6LoWPAN Support on Intel® Galileo Platform

User Guide

September 2015

Document Number: 559445-1.0

Contents

1.0 Introduction **6**
- 1.1 Terminology 7
- 1.2 Reference Documents 7

2.0 Before You Begin **8**

3.0 6LoWPAN Software Overview **9**

4.0 Running the Software **11**
- 4.1 Starting the Network Border Router (NBR) 11
- 4.2 Starting the Router Advertisement Daemon (radvd) 11
- 4.3 Enable Routing in IPv6 13
- 4.4 Starting the 6LoWPAN Demo Server 13

5.0 Accessing 6LoWPAN Network **14**
- 5.1 Access 6LoWPAN Network with Web Browser 14
  - 5.1.1 Device List 14
  - 5.1.2 Pinging a 6LoWPAN Node 15
  - 5.1.3 Toggle LEDs 15
  - 5.1.4 Reading Temperature 15
  - 5.1.5 HTTP Server 16
  - 5.1.6 Network Topology View 16
  - 5.1.7 Configuration Page 17
  - 5.1.8 RSSI Scanner 18
  - 5.1.9 802.15.4 Sniffer 18
- 5.2 Access 6LoWPAN Network with CoAP client 19
  - 5.2.1 Firefox Web Browser GUI CoAP client 19
  - 5.2.2 Command-line Interface CoAP Client 20

6.0 Using Lightweight Machine to Machine (LWM2M) **22**
- 6.1 Configuring and Running the LWM2M (Wakaama) Bootstrap Server 22
- 6.2 Running and Using LWM2M (Wakaama) Server to Access the LWM2M Clients 23
- 6.3 LWM2M Network with Datagram Transport Layer Security (DTLS) Encryption 26

Figures

Figure 1. Overview of 6LoWPAN 6

Figure 2. Overview of the Software Architecture on Intel® Galileo Platform 9

Figure 3. Device List for all 6LoWPAN Nodes 15
Figure 4. IPv6 Address of the Nodes on 6LoWPAN Network 16
Figure 5. Network Topology View 17
Figure 6. Configuration Page 17
Figure 7. RSSI Scanner 18
Figure 8. CoAP GUI Interface 20
Figure 9. LWM2M Protocol Stack 27

Tables

Table 1. Terminology 7
Table 2. Reference Documents 7
Table 3. Command Line Interface 21
Table 4. Objects Supported by IoT-U10 25
Table 5. Objects/Resources 25

Revision History

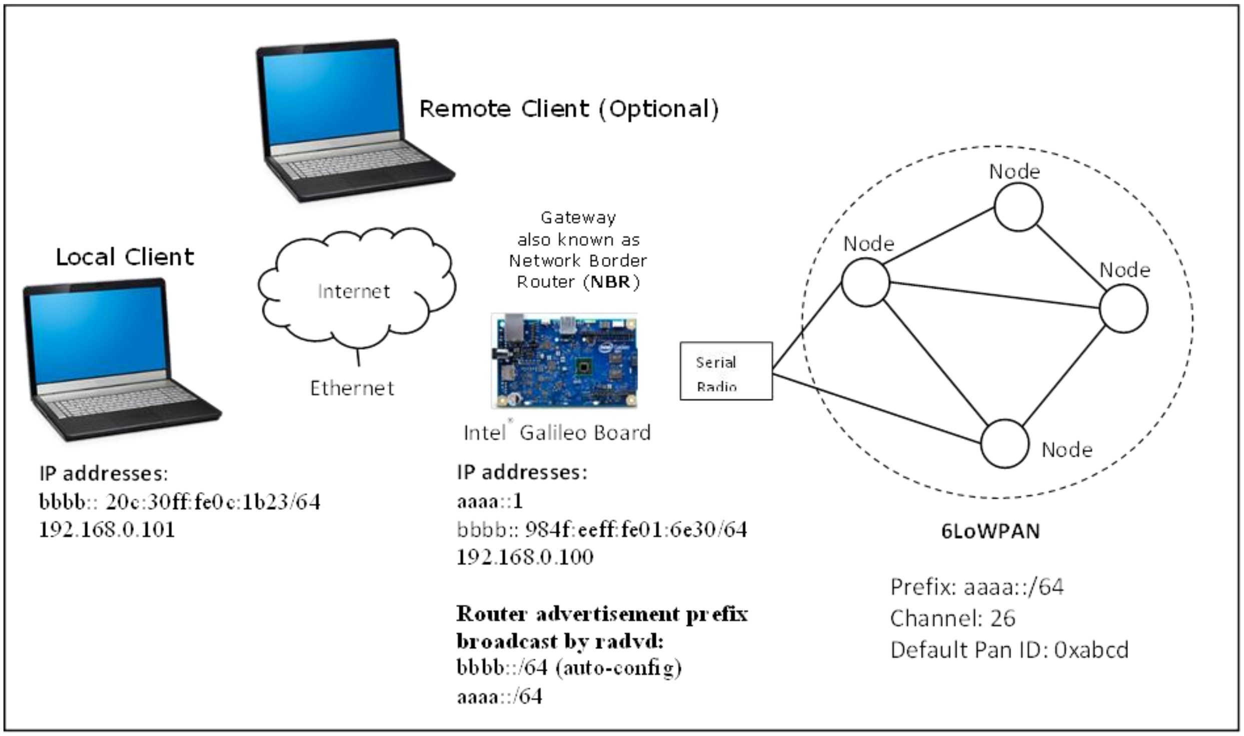
Date	Revision	Description
September 2015	1.0	Initial release.

§

1.0 Introduction

This document describes how to use the software tools available in the Intel® Quark™ BSP 1.2.0 to set up a simple 6LoWPAN network. The Intel® Galileo board running the Intel® Quark™ BSP 1.2.0 acts as IPv6 gateway or 6LoWPAN network border router (NBR) between the Ethernet network and the 6LoWPAN network as shown in Figure 1 below.

Figure 1. Overview of 6LoWPAN



Note: Depending on your DHCP server's configurations and your machine's MAC addresses, your local client and Intel® Galileo board's IPv4 and IPv6 addresses might be different from the ones shown in Figure 1 above.

You can perform simple operations on the 6LoWPAN nodes such as sending commands to toggle the LED, reading temperature, and so on through your local clients or remote clients.

