ANRC

# RedHat Linux Kernel Internals Laboratory Exercises

## *Lab 2: Configuring and Compiling the Linux Kernel*

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

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

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

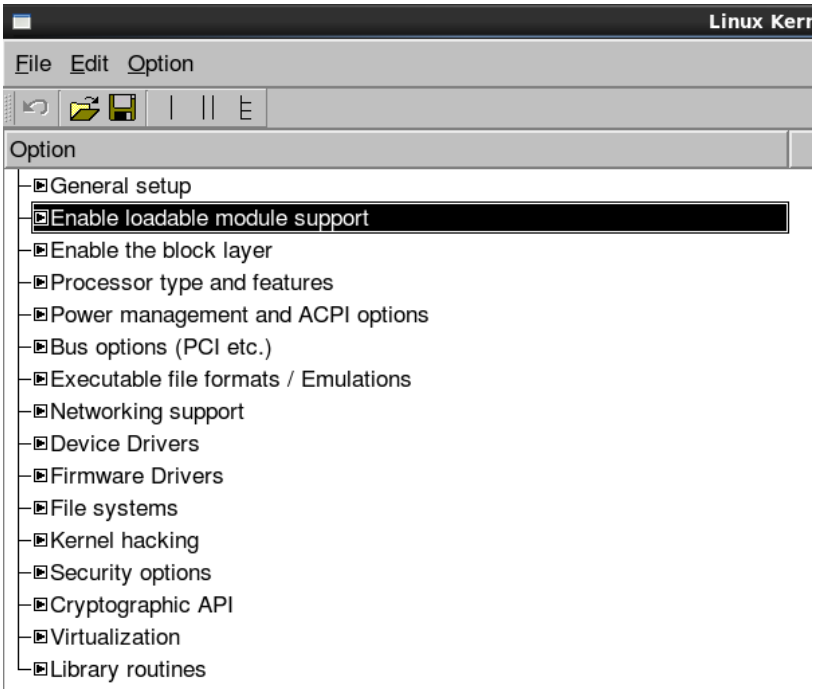
```
[user@localhost rpmbuild]$ cd BUILD/kernel-3.10.0-123.el7/linux-`uname -r`
[user@localhost linux-3.10.0-123.el7.x86_64]$
```

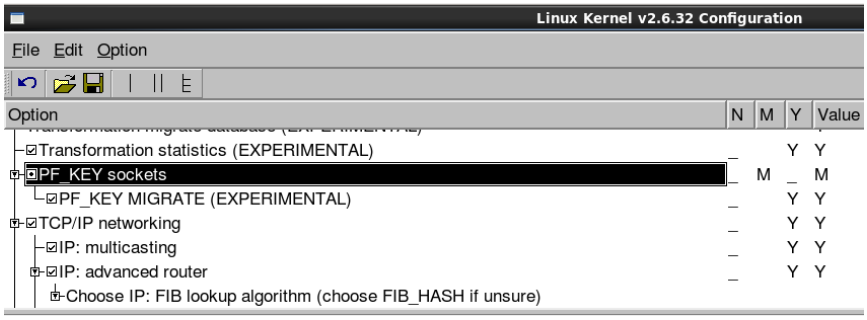
    The XConfig menu system requires qt3 and qt3-devel packages:

    - **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 RHEL Kernel customization.

3. For this lab skip this step: Copy the edited configuration back into the config directory.

    - **sudo cp .config configs/kernel-3.10.0-x86_64.config**

```
[user@localhost linux-2.6.32-431.el6.x86_64]$ cp .config configs/kernel-2.6.32-`uname -m`.config
[user@localhost linux-2.6.32-431.el6.x86_64]$ ls -alrt configs/kernel-2.6.32-x86_64*
-rw-r--r--. 1 user user 105513 May 12 18:24 configs/kernel-2.6.32-x86_64-debug.config
-rw-r--r--. 1 user user 105195 May 12 19:25 configs/kernel-2.6.32-x86_64.config
[user@localhost linux-2.6.32-431.el6.x86_64]$
```

4. Copy the entire contents of the “configs/” directory to the “/usr/src/redhat/SOURCES/“ directory.

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

5. Navigate to "**/home/user/rpmbuild/SPECS/**" and edit the file **kernel.spec** (nano, gedit, vi, ..etc.)

6. 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.

7. For this lab skip this step: If you are developing a custom patch, you need to add that file after default patch list. Search for "Patch999", add your declaration starting with the number 40000 before this line, so that your patch is not in any danger of conflicting with the RHEL kernel patch space. Look for:

```
Source84: config-s390x-generic-rhel
Source85: config-powerpc64-debug-rhel
Source86: config-s390x-debug-rhel

# My Custom Patches
Patch40000: my-awesome-kernel.patch

# empty final patch file to facilitate testing of kernel patches
Patch999999: linux-kernel-test.patch
```

- ex. **Patch40000: my-awesome-kernel.patch**

Add your patch file to the SOURCES directory in order for the "ApplyPatch" function to locate and apply it:

```
[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]$
```

Since we are not doing any direct kernel patching we can safely skip these steps.

8. For this lab skip this step: Save the kernel.spec file (nano CTRL+O, vi :wq!)

9. Now we are ready to build the kernel based on our configuration and patch options. Change directory to the /home/user/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 .
```

10. 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/user/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]$
```

11. Skip this step for this lab: Make sure that you install those files, as root, using a rpm -ivh kernel-*.rpm command.

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.