Laboratorio 2:

Comunicación UDP en IPv6

Redes de Sensores Inalámbricos
IIE, Facultad de Ingeniería, UDELAR

Tabla de Contenidos.

1 Introducción

2 Objetivo
    2.1 Objetivos de aprendizaje

3 Lecturas y actividades previas

4 Procedimientos y Tareas

5 Entregables

6 Referencias

1 Introducción

En este laboratorio se realizará una primera aproximación al uso de comunicación inalámbrica entre nodos de una red de sensores utilizando Contiki OS. Se utilizarán diferentes canales de radio y se experimentará con mensajes multicast y unicast.

El trabajo en este laboratorio es grupal, sin embargo sugerimos que cada participante trabaje con su propia computadora, implementando las funcionalidades discutidas dentro del grupo.

2 Objetivo

El objetivo de este laboratorio es familiarizarse con el envío de mensajes de radio en Contiki, y profundizar en el uso de Contiki OS. Se verá también cómo modificar los parámetros por defecto del sistema operativo.

2.1 Objetivos de aprendizaje

Se espera que el estudiante logre como mínimo:

- Manejo básico de make para compilación de aplicaciones de Contiki OS (redefinición de parámetros por defecto).
- Uso básico de radio y configuración de stack de comunicaciones.
- Comprensión del uso de la API UDP de Contiki OS .

3 Lecturas y actividades previas

Previo al laboratorio el estudiante deberá leer las siguientes secciones de [1]:

- Sección 4.1 - Addressing and Radio Frequency Basics.
- Sección 4.5.1 - The UDP API.
- Sección 4.5.2 - UDP Link-Local multicast example.

4 Procedimientos y Tareas

1) Prueba de envío de mensajes UDP multicast.
   i) Copiar el archivo: *./contiki/examples/ipv6/simple_udp_rpl/broadcast-example.c* a una nueva carpeta, por ejemplo: *./contiki/examples/lab2*.
   ii) Crear un archivo *project-conf.h* e incluirlo en el `Makefile`. El mismo debe desactivar el protocolo de ruteo para facilitar el uso de la API UDP. Para lograr este objetivo, se deben agregar las siguientes líneas:

```
#ifdef UIP_CONF_IPV6_RPL
#undef UIP_CONF_IPV6_RPL
#endif

#define UIP_CONF_IPV6_RPL 0
```

   iii) Modificar el código del callback de recepción para que cada vez que se reciba un nuevo paquete, imprima:
      1) el contenido de los mensajes recibidos,
      2) la dirección del nodo que envía el mensaje,
      3) la dirección a la cual es enviado el mensaje.

Para imprimir las direcciones puede usar la función auxiliar:

```
uip_debug_ipaddr_print(&addr);
```

Se recomienda imprimir un fin de línea a continuación para mejor visualización.

iv) Modificar el código del programa para que el texto a enviar sea “Grupo X”, donde X es el número de grupo.

v) Cargar el programa en dos nodos y verificar que:
   1) La dirección del nodo que envía el mensaje es correcta.
   2) La dirección a la que se envía el mensaje es una dirección multicast válida.
   3) Se imprime correctamente el contenido de los mensajes del otro nodo.
   4) Verificar y tomar nota de las direcciones MAC e IPv6 con las que quede configurado cada nodo. ¿De qué tipo es la dirección IPv6? ¿Puede deducir cómo se genera a partir de la dirección MAC?

Nota: Si se trabaja en simulación, COOJA asignará automáticamente el NODE ID en tiempo de compilación. Si se trabaja en hardware, puede especificarlo manualmente al momento de compilar y cargar el nodo de la siguiente manera, donde XX es el ID asignado al nodo:

```
make TARGET=zoul BOARD=remote-revb NODEID=0xXX lab2.upload
```

2) Configuración y uso de canal exclusivo de cada grupo
   i) Configurar el canal de la radio para que cada grupo utilice un canal exclusivo. El número de canal del grupo X será 10+X.
   Se puede agregar el siguiente código para verificarlo:

```
static uint8_t ch_num;
NETSTACK_RADIO.get_value(RADIO_PARAM_CHANNEL,&ch_num);
printf("RF_CHANNEL: %d\n", ch_num);
```

   ii) Configurar un nodo con el canal del grupo, y dejar el otro en el canal por defecto. Comprobar que en esa situación, los nodos no pueden comunicarse.
   iii) Configurar los dos nodos en el canal del grupo. Comprobar que en esa situación, los nodos pueden comunicarse correctamente.

3) Envío de mensajes UDP unicast
   i) Modificar el programa para que mande mensajes unicast a otro nodo. Para este objetivo, puede ser útil la siguiente función para declarar el valor de una dirección IPv6:

```
uip_ip6addr(&addr, 0x1111, 0x2222, 0x3333, 0x4444, 0x5555,
0x6666, 0x7777, 0x8888);
```

   ii) Verificar que se reciben mensajes correctamente y que la dirección de destino de los mensajes es una dirección unicast.

5 Entregables

Se deberá entregar un archivo comprimido que contenga lo siguiente:

1. carpeta Tarea 1 incluyendo Makefile, project-conf.h, .c, .h,
2. carpeta Tarea 2 incluyendo Makefile, project-conf.h, .c, .h,
3. carpeta Tarea 3 incluyendo Makefile, project-conf.h, .c, .h.

Es imprescindible que el código esté comentado adecuadamente: cada línea o líneas consecutivas que sean agregadas, deberán ser acompañadas de un comentario que explique su propósito.

En caso de haber realizado simulaciones, los archivos de simulación generados se agregarán al archivo comprimido a entregar.

Las entregas se realizarán a través de la **plataforma EVA del curso**.

6 Referencias

[1] https://github.com/marcozennaro/IPv6-WSN-book/blob/master/Releases