Refer to the next chapter on additional firmware and hardware required to set up the 6LoWPAN network.

1.1 Terminology

Table 1. Terminology

Term	Description
6LoWPAN	IPv6 over Low power Wireless Personal Area Network
BSP	Board Support Package
DHCP	Dynamic Host Configuration Protocol
MAC	Media Access Control
NBR	Network Border Router
UART	Universal Asynchronous Receiver/Transmitter

1.2 Reference Documents

Table 2. Reference Documents

Document	Document No./Location
Contiki: The Open Source OS for the Internet of Things	http://www.contiki-os.org/

§

2.0 Before You Begin

This is not an introductory document for 6LoWPAN. Many online resources out there cover the concepts and specifications of 6LoWPAN. This document does walk you step by step on setting up the gateway or network border router (NBR) on the Intel® Galileo board to enable the communication between the local/remote PC on the Ethernet network and the nodes on the 6LoWPAN network. It also covers different methods to access the 6LoWPAN nodes with local PC through HTTP and CoAP protocols.

This document assumes you have some experience with IPv4 and IPv6 networking, Linux*/Windows* OSes, and using software/hardware tools to flash firmware into the hardware platform.

What you need:

- A host PC as local/remote client running either of the following:
  - Linux; Ubuntu* Linux with IPv6 support is recommended.
  - Microsoft* Windows 7, Windows 8 with IPv6 support.
- An Intel® Galileo board running Intel® Quark™ BSP 1.2.0.
- A web browser installed on host PC. The Firefox* browser is recommended.
- An Ethernet network with a DHCP server. This network connects the local client and Intel® Galileo board through the LAN. Connection to Internet and remote client are optional.
- A 6LoWPAN serial radio device, IoT-U10, from Yanzi Networks AB, which will be connected to the Intel® Galileo board through a USB interface as serial radio.
- One, or more, 6LoWPAN node device, such as IoT-U10 from Yanzi Networks AB.

Note: For more detail on getting the IoT-U10 device, visit http://www.yanzi.se., visit http://www.yanzi.se.

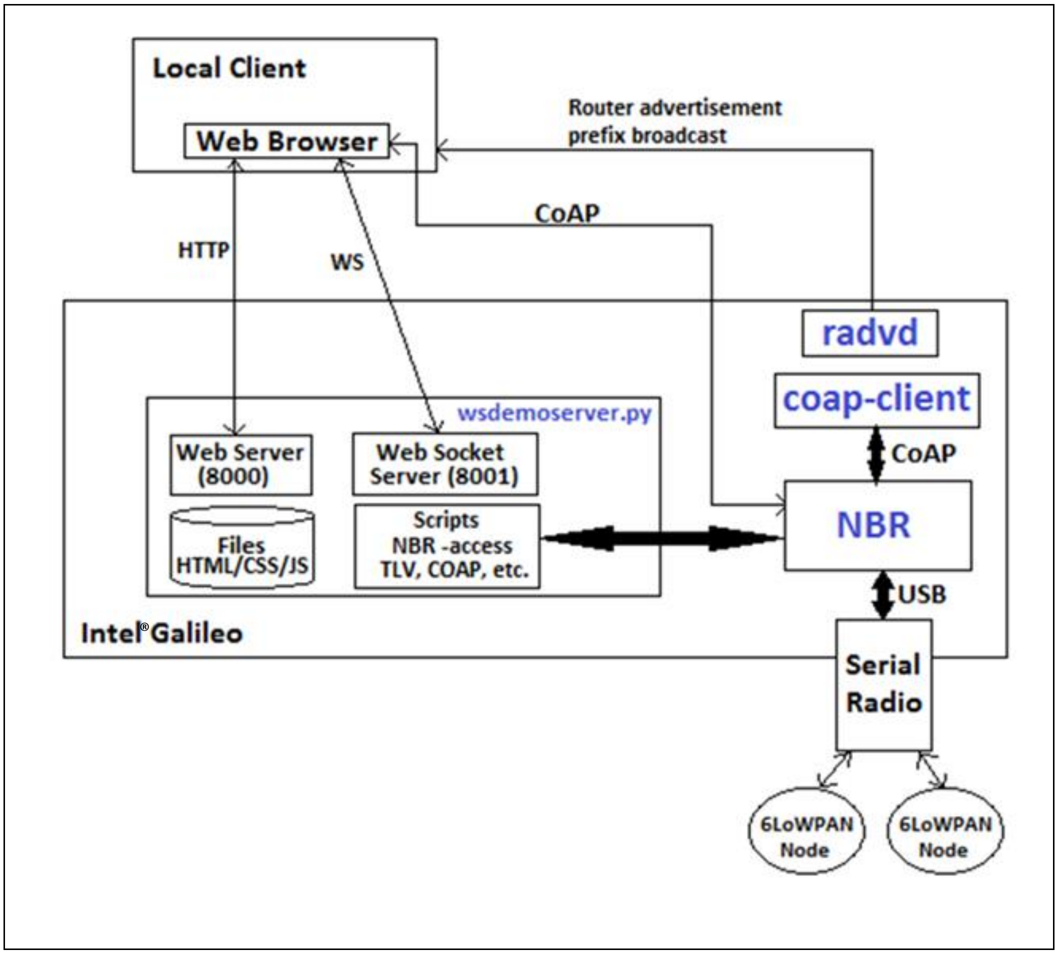
§

3.0 6LoWPAN Software Overview

Below are the main software components and tools for 6LoWPAN support:

- Network border router (NBR)
- Linux IPv6 Router Advertisement Daemon (radvd)
- 6LoWPAN demo server written in Python scripts (wsdemoserver.py)
- CoAP client is a tool that allow users to access 6LoWPAN nodes on the Intel® Galileo platform through the command-line interface (coap-client)

Figure 2. Overview of the Software Architecture on Intel® Galileo Platform



NBR acts as the gateway between the Ethernet network and the 6LoWPAN network. It allows other software such as the 6LoWPAN demo server (**wsdemoserver.py**) and the CoAP client tool (**coap-client**) to access the 6LoWPAN nodes through serial radio.

A 6LoWPAN demo server written in Python scripts (**wsdemoserver.py**) serves as the HTTP and web socket server, which allow users to access the 6LoWPAN network by using a web browser on the local client.

