Laboratorio 4: Capa de red

1 Introducción

En este laboratorio se creará una red IPv6/RPL formada por un nodo *Border Router* y varios nodos *Cliente* para estudiar su formación y mantenimiento. Se simulará la red utilizando la herramienta Cooja y observando los paquetes intercambiados por los nodos se analizará el camino que recorre cada paquete y los tiempos involucrados.

2 Objetivos

2.1 Objetivos generales

Comprender el funcionamiento del armado y mantenimiento de rutas en una RSI mediante el análisis de los mensajes intercambiados por los nodos y su interpretación de acuerdo con lo visto en las clases teóricas de RPL.

2.2 Objetivos específicos

- Utilizar el simulador Cooja para analizar el comportamiento de una red, visualizando los mensajes intercambiados.
- Utilizar Wireshark como herramienta complementaria de análisis de tráfico de una red.
- Interpretar los distintos mensajes de RPL intercambiados por los nodos.
- Ver cómo la posición relativa de un nodo, para cierta topología de enrutamiento, impacta en el tiempo de respuesta.
- Ver el comportamiento de una red real a nivel de RPL.

3 Fundamentos

Durante el laboratorio se analizará una red formada por 3 nodos *Cliente* y un *Border Router*. Los nodos *Cliente* enviarán mensajes a otro nodo destino. La dirección de destino podrá ser modificada con la red funcionando.

Se recomienda fuertemente estudiar la documentación de RPL donde se repasan los conceptos más importantes de RPL, se describen las implementaciones disponibles (RP Classic y RPL Lite que usaremos en el laboratorio) y los modos de operación. Finalmente se describe cómo se forma la topología de la red y los parámetros de configuración (no todos relevantes para el análisis realizado en el laboratorio). De manera complementaria, también leer y tener presente el Tutorial RPL para la realización del laboratorio.

Por otra parte se recomienda repasar el material de referencia Running Contiki-NG in Cooja, teniendo en cuenta que al utilizar docker el simulador se ejecuta simplemente con el comando `cooja` (ya utilizado en el laboratorio 2) y estudiar el uso del Border Router en el Tutorial RPL Border Router.

4 Actividades

4.1 Armado de una red RPL

En esta parte vamos a simular una red RPL utilizando algunas de las ideas ya vistas en los laboratorios 2 y 3. Esta simulación será el punto de partida para las próximas tareas. Para su realización se debe tener en cuenta lo siguiente:

1. La simulación deberá contener un nodo *Border Router* de tipo *cooja* con el siguiente código:

   `contiki-ng/examples/rpl-border-router/border-router.c`

   Este *border router*, al igual que se realizó en el Laboratorio 3, debe propagar el prefijo indicado por `tunslip6` (`fd00`).

2. La red contendrá además tres nodos *Cliente* realizados a partir del código *unicast* desarrollado en el Laboratorio 2, los que deberán tener habilitado el Shell.

   Tener en cuenta que se debe modificar el código de estos nodos para que envíen paquetes a una IP con prefijo `fd00` (puede ser la de cualquier nodo *Cliente* pero debe existir en la red). Recordar que la IP de los nodos puede consultarse mediante el *Shell*.

3. Modificar el radio de interferencia de los nodos para que sea pequeño en relación al alcance. Para ello hay que hacer click izquierdo sobre cualquiera de los nodos y presionar donde dice “Change transmission ranges”. Bajar el INT range a 55 m.

Verificar el correcto armado de la red RPL, corroborando que todos los nodos logran enviar y recibir paquetes del nodo de destino, en particular cuando esto implica realizar uno o más saltos a través de nodos vecinos (variar la ubicación relativa de los nodos y analizar lo que sucede en cada caso).

4.2 Modificación de dirección destino de envío

Al código de los nodos se le adicionará la funcionalidad de modificación de la dirección de destino a la que envía los mensajes, utilizando el botón de usuario. El nodo al que deben ser enviados los paquetes tendrá el ID=n, donde n es el número de veces que se presione el botón[1]. Es decir, cuando el usuario quiera que los mensajes se envien al nodo 3 debe presionar 3 veces el botón de usuario. Si luego quiere cambiar el destino al nodo 2, presionará 2 veces. Para ello:

1. Agregar un nuevo proceso al código de los nodos que cuente el número de veces que cada usuario presiona el botón. Para detectar cuántas veces se presionó, se implementará un timeout luego del cual se dará por finalizado el ingreso del usuario. Se sugiere repasar las actividades del laboratorio 1.

