Creating BT PAN/RNDIS router using OpenWrt

Koichi Okamoto, Ishikawa Masayuki / let’s dive into their mechanism!
Biography (Koichi Okamoto)

Senior System Engineer
at Sony Home Entertainment & Sound Products Inc.

Technical background
- LSI upstream Design, Embedded Systems, Network Configurations

Product development
- Car Audio/Visual/Navi (non OS/Linux/Android/uITRON)
- Wireless Speaker/Sound Bar/Voice Assistant Speaker (Linux)

Public talks
- OpenIoT2018NA (sub speaker)
Table of Contents (Two parts)

• Off-the-self hardware using OpenWrt extends BT PAN in addition to USB RNDIS
  – What is Bluetooth (BT) PAN
  – Learning Linux BT mechanisms
  – Overview
  – Hardware/Software Requirement
  – Create OpenWrt Firmware
  – Write OpenWrt firmware
  – *bluetoothctl* utility flexibility
  – How to control BT PAN on BlueZ
  – Persistence of Pairing Information
  – Make NAP work at cold start

• Introduction to low-end devices running NuttX with OpenWrt router
  – Introduction to NuttX networking features
  – How to run Bluetooth on NuttX
  – Software stack for Bluetooth networking
  – BTstack log example
  – MP3 streaming via Bluetooth
  – Running RNDIS on NuttX
  – Working with Wi-Fi on Spresense
  – Use case for Webserver via Wi-Fi
Off-the-self hardware using OpenWrt extends BT PAN in addition to USB RNDIS
What is Bluetooth (BT) PAN

• PAN stands for Personal Area Networking

  – The following roles are defined within the PAN profile.
  • Network Access Point (NAP) and NAP service
  • Group Ad-hoc Network (GN) and GN service
  • PAN User (PANU) and PANU service
Learning Linux BT mechanisms (1/2)

Dell OptiPlex7040
Core i7-6700 3.4GHz
RAM 8GByte
HDD 1TByte

Ubuntu16.04.5 LTS (xenial)

BT USB Dongle
(BSBT4D09BK)
NAP role

BT PAN Profile connection

https://kernel.googlesource.com/pub/scm/bluetooth/bluez.git
bluez/test (python script)
- test-nap

XperiaXZ (Android7)
PANU roles
Network server hierarchy
========================
Service         org.bluez
Interface       org.bluez.NetworkingServer1
Object path     /org/bluez/{hci0,hci1,...}
Methods         void Register(string uuid, string bridge)

Register server for the provided UUID. Every new connection to this server will be added the bridge.

This API is the essence of NAP role for BT PAN connection
'dbus-send' command can invoke this API
SMP and Networking Support on Nuttx/LC823450

OpenIoT2018 NA Portland

# Nuttx is POSIX-compliant oriented RTOS such as Tiny-Linux.

We have an update of RNDIS in second part

OpenIoT2018 NA Portland
Hardware Requirement

• OpenWrt supporting hardware with USB port
  – BUFFALO WZR-HP-G300NH (USB 2.0 port)
    You can see https://openwrt.org/toh/start

• BT USB Dongle supporting PAN profile
  – BUFFALO BT USB Dongle BSBT4D09BK
Software Requirement

- OpenWrt Firmware for WZR-HP-G300NH with USB RNDIS configuration

I used the below commit at that time.

https://git.opensrtr.org/openwrt/openwrt.git/
commit 8722c52b41d551e768b3cc46049af6657099d59
Author: Christoph Krapp <achterin@googlemail.com>
Date:   Tue Jul 3 11:06:20 2018 +0200

https://git.opensrtr.org/openwrt/openwrt.git/
commit e768b3cc46049af6657099d59
Author: Christoph Krapp <achterin@googlemail.com>
Date:   Tue Jul 3 11:06:20 2018 +0200
Create OpenWrt Firmware

$ git clone https://git.openwrt.org/openwrt/openwrt.git/
$ cd openwrt
$ ./scripts/feeds update -a
$ ./scripts/feeds install -a
$ cp enable_btpan_usbrndis_for_wzr-hz-g300nh.seed .config
$ make defconfig
$ make

These two command should execute once so that a needed software package can be selected.