Users can also run the Linux IPv6 Router Advertisement Daemon (**radvd**) on the Intel® Galileo board to broadcast the IPv6 prefix, which allows the local client to access the 6LoWPAN nodes by referring to their IPv6 address through CoAP protocol. However, **radvd** is not needed if users wish to access the 6LoWPAN nodes with IPv4 address of the HTTP server, which is running on the Intel® Galileo board. Details on how the **radvd** works is out of scope of this document. You can visit http://www.litech.org/radvd/ or https://www.ietf.org/rfc/rfc2461.txt for more details.

Users can access the 6LoWPAN nodes through the CoAP protocol by using the web browser on their local client. This procedure only works if the users' local client operating system supports IPv6 and the **radvd** is running on the Intel® Galileo board. Intel recommends using the Firefox web browser because users can install **Copper(Cu),** a CoAP plugin that is available for the Firefox web browser. With this CoAP plugin, users can access the 6LoWPAN nodes through the GUI interface. For advanced users, they can also access the 6LoWPAN nodes by using **coap-client**, which is a command-line interface tool available in Linux. It is the users' responsibility to install the **coap-client** on their own machine (local client). Users can also use the pre-installed **coap-client** on the Intel® Galileo board to access the 6LoWPAN nodes in the Linux shell.

Note: Serial radio must be attached to the Intel® Galileo board's USB port for the NBR to work. To experiment with the 6LoWPAN network, at least one 6LoWPAN node is needed.

§

4.0 Running the Software

This section guides you step by step on running the 6LoWPAN software on the Intel® Galileo platform. We assume users already have access to the Intel® Galileo Linux shell through SSH (Ethernet port) or serial console (UART port).

4.1 Starting the Network Border Router (NBR)

Attach the serial radio dongle to the USB port on the Intel® Galileo board. You should be able to see `ttyACM0` in the `/dev` directory. Enter the following command to confirm the serial radio is detected by the OS:

```
# ls /dev/ttyA*
/dev/ttyACM0
```

If the serial radio is present, then you can start the NBR. The NBR executable binary is located at `/opt/netcontiki/sbin`. Enter the command as follows:

```
# nohup /opt/netcontiki/sbin/border-router.native -s /dev/ttyACM0 -
L aaaa::1/64 &
```

A tunnel interface tun0 with IPv6 address **aaaa::1** is created by NBR as shown below:

```
# ifconfig
. . .
tun0      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00                   -00
          inet6 addr: aaaa::1/64 Scope:Global
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:0  (0.0 B)  TX bytes:0  (0.0 B)
```

The NBR is up and running now. Next, we start the **radvd**.

4.2 Starting the Router Advertisement Daemon (radvd)

For users who wish to access the 6LoWPAN network through local client with IPv4 address only, this step is optional. The Router Advertisement Daemon (**radvd**) is required if users need to access the 6LoWPAN nodes with IPv6 addresses through the local client. Simply enter the following command to start the **radvd** (make sure the **radvd** configuration file located at `/etc/radvd.conf` exists):

```
# radvd
```

If `/etc/radvd.conf` does not exist, create it by issuing the command "vi /etc/radvd.conf" and enter the following recommended configuration settings and save it:

```
interface enp0s20f6
{
        AdvSendAdvert on;
        prefix bbbb::/64
        {
                AdvOnLink on;
                AdvAutonomous on;
        };

        route aaaa::/64
        {

        };
};
```

With the settings above, **radvd** sends advertisements to all clients on the same LAN. When the clients receive the router advertisement message, they automatically configure addresses with the received prefix (**bbbb::/64**) and the default route. This allows the clients on the LAN network to access the nodes on the 6LoWPAN network. For more details on the **radvd** advanced configurations, visit http://linux.die.net/man/5/radvd.conf and http://www.tldp.org/HOWTO/Linux+IPv6-HOWTO/hints-daemons-radvd.html.

If **radvd** is running successfully, you see the Ethernet interface `enp0s20f6` is auto configured with IPv6 address with prefix **bbbb::/64**.

```
# ifconfig
enp0s20f6 Link encap:Ethernet  HWaddr 98:4F:EE:01:6E:30
          inet addr:192.168.0.100  Bcast:172.30.66.255  Mask:255.255.255.0
          inet6 addr: bbbb::984f:eeff:fe01:6e30/64 Scope:Global
          inet6 addr: fe80::984f:eeff:fe01:6e30/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:37083 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3907 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3118774 (2.9 MiB)  TX bytes:554533 (541.5 KiB)
          Interrupt:41 Base address:0x8000
```

The Ethernet interface on the Intel® Galileo board as shown above is configured with IPv6 address **bbbb::984f:eeff:fe01:6e30/64**.

All the local clients located in the same LAN are auto configured with prefix **bbbb::/64**. For example, in Figure 1, the local client is configured with IPv6 address **bbbb::20c:30ff:fe0c:1b23/64**.

4.3 Enable Routing in IPv6

To enable routing in IPv6, issue the following command:

```
# sysctl -w net.ipv6.conf.all.forwarding=1
```

Now the Intel® Galileo board acts as an IPv6 router, which enables the local client to access the 6LoWPAN nodes with IPv6 addresses.

4.4 Starting the 6LoWPAN Demo Server

Now you need to start the 6LoWPAN Demo Server (wsdemoserver.py) to allow the local client and remote client to access the 6LoWPAN network through HTTP protocol with IPv4 address.

```
# cd /opt/netcontiki/sbin/examples/galileo/demo
# python wsdemoserver.py &
```

The above command starts the 6LoWPAN Demo Server as a background process. Now the local client and remote client are able to access the HTTP server through port 8000. We will get into this part later.

§

5.0 Accessing 6LoWPAN Network

This document covers two methods to access the 6LoWPAN Network:

- HTTP protocol between web server and web browser
- CoAP protocol between CoAP server and CoAP client

5.1 Access 6LoWPAN Network with Web Browser

After the 6LoWPAN Demo Server has started, you can browse to the IP address of the Intel® Galileo board using any modern browser (with support for HTML5 and JavaScript).

