Laboratorio Remoto para Dispositivos IIoT

Manual Arquitectura

2024

Índice

1. Introducción

El LRDIIoT (Laboratorio Remoto para Dispositivos IIoT) es una plataforma que permite el acceso remoto a entornos de desarrollo para utilizar placas IIoT, concretamente las openmote-cc2538. Vale aclarar que estas placas pueden ser programadas y comunicarse entre ellas, lo que permite realizar diversos experimentos desde el punto de vista de tráfico, protocolos, arquitectura y comunicación.

En este documento se desarrollará la arquitectura y forma de comunicación utilizada por las placas openmote-cc2538 con el objetivo de aumentar el entendimiento respecto al funcionamiento de todo el sistema.

2. Placas openmote-cc2538

Es un desarrollo de hardware y un prototipo de plataforma para el Internet de las cosas industrial (IIoT), específicamente para investigadores y desarrolladores que desean trabajar con redes de área de campo inalámbricas de largo alcance y bajo consumo, basadas en el stack IPv6.

Figura 1: Placa Openmote utilizada

Esta construido utilizando el microcontrolador CC2538 ARM-Cortex-M3 de Texas Instrument. Entre sus características se cuenta la posibilidad de operar de forma simultanea en las bandas de 2.4GHz y la banda ISM de 868/915MHz con un soporte completo del standard IEEE 802.15.4, incluida la modulación MR-OFDM del IEEE 802.15.4g-2012.

La placa identificada como OpenMote-CC2538 generalmente se la encuentra anexado a otros desarrollos que le permiten mayor versatilidad:

Figura 2: Conjunto Openmote

De izquierda a derecha se pueden identificar el OpenMote-cc2538 propiamente dicho, OpenBattery, OpenBase y OpenUSB.

2.1. Características de la plataforma

La placa principal incluye 4 LEDs, un botón de usuario para propósitos de debug y un reset de hardware. Adicionalmente suelen contar con tres sensores identificados como sht21 (sensor de humedad y temperatura), max44009 (sensor de luminosidad) y adx1346 (giróscopo).

Gracias a el chip FTDI FT2232H que permite la conversión Serie-USB se puede realizar la comunicación con el dispositivo sin necesidad de contar con un puerto serie desde el equipo que se realizará la programación. Adicionalmente, el chip FTDI permite la programación del CC2538 de forma directa usando el bootloader interno y el script cc2538-bsl de Python.

Se incluye dos conectores de antena SMA para el rango de los sub-GHz y una antena de 2.4 GHz. El conector de la antena de sub-GHz está directamente conectada al chip de radio AT86RF215. La antena de 2.4 GHz realiza una multiplexación utilizando un switch de RF entre el CC2558 y el AT86RF215.

Por último, es posible utilizar las placas OpenMote con otros proyectos open-source como FreeRTOS, RIOT, Contiki y recientemente Contiki-ng (este es el utilizado en el presente proyecto).

3. Protocolos y estándares utilizados

La comunicación entre los motes es inherentemente inalámbrica, y generalmente se los conoce como red de sensores inalámbricos (WSN – Wireless Sensor Network). WSN hace referencia al conjunto de dispositivos autónomos, interconectados entre sí y distribuidos en un entorno, capaces de tomar datos del medio, procesarlos y transmitirlos al resto de la red.

Los elementos que constituyen las redes de sensores inalámbricos se denominan nodos.

La comunicación entre nodos hace uso de IPv6 gracias a las características que este protocolo presenta para redes de cientos o miles de dispositivos que permite que la red escale fácilmente sin necesidad de utilizar protocolos como NAT o DHCP, típicos en redes IPv4.

En redes WSN es común utilizar el estándar IEEE 802.15.4, que garantiza un bajo consumo energético en las redes de sensores inalámbricas, esto se conoce como LowPAN “Low Power Personal Area Networks”.