I provide config.seed file at the bottom of this slide

Generated firmware exists on ./bin/targets/ar71xx/generic/ directory
Write OpenWrt firmware

• Firmware update is done from Web GUI
  – BUFFALO’s original firmware Web GUI can accept “openwrt-ar71xx-generic-wzr-hp-g300nh-squashfs_factory.bin” image
  – After OpenWrt firmware is run, firmware update is done from the OpenWrt Web GUI (Luci) using “openwrt-ar71xx-generic-wzr-hp-g300nh-squashfs_sysupgrade.bin” image
bluetoothctl utility flexibility

Please note that “bluetoothctl” is the file name of Bluetooth utility program.

Interactive mode

```
root@OpenWrt:~# bluetoothctl
Agent registered
[bluetooth]# power on
[CHG] Controller 00:1B:DC:06:61:D4 Class: 0x00020000
Changing power on succeeded
[CHG] Controller 00:1B:DC:06:61:D4 Powered: yes
[bluetooth]# quit
root@OpenWrt:~#
```

“root@OpenWrt:~#” is shell prompt. “[Bluetooth]#” is the prompt of bluetoothctl.

Single command line

```
root@OpenWrt:~# bluetoothctl power on
[CHG] Controller 00:1B:DC:06:61:D4 Class: 0x00020000
Changing power on succeeded
[CHG] Controller 00:1B:DC:06:61:D4 Powered: yes
root@OpenWrt:~#
```

I impressed bluetoothctl utility consideration for both use cases.
I applied my sample control application in my company after I knew that.
How to connect BT PAN by hand (1/6)

• ssh login to OpenWrt and set NAP UUID to `bluetoothd`

```
root@OpenWrt:~# dbus-send --system \    
    --dest=org.bluez /org/bluez/hci0 \    
    --type=method_call \    
    org.bluez.NetworkServer1.Register \    
    string:“00001116-0000-1000-8000-00805f9b34fb” \    
    string:"br-lan"
```

root@OpenWrt:~#
Now **bluetoothd** supports NAP role (2/6)

```
[bluetooth]# show
Controller 00:1B:DC:06:61:D4 (public)
    Name: BlueZ 5.49
    Alias: BlueZ 5.49
    Class: 0x00020000
    Powered: yes
    Discoverable: no
    Pairable: yes
    UUID: Generic Attribute Profile (00001801-0000-1000-8000-00805f9b34fb)
    UUID: NAP                       (00001116-0000-1000-8000-00805f9b34fb)
    UUID: A/V Remote Control        (0000110e-0000-1000-8000-00805f9b34fb)
    UUID: PnP Information           (00001200-0000-1000-8000-00805f9b34fb)
    UUID: A/V Remote Control Target (0000110c-0000-1000-8000-00805f9b34fb)
    UUID: Generic Access Profile    (00001800-0000-1000-8000-00805f9b34fb)
    Modalias: usb:v1D6Bp0246d0531
    Discovering: no

[bluetooth]#
```

Please note that “bluetoothd” is the file name of Bluetooth Daemon program.

Service Discovery

https://www.bluetooth.com/specifications/assigned-numbers/service-discovery/
root@OpenWrt:~# bluetoothctl
[bluetooth]# trust 00:02:5B:00:A5:A5
[CHG] Device 00:02:5B:00:A5:A5 Trusted: yes
Changing 00:02:5B:00:A5:A5 trust succeeded
[bluetooth]#

Specify the MAC address of BT PANU role device to be connected (e.g: low-end device such as second part)
Connect your device to BT on OpenWrt (4/6)

• Make BT discoverable on OpenWrt router
• Pairing your device with OpenWrt router
• Establish BT PAN connection
When PANU role device connects to router, bnep0 interface appears on BT PAN router (NAP role). This node is added “br-lan” bridge interface to which ethernet LAN port connects.