2. El nuevo proceso creado, cada vez que alguien realiza una nueva configuración de destino, envía el número de veces que se presionó el botón mediante *post* de un evento al proceso principal.

3. Comprobar que luego de realizado este cambio es posible cambiar dinámicamente el destino de los mensajes mientras se ejecuta la simulación. Investigar cómo hacer para presionar el botón de un nodo simulado en Cooja, y limitar la velocidad de la simulación a *1X* para que sea posible probar la funcionalidad del botón.

[1] No interesa el número total de veces que se presionó el botón desde el inicio del programa, sino el número de veces que el usuario presiona cada vez que quiere configurar el destino.

4.3 Análisis de la topología: examinando paquetes

Utilizando la simulación anterior se pide:

1. Analizar los mensajes RPL intercambiados entre los nodos (en la ventana Radio messages) y dibujar un diagrama del grafo (DODAG) de la red indicando para cada nodo el rank y el padre preferido elegido.
2. Observar el mecanismo utilizado por el root del DODAG (border-router en este caso) para propagar el prefijo global.
3. Pulsar n veces el botón de un nodo alejado del border router. Observar la secuencia de paquetes intercambiados por los nodos verificando la ruta utilizada hasta llegar al nodo con ID=n.

Nota: Se recomienda utilizar Wireshark para analizar los paquetes a partir del archivo PCAP generado por Cooja. Para ello en la ventana *Radio messages* de *Analyzer* seleccionar "6lowpan Analyzer with PCAP". Los archivos PCAP (`*.pcap`) se guardan en el directorio donde se ejecutó el comando cooja, usualmente en `./contiki-ng/`.

4.4 Análisis de la topología: informacion local

En esta parte se verificará la topología identificada en la parte anterior a partir de la información local de cada nodo a través del servicio *Shell*.

1. Experimentar, ejecutando los diferentes comandos de shell, en particular 'routes' para determinar el padre preferido.
2. Determinar la topología de la red averiguando el padre preferido de cada nodo.

4.5 Análisis de TTL

Desde una aterminal del PC realizar pings al *Border Router* y a diferentes nodos de la red, ejecutando por ejemplo:

```
ping fd00::20X:X:X:X'
```

donde x es el ID del nodo al que se le quiere hacer ping (válido para simulación).

1. Observar las diferencias de tiempos de respuesta entre border-router y el resto de los nodos.
2. ¿Qué valores de período de vida (TTL) se observan para los diferentes nodos de la red? ¿Qué pasa con el tiempo de respuesta?
3. Identificar los paquetes ICMP correspondientes a los pings que se hicieron.

4.6 Prueba en hardware

En esta parte se creará una red en hardware físico utilizando todos los nodos disponibles de la clase[2]. Los docentes dispondrán de un nodo extra que tendrá el rol de *Border Router*. Todos los grupos deberán usar el canal por defecto (0 en CC1350, 26 en CC2650).

Los grupos deberán utilizar el código de los nodos clientes de este laboratorio. Se recomienda enviar datos con baja frecuencia para no generar muchos paquetes de aplicación.

Se realizará colectivamente en el pizarrón diagramas de topología de la red

[2]En función de los tipos de nodos presentes en la clase se podrán crear dos redes, una red en la banda 2.4 GHz (CC2650) y otra red en banda sub-GHz (CC1350).

5 Entregables

Se deberá entregar un archivo comprimido que contenga lo siguiente:

1. Carpeta con el código de los nodos clientes incluyendo Makefile, project-conf.h, .c, .h y simulaciones .csc.
2. El diagrama de la red indicando el rank de cada nodo y el padre preferido.
3. Un archivo de texto incluyendo los resultados de los pings a distintos nodos de la red (Análisis de TTL) y las diferencias observadas.
4. Carpeta con las capturas de pantalla más relevantes de las tareas 4.3, 4.4 y 4.5 con el propósito de adelantar lo que mostrarán en la defensa (no más de una por punto/ítem pedido).

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.

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

6 Referencias

- Programming Contiki-NG: RPL
- Tutorial RPL
- Tutorial Running Contiki-NG in Cooja
- Tutorial RPL Border Router
- Tutorial Shell

Los materiales de este curso fueron parcialmente financiados por:

NEON Network of Competence on Internet of Things

Co-funded by the Erasmus+ Programme of the European Union