Programa Oficial de Postgrado en Ciencias, Tecnología y Computación
Máster en Computación
Facultad de Ciencias - Universidad de Cantabria

Plataforma Virtual para el análisis del rendimiento y la seguridad en Redes de Sensores Inalámbricas

Álvaro Díaz Suárez
adiaz@teisa.unican.es

Director: Pablo Sánchez Espeso

Grupo de Ingeniería Microelectrónica

Dpto. de Tecnología Electrónica, Ingeniería de Sistemas y Automática

Santander, Febrero de 2014

Tabla de contenidos

Lista de Figuras

Lista de Tablas

1 Introducción

Durante los últimos años está creciendo rápidamente el despliegue de redes inteligentes. Las aplicaciones de estas redes abarcan ámbitos tan dispares como control de tráfico y gestión de aparcamientos, seguridad de instalaciones militares, ayuda a personas con minusvalías y mejora de la salud de los ciudadanos, vigilancia y gestión de lugares con valor ecológico, control de instalaciones industriales, gestión del transporte, etc. El funcionamiento de estas redes inteligentes suele estar basado en la adquisición de datos del entorno, la comunicación de estos datos a los centros de procesado y la generación de información basada en esos datos. La información usada por estos entornos inteligentes es normalmente provista por redes de sensores inalámbricas (Wireless Sensor Networks, WSNs), las cuáles son las encargadas de monitorizar distintos aspectos o condiciones del entorno y comunicar los datos recolectados a un "servidor central". Estas redes de sensores inalámbricas están formadas por un conjunto de dispositivos o nodos provistos de sensores, los cuáles están desplegados en distintas localizaciones con capacidad para operar de forma autónoma aunque comunicados y coordinados mediante redes inalámbricas (sistema en red). Normalmente, este tipo de redes integran un gran número de nodos (de cientos a miles) de bajo coste, bajo consumo, elevados tiempos de operación sin mantenimiento ni cambio de batería, con recursos limitados y con múltiples sensores que comúnmente operan en entornos hostiles desatendidos [1].

Cada red de sensores inalámbrica está compuesta por nodos específicos que deben cumplir con los requisitos del sistema, de la red y de los sensores. Durante los últimos años, existe una preocupación creciente por la seguridad de estas redes y su vulnerabilidad en caso de ataques. Por ello, se están empezando a introducir cada vez mas requisitos de seguridad en la especificación de nodos de redes inteligentes. Debido al tipo de información que manejan estos dispositivos y al tipo de despliegue que tienen las redes de las que forman (normalmente un despliegue inalámbrico y desatendido), es necesario mejorar los mecanismos de seguridad actualmente usados. Aunque algunos de los riesgos de seguridad conocidos en las redes tradicionales son aplicables a las redes de sensores inalámbricas, estas últimas tienen vulnerabilidades específicas. Dichas vulnerabilidades son debidas a características propias de estas redes como su canal de comunicación (inalámbrico, compartido, vulnerable y desprotegido), el despliegue en entornos hostiles y desatendidos, su capacidad computacional reducida, su estricto manejo de la energía y la propia complejidad del sistema. Además los atacantes tendrán normalmente acceso directo al hardware de cada nodo, lo que les proporciona un gran número de posibles vías para atacar cada dispositivo o la red.

El nivel de seguridad requerido puede variar de una aplicación a otra dependiendo de la importancia de la información que se va a obtener y/o intercambiar [2]. Por esta razón, es importante identificar y tener en cuenta las debilidades de la red inalámbrica en la fase de diseño. Además, conociendo los efectos potenciales de los ataques típicos a un nodo o a la red será posible prevenir otras vulnerabilidades. Es por ello que el conocimiento de las consecuencias de los

ataques más típicos es un gran valor añadido en el diseño del Hardware y del Software embebido en un nodo o en la red.

Debido a los requisitos de bajo coste y bajo consumo que tienen este tipo de sistemas, los nodos tienen normalmente limitaciones tanto en recursos hardware/software como en capacidad de cómputo. Es por esto que el diseño de la seguridad en redes de sensores inalámbricas tiene que tener en cuenta la memoria, capacidad computacional y disponibilidad de recursos hardware/software. Además de esto, el consumo de energía está limitado. Normalmente el consumo de potencia es la mayor restricción que tienen los diseñadores cuando desarrollan redes de sensores inalámbricas debido a que los nodos suelen ser dispositivos alimentados mediante baterías, en los cuales su vida útil depende de la vida de ésta. Como ya se ha comentado, estos nodos suelen estar en entornos hostiles donde el acceso físico a los nodos es complicado por lo que el acceso a los nodos después del despliegue de la red para el reemplazo de las baterías es muy difícil y costoso. Además si tenemos en cuenta el alto número de nodos con los que pueden contar estas redes y la distribución de los mismos en la red, el coste de reemplazar las baterías aumentaría drásticamente. Por todo ello, es esencial tener en cuenta el impacto en consumo de las medidas y estrategias que se definan para aumentar la seguridad del sistema. Además, unos de los principales efectos que tienen los ataques a redes de sensores inalámbricas es el incremento del consumo en los nodos atacados, lo cual reduce la vida útil del nodo o incluso de la red. Por todo ello, es esencial estimar el impacto en consumo tanto de los ataques como de las estrategias que se adopten para mejorar la seguridad del sistema.

En este trabajo se propone una plataforma virtual (Simulador de redes de sensores inalámbricas mediante software ejecutado en un PC) la cual integra simulación funcional de redes de sensores inalámbricas, integración de RTOS (Real-Time Operating System) y estimación de prestaciones (tiempo de ejecución y consumo), junto con la capacidad de simular de ataques. Además, el simulador ejecuta exactamente el mismo código que se ejecutará en el nodo físico y se generará exactamente el mismo tráfico de paquetes en la red simulada que en la red final, aunque este trabajo se centrará en la simulación de los ataques. La plataforma virtual soporta uno de los sistemas operativos más extendidos en este tipo de redes (FreeRTOS). Además en este trabajo identificaremos los ataques más típicos en WSNs y se propondrán tres tipos de atacantes que permitirán modelar todos los ataques identificados.

Este trabajo se estructura en 7 secciones, iniciadas por esta introducción. La sección 2 presenta un análisis del estado del arte, poniendo el acento en el estudio de las principales vulnerabilidades de las redes de sensores inalámbricas. La sección 3 muestra la técnica propuesta de simulación de redes de sensores inalámbricas, mientras que la sección 4 propone un modelo simulable de los ataques a dichas redes. En la sección 5 se muestran resultados obtenidos con distintas redes de sensores. Por último, la sección 6 presenta las conclusiones del trabajo, y en la sección 7 se recogen las publicaciones que se han generado gracias a este trabajo fin de máster.

2 Estado del arte

Como se ha descrito en la sección anterior, es importante evaluar el comportamiento funcional y el consumo de cada nodo y de la red entera en caso de ataque. Para detectar y corregir potenciales vulnerabilidades es importante tener en cuenta no solo el código software embebido sino también la plataforma hardware y la estructura de la red. Una estimación de alto nivel del rendimiento del sistema es un paso esencial en cualquier metodología de diseño embebido. Las rápidas simulaciones realizadas en las primeras fases del diseño pueden proveer a los desarrolladores información importante que permita la modificación de los algoritmos software o la arquitectura de la red con el objetivo de minimizar los efectos de los ataques. En [3] se investiga un método basado en la probabilidad de ataque para despliegues de redes de sensores buscando principalmente preservar la integridad de la red y su funcionalidad. Dicho trabajo no tiene en cuenta el comportamiento del software que realmente se ejecuta en el nodo.

Por otro lado, muchas de las herramientas existentes de simulación de redes de sensores inalámbricas no ofrecen la posibilidad de simular los típicos ataques que este tipo de redes pueden sufrir. Las referencias [4] y [5] presentan una visión de los simuladores actuales de WSNs. Los más destacados se comentan a continuación. NS-2[6] y OMNET++[7] son simuladores de redes basados en eventos discretos para modelar redes de sensores inalámbricas. Otro entorno de simulación de redes inalámbricas es GloMoSim[8]. TOSSIM[9] es un simulador de eventos discretos a nivel de bit y emulador del sistema operativo TinyOS, un RTOS (sistema operativo de tiempo real) comúnmente usado en redes de sensores inalámbricas. En [10] se presenta una herramienta con un lenguaje propietario. Finalmente Avrora [11] estima con precisión de ciclo de reloj la ejecución de programas que se implementan en nodos de redes de sensores inalámbricas.

Como se ha mencionado anteriormente, uno de los objetivos principales de los ataques es incrementar el consumo de los nodos. Para tener una estimación precisa de los efectos de los ataques, es importante usar una herramienta de análisis de rendimiento que provea estimaciones de consumo de potencia y tiempos de ejecución.

Alguno de los simuladores comentados previamente pueden proporcionar evaluaciones del comportamiento funcional de redes de sensores inalámbricas, soporte para RTOS y estimaciones de potencia y tiempo de ejecución pero, hasta donde nosotros conocemos, no existe ningún simulador que integre todas estas características y/o simulación de ataques. Este trabajo presenta un simulador que une a la capacidad de simular la funcionalidad del nodo y la red, la estimación de prestaciones (tiempo de ejecución y consumo), el modelado del RTOS (FreeRTOS) y la simulación de ataques.

Además en este trabajo identificaremos los ataques más típicos en WSNs y se propondrán tres tipos de atacantes que permitirán modelar todos los ataques identificados.

2.1 Restricciones de energía

Las redes de sensores actuales se caracterizan por su bajo consumo y coste por nodo. Por ello, la mayoría de los nodos limitan la cantidad de recursos hardware y software disponibles para la aplicación embebida. Esto influye directamente en el grado de seguridad que se puede implementar en la red. Por lo tanto el diseño de la seguridad de WSNs debe considerar no solo el consumo de energía de los nodos, sino que también su memoria, capacidad computacional, y por supuesto, la disponibilidad de recursos orientados a seguridad (por ejemplo, codificadores hardware).

Normalmente el consumo es la mayor restricción a la hora de diseñar redes de sensores inalámbricas. Generalmente los nodos son dispositivos alimentados por baterías. Por lo tanto, su ciclo de vida viene limitado por la vida de la batería. Como ya se explicó anteriormente, debido al modo de despliegue que suelen tener estas redes, normalmente es muy difícil y costoso el reemplazo de las baterías para aumentar la vida de la red. Es por esto que a la hora de introducir nuevas medidas de seguridad hay que considerar esta restricción de consumo.

De acuerdo con los estudios sobre consumo de energía presentados en [16] y [17], el rango de comunicación de los nodos está limitado por la necesidad de conservar la energía. Sin embargo, en casi todas las topologías de redes existen un pequeño número de nodos con una capacidad mayor que permiten hacer de puente entre distintas subsecciones de la red o entre distintas redes. Estos dispositivos se denominan “Gateways”.

Excluyendo los “Gateways”, el resto de los dispositivos que forman parte de la red están la mayoría de su tiempo operando en un modo de bajo consumo o durmiendo y solo se despiertan cuando necesitan procesar un evento. Por esta razón, la disponibilidad de un nodo en una red de sensores puede ser limitada.

2.2 Seguridad de redes de sensores inalámbricas

La seguridad afecta a un elevado número de tareas en la red inalámbrica, desde autentificación de nodos, a integridad de mensajes, privacidad y no repudio de información. Cuando se define la seguridad en redes de sensores inalámbricas hay una serie de términos que conviene tener en cuenta:

- **Availability (Disponibilidad):** Se asegura de que los servicios de la red siempre están disponibles.
- **Authorization (Autorización):** Se asegura que sólo los nodos autorizados pueden proveer información a la red.

- **Authentication (Autenticación):** Se segura que la comunicación entre nodos es genuina, esto es que un nodo malicioso no pueda enmascare como un nodo real de la red.
- **Confidentiality (Confidencialidad):** Busca que un mensaje dado no pueda ser entendible por nadie que no sea el destinatario.
- **Integrity (Integridad):** Certifica que un mensaje enviado de un nodo a otro no ha sido modificado por un nodo maligno.
- **Non-repudiation (No repudio):** Se refiere a que un nodo no puede rechazar la retransmision de un mensaje que se ha enviado previamente.
- **Freshness (Frescura):** Implica que los datos son recientes y se asegura de que ningún atacante puede reproducir los mensajes antiguos.

Debido al incremento en el uso de redes de sensores inalámbricas, los riesgos en las transmisiones de información sobre las redes han aumentado. Es por esto por lo que es importante identificar los ataques más dañinos que cada red puede sufrir. A continuación se presenta un estudio de los riesgos típicos de una WSN.

2.2.1 Ataques en redes de sensores inalámbricas

Un ataque puede ser definido como un intento por conseguir un acceso no autorizado a un servicio, recurso o información. También podría ser un intento de comprometer la integridad, disponibilidad o confidencialidad de un sistema. Este trabajo incluye un estudio de los ataques más típicos que una red de sensores inalámbrica puede sufrir. El análisis está basado en las vulnerabilidades descritas en [18]. Los ataques incluidos en este trabajo pueden ser clasificados en 20 categorías distintas.

