Building an image: <http://stackoverflow.com/questions/10814919/how-to-choose-target-and-other-features-in-openwrt-buildroot>

Few [common mistakes](https://forum.openwrt.org/viewtopic.php?pid=36636#p36636) that people make while working with OpenWrt.

CROSS\_COMPILATION

Embedded systems use a different processor and require a cross-compilation toolchain - a compilation toolchain that runs on a host system but that generates code for a target system (and target processor's instruction set architecture (ISA)). For example, if your host system uses x86 and your target system uses MIPS32, the regular compilation toolchain of your host runs on x86 and generates code for x86, while the cross-compilation toolchain runs on x86 and generates code for MIPS32.

To use a program that is not already compile for Openwrt one can cross-compile it for OpenWrt using the [Openwrt-Buildroot](http://wiki.openwrt.org/about/toolchain). It is a highly modified version of standard [buildroot](http://www.buildroot.org/). This tutorial is aimed at explaining how the cross-compilation can be done on a linux machine (Ubuntu to be specific). There is also a link to a tutorial for Hello, World ! for OpenWrt at the end of this document.  
  
Mainly what is required to cross-compile any package for an embedded system running OpenWrt apart from the source code is an OpenWrt Makefile. It looks a bit different from a [conventional makefile](http://mrbook.org/blog/tutorials/make/) and it defines the steps needed to download and install the package.   
  
Steps involved:   
1) Copy the source code of the program in a folder placed in the build root directory. barrier\_breaker in our case.  
2) If the program has any dependencies make sure those dependencies are already compiled by OpenWrt. Many libraries are already included in OpenWrt source code and if required one can choose them by running ‘make menuconfig’ then selecting the particular library inside the Libraries section. In case you can’t find it there and you need to add it manually then it’s like adding any other package to OpenWrt for which this tutorial is anyways.   
3) Create a folder feeds/packages/utils/name\_of\_the\_program and place the OpenWrt makefile for that program there.   
  
Sample OpenWrt Makefile for ‘capwap’: (found on internet) (Do mind the indentations while copying it ! )   
------------------------------------------------------------------------------------------------------------------------------------

# Copyright (C) 2009 OpenWrt.org

# This is free software, licensed under the GNU General Public License v2.

# See /LICENSE for more information.

# $Id: Makefile $

# OpenWrt package for Open Source CAPWAP

# Prepared by Michael Smith, McGill University

include $(TOPDIR)/rules.mk

PKG\_NAME\_SHORT:=capwap

PKG\_NAME:=$(PKG\_NAME\_SHORT)

PKG\_RELEASE:=4

PKG\_VERSION=1.0.0

PKG\_BUILD\_DIR := $(BUILD\_DIR)/$(PKG\_NAME)-$(PKG\_VERSION)

include $(INCLUDE\_DIR)/package.mk

define Package/capwap/Default

SECTION:=net

CATEGORY:=Network

endef

define Package/capwap/Default/description

The project is an implementation of CAPWAP agents for WTPs and ACs.

endefdir -p $(PKG\_BUILD\_DIR)

$(CP) /home1/mayank/Desktop/Openwrt/barrier\_breaker/capwap-0.93.3/\* $(PKG\_BUILD\_DIR)/

endef

define Package/capwap

$(call Package/capwap/Default)

TITLE:=open source capwap

DEPENDS:=+libopenssl +libpthread

endef

define Package/capwap/description

$(call Package/capwap/Default/description)

endef

define Build/Prepare

mk

define Package/capwap/install

$(INSTALL\_DIR) $(1)/usr/capwap/

# $(INSTALL\_DATA) $(PKG\_BUILD\_DIR)/\* $(1)/usr/capwap/

$(INSTALL\_BIN) $(PKG\_BUILD\_DIR)/AC $(1)/usr/capwap/

$(INSTALL\_BIN) $(PKG\_BUILD\_DIR)/WTP $(1)/usr/capwap/

$(INSTALL\_BIN) $(PKG\_BUILD\_DIR)/WUA $(1)/usr/capwap/

endef

$(eval $(call BuildPackage,capwap))  
-----------------------------------------------------------------------------------------------------------------------------------

The makefile is quite self-explanatory but still for the uninitiated here is the line by line explanation :

* The rules.mk file is present in the main source code directory (barrier\_breaker in our case). TOPDIR is defined in the Makefile of OpenWrt which is present in the barrier\_breaker folder in our case and which is where we run all the make commands from.
* BUILD\_DIR is defined in rules.mk that is included at the top. So is the INCLUDE\_DIR
* define Package/capwap/Default tells where can we find the capwap package in menuconfig.

SECTION - The type of package (currently unused)

CATEGORY - Which menu it appears in menuconfig

TITLE - A short description of the package

DESCRIPTION - (deprecated) A long description of the package

URL - Where to find the original software

MAINTAINER - (required for new packages) Who to contact concerning the package

DEPENDS - (optional) Which packages must be built/installed before this package. See [here](http://wiki.openwrt.org/doc/devel/packages#dependencytypes) for the syntax.