It means BT PANU device and ethernet LAN connected device have same LAN side IP network.
“bnep0” interface belongs to “br-lan” bridge (6/6)

LAN is “br-lan” bridge interface

WAN is “eth1” interface

4 ethernet port interface is “eth0.1”

BT PAN connection

NAP role

PANU role

“bnep0”
Persistence of Pairing Information

- Nothing to do for BT pairing
- Once BT pairing is completed by hand, *bluetoothd* (BlueZ daemon) will save its connection data.
Make NAP work at cold start

• Two things need to be executed at startup time on OpenWrt router to enable BT NAP role
  – Set NAP role to `bluetoothd`
  – BT Dongle Power ON

• “Local Startup” is used for the above operation as shell scripts.
  – “Local Startup” locates Web GUI’s “System” tab -> “Startup” screen on the bottom.

```
OpenWrt
Status System Network Logout

Local Startup
This is the content of /etc/rc.local. Insert your own commands here (in front of 'exit 0') to execute them at the end of the boot process.

# Put your custom commands here that should be executed once
# the system init finished. By default this file does nothing.
dbus-send --system --dest=org.bluez.org/bluez/hci0 --type=method_call org.bluez.NetworkServer1.Register string "00001116-0000-1000-8000-08080f9b34fb" string "br-lan"
bluetoothctl power on
```
Introduction to low-end devices running NuttX with OpenWrt router
Biography (Masayuki Ishikawa)

Senior Software Engineer
at Sony Home Entertainment & Sound Products Inc.

Technical background
- 3D graphics, home networking, Internet-to-Home, Embedded Systems

Product development
- Portable Media Player (Linux/Android)
- Digital voice recorder, music player (NuttX)

Public talks
- Arm Techcon 2016, ELC2017NA, OpenIoT2018NA, NuttX2019
Introduction to NuttX networking features

- **Motivation**
  - Confirm interoperability between OpenWrt and NuttX by running network applications on resource limited devices

- **Features**
  - Ethernet and IEEE 802.11 Full MAC
  - 6LoWPAN for IEEE 802.15.4 MAC
  - USB RNDIS, CDC-ECM
  - SLIP, TUN/PPP, loopback devices
  - IPv4, IPv6, TCP, UDP, ARP, ICMP, ICMPv6, IGMPv2
  - BSD compatible socket layer
  - DNS name resolution / NetDB
  - User socket

NuttX 7.31 + H/W (e.g. Cortex-M based)
How to run Bluetooth on NuttX

- Port the BTstack by Bluekitchen to NuttX
  - Based on posix-h4 with H/W flow control
  - UART speed: 921600 baud
  - Free for non-commercial use

- Add TAP mode to the NuttX tun driver
  - TAP mode is used for network bridge
  - NOTE: TUN mode is used for network routing

- HCI_RESET issue in SMP mode
  - CSR’s mode change with HCI_RESET is tricky
  - Still unstable in SMP mode
Software stack for Bluetooth networking

*BNEP: Bluetooth Network Encapsulation Protocol
MP3 streaming via Bluetooth

Realtime Traffic

Inbound: 4.59 kbit/s (0.57 kB/s)
Average: 4.17 kbit/s (0.52 kB/s)

Outbound: 181.04 kbit/s (22.63 kB/s)
Average: 157 kbit/s (19.63 kB/s)

music #1

music #2
Running RNDIS on NuttX

LC823450XGEVK + LEDE (RNDIS) at OpenIoT 2018

Spresense + OpenWrt (RNDIS) at ELC2019NA
Working with Wi-Fi on Spresense

- **Wi-Fi module**: Telit GS2200M
  - Radio protocols: 802.11b/g/n (2.4GHz)
  - Interface: SPI 10MHz with DMA

- Implement GS2200M driver from scratch*
  - Based on the Nuttx usrsock
  - Both STA and AP modes are supported
  - Fix cxd56_gpioint.c for interrupt handling
  - TCP and UDP are supported

- Modify the uIP webserver app for Nuttx
  - Add a directory listing feature

* The code is available at https://bitbucket.org/nuttx/nuttx
Use case for Webserver via Wi-Fi