2.2.1.1 Ataque "Jamming"

Este ataque produce una denegación del servicio a usuarios autorizados, atascando el tráfico legítimo con una cantidad abrumadora de trafico ilegítimo. Se interrumpe la funcionalidad de la red al transmitir señales con mucha energía que colisionan con los paquetes verdaderos. En definitiva, lo que busca es introducir ruido en algunos canales para romper la comunicación entre nodos. Hay muchas estrategias de ataque por Jamming, pero podemos clasificarlas según su modo de introducir el ruido en la red.

- Ruido
- Tonos
- Pulsos
- Barrido
- Seguimiento

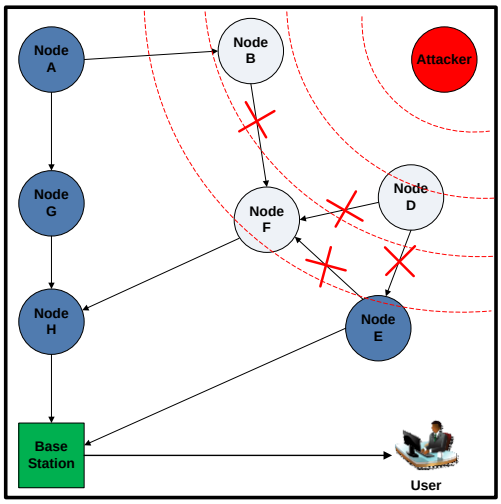

Figura 1. Ataque Jamming

Además de esto, un ataque jamming puede ser por:

- Banda-ancha o BBN (Broad Band Noise)
  - Mete ruido en todos los canales.
- Banda-parcial o PBN (Partial Band Noise)
  - Introduce ruido solo en una parte del espectro.
- Banda estrecha
  - Introduce ruido en un único canal.

La Figura 1 ilustra cómo se comporta el ataque Jamming. El atacante produce ruido de manera intermitente o persistente produciendo interferencias. Como se puede ver en la figura, esta interferencia afecta directamente al menos a 3 nodos de la red (B, D y F).

2.2.1.2 **Ataque "Tampering"**

Los ataques por Tampering [19] requieren acceso físico al nodo para robar sus datos internos. A través de estos ataques es posible obtener información sensible, como por ejemplo llaves criptográficas.

2.2.1.3 **Ataque "Collision"**

En un ataque por collision [20], el nodo atacante no sigue el protocolo de control de acceso al medio y produce colisiones con las transmisiones del nodo vecino mediante el envío de un paquete corto ruidoso. Los paquetes colisionan cuando dos nodos intentan transmitir en la misma frecuencia al mismo tiempo, produciendo la corrupción de los paquetes. Este ataque puede causar una gran cantidad de interrupciones en el funcionamiento de la red.

2.2.1.4 **Ataque "Resource Exhaustion"**

Este ataque consiste en repetir colisiones y múltiples retransmisiones hasta que el nodo atacado se queda sin energía o recursos para procesar la información. El nodo malicioso transmite o solicita paquete continuamente a través del canal de comunicación.

2.2.1.5 **Ataque "Energy Drain"**

Debido a la dificultad de reemplazar las baterías de los nodos y los requisitos de consumo de energía que suelen tener este tipo de dispositivos, los ataques pueden comprometer los nodos inyectando mensajes en la red o generando una gran cantidad de tráfico. Estos falsos mensajes causan falsos requisitos en los nodos de la red y estos contestan consumiendo la energía de sus baterías. Un objetivo de este ataque podría ser eliminar nodos de la red, degradando el rendimiento y finalmente dividiendo la red en dos partes. Esto permite tomar el control de una parte de la red insertando un nuevo nodo atacante como enlace.

La figura 2 muestra un nodo atacante generando mensajes falsos continuamente. Sus nodos vecinos (C y G) responden al ataque y finalmente se quedan sin batería.

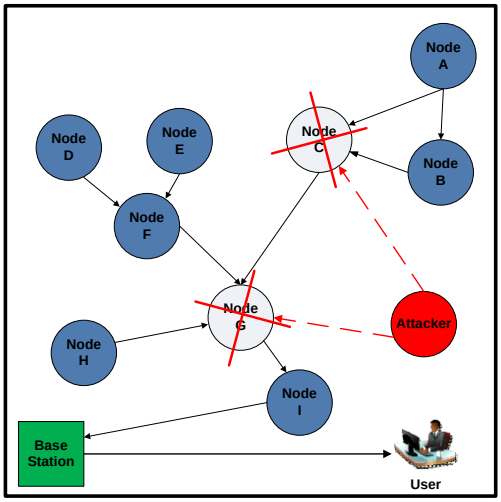

Figura 2. Ataque Energy Drain

2.2.1.6 **Ataque "Interrogation"**

Este ataque explota el mecanismo de comunicación basado en "handshake" que muchos protocolos usan para evitar problemas de pérdida de integridad de los mensajes en la red. El ataque consiste en enviar mensajes RTS (Require To Send) para obtener respuestas CTS (Clear to Send) de un nodo vecino.

2.2.1.7 **Ataque "Sniffing"**

Los ataques por Sniffing capturan paquetes de la red. Si los paquetes de la red no viajan encriptados, los datos de los mensajes pueden ser capturados por un sniffer y ser leídos fácilmente. Sniffing se refiere al proceso usado por el atacante para capturar el tráfico de la red usando un sniffer (programa y hardware específicos para capturar mensajes en redes). Cuando un paquete es capturado por un sniffer, el contenido de este puede ser analizado. Los sniffers pueden ser utilizados por los atacantes para capturar datos importantes como contraseñas, información, etc.

2.2.1.8 **Ataque "Black Hole"**

El ataque por agujero negro consiste básicamente en alterar el routing de la red con el objetivo de atraer todos los paquetes hacia el nodo atacante y, sigilosamente, este los vaya descartando o retirándolos del tráfico de la red. En la Figura 3 se muestra un ataque con esta estrategia, donde todos los paquetes enviados por los nodos I, G, D y E D son capturados por el atacante y desechados

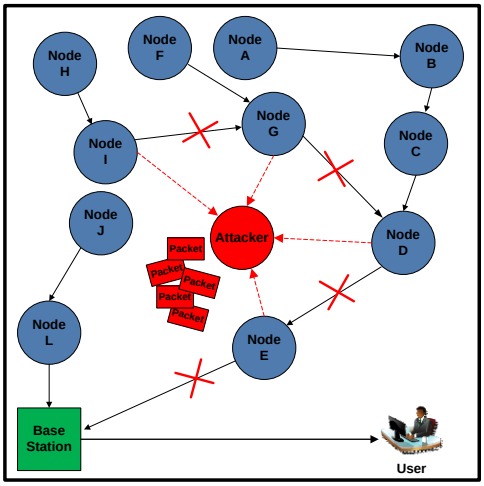

Figura 3 Ataque Black Hole

2.2.1.9 **Ataque "Selective Forwards"**

En las redes multi-salto, los nodos asumen que cuando envían un mensaje a un nodo que no tiene comunicación directa con él éste llega correctamente. Para enviar mensajes a dicho nodo no accesible directamente, el emisor tiene que enviar el paquete a nodos accesibles para que éstos se lo envíen al destinatario final mediante una serie de "saltos" (transmisiones) entre nodos accesibles directamente. Sin embargo, un nodo malicioso podría ser el encargado de recibir los mensajes y simplemente desechar o no propagar los que él crea conveniente.

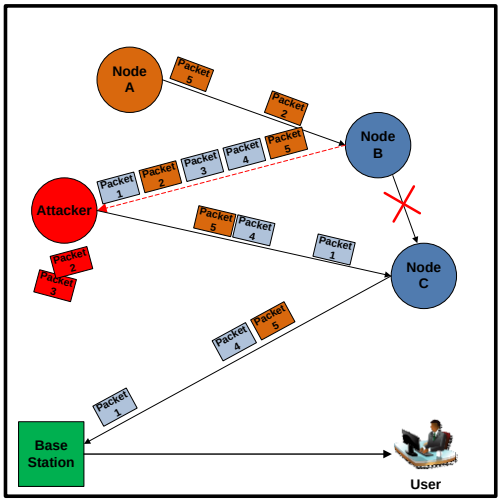

Figura 4 Selective Forward 1

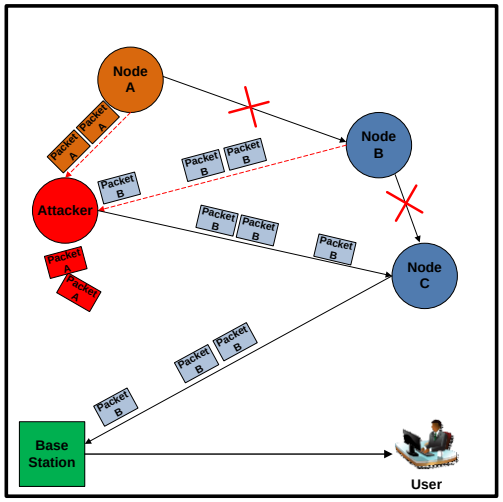

Figura 5 Selective Forward 2

El procedimiento que se sigue en este tipo de ataques es muy similar al ataque descrito en el apartado 2.2.1.8 (Ataque por agujero negro). Primero, un nodo atacante convence a la red de que él es el nodo más cercano a la estación base, atrayendo todos los paquetes a él mismo. Una vez que los nodos de la red confían en él y los mensajes pasan por el nodo malicioso, éste solo tendrá que desechar todos los paquetes (Figura 3), eliminar un número aleatorio de paquetes (Figura 4) o suprimir unos paquetes específicos (Figura 5)

2.2.1.10 Ataque "Homing"

En un ataque por Homing, el atacante estudia el tráfico de la red para deducir la localización de los nodos críticos, tales como nodos coordinadores ("router") o los vecinos de la estación base. El ataque consiste en deshabilitar esos nodos. Este ataque puede tener similitudes con el ataque por agujero negro.

2.2.1.11 Ataque "Sink Hole"

En el ataque por Sink Hole, el adversario intenta atraer todo el tráfico cercano a una cierta área a través de un nodo comprometido. Ese nodo comprometido se coloca en el centro del área que se desea atacar y crea una “esfera de influencia” atrayendo todo el tráfico de los nodos destinado a una estación base.

2.2.1.12 Ataque "Hello Flood"

En este ataque, el atacante típicamente lo que intenta es consumir la energía de los nodos atacados. Un atacante con un alto poder de transmisión puede enviar paquetes “HELLO” (Usado en múltiples protocolos para descubrir nodos nuevos) para convencer a los nodos de la red que el adversario se encuentra a un salto, causando que muchos nodos le contesten y gasten su energía enviando paquetes a este nodo imaginario.

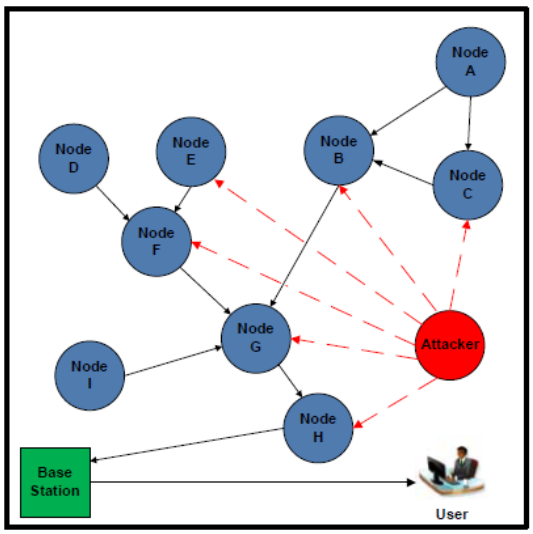

Figura 6 Ataque Hello flood

En la Figura 6 se muestra un ejemplo de este ataque, donde el nodo atacante envía en "broadcast" paquetes "Hello" para convencer a los nodos de la red que él es un nodo vecino.

2.2.1.13 Ataque "Sybil"

Este ataque consiste en la modificación de los caminos ("routing") de la red buscando atraer el tráfico hacia los nodos atacantes, con el objetivo de aislar de la red ciertos nodos. Cuando estos nodos no se pueden comunicar más, el atacante procede a enviar paquetes falsos con el objetivo de suplantar la identidad de los nodos aislados o incomunicados.