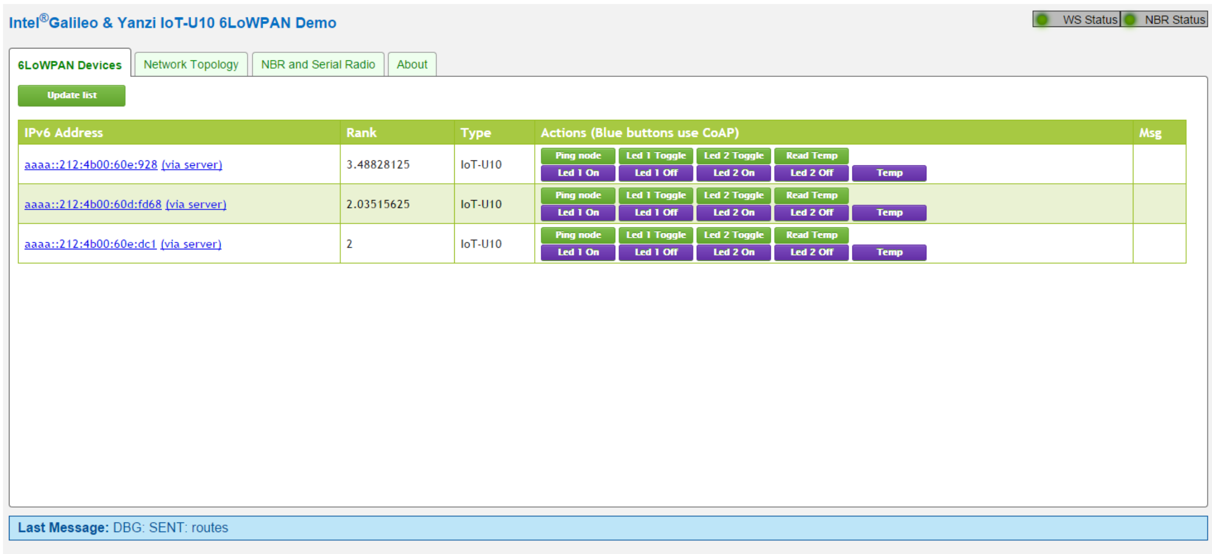
Note: The default port is 8000, so the link to the index page is (assuming Intel® Galileo board's IP address is 192.168.0.100) the following:

http://192.168.0.100:8000/index.html

5.1.1 Device List

The 6LoWPAN Demo page contains a device list where all the 6LoWPAN nodes can be seen and all the basic function can be demonstrated by clicking on the buttons. The device list is shown in the screenshot below.

Figure 3. Device List for all 6LoWPAN Nodes



In the screenshot above, you can see there are two 6LoWPAN nodes discovered by the NBR. Users can click on the “Update list” button to refresh the device list if the 6LoWPAN nodes are not listed on the page. Each of the devices is listed with IPv6 address together with their type.

5.1.2 Pinging a 6LoWPAN Node

Any node that connects to the 6LoWPAN network is listed in the device list of the page. Each of the nodes has a ping button so that it is possible to ping the nodes from the web application.

After clicking on the ping button, a window pops up within the web page and the response from the ping command is shown.

5.1.3 Toggle LEDs

If the node is a Yanzi IoT-U10 radio device, there are two Toggle LED buttons on the same row as the IPv6 address of the node. Clicking on these buttons toggles the corresponding LED on the IoT-U10 node.

5.1.4 Reading Temperature

There is a button for reading the temperature of the device. If clicked, the request is sent to the 6LoWPAN web server, which then sends a request for reading the temperature variable. The temperature is displayed at the end of the same row.

5.1.5 HTTP Server

In the IoT-U10 node, there is also a tiny web server that shows some information on the node. If your OS supports IPv6 and you are directly connected to the same network, you can either browse to http://[IPv6-address-of-node]/ or do a wget to the same address to access the node's information through HTTP protocol.

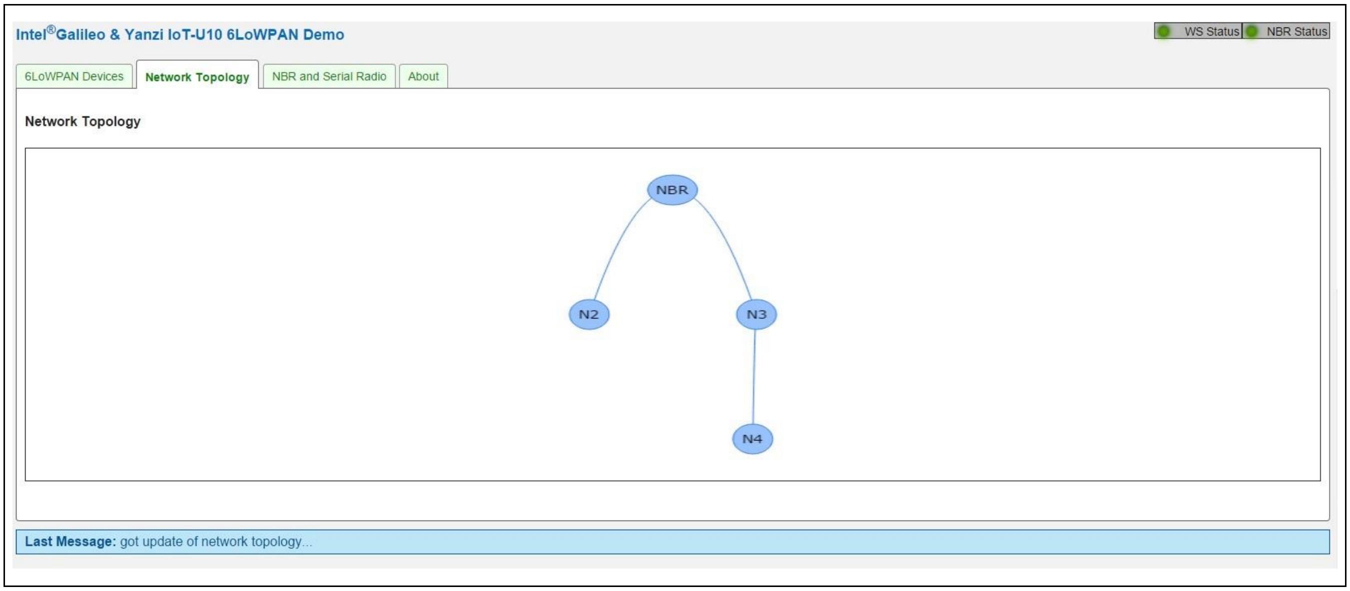
The screenshot below shows that we can browse to the IPv6 address of the nodes on 6LoWPAN network with web browser. A simple web page with some links is shown on the browser. Users can click on the links displayed on the web page to retrieve some basic network information about the node.