Network applications

- DHCP client
- DNS client
- telnetd
- webserver
- gs220m daemon

/dev/mmcscd0 /dev/usrsock /dev/gs2200m

SDHCI Interrupt SPI
micro SD GS2200M

The latest NuttX upstream + Spresense

Web browsing with Firefox

OpenWrt

Wi-Fi

WZR-HP-G300NH

VirtualBox + Ubuntu

VirtualBox + Ubuntu
Conclusion

- This work shows how to easily extend an OpenWrt router
- Also, NuttX networking is feasible on resource limited devices
I knew about LEDE at ELC2017 session in Portland

OpenWrt/LEDE: when two become one, presented by Florian Fainelli, (https://sched.co/9luP)

https://elinux.org/images/0/0a/ELC_OpenWrt_LEDE.pdf

http://events17.linuxfoundation.org/events/embedded-linux-conference/program/slides
Reference (2/3)

Developing Audio Products with Cortex-M3/NuttX/C++11
(https://sched.co/9O0s)

(ELC2017 North America)

SMP and Networking Support on NuttX/LC823450
(https://sched.co/DYML)

(OpenIoT2018 North America)
OpenWrt documents

- Table of Hardware (supported hardware list)
  https://openwrt.org/toh/start

- Official Documents starting point
  https://openwrt.org/docs/start

- Developer Guide
  https://openwrt.org/docs/guide-developer/start

- Creating packages
  https://openwrt.org/docs/guide-developer/packages

OpenWrt manages software components on a package basis. For this reason, it is better to create a package to import (Porting software component to OpenWrt). This URL explains about it. The template of Makefile realizes easy porting explanation by the example of bridge package.

- Quick Image Building Guide (Image build guide)
  https://openwrt.org/docs/guide-developer/quickstart-build-images

BUFFALO product information

- Wi-Fi router
  http://buffalo.jp/products/catalog/network/wzr-hp-g300nh/

- BT Dongle
  http://buffalo.jp/product/peripheral/wireless-adapter/bsb4d09bk/

- IBM WATSON IOT iot-raspberrypi
  https://github.com/ibm-watson-iot/device-raspberrypi

- dbus
  https://www.freedesktop.org/wiki/Software/dbus/

- dbus-monitor
  https://dbus.freedesktop.org/doc/dbus-monitor.1.html
dump D-Bus message
Any Questions?
Thank you for your participation and interest
Supplemental Material

• What’s OpenWrt
• Next two slides come from our OpenIoT2018 North America “SMP and Networking support on NuttX / LC823450”.
Introduction to LEDE

- **Motivation**
  - Build a shareable network testing environment for NuttX

- **Software**
  - LEDE project as of ELC2017 session
  - The project was forked from OpenWRT that is famous OSS for the router world as a turn key solution but they became one again (at the beginning of 2018)

- **Hardware**
  - WZR-HP-G300NH (buffalo) Wi-Fi router with USB 2.0 port

---

![Image](image.png)

Announcing the OpenWrt/LEDE merge

Both the OpenWrt and LEDE projects are happy to announce their unification under the OpenWrt name. After long and sometimes slowly moving discussions about the specifics of the re-merge, with multiple similar proposals but little subsequent action, we're happy to announce that both projects are about to execute the final steps of the merger.

The new, unified OpenWrt project will be governed under the rules established by the LEDE project. Active members of both the former LEDE and OpenWrt projects will continue working on the unified OpenWrt.

LEDE's fork and subsequent re-merge into OpenWrt will not alter the overall technical direction taken by the unified project. We will continue to work on improving stability and release maintenance while aiming for frequent minor releases to address critical bugs and security issues like we did with LEDE 17.01 and its four point releases until now.

Old pre-15.05 OpenWrt CC releases will not be supported by the merged project anymore, leaving these releases without any future security or bug fixes. The OpenWrt CC 15.05 release series will receive a limited amount of security and bug fixes, but is not yet fully integrated in our release automation, so binary releases are lacking behind for now.

The LEDE 17.01 release will continue to get full security and bug fix support for both source code and binary releases. We are planning a new major release under the new name in the next few months.
Support RNDIS on LEDE

- How to setup
  - Modify configuration
  - Add network USB0 (RNDIS) via LuCI
  - Change the network setting of USB0
Embedded Linux Conference North America