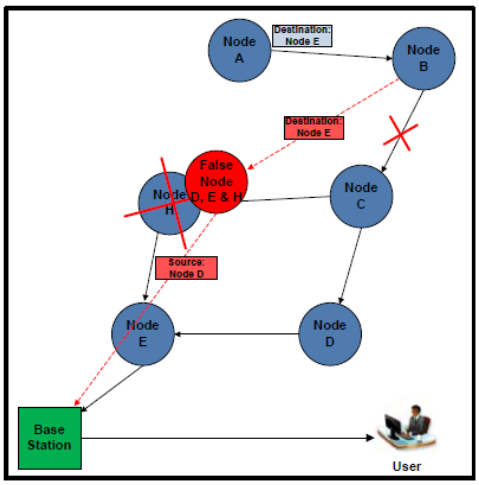

Figura 7 Ataque Sybil

2.2.1.14 Ataque "Misdirection"

Este ataque [21] consiste en la introducción de mensajes sin destino conocido en la red o con destino inalcanzable. El atacante introduce el paquete con un destino que no existe en la red. El objetivo es que el paquete intruso viaje sin dirección por la red. Esto aumenta el consumo de los nodos al tener más paquetes que procesar y aumenta la latencia de los paquetes reales haciendo que unos pocos paquetes no alcancen la estación base.

2.2.1.15 Ataque "Node Replication"

En el ataque por replicación se busca introducir diferentes nodos que suplanten la identidad de un nodo existente en la red. Aunque el ataque por replicación pueda parecerse mucho al ataque descrito en la Sección 2.2.1.13 (Ataque Sybil), éstos difieren en una característica clave. En el ataque por Sybil, es un único atacante el que se hace pasar por múltiples identidades, mientras que en el ataque por replicación son múltiples nodos los que se hacen pasar por un único nodo de la red. Por lo tanto, el ataque Sybil es más fácil llevarlo a cabo ya que únicamente es necesario un atacante para tener éxito. En cambio, el ataque por replicación requiere desplegar múltiples atacantes para tener éxito. Debido al mayor número de atacantes, las posibilidades de detectar este ataque aumentan. La Figura 8 muestra un ejemplo de ataque por replicación.

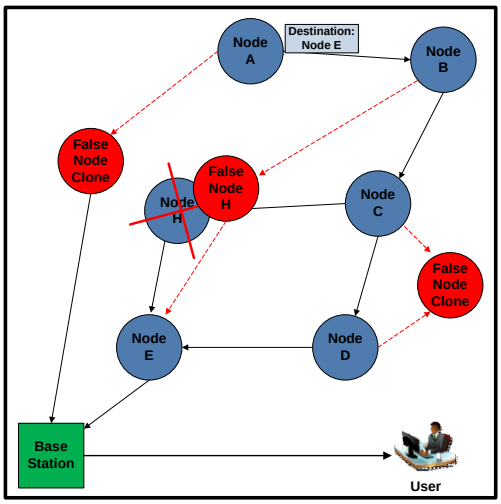

Figura 8 Ataque Node Replication

2.2.1.16 Ataque "Spoofing"

En un ataque por suplantación un nodo consigue suplantar a otro nodo de la red falsificando sus datos y consiguiendo una ventaja ilegitima. Este ataque consiste en la alteración del routing de la red para debilitarla. Existen tres diferentes técnicas básicas con distintos objetivos a la hora de llevar a cabo este tipo de ataque.

- *Lazo en la red*
    - Consiste en modificar el routing de la red para que un paquete realice un bucle infinito y este no llegue nunca a su destino.
- *Atraer el tráfico*
    - Mediante esta técnica conseguimos que el tráfico pase por el atacante y este lo modifique a su antojo.
- *Repeler el trafico*
    - Se consigue sobrecargar una parte de la red y disminuir el uso en otra.

2.2.1.17 Ataque "Flooding"

El atacante mandará continuamente solicitudes de nueva conexión, hasta que los recursos del nodo al que le solicitan la conexión queden exhaustos. Esto puede producir serios daños al nodo atacado al reducirle su ciclo de vida o dejarle en un estado de inanición.

2.2.1.18 Ataque "Denial of Service"

Los ataques de denegación de servicio (DoS, Denial of service) pueden causar mucho daño a sistemas alimentados por batería. En este trabajo se presta especial atención a un tipo de ataque DoS llamado "Path-Based DoS". En un ataque DoS, se envía al nodo atacado una gran cantidad de paquetes, haciendo que el nodo se sobrecargue y no pueda seguir prestando servicios; por eso se le denomina "denegación", pues hace que el servidor no dé abasto a la cantidad de solicitudes.

2.2.1.19 Ataque "Application"

Este ataque consiste en la modificación del firmware o del software embebido en el nodo de la red. El atacante normalmente necesita tener acceso directo al nodo y modificar su programación para producir un funcionamiento erróneo.

2.2.1.20 Ataque "Overwhelm"

Es un ataque directo al sensor del nodo, para que el envío de paquetes aumente o mande información errónea y, en consecuencia, el consumo del nodo o de la red aumente. El atacante normalmente necesita acceso directo al nodo para provocar estímulos en el sensor o una lectura errónea.

2.3 Vulnerabilidades en redes de sensores inalámbricas

Existen distintas estrategias a la hora de atacar una red de sensores, con métodos que consisten en interceptar mensajes, modificarlos para corromperlos y volverlos a inyectar en la red, hasta estrategias más sofisticadas en las cuales un nodo malicioso actúa como un Gateway falso. Por esta razón no es fácil definir una clasificación general que incluya todos los posibles tipos de ataques. En [13], Mohammadi y Jadidoleslamy clasifican los ataques en dos grandes categorías: ataques activos y pasivos.

TABLA 1–ATAQUES ACTIVOS Y PASIVOS

Jamming	Activo
Tampering	Pasivo
Collision	Activo
Resource exhaustion	Activo
Energy drain	Activo
Interrogation	Activo
Sniffing	Pasivo
Selective Forwarding	Activo
Black Hole	Activo
Sinkhole	Activo
Hello Flood	Activo
Misdirection	Activo
Sybil	Activo
Node Replication	Activo
Spoofing	Pasivo
Flooding	Activo
Homing	Activo
Path-based DOS	Activo
Application	Activo
Overwhelm	Activo

Mientras que los ataques pasivos están relacionados con vulnerabilidades de privacidad (espionaje y robo de la información mediante la interceptación de las comunicaciones de paquetes intercambiados dentro de una WSN) los ataques activos buscan la inyección de paquetes erróneos/falsos de datos suplantando nodos o modificando datos, creando agujeros de seguridad en los protocolos, destruyendo nodos, degradando el rendimiento, corrompiendo la funcionalidad y sobrecargando la red.

El estudio presentado en este trabajo se centra en los ataques activos que más afectan al rendimiento de la red. Estos ataques son los candidatos a ser caracterizados en la herramienta de simulación, según [14]. Siendo más precisos, es necesario modelar los ataques que intentan alterar la disponibilidad de la red,

corrompiendo total o parcialmente el flujo de comunicación entre los nodos de la red. La Tabla 1 muestra una primera división de los ataques entre activos y pasivos.

3 Técnica de simulación

En este trabajo se presenta una novedosa metodología para simular aspectos de seguridad en redes de sensores inalámbricas. Con este objetivo, y teniendo en cuenta las amenazas estudiadas en el apartado anterior, se ha implementado en un entorno de análisis de prestaciones, un módulo de simulación de redes inalámbricas que permite la simulación de ataques en las primeras etapas del desarrollo del sistema.

3.1 Co-Simulación HW/SW

La Co-Simulación HW/SW es una parte fundamental en el proceso de verificación de sistemas electrónicos complejos. Durante las primeras fases del proceso de diseño, se pueden utilizar distintos lenguajes para describir el comportamiento del sistema. En este trabajo se utiliza el lenguaje SystemC para realizar la simulación conjunta de la parte Hardware con la Software a nivel de sistema. A este nivel se pueden utilizar distintas técnicas de simulación. Uno de los métodos de co-simulación más utilizados está basado en el uso de ISS (Instruction Set Simulator), los cuáles son capaces de simular la ejecución directa del código binario de los componentes SW. Esta solución es muy precisa, pero muy lenta para ser utilizada en primeras fases del proceso de diseño (nivel sistema). La utilización de metodologías de simulación nativas permite obtener resultados en tiempos más breves que utilizando un ISS. Dichas metodologías se basan en la instrumentación del código fuente del sistema a simular y su compilación en la plataforma en donde se realiza la simulación. En este trabajo se presenta una herramienta de análisis de prestaciones basada en simulación nativa. En dicho entorno, la ejecución del SW se simula en el ordenador (plataforma "host") usando avanzados modelos abstractos del procesador, del RTOS ("Real Time Operating System") y de la plataforma HW objetivo (plataforma "target").

La metodología de Co-Simulación descrita en esta sección está basada en la técnica de simulación nativa [22] presentada en la Figura 9. Dicha metodología se basa en la anotación (o instrumentación) del código fuente del software embebido con información relacionada con el hardware sobre el que se ejecuta (plataforma "target"). El sistema utiliza el núcleo del simulador de código abierto de SystemC, lo que permite añadir componentes SystemC a la simulación. Gracias a las anotaciones, es posible obtener una estimación precisa y veloz de la energía y potencia consumidas, tiempo de ejecución, número de misses/ hits de las cachés de datos e instrucciones, número de accesos a buses, etc.

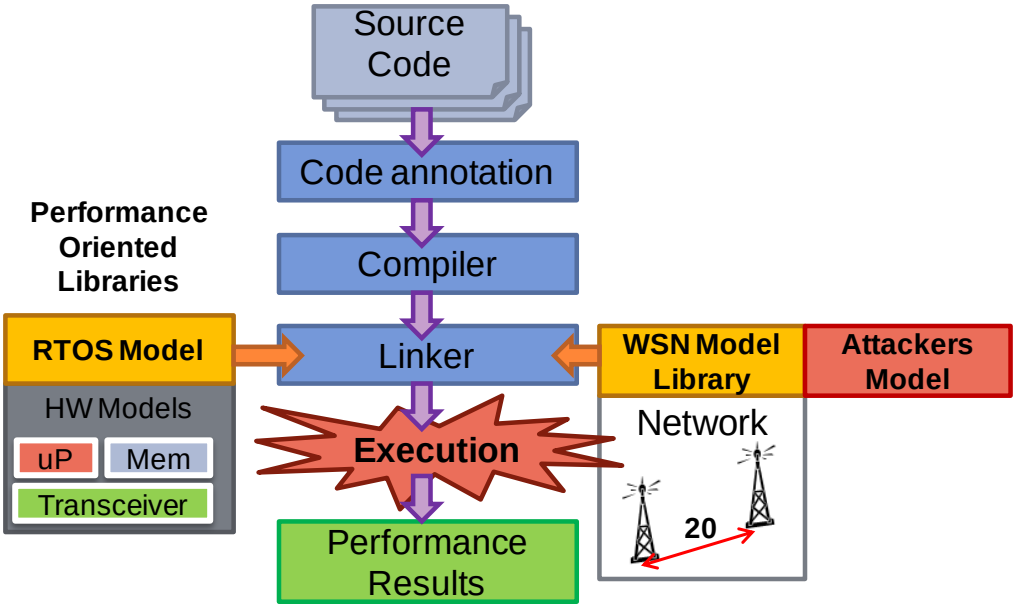

Figura 9: Proceso de co-Simulacion Nativa en la plataforma virtual

El proceso de compilación del código fuente conlleva 2 pasos:

1. El código fuente es compilado, siendo identificados los bloques básicos en el código. Para cada bloque básico se le añade una anotación con el coste (tiempo de ejecución, consumo, etc) que cada bloque básico tiene en la plataforma en la que se va a ejecutar. Este proceso permite obtener el código instrumentado.
2. El código instrumentado obtenido en la fase anterior es compilado nativamente en el ordenador en donde se realizará la simulación (plataforma "host"). La ejecución de dicho código simulará el comportamiento del sistema al tiempo que producirá estimaciones basadas en los costes añadidos (tiempo de ejecución, consumo, etc).

En la Figura 9 también se muestran algunas de las librerías que se incluyen en el simulador. Dichas librerías incluyen modelos de simulación del RTOS y la red, así como el modelo de los ataques que se presentará en las secciones siguientes.

3.2 Modelo de la red de sensores inalámbrica

En la Figura 10 se puede ver representado un ejemplo de modelado de WSN. En dicho modelo se incluyen dos nodos en la red, así como el modelo de red. Como se puede observar, la arquitectura y la aplicación de cada nodo es independiente de los otros nodos, por lo tanto se puede simular una red heterogénea con nodos con distintas arquitecturas, firmwares o incluso Sistemas Operativos. El software de aplicación de cada nodo (App SW Code) será instrumentalizado con información correspondiente al nodo en el cual se ejecuta el código. El simulador proporcionará un modelo de RTOS (FreeRTOS) que permitirá abstraer detalles hardware de la

plataforma objetivo (o "target"). Además, el entorno de simulación proporcionará un modelo de red de comunicaciones.