USERID - (optional) a username:groupname pair to create at package installation time.

* ‘mkdir - p’ --parents no error if existing, make parent directories as needed
* DEPENDS:= line is a way of using dependencies. +libopenssl and +libpthread mean that whenever capwap is selected in menuconfig these packages will be automatically selected. To know more check [this](http://wiki.openwrt.org/doc/devel/dependencies).

4) Now in the Makefile of the program (located in barrier\_breaker/capwap/ directory in our case) make changes to it to change the location of dependencies to the toolchain folder. **To check the toolchain folder for your device go to downloads.openwrt.org and find the toolchain at the top of the index where the OpenWrt binary for your specific router is located.**

eg. in our case we made the following changes to the capwap Makefile (present in barrier\_breaker/capwap/ folder) .   
  
LDFLAGS = /home1/mayank/Desktop/Openwrt/barrier\_breaker/build\_dir/target-mipsel\_24kec+dsp\_uClibc-0.9.33.2/openssl-1.0.1j/libssl.a /home1/mayank/Desktop/Openwrt/barrier\_breaker/build\_dir/target-mipsel\_24kec+dsp\_uClibc-0.9.33.2/openssl-1.0.1j/libcrypto.a -lpthread -ldl -D\_REENTRANT

CFLAGS = -Wall -g -O0 -D\_REENTRANT -I/home1/mayank/Desktop/Openwrt/barrier\_breaker/build\_dir/target-mipsel\_24kec+dsp\_uClibc-0.9.33.2/openssl-1.0.1j/include/ #-DCW\_NO\_DTLS

5) Once you have the makefile, the steps needed to build the package are as follows:

1. ./scripts/feeds update -i
2. ./scripts/feeds install NetlinkServer
3. make menuconfig
4. make package/capwap/{clean,compile,install} V=s

If the dependencies that you require also need to be compiled for OpenWrt follow the same process.

Building a new package with OpenWRT

Package information

Basefile written in C. Dependency on libnl-3 libnl-genl-3.

Steps:

1. Go to the openwrt buildroot (barrier\_breaker in my case)

2. Copy your files into a directory in buildroot. Your package should contain the Makefile to make your program.

3. Taking care of dependencies:

a. Now this is the tricky part. Firstly, the base buildroot of openwrt does not include libnl-3. So we have to include libnl-3 first. To do this, we need the libnl-3.2.21.tar.gz and corresponding Makefile. Both can be found on net. (the process is same for any other dependency you have to install)

(find the files here and here)

- create a folder libnl-3 inside feeds/package/libs/ folder.

- keep libnl-3.2.21.tar.gz file in dl/ directory in the buildroot

- run

$> ./scripts/feed update -a

(this will update the feed folder)

$> ./scripts/feed install -a

(this will install the libnl-3)

$> make menuconfig

- when menuconfig opens, go to libraries option and select libnl-3

- now run

$> make package/libnl-3/compile V=s

Now the libnl-3 is compiled

4. Now we have to make changes to Makefile of our program

First lets add library dependencies. In your Makefile, change -libnl -lib-nl or -lib-genl to the exact location of .a files.

In our case here, libnl-3.a and lib-genl-3.a files must be present in the build\_root/<target\_mips>/libnl-3/ipkg-install/usr/lib/ directory. Add this to the gcc command to add the libraries. Also include -lpthread -ldl -lm

Second, add the include dir. In our case it is

-I<path\_here>/build\_dir/<target-mips>/libnl-3.2.21/include/

So our makefile will look something like..

################################################################################

# Automatically-generated file. Do not edit!

################################################################################

-include ../makefile.init

RM := rm -rf

# All of the sources participating in the build are defined here

O\_SRCS :=

C\_SRCS :=

S\_UPPER\_SRCS :=

OBJ\_SRCS :=

ASM\_SRCS :=

OBJS :=

C\_DEPS :=

EXECUTABLES :=

# Every subdirectory with source files must be described here

SUBDIRS := \

. \

# Add inputs and outputs from these tool invocations to the build variables

C\_SRCS += \

./netlinkTest.c

OBJS += \

./netlinkTest.o

C\_DEPS += \

./netlinkTest.d

CC = mips-openwrt-linux-uclibc-gcc

# Each subdirectory must supply rules for building sources it contributes

%.o: ./%.c

@echo 'Building file: $<'

@echo 'Invoking: GCC C Compiler'

$(CC) -O0 -g3 -Wall -c -fmessage-length=0 -MMD -MP -MF"$(@:%.o=%.d)" -MT"$(@:%.o=%.d)" -o"$@" "$<" -I/home1/mayank/Desktop/Openwrt/barrier\_breaker/build\_dir/target-mipsel\_24kec+dsp\_uClibc-0.9.33.2/libnl-3.2.21/include/

@echo 'Finished building: $<'

@echo ' '

