4. Modifying the kernel specification file

The kernel specification file should now be modified. This will allow you do determine which kernel you are booting when you see the boot menu.

```
NOTE: You only need to do this ONE time. You only need to re-do it if you want to create
a new kernel with a different ID. If this is NOT the first time, go to the end of patch
line.
```

The line numbers mentioned in this section relate only to the **current** CentOS kernel specification file.

First, we have to save the “build specifications” file. Then we can modify it later. Instead of “nano”, you can use your favorite text editor.

```
[user@host]$ cd ~/rpmbuild/SPECS/
[user@host SPECS]$ cp kernel.spec kernel.spec.distro
[user@host SPECS]$ nano kernel.spec
```

At line 18 in *kernel.spec*, for CentOS-**6**, (for CentOS-7 it is line **8**) the definition of *buildid* is commented out. This must be uncommented and given a value to avoid a conflict with your currently installed kernel. Change the line in a manner *similar* to the example below:

```
%define buildid  .your_identifier
```

Leave no space between the "%" and "define". I used my initials (djf) and a version# as my *identifier*. Note the period (the dot) preceding *your* identifier!! Keep it!!

If you have any patches to apply, you need to make reference to them in **two** places. If you do *not* have any patches to apply, then just proceed to the line below that reads "End of the patch section" and to change the *kernel.spec* file. To create your patches see: Creating the patch files.docx. Note that you don’t have to do ALL of the steps that are there. READ the instructions as you go through that file, then continue below.

RESUME HERE (from creating your patch files):

End of the patch section. Now continue changing the *kernel.specs* file *as needed* and save it with the same name.

For CentOS-**6** only: do the following 2 things in your *kernel.spec* file:

1. locate a line that starts with
   "# Drop some necessary files", then **replace** the following line:

   ```
   cp $RPM_SOURCE_DIR/config-* .
   ```

   with the following line (DON’T forget the ending *dot* as highlighted) and leave the * as it is:

   ```
   cp $RPM_SOURCE_DIR/kernel-*.config .
   ```

2. Locate the following line and comment it out by putting a “#” (without the “”s) in front of it:

   ```
   make -f %{SOURCE20} VERSION=%{version} configs
   ```

All users: If you have changed the kernel.spec file, save it now.

5. Building & Installing the new kernel

Start the build: (Be careful to use the **backward** quote signs!!!) The rpmbuild command is one continuous line. FIRST get to the rpmbuild/SPECS folder, then do the rpmbuild. The rpmbuild command below can be safely copied and pasted into your command line.

```
[user@host SOURCES]$ cd  ~/rpmbuild/SPECS

[user@host SPECS]$ rpmbuild  -bb --without xen   --without debug   --without debuginfo
--without kabichk   --without fips --target=`uname -m` kernel.spec 2> build-err.log | tee
build-out.log
```

The “--without” options make it a smaller kernel and builds it faster.

SKIP the remainder of this step and continue with step 6.

For those who are experienced: for kernels >= 2.6.18-53.el5, you can add some other useful options to the *rpmbuild* command by using the --with and/or --without flags and associated arguments. The main options to note are:

```
--with baseonly

--with xenonly

--without up
```

For example, to build just the base kernel packages use just these options:

```
--with baseonly --without debug --without debuginfo
```

6. (Resume here) Installing the new kernel (make it bootable)

When the build completes, all your custom kernel rpm files will be found here
(note the backquotes) ~/rpmbuild/RPMS/`uname  -m`

For the TJW system, `uname -m` will be replaced with s390x
For VMWare systems, `uname -m` will be replaced with x86_64

Install your custom kernel files as **root**, by using the 'yum' command below.

6.1. Using the yum command (preferred)

To install a kernel-* package (where * represents the *specific* kernel you want to install), use the command below, but REPLACE *my* initials with YOUR *buildid* as used in the changes to *kernel.spec* in step 4 above.

```
For TJW:
[root@host SPECS]$ yum localinstall kernel-2.6.32-504.23.4.cl6.djf2.s390x.rpm

For VMWare:
[root@host SPECS]$ yum localinstall kernel-3.10.0-514.26.2.el7.djf2.x86_64.rpm
```

6.2. (You should not need this, if 6.1 worked) Using the rpm command (alternate)

When installing the kernel-* packages using the rpm command, you should note that 'kernel' and 'kernel-devel' must be **installed**, not updated, by using an *rpm -ivh* 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.

NEVER use an *rpm **-U**vh* 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. (i.e.; you will be stuck with your non-working system).

FYI: Packages *other than* 'kernel' and 'kernel-devel' DO need to be installed (updated) with an *rpm -Uvh* command. (We won’t be doing that.) Please note the difference in the flag between *-i* (install) and *-U* (update).