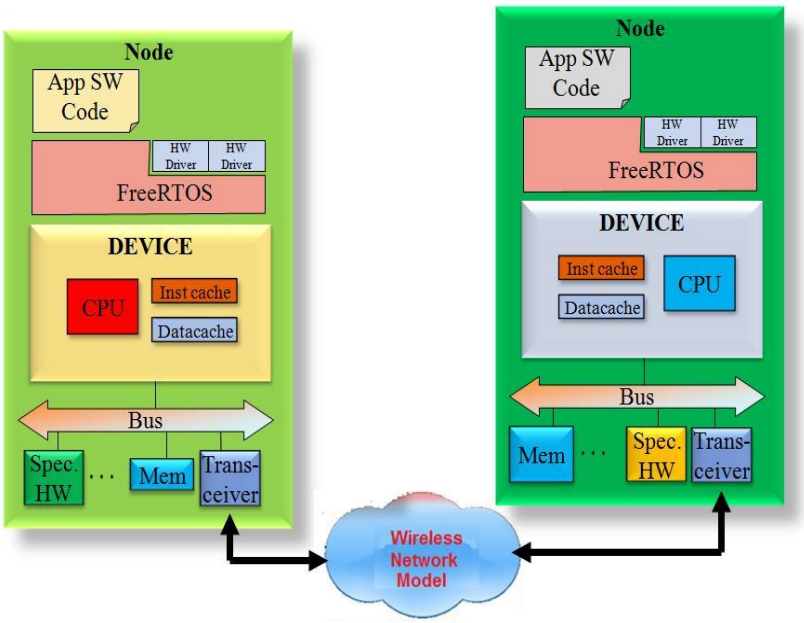

Figura 10: Representacion de la arquitectura de los nodos + WSN

En una transmisión inalámbrica, el canal de transmisión entre dos nodos es el aire. Dicho canal es compartido, con ruido e interferencias, donde se tiene que tener en cuenta que el rango de alcance de un nodo es limitado. Además los mensajes pueden ser escuchados por nodos que no son los receptores de los mismos. Como consecuencia de esto, los desarrolladores deben determinar la visibilidad y la probabilidad de recepción correcta de un paquete enviado entre nodos o la probabilidad de pérdida de paquetes. Con el objetivo de obtener simulaciones precisas, los desarrolladores deben estudiar la zona de despliegue de la red de sensores inalámbrica y definir una matriz con las probabilidades de pérdidas de paquetes entre nodos. Esta probabilidad también debe incluir la probabilidad de perdida de paquete debido a ruidos externos. Esta probabilidad puede ser calculada con simuladores de propagación electromagnética, como por ejemplo la herramienta Cindoor [23]. Gracias a Cindoor los desarrolladores pueden determinar la visibilidad de los nodos de una red con sus rangos de alcance y las probabilidades de error. Con la matriz de probabilidades de error, el simulador podrá conocer la efectividad de los enlaces entre nodos.

Un esquema de alto nivel del modelo de red se puede ver representado en la Figura 11. Como se puede observar, la interfaz de red ("hardware network interface") es el responsable de transmitir/recibir los paquetes con destino/origen en otro nodo. Este modo de operación es muy similar al usado en un sistema real. En un sistema inalámbrico real, cuando un nodo envía un mensaje a otro, el paquete es transferido al transmisor de radio ("transceiver" de radio) para que lo envíe por radio frecuencia (RF). El mensaje no solo llega al receptor de radio del destino, sino a todos los nodos a los que tenga al alcance. Será el receptor de radio con su interfaz de red la encargada de recibir los paquetes para que el firmware del nodo solo procese los paquetes con destino el mismo.

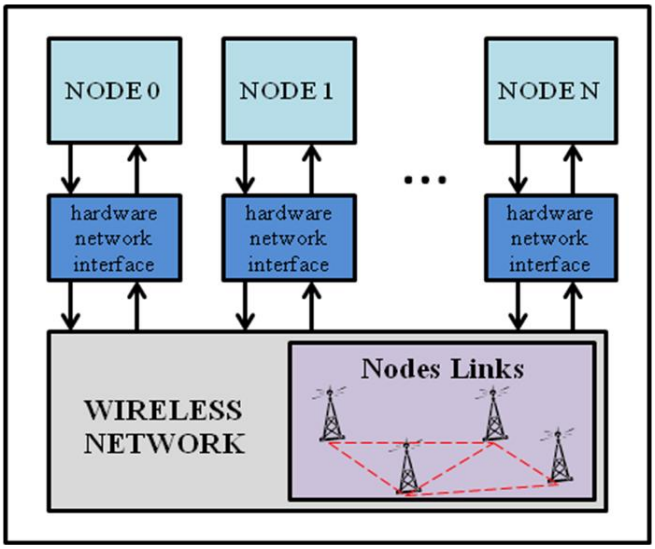

Figura 11: Modelo de la red

Con la matriz de probabilidades anteriormente comentada, el modelo de red del simulador conoce el alcance de cada nodo y la probabilidad de recepción de mensajes. Por ejemplo, si el desarrollador define la probabilidad asociada a un enlace ("link") entre dos nodos como 100, significaría que el rango del nodo emisor no es lo suficiente para alcanzar al nodo destino directamente (100% de probabilidad de pérdida de mensaje). Sin embargo, si el desarrollador asocia al enlace un valor 0, la probabilidad de error en la transmisión será 0 y todos los paquetes transmitidos entre esos dos nodos llegaran a su destino. Por otro lado, si la probabilidad del enlace es un número entre 0 y 100, este número indica la probabilidad de que el paquete no llegue a su destino. Por ejemplo, en la Figura 12, el enlace entre el nodo 0 y el nodo 1 tiene asociado un valor del 10%, esto indica que de cada 100 paquetes enviados a través de ese enlace (del nodo 0 al 1) solo 90 llegaran a su destino. En el caso de los paquetes enviados entre el nodo 3 y 4, la red perderá 4 paquetes de cada 100. Esta matriz la podemos ver reflejada en las Figuras 11 y 12, identificada como “NODE LINKS”

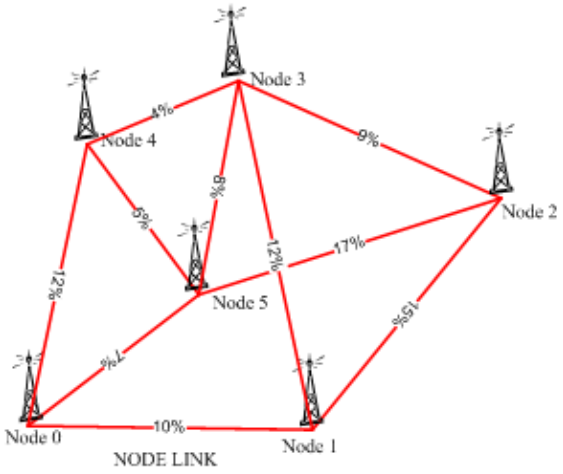

Figura 12: Representación de la red inalámbrica

El esquema del modelo de simulación de la red inalámbrica está representado en la Figura 13. El modelo de red es el responsable de transmitir los paquetes a su destino. Cuando un nodo envía un paquete, el modelo lo añade a su cola de mensajes que se ordena por el tiempo de llegada al nodo receptor. Cuando el tiempo de simulación coincide con el tiempo de llegada del paquete, el modelo de red inalámbrica saca el paquete de la cola y genera un número aleatorio entre 0 y 100. Si la probabilidad de recepción correcta ("success") es más grande que el número aleatorio, la red transmitirá el paquete a su destino. En otro caso, la red lo descartará. La probabilidad de recepción correcta ($P(success)$) se entiende como 100 menos la probabilidad de error ($P(error)$) definida en la matriz de probabilidades.

$$P(success) = 100 - P(error)$$

Por ejemplo, en la Figura 12, en un paquete enviado entre el nodo 0 y 1 será recibido únicamente en el caso de que el número aleatorio generado por el simulador sea mayor que la probabilidad de error definida en la matriz (10 en este caso). Sin embargo, en una red inalámbrica real la transmisión se produce a través de un canal compartido y por ello el paquete será enviado a todos los nodos que están dentro de su rango (nodos 1, 5 y 4, con probabilidad menor de 100). Por esta razón, este paquete es enviado no únicamente a su destino en la simulación, sino también a los destinos para los que es visible el nodo emisor. En el caso de que ese paquete llegue a un nodo que no es su destino, será el interfaz de red y/o el software embebido el encargado de desecharlo. El interfaz de red identifica de forma automática el destino del paquete y rechaza los mensajes que no son dirigidos al nodo, computando el coste en tiempo y consumo del proceso.

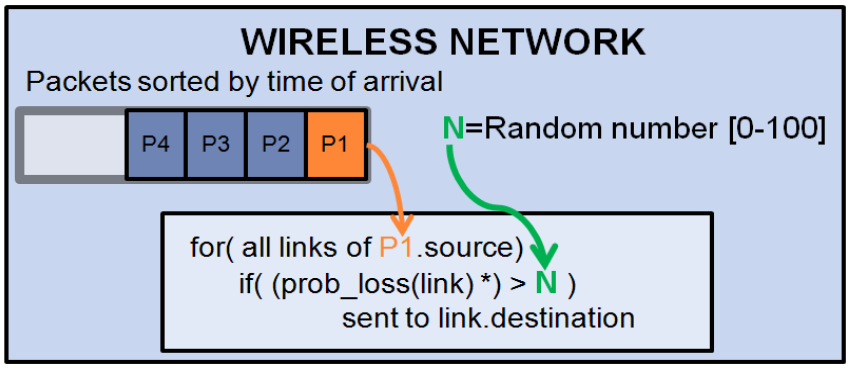

Figura 13: Esquema del funcionamiento de la red en el simulador

Este modelo de red permite la exploración del espacio de diseño teniendo en cuenta la topología de la red, sin la necesidad de recompilar el código cada vez que se modifican los parámetros el despliegue. Esto es posible porque la matriz de probabilidades se gestiona dinámicamente, lo que ayuda a reducir el tiempo de análisis de la red.

3.2.1 Componentes Hardware específicos de un nodo

Hay dos componentes en un nodo de una red de sensores inalámbrica que normalmente otros sistemas no tienen. Estos componentes son el sensor y el módulo de radio ("Transceiver" de RF). El sensor es el responsable de leer la información del exterior con una cierta frecuencia o cuando un evento ocurra. Este componente se implementa en la simulación como un componente externo con un consumo especifico. Se pueden simular todos los tipos de sensores que se desee. Sus modelos únicamente diferirán en la información enviada al nodo, su consumo, su tiempo de lectura y la frecuencia de muestreo. El otro componente fundamental es el módulo de radio. Dicho componente hardware es bastante más complejo que un sensor, integrando normalmente registros de configuración que controlan dinámicamente su modo de operación. Los registros implementados en este trabajo modelan módulos de Digi [32] que usan el protocolo de radio XBee 802.15.4. Entre los registros implementados destacan los siguientes:

- **"Destination Address High and Low"** -> Definen la dirección destino de un envío.
- **"Baud Rate"** -> Especifican la velocidad de transferencia entre el modulo radio y el nodo (plataforma "target").
- **"Mac Retries"** -> Número de reintentos que se pueden utilizar para enviar un mensaje de tipo "unicast" (mensaje con un único destino). Si un mensaje no llega a su destino el módulo de RF lo re-enviará hasta dicho número de veces, después de lo cual considerará que la transmisión es imposible/errónea.
- **"Multiple Transmissions"** -> Número de transmisión que hace en un envío "broadcast" (recordemos que un envío "broadcast" no necesita confirmación y el nodo emisor no recibe confirmación de paquete llegado a su destino).
- **"Power Level"** -> El nivel de potencia a la cual el modulo RF transmite.

3.3 Modelo RTOS

Un Sistema operativo de tiempo real (RTOS) está destinado a servir las solicitudes de recursos del sistema de aplicaciones en tiempo real. Estos sistemas no son normalmente sencillos. Como Pronk explica en [24], normalmente un RTOS crea y opera un cierto número de tareas de usuario concurrentes, planifica estas tareas garantizando una respuesta en un tiempo limitado usando mecanismos de prioridad y garantiza la no interferencia entre estas tareas y el núcleo (kernel) del RTOS. Además permite la comunicación de las tareas mediante colas, su sincronización mediante mutex o semáforos, y la interacción directamente con el hardware a través de drivers de los dispositivos, relojes y mecanismos de interrupción.