Figure 4. IPv6 Address of the Nodes on 6LoWPAN Network

5.1.6 Network Topology View

The screenshot shown below is the network topology view that shows two 6LoWPAN nodes that are connected to the NBR. All these nodes support the network statistics functional instance. This instance makes it possible to read out routing layer data such as Rank (position in the network) and Parent (the default route of the node) and some other information. This information is used to create a network topology graph.

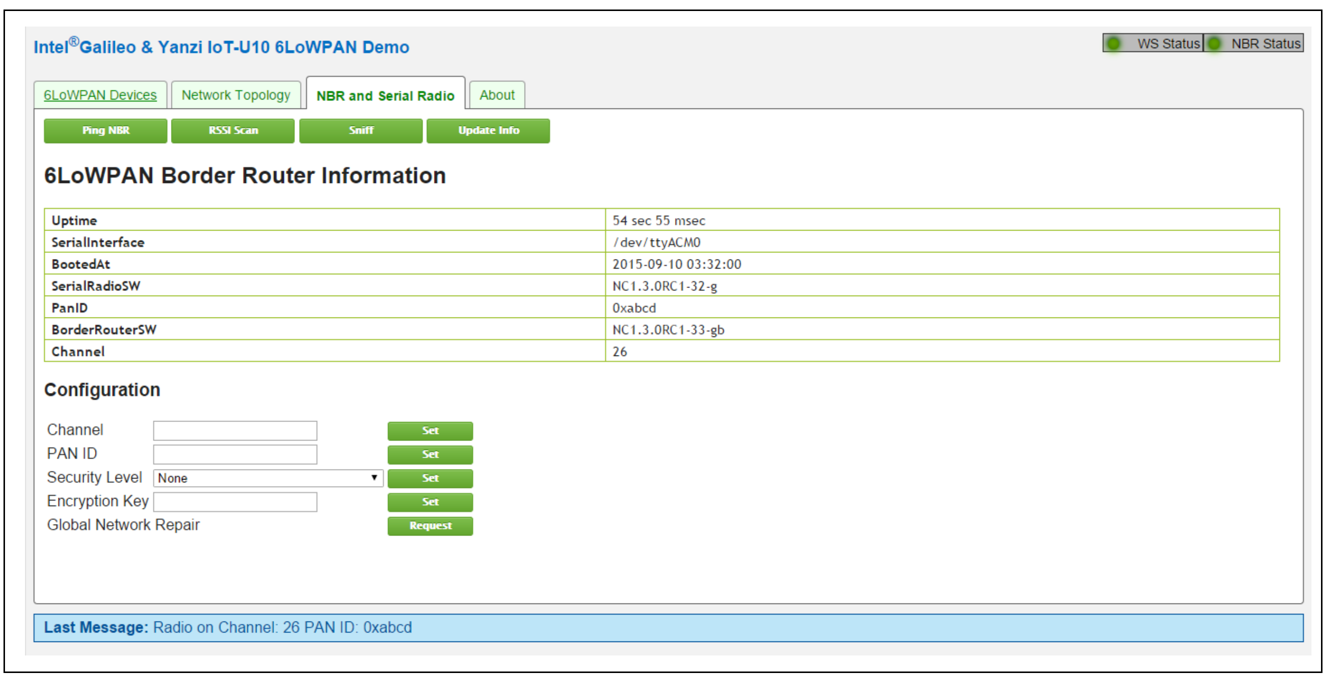
Figure 5. Network Topology View



5.1.7 Configuration Page

The configuration page of the 6LoWPAN demo server page shows some information about the network border router (**NBR**) and serial radio. It also allow configuration of channel, PAN ID, and link layer security key as shown in the screenshot below.

Figure 6. Configuration Page



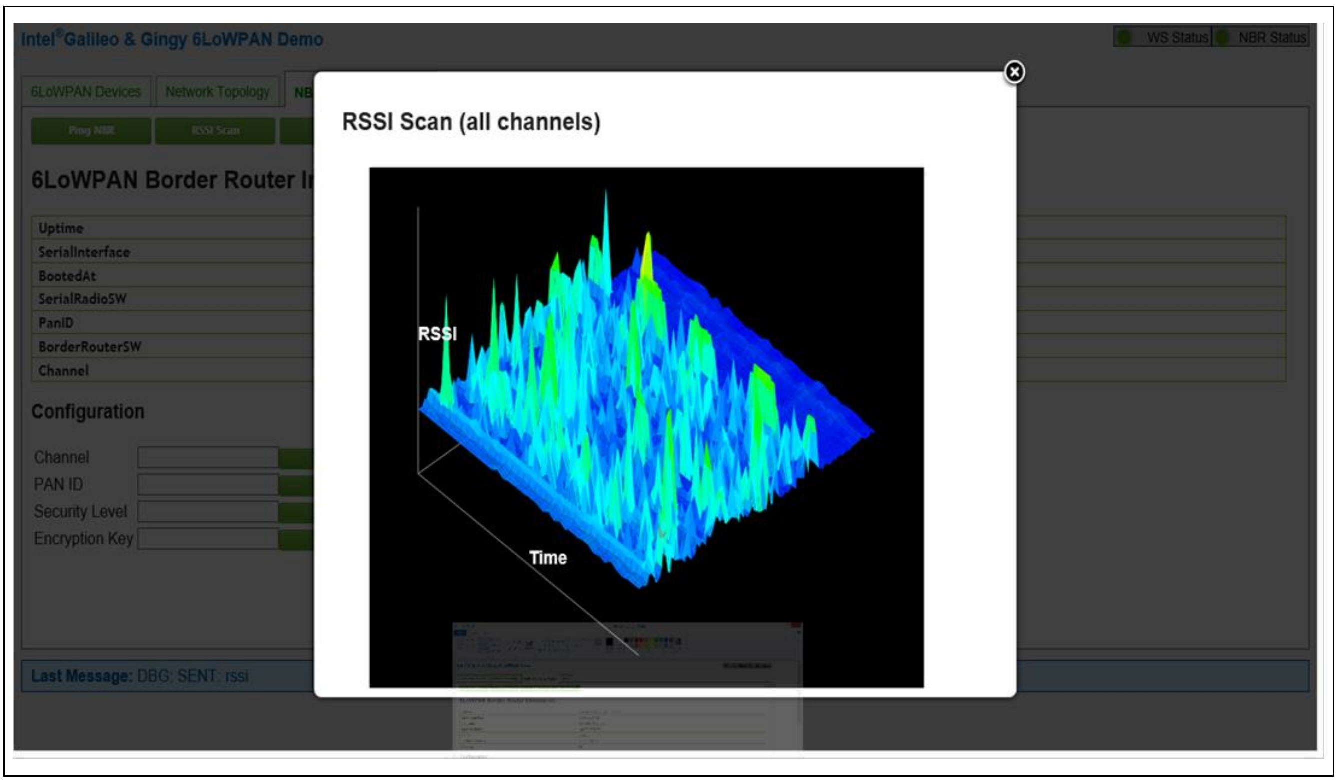
Note: Currently, the channel, PAN ID, security level, and encryption key of the nodes in the 6LoWPAN network are set during firmware compilation time. Therefore, changing these configuration parameters on the NBR will risk losing communication between the 6LoWPAN nodes and the NBR.

