# Client-Server Image for Yocto Project

## Table of Content

---

## Project Description

Project has four phases:

1. Download Build Yocto Project

    - Set up an OpenEmbedded environment
    - Configure the project and choose a target
    - Build your Poky image

2. Creating Your Own Yocto Layer

    - Implement client application using sockets that logs a message “Hello from Yocto” every 5 seconds when connected to server.
    - Implement the server application using sockets.
    - Create a new Yocto layer.
    - Interface this custom layer to the existing Yocto project.
    - Add your client application to your layer.
    - Add your server application to your host machine.
    - Build your image.

3. Make kernel automatically starting your client application.

4. (Bonus) Build a full embedded linux image using yocto for raspberry pi board (or any high end board ) and boot image on this board.

---

## Creation Steps

# Setup the Environment

1. Create starter structure of the project directory

```
mkdir client-server-yocto-project
cd client-server-yocto-project
mkdir sources
```

2. Clone required repositories to the `sources` directory

```
cd sources
git clone git://git.yoctoproject.org/poky -b kirkstone
git clone https://git.yoctoproject.org/meta-raspberrypi/ -b kirkstone
git clone git://git.openembedded.org/meta-openembedded -b kirkstone
cd ..
```

> *Note 1: you can chose different yocto release from [Yocto Releases Wiki Page (https://wiki.yoctoproject.org/wiki/Releases)](https://wiki.yoctoproject.org/wiki/Releases)*

1. Install and activate python v3.8 as requested for `kirkstone` release

```
pyenv install 3.8
pyenv local 3.8
python --version
```

> *Note 2: make sure you have the same python version based on the selected release to avoid issues during building process*

4. Setup build environment and required commands

```
source sources/poky/oe-init-build-env
```

# Create Custom Layer

1. Create custom layer for the client application called `meta-client` and add it to bitbake layers

```
cd build
bitbake-layers create-layer ../meta-client
bitbake-layers add-layer ../meta-client
```

the `build/bblayers.conf` should look like the following:

```
# POKY_BBLAYERS_CONF_VERSION is increased each time build/conf/bblayers.conf
# changes incompatibly
POKY_BBLAYERS_CONF_VERSION = "2"

BBPATH = "${TOPDIR}"
BBFILES ?= ""

BBLAYERS ?= " \
  /{PROJECT_LOCATION}/client-server-yocto-project/sources/poky/meta \
  /{PROJECT_LOCATION}/client-server-yocto-project/sources/poky/meta-poky \
  /{PROJECT_LOCATION}/client-server-yocto-project/sources/poky/meta-yocto-bsp \
  /{PROJECT_LOCATION}/client-server-yocto-project/meta-client \
  "
```

2. Delete `recipes-example` directory and create `recipes-core` for image, and `recipes-packages` for custom packages

```
cd ../meta-client
rm -rf recipes-example
mkdir recipes-core
mkdir recipes-packages
```

3. Create a recipe for the image inherits from the `core-minimal-image`, which contains the client package

```
mkdir recipes-core/images
nano recipes-core/images/client-image.bb
```

the `client-image.bb` content should be:

```
inherit core-image

# Base this image on core-image-minimal
include recipes-core/images/core-image-minimal.bb

# Include modules in rootfs
IMAGE_INSTALL += " \
        client \
"
```

4. Create a recipe for the client package, defines the procedures of compile the source code, setup init startup services, and set the initial configuration

```
mkdir recipes-packages/client
mkdir recipes-packages/client/files # put there client.c, client_service, and client.conf
nano recipes-packages/client/client_0.1.bb
```