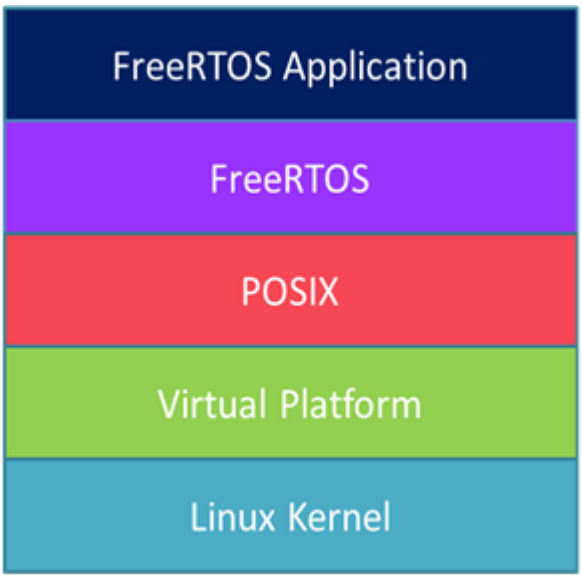

Figura 14: Modelado de FreeRTOS en el Simulador Virtual

La mayoría de las aplicaciones de los nodos están implementadas sobre un RTOS. Por esta razón, es muy importante modelar un sistema operativo en nuestra plataforma de simulación. En [25] se presenta un estudio sobre los distintos RTOS que normalmente se utilizan en WSNs. Para conseguir una mayor precisión en las estimaciones y poder simular el código final introducido en los nodos es muy importante tener modeladas las funciones del Sistema Operativo. Una de las mayores ventajas de la simulación propuesta es el uso de la misma API que la plataforma física utiliza. Con este propósito, se ha utilizado un modelo basado en el API de POSIX [26]. El modelo que el simulador virtual utiliza facilita la integración de nuevos RTOS gracias a una implementación completa del API POSIX. Este modelo provee hilos ("threads"), mutexes, semáforos, colas, timers y otros servicios comunes de POSIX. A partir de esta completa implementación de la infraestructura POSIX se puede modelar la mayoría de RTOS utilizados en WSNs. Como se puede ver reflejado en la Figura 14, la capa POSIX es soportada por el simulador Virtual. Uno de los RTOS más populares en redes de sensores inalámbricas es FreeRTOS[27]. Para la integración de este RTOS en la plataforma virtual se ha colocado una nueva capa (Capa FreeRTOS) sobre la capa existente de POSIX. Gracias a esta capa, es posible simular aplicaciones desarrolladas para este Sistema Operativo. Esta capa implementa la funcionalidad de FreeRTOS utilizando funciones del API POSIX, como se puede observar en la Figura 14. El modelo actual implementa la última versión disponible de FreeRTOS (V7.6.0). En muchos casos, la implementación de las funciones con el enfoque adoptado únicamente requiere adaptar la interfaz de la API FreeRTOS para llamar a funciones similares de la API POSIX.

Para usar esta funcionalidad de los nuevos Sistemas Operativos se deben escribir las aplicaciones para que ejecuten las funciones de la API de FreeRTOS. En el estándar de codificación de FreeRTOS hay convecciones de nombres que deben ser respetadas. Por ejemplo, las funciones de la API tiene un sufijo que define el tipo

de dato que la función retorna. De esta forma, las funciones cuyo nombre empieza por "x" retornan una estructura, las que empiezan por "s" retornan un valor de tipo "short", etc. Algunas funciones típicas de esta API son las siguientes:

- **Task Creation**
  - *xTaskCreate()* -> Crea una nueva tarea
  - *xTaskDelete()* -> Borra una tarea del manejador de tareas del kernel.

- **Task Control**
  - *vTaskDelay()* -> Retrasa una tarea durante un determinado número de ticks de reloj.

- **Queues**
  - *xQueueCreate()* -> Crea una nueva instancia de una cola.
  - *vQueueDelete()* -> Borra una cola liberando toda la memoria reservada.
  - *xQueueSend()* -> Añade un objeto a la cola
  - *xQueueSendToFront()* -> Añade un objeto al principio de la cola.
  - *xQueueSendToBack()* -> Añade un objeto al final de la cola.
  - *sQueueReceive()* -> Recibe un objeto de una cola.

- **Semaphores / Mutexes**
  - *vSemaphoreCreateBinary()* -> Crea un semáforo binario.
  - *xSemaphoreCreateMutex()* -> Crea un semáforo mutex
  - *xSemaphoreTake()* -> Obtiene el semáforo.
  - *xSemaphoreGive()* -> Libera el semáforo.

Además de modelar las funciones del sistema operativo, también es necesario modelar las funciones de acceso al hardware. El nodo tiene una serie de registros que permiten configurar y acceder a los distintos periféricos hardware de la plataforma "target". Para acceder a ellos se suelen utilizar API específicas del fabricante el micro-controlador del nodo. Por ejemplo, la librería STM32LIB de STMicroelectronics proporciona una API común de acceso al hardware de todos los dispositivos de la familia STM32 de dicho fabricante. Dicho API permite portar el código embebido entre diferentes dispositivos de la misma familia sin tener que cambiar el código. Este API ha sido modelado en el simulador, lo que permite acceder al modelo de hardware desde el código de aplicación. Además, el simulador puede ejecutar el mismo código se será implementado en los nodos físicos y que usa dicha librería.

4 Técnica del modelado de ataques

Con el objetivo de simular ataques en la plataforma virtual, este trabajo propone introducir nuevos nodos (atacantes) en el modelo de WSN. Estos nuevos nodos simulan los atacantes con modelos sencillos, para que el tiempo de simulación no se vea afectado. Se proponen tres tipos básicos de nodos atacantes:

- **"Link-Noise node" ->** Introduce ruido en el canal
- **"Fake packet injection node" ->** Inyecta paquetes en la red
- **"Direct Attack node"** -> Ataca a un nodo directamente

4.1 Atacante de tipo "Link-Noise"

Un nodo "Link-Noise" modifica dinámicamente la probabilidad de pérdida de paquete entre dos nodos reduciendo la calidad del enlace, "link", de comunicación. Por lo tanto durante un ataque los paquetes de los enlaces atacados se recibirán con mayor dificultad o incluso será imposible que lleguen a su destino.

LinkNoise (links[ ], power[ ],numPackets, time[ ], typePackets)

Figura 15: Definición del atacante Link-Noise

Como se muestra en la Figura 15, el nodo "Link-Noise" permite la definición de distintos parámetros:

- **Links:** Lista de los enlaces de comunicación o pares de nodos que son afectados por este nodo atacante.

- **Power**: Lista con la cantidad de ruido que será inyectada en cada enlace definido en el anterior parámetro. El ruido será el porcentaje adicional de probabilidad de pérdida de paquete que se añadirá a la probabilidad original del enlace afectado.

- **NumPackets:** El porcentaje de paquetes que son afectados por el incremento de probabilidad. Para los otros paquetes la probabilidad de pérdida de mensajes no será afectada por el nodo atacante.

- **Time:** Define los rangos de tiempo en los cuales el atacante afecta a la red. El atacante se puede encender o apagar múltiples veces durante la simulación.

- **TypePackets**: El ataque puede solo podría afectar a un determinado tipo de paquetes. Esto permite la simulación de ataques selectivos.

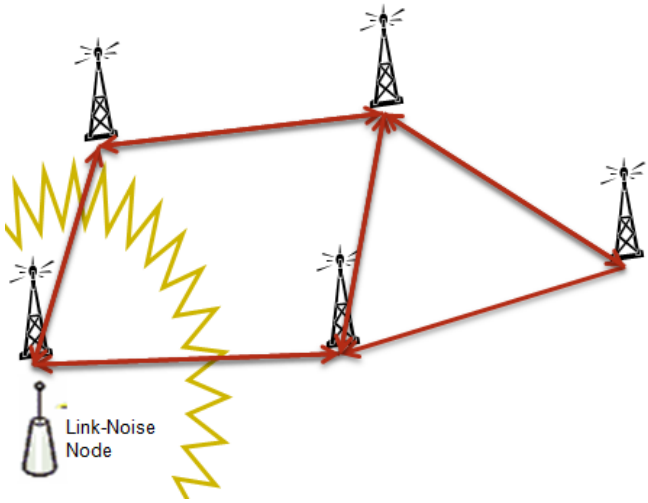

Figura 16: Atacante de tipo "Link-Noise"

Como se representa en la Figura 16, el nodo atacante por ruido es el responsable de introducir una pérdida de la calidad del canal en los enlaces especificados. Para la simulación eficiente de los nodos "Link-Noise", se ha modificado el modelo de red comentado anteriormente (Figura 13). Básicamente, el atacante modifica la probabilidad de pérdida de paquete para un cierto número de mensajes durante un periodo de tiempo predefinido. En la Figura 17 se puede ver el nuevo modelo de red en el que se incluye esta nueva probabilidad. Cuando un paquete tiene que ser transferido al nodo receptor, la probabilidad de recepción incluye la probabilidad del link original ("prob_loss" en la Figura 17) más el ruido adicional introducido por el atacante ("attackers" en la Figura 17).

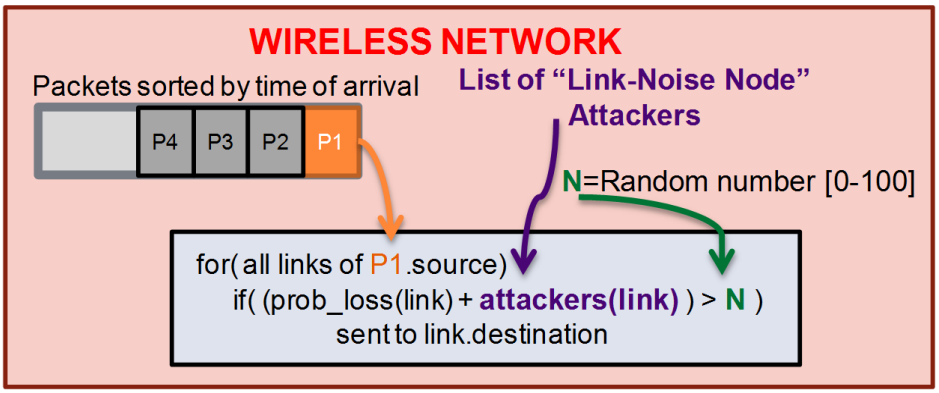

Figura 17: Simulación con atacantes Link-Noise

4.2 Ataque inyector de paquetes falsos

Este nodo atacante introduce paquetes falsos en la red con diferentes motivaciones. Los paquetes se reciben porque el nodo atacante forma un paquete con una estructura formalmente correcta. En la Figura 18 se muestra la definición de este tipo de atacantes con sus distintos parámetros. La estructura del paquete inyectado en la red es definido por los parámetros con los que se especifica el

atacante. Dichos parámetros permiten definir desde la frecuencia de inyección de paquetes hasta su tipo, el modo de ataque, etc.

```
FakeInjec(frequency, typePacket, time[ ], nodesDestine[ ], broadcast)
```

Figura 18: Definición del atacante inyector

Los parámetros que definen a este atacante son los siguientes:

- **Frequency:** Define la tasa de paquetes falsos inyectados en la red
- **TypePackets**: Define el tipo de paquetes que se van a inyectar. Pueden ser implementados distintos tipos de paquetes para abarcar todos los ataques descritos en secciones anteriores: desde paquetes "HELLO", RTS o CT a paquetes con un "pay-load" aleatorio.
- **Time**: Define el rango de tiempo en el cual el atacante está funcionando. El atacante puede encenderse y apagarse las veces que sean necesarias durante la simulación.
- **Nodes Destine:** Esta lista específica los nodos destino a los que se les va a inyectar paquetes.
- **Broadcast:** Si este atributo está definido, cada paquete inyectado será enviado en modo "broadcast" y llegará a todos los nodos visibles de la red

Para la simulación de este atacante, el nodo atacante inyectara paquetes en la cola de transmisión del modelo de la red. Una vez que el paquete se ha introducido en la cola, es el modelo de red el encargado de transmitir este falso paquete, ya que lo trata como a un paquete genuino. En la Figura 19 se puede ver una ampliación de la Figura 17, con la introducción de este nuevo atacante en el modelo de la red.

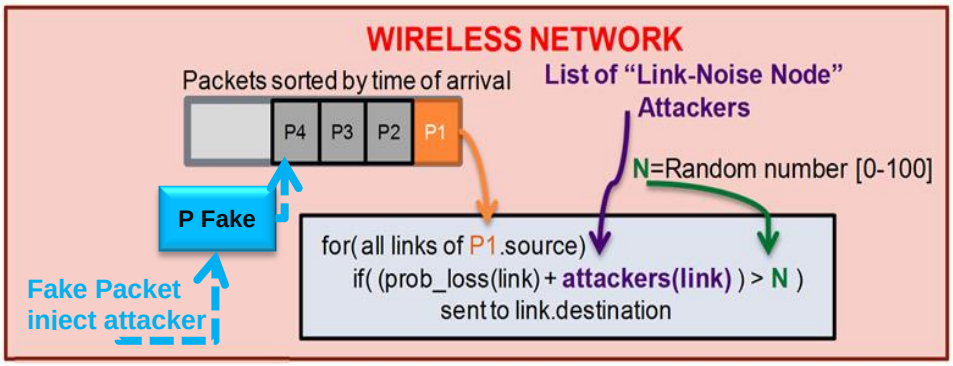

Figura 19: Simulación con atacantes inyectores

4.3 Nodo de ataque directo

El nodo por ataque directo modifica el software embebido en el nodo atacado. El modelo sustituye la programación de un nodo genuino por un programa falso. La definición del ataque incluye la nueva aplicación que será cargada en el nodo atacado y sus parámetros de red ("packet-loss probabilities"). En definitiva, lo que se hace es modificar el firmware de un nodo genuino por el software maligno y simular normalmente.