5.1.8 RSSI Scanner

The RSSI Scanner demo is started by clicking on “RSSI Scan” on the “NBR and Serial Radio” web page. The NBR then re-configures the serial radio to act as a RSSI Scanner and it pops up a window within the web page that show the RSSI scanning results in a 3D surface plot. The graph is updated as long as the sniffer mode is running. While the scanner is running, the NBR will not manage the 6LoWPAN network.

To exit the scanner mode, click outside the window or at the closing button on the upper right corner.

Figure 7. RSSI Scanner



5.1.9 802.15.4 Sniffer

The sniffer demo is started by clicking on the “Sniffer” button on the “NBR and Serial Radio” web page. It sniffs a set of packets for a short time and displays the packets in hexadecimal in a pop-up window. While the sniffer is running, the NBR will not manage the 6LoWPAN network.

To exit the sniffer mode, click outside the window or on the closing button on the upper right corner.

5.2 Access 6LoWPAN Network with CoAP client

In this section, we focus on accessing the 6LoWPAN nodes directly with IPv6 address through CoAP protocol. Keep in mind that every one 6LoWPAN nodes is a CoAP server. Users access the 6LoWPAN node as a CoAP client.

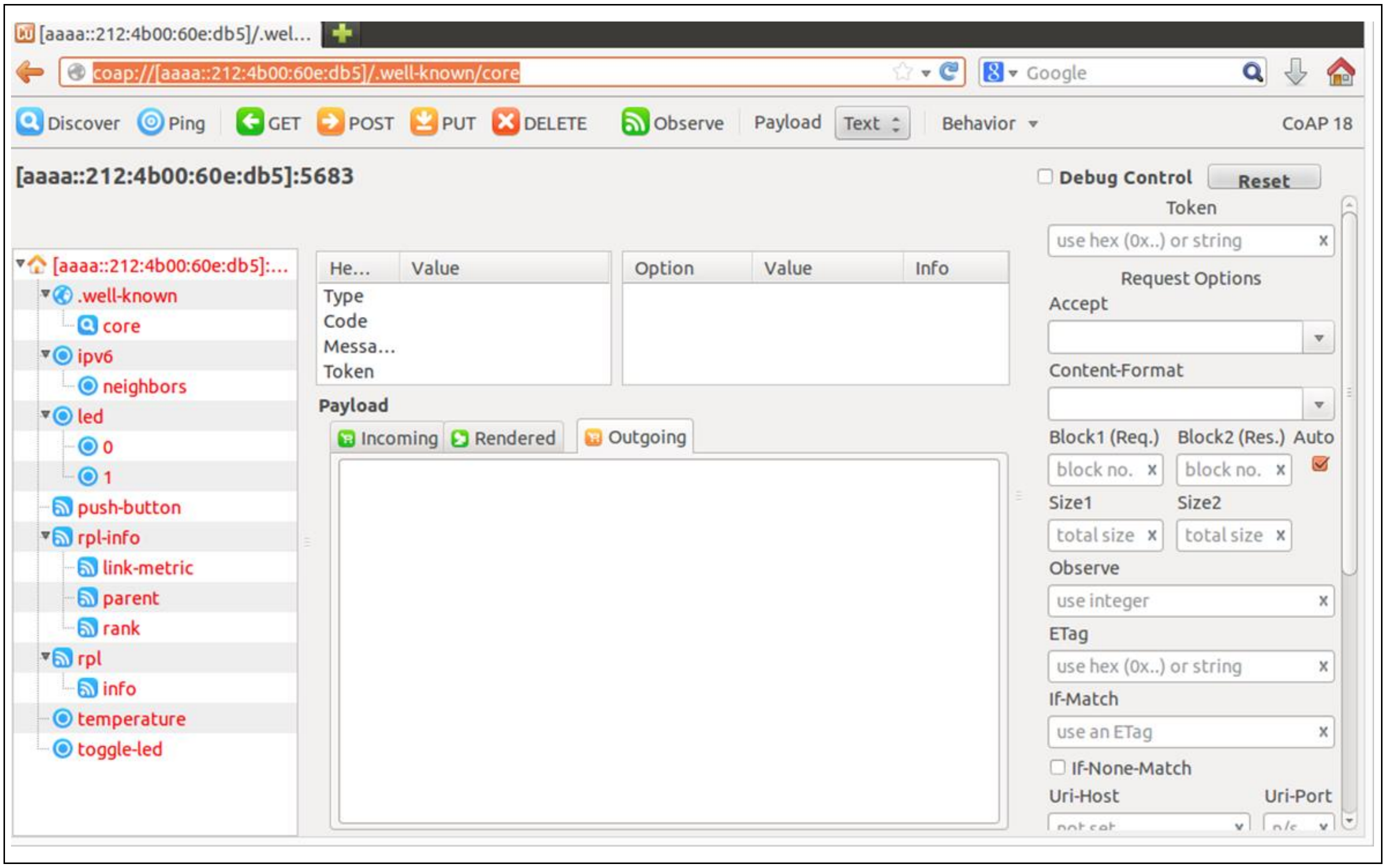
5.2.1 Firefox Web Browser GUI CoAP client

Use your Firefox web browser and visit the https://addons.mozilla.org/en-US/firefox/addon/copper-270430/ to install the CoAP plugin **Copper (Cu)** into it. Make sure the plugin is installed properly and enabled in Firefox web browser. Then, find out the IPv6 address of the 6LoWPAN node you wish to access from the 6LoWPAN Demo server's device list. For example, if the 6LoWPAN node's IPv6 address is **aaaa::212:4b00:60e:db5**, enter the following CoAP URI into the Firefox address bar as below:

coap://[aaaa::212:4b00:60e:db5]/.well-known/core

A CoAP GUI interface is shown as seen in the screenshot below.

