.rpm
Wrote: /home/user/rpmbuild/RPMS/x86_64/kernel-tools-libs-devel-3.10.0-123.el7.x86_64.rpm
Wrote: /home/user/rpmbuild/RPMS/x86_64/kernel-tools-debuginfo-3.10.0-123.el7.x86_64.rpm
Wrote: /home/user/rpmbuild/RPMS/x86_64/kernel-devel-3.10.0-123.el7.x86_64.rpm
Wrote: /home/user/rpmbuild/RPMS/x86_64/kernel-debuginfo-3.10.0-123.el7.x86_64.rpm
Wrote: /home/user/rpmbuild/RPMS/x86_64/kernel-debug-3.10.0-123.el7.x86_64.rpm
Wrote: /home/user/rpmbuild/RPMS/x86_64/kernel-debug-devel-3.10.0-123.el7.x86_64.rpm
Wrote: /home/user/rpmbuild/RPMS/x86_64/kernel-debug-debuginfo-3.10.0-123.el7.x86_64.rpm
Executing(%clean): /bin/sh -e /var/tmp/rpm-tmp.5fANo7
+ umask 022
+ cd /home/user/rpmbuild/BUILD
+ cd kernel-3.10.0-123.el7
+ rm -rf /home/user/rpmbuild/BUILDROOT/kernel-3.10.0-123.el7.x86_64
+ exit 0
[user@localhost SPECS]$
```

Note: If you have built a kernel version that is older than a currently installed version you will also have to use the --oldpackage flag with the rpm command.

Warning! **UNDER NO CIRCUMSTANCES** use a rpm -Uvh command to install your kernel as this will update (overwrite) the currently installed version. Hence if you have a problem with your custom kernel, you will not be able to revert to the previous, working, version.