4.4 Relación entre los modelos de atacante propuestos y las vulnerabilidades detectadas.

En esta subsección se va a analizar la relación entre la Sección 2.2 de este documento (Seguridad en redes inalámbricas) y los modelos de atacantes descritos. En la Tabla 2 se puede ver la relación entre las vulnerabilidades en WSNs detectadas en la Sección 2 y los modelos de atacantes propuestos. Estos tres modelos cubren todas las vulnerabilidades detectadas. Por ejemplo, el nodo por ataque directo permite la simulación del ataque por "Overwhelm" (Sección 2.2.1.20) o por aplicación (Sección 2.2.1.19).

TABLA 2–ATACANTES UTILIZADOS PARA CADA ATAQUE

	Types of attacks		
Link-Noise node	Jamming	Interrogation	Fake packet injector node
	Collision	Energy Drain	
	Resource Exhaustion	Hello Flood	
	Black Hole Attack	Misdirection	
	Path-based DoS	Flooding	
	Homing	Overwhelm Attack	Direct attack
	Selective Forwarding	Application Attack	
Link + Injectorr	Spoofed	Sink Hole	
	Sybil	Node Replication	
	Tampering	Sniffing	Not affected

Como se puede ver en la Tabla 2, un nodo atacante de tipo "Link-Noise" puede modelar distintos ataques dependiendo de los parámetros con que definamos al atacante. Algunos de estos ataques pueden ser, por ejemplo, Jamming, Collision, Black Hole y Path-based. Por ejemplo, los ataques por Jamming y por Collision pueden ser modelados con el mismo nodo atacante (Link-Noise) pero definiendo distintos parámetros. Estos parámetros no solo se usan para definir ataques, sino también para definir distintas estrategias de cada ataque. Por ejemplo, en el ataque por Jamming hay múltiples estrategias [28]. Si por ejemplo, lo que se quiere es hacer un ataque por Jamming por Banda Parcial [28], el atacante podrá ser

modelado como un nodo “Link-Noise” en el cual el número de paquetes afectado sea el 50%

Otra vulnerabilidad que se puede modelar con este atacante es el ataque por Collision. Este ataque puede ser implementado de dos maneras distintas. La manera más directa es aquella en la cual el atacante conoce el canal de transmisión de paquetes. La manera indirecta es utilizar fuerza bruta: la probabilidad de corromper el paquete es directamente proporcional al número de canales inalámbricos.

Otro ejemplo es el ataque por interrogación. Este ataque se simula con el nodo atacante inyector de paquetes. En este caso, es necesario configurar el nodo atacante para que envíe constantemente mensajes CTS y RTS.

El modelado del ataque por "Sybil" es un caso especial, debido a que se necesita la introducción de dos tipos de nodos atacantes distintos: un "Noise-Link" y un inyector. El atacante “Noise-Link” aísla el nodo atacado y el Atacante inyector se encarga de insertar paquetes alterados.

En la Sección 2 de este documento, se presentan ataques por Tampering y por Sniffing. Sin embargo, estos ataques no se muestran en la Tabla 2. Esto es debido a que estos ataques no afectan al modo de operación del nodo ni de la red, sino que implican robo de información. Son ataques pasivos.

5 Caso de estudio

5.1 Informes del Simulador Virtual

Es necesario seguir una serie de pasos para utilizar el simulador en el modelado de plataformas hardware y redes inalámbricas que permitan la ejecución de aplicaciones SW en el modelo virtual. Normalmente, las aplicaciones software que se ejecutarán en los nodos están escritas en C/C++, siendo compiladas con el compilador del simulador virtual. Además, se requieren librerías y estimadores que modelen el hardware de cada nodo. Por supuesto también será necesario el archivo con la definición de la topología de la red. Una vez que todos estos archivos han sido procesados, el entorno genera un modelo ejecutable, el simulador virtual. Cuando se ejecuta este modelo, se podrán obtener las siguientes estimaciones para cada nodo de la red:

- **RTOS ->** Datos del Sistema operativo
  - *Processes* ->
    - *Created* -> Número de procesos creados
    - *Destroyed* -> Número de procesos destruidos
    - *Mean process duration* ->Tiempo del proceso principal
  - *User Time* -> Tiempo total de usuario
  - *Kernel Time* -> Tiempo total de Kernel.

- **Transceive**r -> Datos del interfaz de red
  - *Energy Standby* ->Energía del transceiver en modo standby (en Julios)
  - *Energy transmission* -> Energía del transceiver durante las transmisiones (en Julios)
  - *Power* -> Potencia del transceiver in Watts.
  - *Packets send* -> Número de paquetes enviados.
  - *Packets received* -> Número de paquetes recibidos.

- **Processor ->** Datos de cada procesador del sistema
  - *Number of switches*
    - *Thread switches* -> Número de cambios de hilo o thread
    - *Context switches* -> Número de cambios de contexto.
  - *Time estimations*
    - *Running time* -> Tiempo en ns de la CPU corriendo
    - *Use of CPU* -> % de uso de la CPU
  - *Instructions*
    - *Executed instructions* -> Número de instrucciones ejecutadas
    - *Cache misses* -> Número de misses en la cache de instrucciones

- *Data*
  - *Cache misses* -> Número de misses en la cache de datos

- *Energy and Power*
  - *Core* -> Energía y potencia del procesador en Julios y Watios respectivamente
  - *Instruction cache* -> Energía y potencia de la cache de instrucciones en Julios y Watios respectivamente
  - *Data cache* -> Energía y potencia de la cache de datos en Julios y Watios respectivamente

- **Simulation time** -> Tiempo de simulación en segundos

En la Figura 20, se puede ver una captura de pantalla de la interfaz de la plataforma virtual. Desde dicha interfaz es posible configurar el modelo de la red de forma sencilla, añadiendo nodos con sus características e indicando los códigos fuentes que ejecuta cada nodo. Además se podrán añadir los atacantes que se deseen. Los resultados de las estimaciones obtenidas para cada uno de los nodos se podrán ver dinámicamente en la interfaz, conforme avanza la simulación. En la parte inferior de la figura es posible apreciar una pequeña ventana con los informes finales generados por la herramienta, con resultados de las estimaciones comentadas anteriormente.

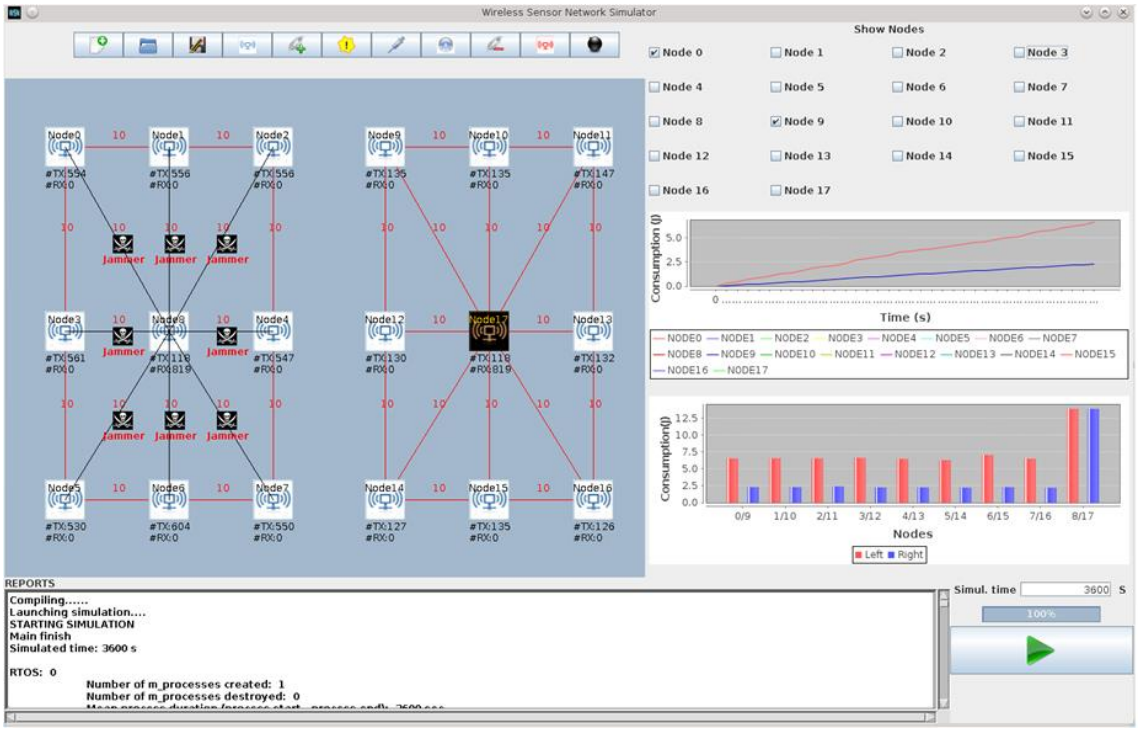

Figura 20: Captura de pantalla de la interfaz del simulador

En la Figura 20 se presenta la simulación de dos redes iguales, con el mismo número de nodos, el mismo software en los nodos de las redes y la misma topología

de la red. La diferencia es que a la primera red se la ha atacado mediante un atacante de tipo jammer, por lo que el consumo y prestaciones de la primera red son peores. A la derecha, en la parte superior, se puede seleccionar los nodos a representar gráficamente. Se mostrá la energía consumida por cada nodo seleccionado durante todo el tiempo de simulación. En la gráfica de la parte inferior se puede observar el consumo de cada nodo en cada red (red con nodos atacados versus red con nodos no atacados).

5.2 Caso de uso: Ataques sobre WSNs

En esta sección, se presentan resultados del simulador de redes inalámbricas con diferentes ejemplos.

5.2.1 Escenario

Esta sección describe los tres escenarios que se van a utilizar para demostrar la versatilidad y eficacia de la simulación de ataques. Se han diseñado diferentes escenarios donde se han introducido diferentes ataques con el objetivo de comparar los diferentes efectos de un mismo ataque tiene sobre distintas redes. Esto permite demostrar como un ataque que no parecía peligroso puede ser muy dañino para un tipo de redes o, al contrario, como un ataque peligroso para un tipo de redes no lo es para otros.

La funcionalidad del sistema simulado es la una monitorización periódica de la temperatura ambiental. Los nodos de los tres sistemas tienen arquitecturas hardware y aplicaciones software similares.

La red de sensores que vamos a simular integra dos tipos de nodos diferentes. Estos nodos tienen distintas arquitecturas hardware y distintas aplicaciones corriendo en cada uno. El primer tipo de dispositivo es el nodo central o Gateway. Él será el responsable de comunicarse con el “exterior” y será el único capaz de mandar mensajes fuera de la red, al tener un módulo GPRS. Además de esto, tendrá que recoger la información de los sensores de los otros nodos. Para ello se comunicará con el segundo tipo de nodo, el Sensor Board, o el End-device. Este nodo tendrá un sensor de temperatura con el que podrá leer la temperatura ambiental. Cuando el End-device lea esta información se encargará de enviar esta información al Gateway. El Gateway esperará para recibir esta información de los sensores para componer un mensaje con los datos. Este mensaje lo enviará el Gateway a través de su módulo GPRS a la central de datos.

Cuando un nodo termine su funcionalidad pasará a un estado dormido con el objetivo de reducir el consumo e incrementar el tiempo de vida de la batería.

5.2.2 Modelos de red

A continuación se van a describir tres modelos distintos de red, con nueve nodos en cada red interconectados con el esquema que se puede ver en las figuras 21 ,22 y 23. El primer ejemplo es una red inalámbrica mallada (Figura 21). La figura 22 representa el despliegue de una red Lineal y por último la figura 23 muestra una red circular. Cada nodo toma su medida y envía su lectura al nodo Gateway. En caso de

que el nodo no tenga visibilidad directa con el Gateway, la lectura será enviada a un nodo vecino para que ésta sea retransmitida al Gateway. Los porcentajes de las líneas rojas representan la probabilidad de perder un paquete del canal inalámbrico. Si no hay línea roja entre dos nodos, podemos asumir que la probabilidad es del 100% (No hay conexión directa)

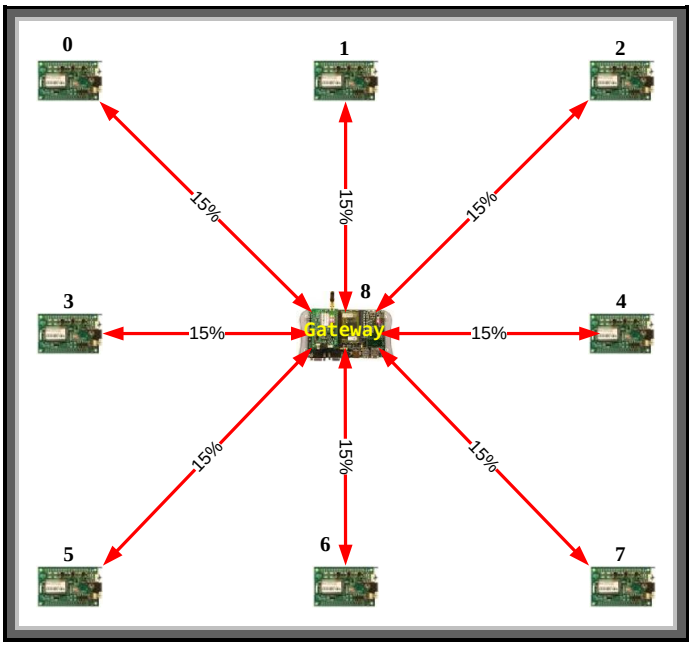

Figura 21: Representación del modelo de la red mallada que se va a estudiar.

