Building a customised Linux kernel for the RPI

A customised Linux kernel can be built with just the functionality really needed and without all the unnecessary stuff such as a windowed OS, audio, video etc etc.

A really simple tool that allows you to define and create a customised kernel is buildroot. Buildroot is a set of Makefiles and configurations together with a number of predefined system configs, for example Beagleboard configs, RPI etc etc.

1) To use buildroot and create a build directory suitable for multiple board types you will need to install buildroot and then:
2) cd to the buildroot directory
3) create the sub directories for the boards you wish to configure, eg mkdir RPI3
4) cd to the created directory and configure the ../Makefile and create the new .config for the target board with something like the following for the RPI3-64:
5) *make -C ../buildroot O=$(pwd) raspberrypi3-64_defconfig*
6) Customise the configuration with *make menuconfig (make xconfig*, your choice).
7) Select the 'system configuration' option and change:
    1) System hostname & system banner
    2) Root password and whether to allow root logins.
    3) Paths to user table, root file system overlays, post build and post image scripts
8) In target packages, enable show packages provided by *Busybox*
9) in target packages→Hardware Handling→Firmware enable *B43 firmware, Broadcom bcm43xxx, rpi-bt-firmware, rpi-firmware, rpi-wifi-firmware*
10) in target packages→Hardware Handling enable *i2c-tools* and *spi-tools*
11) in target packages→Networking applications enable *dropbear* and *tinyhttpd, wireless tools, wpa-supplicant* and enable *nl80211 support*.
12) Create a directory inside *buildroot/board* in which create the kernel-patch, rootf-overlay, users directories.
13) In the users directory create your space delimited users file containig things like:

    ptr -1 PTR -1 !=printer /home/PTR /bin/sh users,operator printer account

    andrew -1 users -1 =PASSWORD /home/andrew /bin/sh users,sudoers Andrew account
14) In the rootfs-overlay directory create the /etc, /var, /usr directories.

15) In the /etc directory create the modules.conf file and wpa_supplicant.conf file. Modules.conf will list the modules to be loaded at startup time. Wpa_supplicant contains the wireless network ssid, psk info to connect to your home wifi.

16) In /etc create the init.d directory create a file to parse modules.conf and load the modules. I use the following:

```
#!/bin/sh
#
# This script loads up a list of drivers at system boot time.
# It will test for the existence of /etc/modules.conf.
# if file exists then read each line one at a time and
# process it with modprobe
# output a log message to the console
#
# lines beginning with # should be discarded
#
#

CONF="/etc/modules.conf"
if  ! [ -f $CONF ]  # file  does not exist
then
        echo 'file does not exist';
        exit 0;

fi
#continue to process the file line by line
while read f
do
LEN=$(echo -n $f | wc -m )
# LEN is th length of the string, dump any 1 char or less
if [ $LEN -gt "2" ] ; then

echo $f | grep "#" > /dev/null 2>&1
if [ "$?" -eq "1" ] ; then
# now modprobe the line read to load the driver
        modprobe -s -v $f

fi
fi

done < $CONF

```

Illustration 1: S00modules script to load modules at startup

17) Create the /var/www directory and put any html pages in there.

18) Make sure that your post-image and post-build scripts as well as the genimage script are all in the RPI3 directory

19) Then *make all* – this takes a long time and will create the target file system and boot images.

20) To put the boot image to a SD card use something like the following: *sudo dd if=images/sdcard.img of=/dev/sdh ; sync ; sync*

21) Put the cd card in the RPI and boot. The console messages should tell you more or less what's going on. Once booted, use a few standard commands to see how it all went eg:
    1) ls / to see if the /home directory has been created. This indicates that the users file was processed. Ls /home lists the users that have home directories created.
    2) Cat /etc/passwd lists the users, again to see if they have been created
    3) lsmod to see if the modules listed in modules.conf have been loaded
    4) ifconfig to see if the wireless network is up and running
    5) ps -e to see if dropbox and tinyhttpd are running
    6) browse to the address given in ifconfig to see if the index.html page is displayed
    7) ssh to one of the accounts created
    8) etc