Sin embargo, IPv6 ha sido diseñado para infraestructuras de internet con un gran ancho de banda, ya que generalmente los dispositivos en los que se utiliza este protocolo de red no presentan las limitaciones típicas de una red de sensores WSN. Atendiendo a esta problemática, la IETF (Internet Engineering Task Force) desarrolló el estándar 6LoWPAN, que permite la transmisión de paquetes IP en redes basadas en el estándar IEEE 802.15.4. IETF lo ha conseguido introduciendo una capa de adaptación entre las capas de enlace y de red del modelo OSI y reestructurando el formato de los paquetes IPv6.

Figura 3: Stack utilizado

3.1. Estándar IEEE 802.15.4

Es un estándar de comunicación inalámbrica de corto alcance diseñado para redes de área personal inalámbricas que requieren bajo consumo de energía y baja tasa de transmisión de datos.

3.1.1. Características

Su objetivo principal es brindar una tecnología de comunicación inalámbrica de bajo consumo energético y baja tasa de bits para aplicaciones de redes de sensores inalámbricos (WSN), control industrial, automatización del hogar, salud conectada y otras aplicaciones de IoT.

- Capa Física: IEEE 802.15.4 define las especificaciones para la capa física que incluyen frecuencias de operación en las bandas de 2.4 GHz, 868 MHz y 915 MHz, modulación de señales, tasas de transmisión de datos (hasta 250 kbps), rangos de alcance típicos y requisitos de consumo de energía.
- Capa de Enlace de Datos: La capa de enlace de datos definida por IEEE 802.15.4 proporciona mecanismos de acceso al medio eficientes, como CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) y GTS (Guaranteed Time Slot), para garantizar una comunicación confiable en entornos con múltiples dispositivos.
- Modos de Operación: El estándar IEEE 802.15.4 admite diferentes modos de operación, incluyendo el modo de dispositivo simple (sin coordinación de red), el modo de coordinador de pan (Personal Area Network) que gestiona las actividades de red y el modo de dispositivo final (end device) que se comunica a través de un coordinador.
- Topologías de Red: IEEE 802.15.4 admite varias topologías de red, como redes en estrella, redes de malla (mesh networks) y redes de árbol, lo que permite diseñar redes flexibles y adaptativas según los requisitos de la aplicación.
- Seguridad: El estándar incluye mecanismos de seguridad como cifrado AES (Advanced Encryption Standard) de 128 bits, autenticación y control de acceso para proteger la integridad y la confidencialidad de los datos transmitidos.
- Aplicaciones: IEEE 802.15.4 se utiliza ampliamente en aplicaciones de IoT, WSN, domótica, monitorización y control industrial, sistemas de seguimiento y localización, dispositivos médicos y otras aplicaciones que requieren comunicaciones inalámbricas eficientes y de baja potencia.

3.2. IPv6

IPv6 o protocolo de internet versión seis (“internet Protocol version six”) es la versión más reciente del protocolo de internet que se utiliza para identificar y localizar dispositivos en redes de comunicaciones.

3.2.1. Características

- Direcciones IPv6: IPv6 utiliza direcciones de 128 bits, a diferencia de IPv4 que utiliza direcciones de 32 bits. Esto permite un espacio de direcciones mucho más grande, lo que resuelve el problema de agotamiento de direcciones IPv4.
- Formato de Direcciones: Una dirección IPv6 se representa en formato hexadecimal separado por dos puntos. Por ejemplo, una dirección IPv6 típica podría ser 2001:0db8:85a3:0000:0000:8a2e:0370:7334.

- Notación de Prefijo: En IPv6, los prefijos de red y subred se especifican utilizando la notación de prefijo CIDR (Classless Inter-Domain Routing). Por ejemplo, 2001:db8::/32 indica un prefijo de red de 32 bits.
- Tipos de Direcciones IPv6:
    - Dirección Unicast: Identifica una única interfaz de red.
    - Dirección Multicast: Utilizada para enviar paquetes a múltiples interfaces de red.
    - Dirección Anycast: Identifica un grupo de interfaces y el paquete se envía a la interfaz más cercana dentro del grupo.
- Mayor Espacio de Direcciones: Permite un número mucho mayor de dispositivos conectados a Internet.
- Autoconfiguración de Direcciones: IPv6 tiene la capacidad de autoconfigurarse, lo que simplifica la configuración de direcciones en dispositivos.
- Soporte para Seguridad: IPv6 incluye soporte integrado para IPSec (Internet Protocol Security) que proporciona seguridad en las comunicaciones.
- QoS (Quality of Service): IPv6 incluye soporte para calidad de servicio para priorizar ciertos tipos de tráfico.