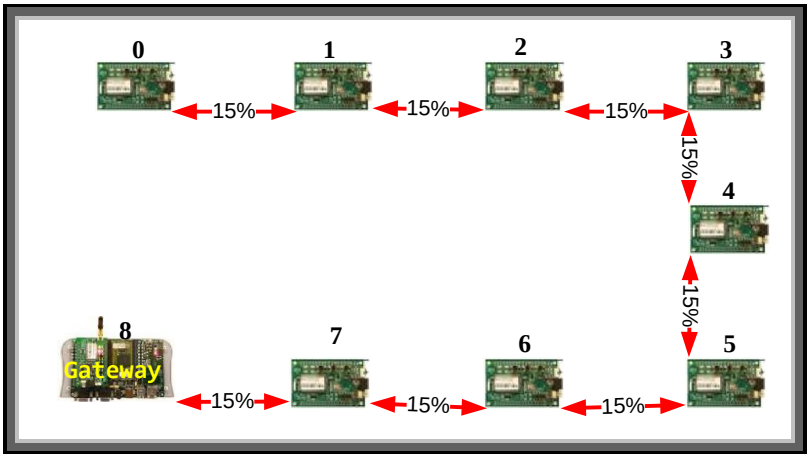

Figura 22: Representación del modelo de la red lineal que se va a estudiar.

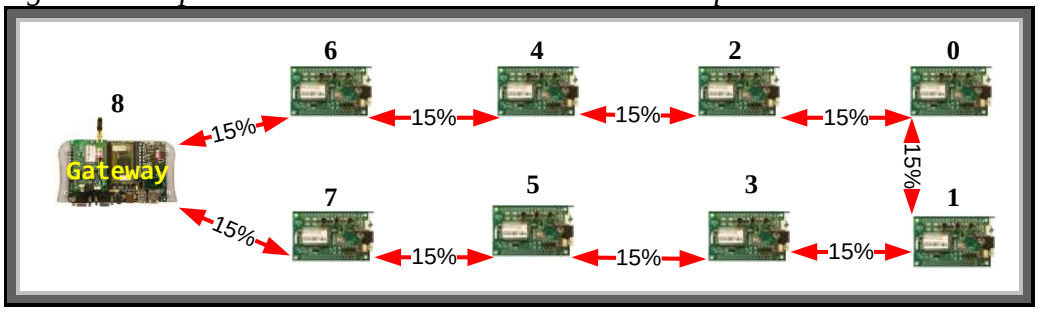

Figura 23: Representación del modelo de la red circular que se va a estudiar.

En este ejemplo, la probabilidad de error entre las transmisiones de los paquetes es siempre la misma (15%) entre los nodos interconectados. El uso de la misma probabilidad ayudará al lector a comprender más fácilmente los resultados obtenidos cuando se comparen los efectos de los distintos ataques sobre los tres tipos de redes. En ejemplos reales, donde hay disponibles datos exactos, se podrá definir cada link con la probabilidad que sea necesaria. En un caso real, sería necesario calcular la probabilidad de error con más precisión si se quisiesen obtener datos precisos con la simulación. Es importante incluir la probabilidad de interferencia causada por la acumulación de muchos nodos en un limitado espacio o por la presencia de otras redes. Por supuesto también es importante tener en cuenta los obstáculos físicos (por ejemplo paredes) que una señal de radio enviada por un nodo se pueda encontrar hasta llegar a su destino.

5.2.3 Modelo de la plataforma

Esta sección describe el modelo de la arquitectura hardware de cada nodo de la WSN. Todos los nodos tendrán el mismo hardware. Los componentes más importantes serán un procesador Cortex M3 fabricado por ARM [33], con un reloj de 90 MHz, una memoria y un módulo de radio XBee 802.15.4 [32]. Todos estos componentes se interconectarán mediante un bus de sistema.

Las diferencias principales entre los dos tipos de nodos se pueden ver en la Figura 24, y son los siguientes componentes:

- **Gateway**: Este nodo integra el módulo GPRS. Este componente se modela con un consumo y un tiempo asociado con cada mensaje enviado.
- **End Device:** El sensor responsable de tomas las medidas ambientales está integrado en este nodo. El sensor está implementado como un sensor genérico Lee la temperatura exterior cuando el sistema se lo requiere.

Un componente importante en todos los nodos es el módulo de radio o "transceiver". Está implementado junto con los registros de configuración para controlar su modo de operación. Estos registros los puede modificar la aplicación para configurar la funcionalidad del “transceiver” dinámicamente. Para leer y modificar estos registros se utilizara un API que implementa los comandos AT del XBee. Dicho API ha sido desarrollado e integrado en el simulador en el marco de este trabajo. Un ejemplo de los cambios en los resultados por la modificación del funcionamiento del “transceiver” se puede ver en la publicación [29], presentada por el autor de este mismo trabajo. En los ejemplos que se van a ejecutar se utiliza la configuración por defecto del módulo XBee-pro.

Los parámetros del módulo de radio que mas afectan a la simulación son los siguientes:

- Baud Rate: 9600bps
- Mac Retries: 10
- Multiple Transmissions: 0x3
- Power Level: 4 or 18dBm.

Como ya se ha explicado anteriormente, estos datos podrían ser modificados en cualquier momento por la aplicación a través de los comandos AT.

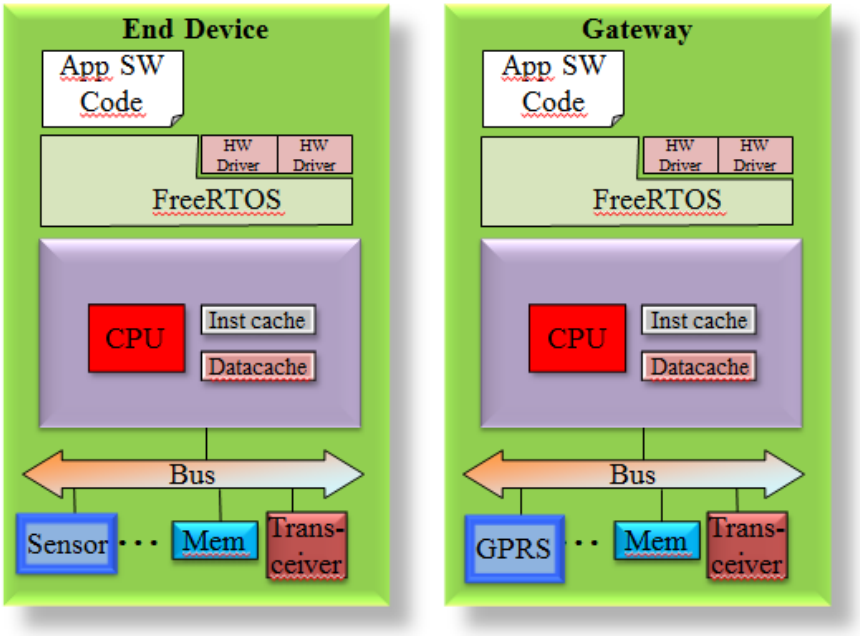

Figura 24: Representación de las arquitecturas de los dos tipos de nodos

5.2.4 Modelo de aplicación

Esta sección describe el software ejecutado por cada tipo de nodo. Como se explicó anteriormente, cada nodo corre un software distinto, ya que cada uno tiene una funcionalidad distinta. Sin embargo, como elemento común entre todas las aplicaciones, absolutamente todas están desarrolladas para correr con FreeRTOS.

A continuación, hay una pequeña explicación de cada aplicación:

- **"Gateway"**: El "Gateway" es el responsable de recibir los mensajes de los "End-Devices" para transmitir esa información más allá de la WSN. Cuando todos los "End-devices" están despiertos, el Gateway espera recibir todos los datos con cada temperatura de cada "End-device". Cuando el Gateway ha conseguido recopilar toda esta información, compone un mensaje y lo envía a través de su módulo GPRS. Después de esto, la aplicación manda una interrupción al nodo para que se duerma durante 30 segundos.

- **"End Device"**: La función del "End-Device" cuando se despierta es leer la temperatura del sensor y enviar esta información al Gateway o a otro sensor con el objetivo de que esa información llegue al Gateway. Cuando finaliza su función, duerme durante 30 segundos. Por lo tanto, la frecuencia de la adquisición de los datos es cada 30 segundos.

5.2.5 Ataques simulados

En cada una de las tres redes que se van a simular vamos a introducir 4 ataques en 5 simulaciones distintas (un ataque por simulación y una simulación sin ataques). En este caso de estudio, solo vamos a considerar 4 ataques ya que la variedad de ataques disponibles es muy amplia (Tabla 2), pero el simulador virtual permite la simulación de todos los ataques descritos. El primer ataque será un ataque por "Jamming" (Figura 25) en los "Gateways", con una efectividad del 60%. La figura 25A muestra el ataque en la red mallada. En la figura 25B el "Jammer" ataca al Gateway de la red lineal y en la Figura 25C es sobre la red circular.

El segundo ataque a simular será un ataque por "Sybil" (Figura 26) sobre el nodo 3. El tercer ataque será un ataque por interrogación (HELLO Flow) al nodo Gateway. Este ataque lo podemos ver representado en la Figura 27. Por último el cuarto ataque consiste en un ataque doble por "Jamming" + inyección de paquetes sobre el nodo 3 (Figura 28).

El objetivo de estas simulaciones es estimar cuál de estos ataques es más problemático (en términos de consumo de energía) para cada red.

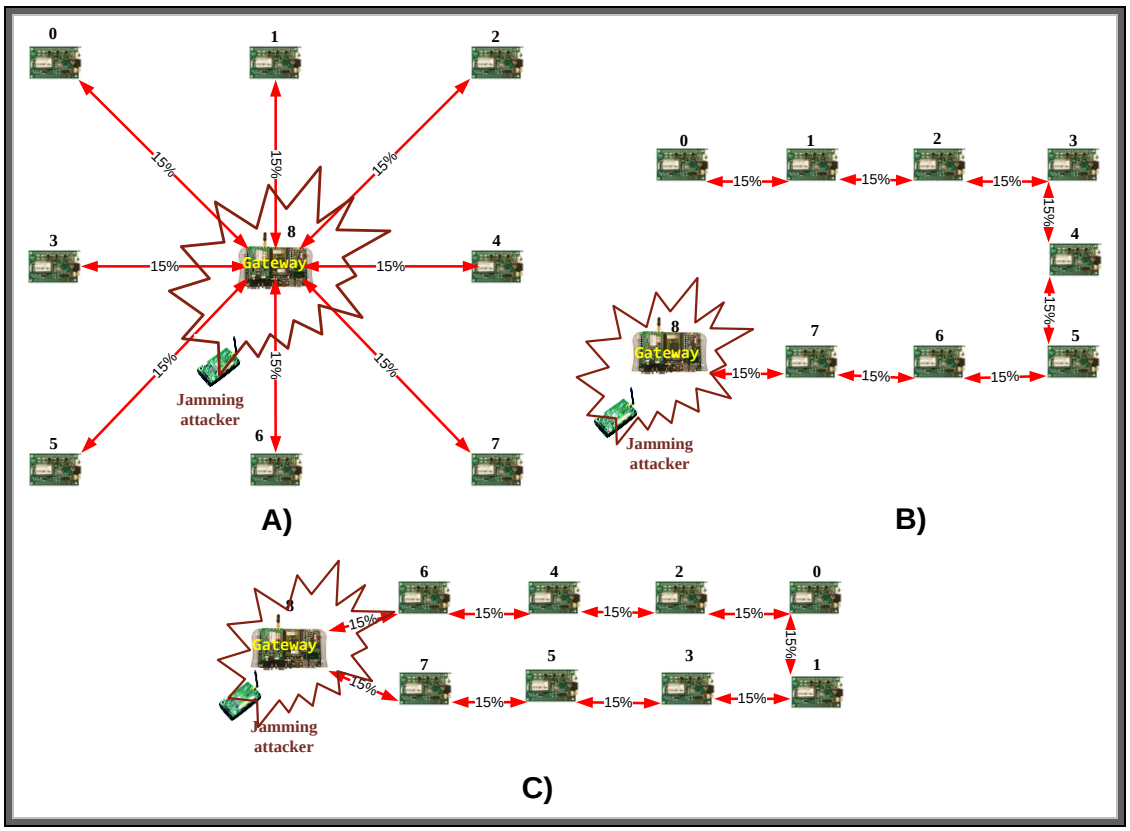

Figura 25: A)Ataque por Jamming sobre la red mallada. B) Ataque por Jamming sobre la red lineal C) Ataque por Jamming sobre la red circular.