Figure 8. CoAP GUI Interface



Use the buttons **GET, POST, PUT, DELETE** to perform an action on the resources listed in the left pane on the web page. For example, at a resource location (coap://[aaaa::212:4b00:60e:db5]:5683/temperature), use the **GET** button to read the temperature. The temperature reading is displayed in the browser. To toggle the LED on the 6LoWPAN node, click on the link "0" under the "led" in the left pane. You are redirected to URI coap://[aaaa::212:4b00:60e:db5]:5683/led/0. Click on the **POST** button to toggle the state of LED. The state of LED is displayed in the browser.

5.2.2 Command-line Interface CoAP Client

For users who prefer command-line interface, they can use the **coap-client** preinstalled in the Intel® Galileo board. Below are some examples on using the command-line interface to access the 6LoWPAN node:

Table 3. Command Line Interface

CoAP Resource Name	Method	Command
/.well-known/core	GET	root@quark:~# coap-client -m get coap://[aaaa::212:4b00:60e:db5]/.well-known/core v:1 t:0 tkl:0 c:1 id:49657 </.well-known/core>;ct=40,</led>;title="LED(default): led=toggle ?len=0..";rt="Text",</led/0>;title="LED0: led=toggle ?len=0..";rt="Text",</led/1>;title="LED1: led=toggle ?len=0..";rt="Text",</ipv6/neighbors>;title="IPV6-Neighbors: ?len=0..";rt="Text",</rpv:1 t:0 tkl:0 c:1 id:49658 l-info>;title="RPL Information";obs,</rpl-info/parent>;title="RPL Parent";obs,</rpl-info/rank>;title="RPL Rank";obs,</rpl-info/link-metric>;title="RPL Link Metric";obs,</push-button>;title="Push event";obs,</temperature>;title="Temperature: ?len=0..";rt="Tv:1 t:0 tkl:0 c:1 id:49659 ext"
/temperature	GET	root@quark:~# coap-client -m get coap://[aaaa::212:4b00:60e:db5]/temperature v:1 t:0 tkl:0 c:1 id:19162 Temperature (C): 28.5
/led/0	POST	root@quark:~# coap-client -m post -e led=toggle coap://[aaaa::212:4b00:60e:db5]/led/0 v:1 t:0 tkl:0 c:2 id:46220 LED0 Toggle: ON

6.0 Using Lightweight Machine to Machine (LWM2M)

Lightweight Machine to Machine (LWM2M) is a protocol from the Open Mobile Alliance for M2M or IoT device management. It is a new, still on-going effort to create a new technical standard for remote management of machine-to-machine devices, service enablement, and application management. For more detailed information about how this protocol works, obtain a free copy of specification document from URL below:

http://openmobilealliance.hs-sites.com/lightweight-m2m-specification-from-oma

To manage 6LoWPAN nodes using this protocol, the nodes should support LWM2M protocol. The nodes act as LWM2M clients and communicate with the LWM2M server running on Intel® Galileo. Commands and queries can be sent to the nodes through the LWM2M server's command prompt interface. The following sections describe the required steps to configure the LWM2M clients and manage them through the LWM2M server.

6.1 Configuring and Running the LWM2M (Wakaama) Bootstrap Server

LWM2M (Wakaama) Bootstrap Server provides a **Bootstrap Interface** for provisioning the essential information into the LWM2M clients to enable the LWM2M clients to register themselves with one or more LWM2M servers.

The LWM2M Bootstrap Server's executable binary (**bootstrap_server**) and its configuration file (**bootstrap_server.ini**) can be found under directory `/opt/netcontiki/sbin/`.

To start the LWM2M (Wakaama) Bootstrap Server, perform the following:

```
# cd /opt/netcontiki/sbin
# ./bootstrap_server -f bootstrap_server.ini
```

Messages are displayed in the console when provisioning happen between the LWM2M clients and the LWM2M Bootstrap server. The LWM2M clients are ready to connect to LWM2M server after being successfully configured by the LWM2M Bootstrap Server.

Here is an example **of bootstrap_server.ini**:

```
# Standard non-encrypted LWM2M server
[Server]
id=1
uri=coap://[aaaa::1]:5683
bootstrap=no
lifetime=300
security=NoSec

# DTLS encrypted LWM2M server with Pre-Shared key
# using identity 'IPSO_Interop' and password 'Not_So_Secret_Key'.
# The identity and password are specified in hexadecimal form.
[Server]
id=2
uri=coaps://[aaaa::1]:5684
bootstrap=no
lifetime=300
security=PSK
public=4950534f5f496e7465726f70
secret=4e6f745f536f5f5365637265745f4b6579

# For any client, we delete all server accounts and
# provision all the server info.
[Endpoint]
Delete=/0
Delete=/1
Server=1
```

Settings above provision the LWM2M clients to connect to the first entry of the LWM2M server listed in the configuration file, which is a non-DTLS encrypted LWM2M server with URI (coap://[aaaa::1]:5683).This is the ipv6 address of Intel® Galileo board.

6.2 Running and Using LWM2M (Wakaama) Server to Access the LWM2M Clients

Start the LWM2M (Wakaama) Server:

```
# cd /opt/netcontiki/sbin
# ./lwm2mserver 2> messages.log
```

This starts the LWM2M (Wakaama) Server with all the low level messages received from the LWM2M clients being redirected to the file **messages.log**.