3.3. El problema de IPv6 e IEEE 802.15.4

El estándar IEEE802.15.4 impone una restricción respecto a la información que se desea transmitir. Al tratarse de un medio propenso a pérdidas y poco fiable, es razonable tener un límite en la longitud de los paquetes a enviar. Por tanto, existe un tamaño máximo de los paquetes, o Unidad Máxima de Transmisión (MTU), fijado por el estándar. Este tamaño máximo es de 127 bytes. El problema aparece cuando IPv6 obliga a soportar MTUs de al menos 1280 bytes, llegando a un tamaño de carga normalmente de hasta 64 kB, además de la posibilidad de transmitir jumbogramas que pueden llegar a longitudes de hasta 4 GB menos 1 byte en la carga (payload).

Teniendo en cuenta lo anterior, surge la necesidad de trocear el paquete a transmitir y enviarlo por partes. Es aquí donde aparecen protocolos de fragmentación y desfragmentación, como 6LoWPAN.

Mediante 6LoWPAN el paquete se fragmenta en distintos trozos, siendo necesario imponer un orden para poder desfragmentarlo y recomponerlo en el destino. Además de esta trazabilidad en los fragmentos, 6LoWPAN presenta otra ventaja en forma de ahorro de espacio: la compresión de encabezados.

Las capas que recibe el stack de 6LoWPAN (IPv6 y el protocolo de transporte elegido) tienen encabezados relativamente grandes y pesados, algo que resulta difícil de mover tal cual si se quiere respetar el tamaño límite impuesto por la capa física. Para ello es capaz de comprimir dichos encabezados y por lo tanto, la cantidad de bytes que se envía. Normalmente se utiliza IPHC (IP header compression) para los encabezados IP y NHC (Next Header compression) para los de TCP/UDP. La ventaja de este método

de compresión es que toda la información necesaria para descomprimir el encabezado está contenida en el propio paquete enviado, y por lo tanto cada paquete se puede comprimir y descomprimir de forma independiente sin necesidad de otros paquetes adicionales.

En la siguiente imagen se observa la compresión realizada en la que se pasa de 48 bytes de encabezado IP y UDP a solamente 7 bytes, sin perder información. Gran parte de dicha compresión se basa en que todos los nodos se encuentran en la misma red y por lo tanto existe mucha información repetida y no necesaria.