the `client_0.1.bb` content should be:

```
DESCRIPTION = "Client application send periodic messages to server"
LICENSE = "MIT"
LIC_FILES_CHKSUM = "file://${COREBASE}/meta/COPYING.MIT;md5=3da9cfbcb788c80a0384361b4de20420"

inherit update-rc.d

SRC_URI = " \
  file://client.c \
  file://client_service \
  file://client.conf \
  "

S = "${WORKDIR}"

do_compile() {
    ${CC} ${CFLAGS} ${LDFLAGS} client.c -o client
}

do_install() {
    # Hook the client to init services
    install -d ${D}${sysconfdir}/init.d
    install -m 0755 client_service ${D}${sysconfdir}/init.d/client_service

    # Install client binary to /usr/bin directory
    install -d ${D}${bindir}
    install -m 0755 client ${D}${bindir}

    # Move initial config file to /etc/ directory
    install -d ${D}${sysconfdir}/client
    install -m 0644 client.conf ${D}${sysconfdir}/client/client.conf
}

INITSCRIPT_NAME = "client_service"
INITSCRIPT_PARAMS = "start 99 1 2 3 4 5 . stop 20 0 1 6 ."
RDEPENDS_${PN} = "initscripts"
CONFFILES_${PN} += "${sysconfdir}/init.d/client_service"
```

## Build for Qemu

1. Set machine name in `build/local.conf` to `qemux86-64`

```
[..]
MACHINE ??= "qemux86-64"
[..]
```

2. Build the image

```
bitbake client-image
```

3. Run the image in qemu

```
runqemu qemux86-64
```

## Build for Raspberry Pi 4

6. Add `meta-raspberrypi`, `meta-oe`, `meta-multimedia`, `meta-networking`, and `meta-python` layer manually or using bitbake

```
cd build
bitbake-layers add-layer ../sources/meta-raspberrypi
bitbake-layers add-layer ../sources/meta-openembedded/meta-oe
bitbake-layers add-layer ../sources/meta-openembedded/meta-python
bitbake-layers add-layer ../sources/meta-openembedded/meta-multimedia
bitbake-layers add-layer ../sources/meta-openembedded/meta-networking
```

the `build/bblayers.conf` should look like the following:

```
# POKY_BBLAYERS_CONF_VERSION is increased each time build/conf/bblayers.conf
# changes incompatibly
POKY_BBLAYERS_CONF_VERSION = "2"

BBPATH = "${TOPDIR}"
BBFILES ?= ""

BBLAYERS ?= " \
  /home/darkknight/Projects/client-server-yocto-project/sources/poky/meta \
  /home/darkknight/Projects/client-server-yocto-project/sources/poky/meta-poky \
  /home/darkknight/Projects/client-server-yocto-project/sources/poky/meta-yocto-bsp \
  /home/darkknight/Projects/client-server-yocto-project/sources/meta-raspberrypi \
  /home/darkknight/Projects/client-server-yocto-project/meta-client \
  /home/darkknight/Projects/client-server-yocto-project/sources/meta-openembedded/meta-oe \
  /home/darkknight/Projects/client-server-yocto-project/sources/meta-openembedded/meta-python \
  /home/darkknight/Projects/client-server-yocto-project/sources/meta-openembedded/meta-multimedia \
  /home/darkknight/Projects/client-server-yocto-project/sources/meta-openembedded/meta-networking \
  "
```

2. Set machine name in `build/local.conf` to `raspberrypi4-64`

```
[..]
MACHINE ??= "raspberrypi4-64"
[..]
```

3. Build the image

```
bitbake client-image
```

4. Decompress the image using `bzip2`

```
bzip2 -d -f tmp/deploy/images/raspberrypi4/core-image-sato-raspberrypi4.wic.bz2
```

5. Flash the image to the SD card (make sure you already connect it to the machine)

```
sudo dd if=tmp/deploy/images/raspberrypi4-64/core-image-sato-raspberrypi4-64.rpi-sdimg of=/dev/sdx
```

> *To check the SD card partition run `sudo fdisk -l` and replace x with your sd card partition id*

---

# Build the server application

To build the server application you should have `cc` or `gcc` compiler, and before that don't forget to set the port on the code `SERVER_PORT`

```
cc server.c -o server
```

---

# Configure the client application after building the image

The client application can be configured after building the custom image by using one of the following methods

1. [Method 1] Modify the config file `/etc/client/client.conf` with the server info

```
echo "SERVER_IP 192.168.1.8" > /etc/client/client.conf
echo "SERVER_PORT 1236" >> /etc/client/client.conf
```

the file content will be:

```
SERVER_IP 192.168.1.8
SERVER_PORT 1236
```

1. [Method 2] Use client options

```
client --ip 192.168.1.8 --port 1236
```

2. Restart the client service to close the current connection, and open new one with the new config

```
service client_service stop
service client_service start
```