USER\_OBJS :=

#LIBS := -lnl-3 -lnl-genl-3

LIBS := -I/home1/mayank/Desktop/Openwrt/barrier\_breaker/build\_dir/target-mipsel\_24kec+dsp\_uClibc-0.9.33.2/libnl-3.2.21/include/

LDFLAGS = /home1/mayank/Desktop/Openwrt/barrier\_breaker/build\_dir/target-mipsel\_24kec+dsp\_uClibc-0.9.33.2/libnl-3.2.21/ipkg-install/usr/lib/libnl-3.a /home1/mayank/Desktop/Openwrt/barrier\_breaker/build\_dir/target-mipsel\_24kec+dsp\_uClibc-0.9.33.2/libnl-3.2.21/ipkg-install/usr/lib/libnl-genl-3.a -lpthread -ldl -lm -D\_REENTRANT

ifneq ($(MAKECMDGOALS),clean)

ifneq ($(strip $(C\_DEPS)),)

-include $(C\_DEPS)

endif

endif

-include ../makefile.defs

# Add inputs and outputs from these tool invocations to the build variables

# All Target

all: wtpAccess

# Tool invocations

wtpAccess: $(OBJS) $(USER\_OBJS)

@echo 'Building target: $@'

@echo 'Invoking: GCC C Linker'

$(CC) -o "wtpAccess" $(OBJS) $(USER\_OBJS) $(LIBS) $(LDFLAGS)

@echo 'Finished building target: $@'

@echo ' '

# Other Targets

clean:

-$(RM) $(OBJS)$(C\_DEPS)$(EXECUTABLES) wtpAccess

-@echo ' '

.PHONY: all clean dependents

.SECONDARY:

-include ../makefile.targets

5. now create the feed makefile

- make a folder <wtpAccess> in feeds/package/utils/

- Create Makefile (should look as shown below)

include $(TOPDIR)/rules.mk

# Name and release number of this package

PKG\_NAME:=wtpAccess

PKG\_RELEASE:=1

# This specifies the directory where we're going to build the program.

# The root build directory, $(BUILD\_DIR), is by default the build\_mipsel

# directory in your OpenWrt SDK directory

PKG\_SOURCE:=wtpAccess.tar.gz

#PKG\_SOURCE\_URL:=http://www.infradead.org/~tgr/libnl/files/

#PKG\_MD5SUM:=6fe7136558a9071e70673dcda38545b3

PKG\_BUILD\_DIR := $(BUILD\_DIR)/$(PKG\_NAME)

include $(INCLUDE\_DIR)/package.mk

# Specify package information for this program.

# The variables defined here should be self explanatory.

define Package/wtpAccess

SECTION:=utils

CATEGORY:=Utilities

TITLE:=WTP\_ACCESS -- prints a snarky message

DEPENDS:=+libnl-3

endef

define Package/wtpAccess/description

This package is made for openCAPWAP software to obtain WiFi radio and station information.

endef

# Specify what needs to be done to prepare for building the package.

# In our case, we need to copy the source files to the build directory.

# This is NOT the default. The default uses the PKG\_SOURCE\_URL and the

# PKG\_SOURCE which is not defined here to download the source from the web.

# In order to just build a simple program that we have just written, it is

# much easier to do it this way.

define Build/Prepare

/bin/rm -rf $(PKG\_BUILD\_DIR)/

mkdir -p $(PKG\_BUILD\_DIR)

#tar -xvf $(TOPDIR)/dl/$(PKG\_SOURCE) -C $(PKG\_BUILD\_DIR)/

cp -r $(TOPDIR)/wtpAccess/\* $(PKG\_BUILD\_DIR)/

endef

define Build/Compile

$(call Build/Compile/Default)

make -C $(PKG\_BUILD\_DIR)

endef

# We do not need to define Build/Configure or Build/Compile directives

# The defaults are appropriate for compiling a simple program such as this one

# Specify where and how to install the program. Since we only have one file,

# the helloworld executable, install it by copying it to the /bin directory on

# the router. The $(1) variable represents the root directory on the router running

# OpenWrt. The $(INSTALL\_DIR) variable contains a command to prepare the install

# directory if it does not already exist. Likewise $(INSTALL\_BIN) contains the

# command to copy the binary file from its current location (in our case the build

# directory) to the install directory.

define Package/wtpAccess/install

$(INSTALL\_DIR) $(1)/bin

$(INSTALL\_BIN) $(PKG\_BUILD\_DIR)/wtpAccess $(1)/bin/

endef

# This line executes the necessary commands to compile our program.

# The above define directives specify all the information needed, but this

# line calls BuildPackage which in turn actually uses this information to

# build a package.

$(eval $(call BuildPackage,wtpAccess))

6. Now run

$> ./scripts/feed update -a

$> ./scripts/feed install -a

$> make menuconfig

(Select wtpAccess package)

$> sudo make package/wtpAccess/compile V=s

7. If everything goes well, your .ipk file will be stored in

bin/<ar7xxx>/packages/packages/

Hope this tutorial helps.. Please ask for any questions..