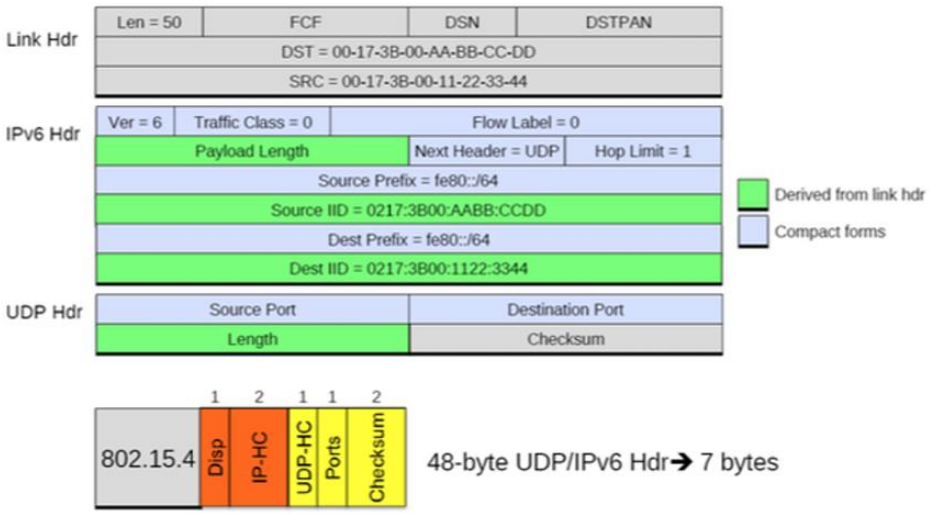


Figura 4: Compresión de encabezado

3.4. Protocolo RPL

RPL (Routing Protocol for Low-Power and Lossy Networks) es un protocolo diseñado específicamente para redes de baja potencia y alta pérdida de paquetes (LLN por sus siglas en inglés). Estas redes suelen estar compuestas por dispositivos con recursos limitados, como sensores y actuadores, que se comunican a través de tecnologías como IEEE 802.15.4 (usada en redes inalámbricas de sensores).

3.4.1. Características

- Objetivo Principal: proporcionar un enrutamiento eficiente y confiable en redes LLN, optimizando el uso de energía y adaptándose a las condiciones de alta variabilidad y pérdida de paquetes.
- Jerarquía de Enrutamiento: RPL organiza los nodos de la red en una jerarquía de rutas llamada DODAG (Destination-Oriented Directed Acyclic Graph). Esta jerarquía permite el enrutamiento eficiente desde los nodos hoja hasta el nodo raíz (root), minimizando el consumo de energía y la sobrecarga de enrutamiento.

- Padre y Hijo en DODAG: Cada nodo en la jerarquía DODAG tiene un padre (parent) y puede tener varios hijos (children). Los nodos hijos envían sus datos al nodo padre, que luego los reenvía hacia arriba en la jerarquía, siguiendo el camino óptimo hacia el nodo raíz.
- Métricas de Enrutamiento: RPL utiliza métricas de enrutamiento para determinar las rutas óptimas en la red. Estas métricas pueden incluir la calidad de la conexión, el consumo de energía, el número de saltos, entre otros.
- Soporte para Topologías Múltiples: RPL es flexible y puede adaptarse a diferentes topologías de red, incluyendo topologías en estrella, en malla y en árbol.
- Compatibilidad con IPv6: RPL está diseñado para trabajar con IPv6, lo que lo hace adecuado para su implementación en entornos IoT (Internet of Things) y redes de sensores donde IPv6 es ampliamente utilizado.

3.5. Protocolo CoAP

CoAP (Constrained Application Protocol) es un protocolo de software que se encuentra en el nivel de capa de aplicación del modelo OSI y está apuntado a correr en dispositivos simples, permitiendo que puedan comunicarse sobre internet. Su topología de red se asemeja a una malla, en la que pueden existir múltiples servidores y clientes que interaccionan entre si de forma conjunta o independiente, de acuerdo al formato en que se hayan programado y configurado.

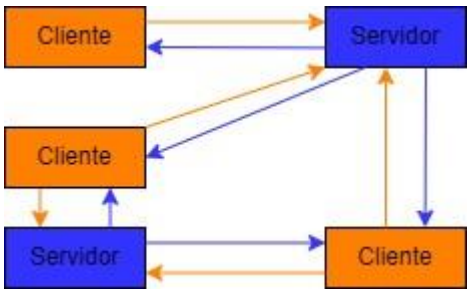


Figura 5: Formato comunicación CoAP

3.5.1. Características

- Diseñado para Dispositivos Constrained: CoAP está diseñado para su uso en dispositivos con recursos limitados, como sensores, actuadores y otros dispositivos IoT que operan en redes de baja potencia y ancho de banda limitado.
- Basado en UDP: CoAP se basa en el protocolo de transporte UDP (User Datagram Protocol), lo que lo hace liviano y adecuado para aplicaciones en tiempo real y con baja latencia.