IPSO object information as shown below are displayed on screen automatically when a LWM2M client has successfully registered itself to the LWM2M server:

```
======== IPSO ========
Client "IoT-U10plus4B00060E0DB5" Light Control #1:
Client "IoT-U10plus4B00060E0DB5" Light Control #0:
Client "IoT-U10plus4B00060E0DB5" Temperature Sensor #0:
Client "IoT-U10plus4B00060E0DB5" Digital Input #0:
```

Depending on the type of services or functions supported by the LWM2M clients, different object information may be shown.

Enter "**help**"command in the LWM2M server's command prompt to get a list of commands supported by the server as shown below:

```
> help
help    Type 'help [COMMAND]' for more details on a command.
list    List registered clients.
read    Read from a client.
write   Write to a client.
exec    Execute a client resource.
del     Delete a client Object instance.
create  create an Object instance.
observe Observe from a client.
cancel  Cancel an observe.
ipso    Toggle IPSO UI.
q       Quit the server.
```

Now, enter "**list**" command to get a list of LWM2M clients registered to the server.

```
> list
Client #0:
        name: "IoT-U10plus4B00060E0DB5"
        binding: "Not specified"
        lifetime: 86400 sec
        objects: /0/1, /1/0, /3/0, /3200/0, /3303/0, /3311/0,
/3311/1,
```

From the example above, there is only one LWM2M client registered to the server and the client's name is "**IoT-U10plus4B00060E0DB5**". The ID of the client is **0**. There are seven IPSO objects reported by the LWM2M client.

IPSO objects are presented with **/{Object ID}/{Object Instance}**.

The following objects are supported by IoT-U10.

Table 4. Objects Supported by IoT-U10

/0/1	LWM2M Security	Instance 1	
/1/0	LWM2M Server	Instance 0	
/3/0	Device	Instance 0	
/3200/0	Digital Input	Instance 0	e.g., switch/button
/3303/0	Temperature Sensor	Instance 0	
/3311/0	Light Control	Instance 0	e.g., LED/light bulb
/3311/1	Light Control	Instance 1	e.g., LED/light bulb

The first three IPSO objects (/0/1, /1/0, and /3/0) are out of scope in this document. From the information above, the LWM2M client has one digital, one temperature sensor, and two light controls.

Object is a collection of **Resources**. A **Resource** is an atomic piece of information or interface that can be accessed or manipulated (for example, Read, Written or Executed) by user. Objects/Resources can be accessed with simple URI **/{Object ID}/{Object Instance}/{Resource ID}**, as in the following table.

Table 5. Objects/Resources

URI	Object ID	Instance ID	Resource ID
/3200/0/5500	Digital Input	Instance 0	Digital Input State
/3303/0/5700	Temperature Sensor	Instance 0	Sensor Value
/3311/0/5850	Light Control	Instance 0	On/Off

For more information about the definition of Object and Resource, refer to the following URL:

http://technical.openmobilealliance.org/Technical/technical-information/omna/lightweight-m2m-lwm2m-object-registry

Below are some of the basic examples on how the LWM2M client is accessed through the URIs as shown above.

To read the state of input button of the LWM2M client (Client ID **0**), enter the following command in LWM2M (Wakaama) server's command prompt:

```
> read 0 /3200/0/5500
OK
>
Client #0 3200/0/5500 : 2.05 (COAP_205_CONTENT)
4 bytes received:
    E1 15 7C 00   ..|.
```

To get the reading of temperature sensor of the LWM2M client:

```
> read 0 /3303/0/5700
OK
>
Client #0 3303/0/5700 : 2.05 (COAP_205_CONTENT)
7 bytes received:
    E4 16 44 41  FF 7E 00  ..DA.~.
```

To turn on the first LED of the LWM2M client:

```
> write 0 /3311/0/5850 1
OK
>
Client #0 3311/0/5850 : 2.04 (COAP_204_CHANGED)
```

To turn off the first LED of the LWM2M client:

```
> write 0 /3311/0/5850 0
OK
>
Client #0 3311/0/5850 : 2.04 (COAP_204_CHANGED)
```

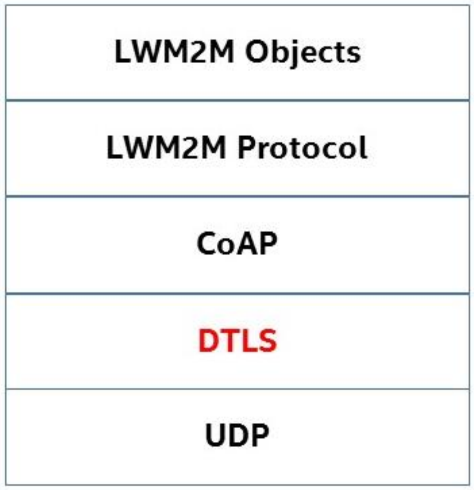
For more advance operations on LWM2M object's resources, refer to the LWM2M specification document.

Note: All data received from LWM2M client are in raw format (Hexadecimal). Refer to the 6LoWPAN device (for example, IoT-U10) manual for the interpretation of the data.

6.3 LWM2M Network with Datagram Transport Layer Security (DTLS) Encryption

Datagram Transport Layer Security (DTLS) is a protocol for securing network transmission for datagram-oriented transport such as UDP. It provides security for data interchanged between the LWM2M server and LWM2M client. Refer to Figure 9 below.

Figure 9. LWM2M Protocol Stack



DTLS can be enabled easily by changing the configuration file of LWM2M (Wakaama) Bootstrap Server. Just change the settings under **Endpoint** in **bootstrap_server.ini** as below:

```
[Endpoint]
Delete=/0
Delete=/1
Server=2
```

The settings above provision the LWM2M clients to connect to LWM2M server with id = 2. Refer to Section 6.1, the 2nd entry of the server in **bootstrap_server.ini** are configured using the DTLS Pre-Shared Key security.

Start the LWM2M (Wakaama) Bootstrap Server:

```
# cd /opt/netcontiki/sbin
# ./bootstrap_server -f bootstrap_server.ini
```

Then, start the LWM2M (Wakaama) server with DTLS:

```
# cd /opt/netcontiki/sbin
# ./lwm2mserver-dtls 2> messages.log
```

The LWM2M clients should connect to the LWM2M (Wakaama) server with secure encryptions and available for access through the LWM2M (Wakaama) server as described in Section 6.2.§

§