IoT Light control demo - Requirements Specification

Purpose of this document is to describe the requirements and instructions for work offered to “Ohjelmistoprojektit” -course.

Description of work

Project work consist of planning, implementation and testing of demo application which consist of Web user interface and several embedded devices. Embedded devices are running Mbed OS demonstrating its networking capabilities. Applications way be based on examples published by ARM on the Github.

Network traffic between nodes and user interface is implemented as IP multicast. Each node and user interface joins same multicast group which allows effective broadcasting of messages.

Description of the environment

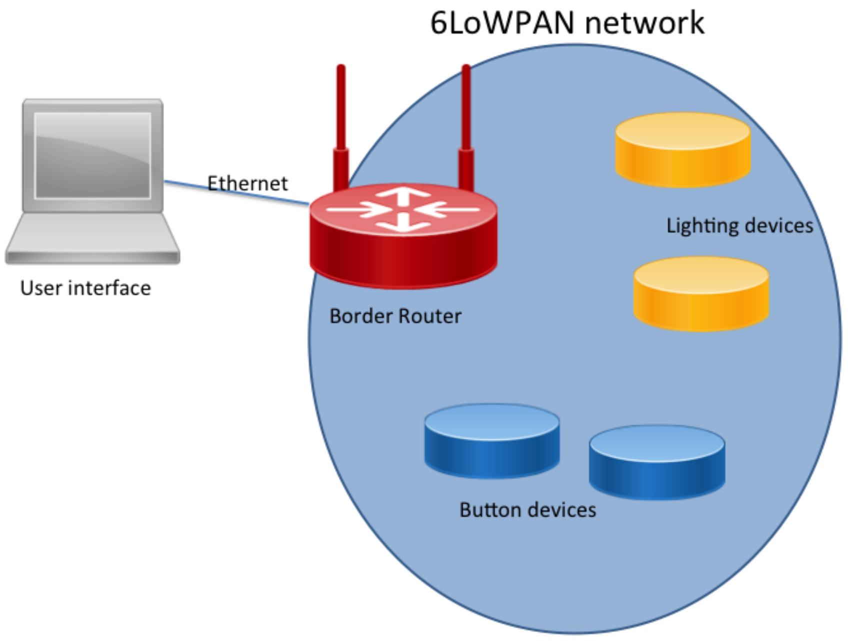

Figure 1: Network

Example platform, illustrated in figure 1, consist of following devices:

- Linux laptop running user interface
- Border Router device running on K64F development platform
- Few wireless nodes, having a button and IEEE 802.15.4 radio
- Few wireless nodes, having a light and IEEE 802.15.4 radio
- Ethernet cable connected between Border Router and laptop

Background information

Mbed OS

Arm Mbed OS is a operating system targeting for Internet of Things segment.

Mbed simplifies and speeds up the creation and deployment of IoT devices based on Arm microcontrollers.

For more information, see Getting Started guide from http://os.mbed.com

6LoWPAN based mesh

Please study the information from Mbed OS Handbook. https://os.mbed.com/docs/v5.6/tutorials/mesh.html

Mbed OS has fully mesh capable IPv6 networking stack and we provide two example applications for evaluating this functionality:

- Border Router
- Mesh example application

Those example applications can be used as a base for this demo and then extended to support functionalities required in this document.

Multicast traffic

Multicast traffic is a form of broadcast that flows for only nodes which have registered to specific multicast group.

For background information, please study https://en.wikipedia.org/wiki/IP_multicast

When enabling the multicast traffic, modifications are required for both node and border router software. Also the UI software running on Linux requires registration into the multicast groups. For following examples, we use `ff15::abba:abba` as our multicast group address.

Posix example for registering into the multicast group

Following example registers to multicast group by first converting the group address to binary, using `inet_pton()` then creating request structure used for `setsockopts()` call. Example also shows how to start listening for incoming packets using `bind()` and `recvfrom()` calls.

```
#define MULTICAST_GROUP "ff15::ABBA:ABBA"

int main(void)
{

    int s = socket(AF_INET6, SOCK_DGRAM, 0);

    struct sockaddr_in6 saddr;
    struct sockaddr_in6 addr_from;

    struct in6_addr mcast;
    if (inet_pton(AF_INET6, MULTICAST_GROUP, &mcast) != 1) {
        perror("inet_pton()");
        return 1;
    }

    saddr.sin6_family = AF_INET6;
    saddr.sin6_addr = in6addr_any;
    saddr.sin6_port = htons(PORT);
    if (bind(s, (struct sockaddr *)&saddr, sizeof saddr) == -1) {
        perror("bind()");
        return 1;
    }

    struct ipv6_mreq mreq = {
        .ipv6mr_multiaddr = mcast,
        .ipv6mr_interface = 0,
    };
    setsockopt(s, IPPROTO_IPV6, IPV6_JOIN_GROUP, &mreq, sizeof mreq);

    printf("Listening.\n");
    while (1) {
        char buf[100];
        socklen_t alen = sizeof addr_from;
        int len = recvfrom(s, buf, 100, 0, (struct sockaddr *)&addr_from, &alen);
        if (len == -1) {
            perror("recvfrom()");
            return -1;
        }
        printf("Received %s\n", buf);
    }
    close(s);
}
```

Mbed OS example for registering into the multicast group

For Mbed OS the API is similar as the standard Posix one.

```
static void init_socket()
{
    my_socket = new UDPSocket(network_if);
    my_socket->set_blocking(false);
    my_socket->bind(UDP_PORT);
    my_socket->setsockopt(SOCKET_IPPROTO_IPV6, SOCKET_IPV6_MULTICAST_HOPS, &multicast_hops,

    ns_ipv6_mreq_t mreq;
    memcpy(mreq.ipv6mr_multiaddr, multi_cast_addr, 16);
    mreq.ipv6mr_interface = 0:

    socket_setsockopt(my_socket, SOCKET_IPPROTO_IPV6, SOCKET_IPV6_JOIN_GROUP, mreq, sizeof m

    my_socket->sigio(callback(handle_socket));
}
```

Enabling multicast routing in the Border Router

Insert following code snipped into `borderrouter_tasklet.c` file in function `start_6lowpan()`

```
        multicast_fwd_add(net_6lowpan_id, multicast_addr, 0xFFFFFFFF);
        multicast_fwd_add(backhaul_if_id, multicast_addr, 0xFFFFFFFF);
        multicast_fwd_set_proxy_upstream(backhaul_if_id);
```

Multicast address is defined in `mbed_app.json` configuration file.

Requirements

Requirements for User Interface

1. Joins a multicast group address which is defined for the demo.
2. Listens for incoming advertisements from nodes.
3. Displays a list of found nodes for user, separate by function (button/switch, light)
4. Allows user to create connection between a button and a light.

5. When creating connection, send a unicast message for node telling which button to listen to.

Requirements for Node applications

Node may refer to a button or a lighting node which both contain very similar software.

1. Application connects to 6LoWPAN-ND based network
2. Application receives IPv6 address from network
3. Once connected node joins to a specified multicast group
4. Node listens for UDP messages on port 1030

Requirements for Light node

This is a node which contains a light bulb or LED connected to it.

1. When receiving a controlling message from the user interface, records the address which this node should listen to for incoming switch commands.
2. When receiving a multicasted message from a button node, compares the source address to one received from UI. If matches, switches the light.