- Protocolo RESTful: CoAP sigue el modelo de arquitectura REST (Representational State Transfer), lo que significa que utiliza los métodos HTTP (GET, POST, PUT, DELETE) para la comunicación entre cliente y servidor.
- URI y Operaciones: En CoAP, los recursos se identifican mediante URI (Uniform Resource Identifier) y se pueden realizar operaciones como GET (obtener), POST (crear), PUT (actualizar) y DELETE (eliminar) sobre esos recursos.
- Soporte para Observación: CoAP incluye soporte nativo para la observación de recursos, lo que permite a los clientes suscribirse a cambios en los recursos y recibir notificaciones cuando estos cambian.
- Seguridad: CoAP proporciona mecanismos de seguridad como DTLS (Datagram Transport Layer Security) para proteger las comunicaciones entre cliente y servidor, asegurando la autenticación y la confidencialidad de los datos.
- Eficiencia en Reducción de Overhead: CoAP está diseñado para reducir el overhead en las comunicaciones, utilizando encabezados y formatos compactos para minimizar el uso de ancho de banda y energía.
- Integración con IPv6: CoAP se integra fácilmente con IPv6, lo que lo hace compatible con el despliegue de redes IoT basadas en IPv6 y con la comunicación en entornos de Internet de las Cosas.

3.6. Protocolo MQTT

MQTT (Message Queuing Telemetry Transport) es un protocol de mensajería ligero y eficiente diseñado para las comunicaciones entre dispositivos en redes IoT y aplicaciones M2M. A diferencia de CoAP, MQTT utiliza un modelo de publicación/suscripción en el cual los dispositivos pueden publicar mensajes en temas (tópicos) y suscribirse a otros temas para recibir mensajes relevantes. Esto le da versatilidad para que sea una comunicación flexible y rápidamente escalable entre múltiples dispositivos. Su desventaja es que el bróker debe estar siempre en funcionamiento y si por algún motivo se interrumpe la comunicación con este, el resto de los dispositivos no tendrán posibilidad de desplazarse por otra ruta.

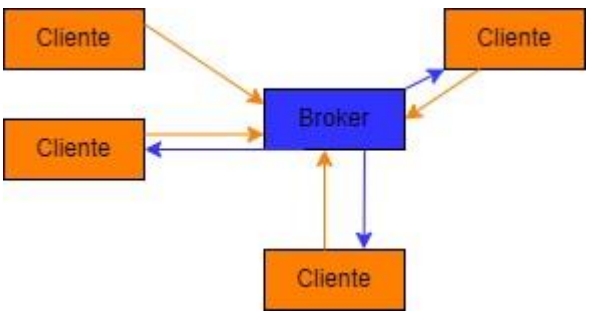


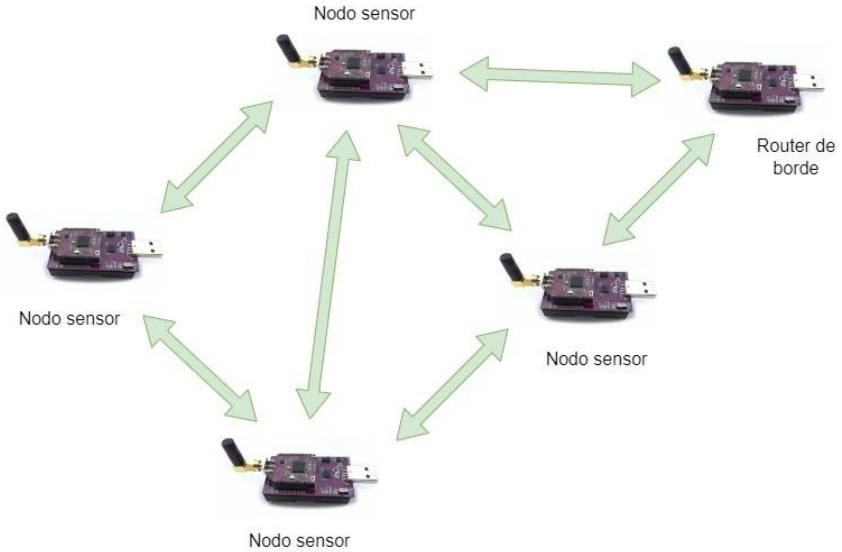
Figura 6: Formato protocolo MQTT

3.6.1. Características

- Protocolo Ligero y Eficiente: MQTT está diseñado para ser ligero y eficiente en términos de uso de ancho de banda y recursos de red. Utiliza un protocolo binario compacto sobre TCP/IP, lo que lo hace adecuado para dispositivos con recursos limitados, como sensores y actuadores.
- Calidad del Servicio (QoS): MQTT admite varios niveles de calidad de servicio (QoS) para garantizar la entrega confiable de mensajes. Los niveles de QoS incluyen QoS 0 (entrega a lo sumo una vez), QoS 1 (entrega al menos una vez) y QoS 2 (entrega exactamente una vez).
- Retención de Mensajes: MQTT permite la retención de mensajes en temas. Esto significa que un mensaje publicado en un tema puede ser retenido por el servidor MQTT y entregado a los suscriptores que se conecten posteriormente.
- Última Voluntad y Testamento (LWT): MQTT incluye soporte para la configuración de un mensaje de "última voluntad y testamento" (Last Will and Testament), que se envía automáticamente por el servidor MQTT a los suscriptores cuando un cliente se desconecta inesperadamente
- Seguridad: MQTT puede implementar medidas de seguridad como TLS/SSL para cifrar las comunicaciones entre cliente y servidor, garantizando la confidencialidad e integridad de los datos transmitidos.
- Escalabilidad y Flexibilidad: MQTT es altamente escalable y puede adaptarse a diferentes escenarios y topologías de red. Puede ser implementado en arquitecturas centralizadas, distribuidas o híbridas, según las necesidades de la aplicación.
- Integración con Protocolos Web: MQTT tiene integración con protocolos web como WebSocket, lo que facilita la comunicación bidireccional entre aplicaciones web y dispositivos IoT a través de MQTT.

4. Router de borde

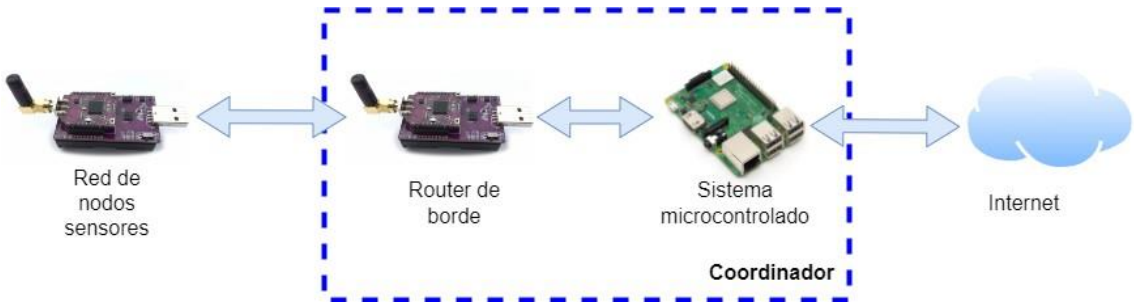
Consiste en un nodo que actúa como interfaz entre la red de nodos IPv6 y el exterior. A través de este se realiza la comunicación con los nodos sensores, administra y registra los dispositivos, entre otros.



5. Nodo Coordinador

El coordinador actúa como interfaz entre el equipo inalámbrico y la red de internet, es decir que funciona como un Gateway. Además, se encarga de realizar las solicitudes de datos a los nodos sensores para luego ser almacenados en una base de datos, y posteriormente ser visualizados.

El router de borde forma parte del bloque coordinador, mientras que la parte restante que trabaja en conjunto con el router de borde es un sistema embebido, que se encarga de realizar las solicitudes de datos, procesarlos y transmitirlos a la base de datos. Generalmente es un microcontrolador, en nuestro caso es una placa RaspberryPI.



6. Bibliografía

- Vilajosana X., Tuset P., Watteyne T. y Pister K. *OpenMote: Open-Source Prototyping Platform for the Industrial IoT.* (PDF) OpenMote: Open-Source Prototyping Platform for the Industrial IoT (researchgate.net)