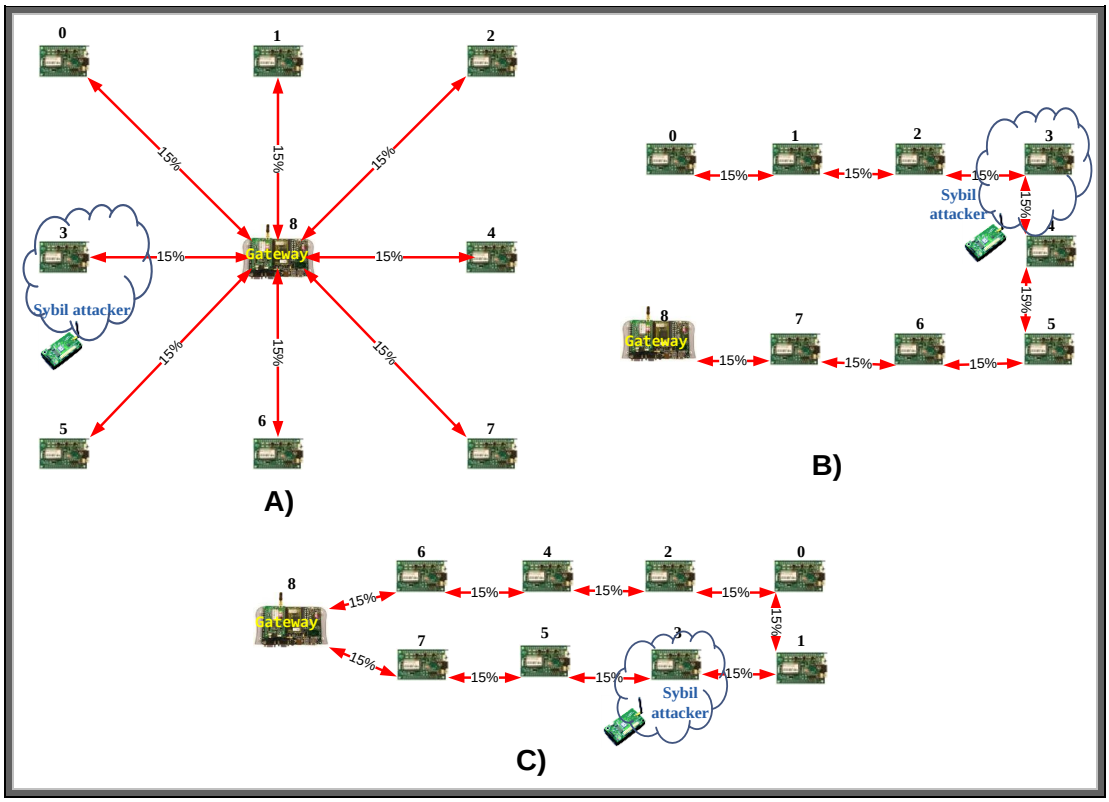

Figura 26: A)Ataque por Sybil sobre la red mallada. B) Ataque por Sybil sobre la red lineal C) Ataque por Sybil sobre la red circular.

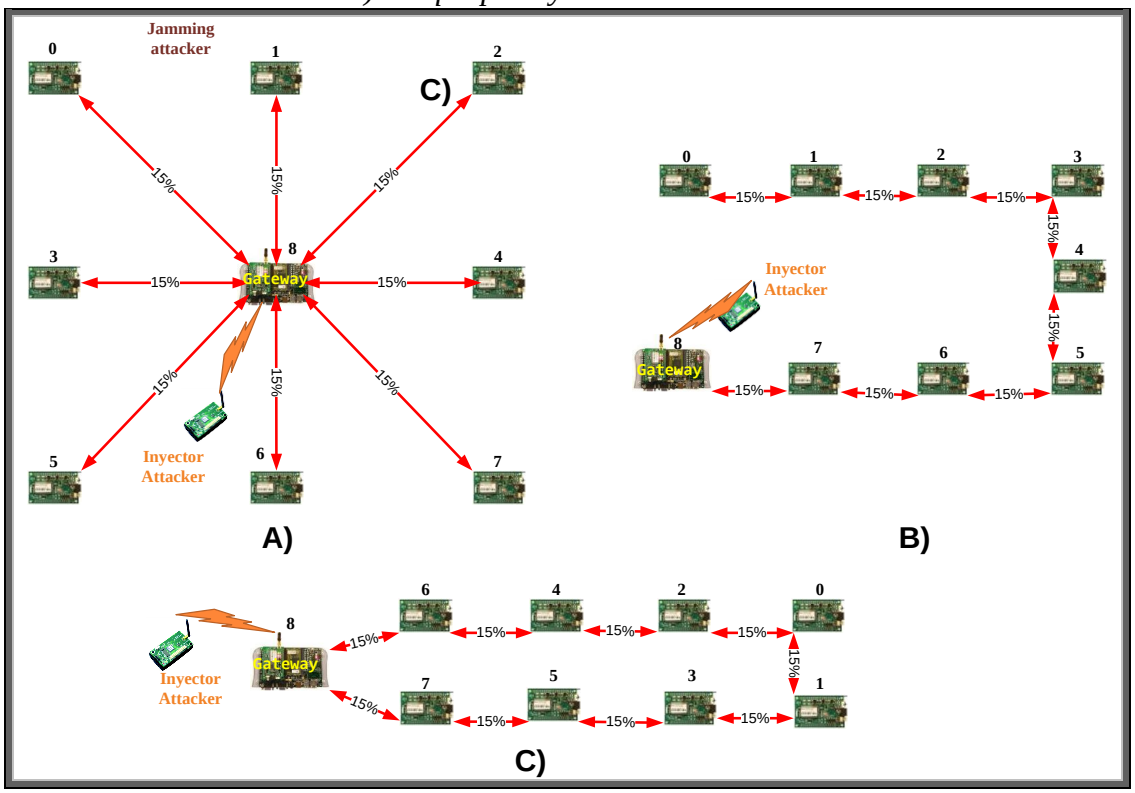

Figura 27: A)Ataque por Inyeccion sobre la red mallada. B) Ataque por Inyeccion sobre la red lineal C) Ataque por Inyeccion sobre la red circular

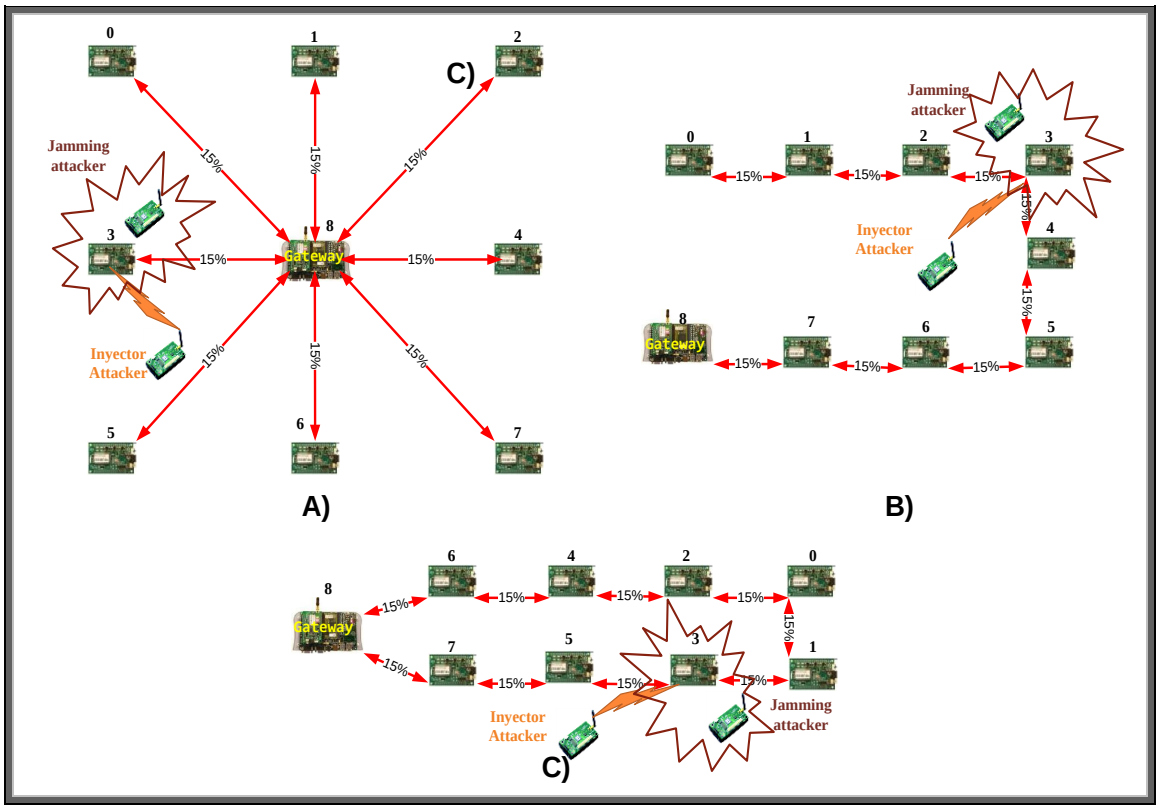

Figura 28: A)Ataque por Inyeccion + Jamming sobre la red mallada. B) Ataque por Inyeccion + Jamming sobre la red lineal C) Ataque por Inyeccion + Jamming sobre la red circular

5.2.6 Resultados Experimentales

En esta sección se representan los resultados obtenidos tras la simulación en el simulador virtual. Para la obtención de estos resultados, se ha simulado la cada red durante una hora. Se han simulado las tres redes bajo las distintas condiciones:

- Sin ataques
- Ataques por “Jamming”
- Ataques por “Sybil”
- Ataques por inyección
- Ataques “Jamming” + inyección

La Tabla 3 muestra el consumo de potencia de cada nodo en la red lineal. La línea 1 muestra el consumo de cada nodo con ataques, y las siguientes filas presentan el incremento de consumo de cada nodo debido al ataque. La Tabla 4 tiene una distribución similar pero para le red mallada, y lo mismo le pasa a la Tabla 5, la cual muestra los resultados de las redes circulares.

TABLA 3-RESULTADOS DE LA SIMULACIÓN EN LA WSN LINEAL

Consumo de la WSN Lineal										
	Node 0	**Node 1**	**Node 2**	**Node 3**	**Node 4**	**Node 5**	**Node 6**	**Node 7**	**Gateway**	**TOTAL**
Sin ataques	2,47J	2,47J	2,47J	2,47J	2,47J	2,47J	2,47J	2,47J	1,33J	21,11J
Jamming	0%	0%	0%	0%	0%	0%	0%	55%	89%	12%
Sybil	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Inyection (Hello Flood)	0%	0%	0%	0%	0%	0%	0%	0%	543%	34%
Jamming + Inyection	0%	0%	77%	326 %	38%	0%	0%	0%	0%	45%

TABLA 4- RESULTADOS DE LA SIMULACIÓN EN LA WSN MALLADA

Consumo de la WSN Mallada										
	Node 0	**Node 1**	**Node 2**	**Node 3**	**Node 4**	**Node 5**	**Node 6**	**Node 7**	**Gateway**	**TOTAL**
Sin ataques	1,91J	1,91J	1,91J	1,91J	1,91J	1,91J	1,91J	1,91J	9,18J	24,48J
Jamming	102%	102%	102%	102%	102%	102%	102%	102%	91%	98%
Sybil	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Inyection (Hello Flood)	0%	0%	0%	0%	0%	0%	0%	0%	28%	11%
Jamming + Inyection	0%	0%	0%	409%	0%	0%	0%	0%	0%	34%

TABLA 5- RESULTADOS DE LA SIMULACIÓN EN LA WSN CIRCULAR

Consumo de la WSN Circular										
	Node 0	**Node 1**	**Node 2**	**Node 3**	**Node 4**	**Node 5**	**Node 6**	**Node 7**	**Gateway**	**TOTAL**
Sin ataques	2,66J	2,66J	2,66J	2,66J	2,66J	2,66J	2,66J	2,66J	2,93J	24,21J
Jamming	0%	0%	0%	0%	0%	0%	0%	0%	86%	11%
Sybil	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Inyection (Hello Flood)	0%	0%	0%	0%	0%	0%	0%	0%	95%	11%
Jamming + Inyection	0%	0%	41%	189%	0%	43%	0%	0%	0%	30%

Como se puede observar en la Figura 29, el consumo total de la red lineal es un 14% inferior que la red mallada y la circular en condiciones normales (sin ataques). Esto es debido a la menor carga de trabajo de los nodos sensores, porque ellos se deben comunicar con sus vecinos. Sin embargo, en la Figura 29 se puede observar que el consumo de la red mallada y la circular es muy similar en la mayoría de los casos. Un caso importante es el ataque por Jamming sobre el Gateway de la red mallada. Este ataque causa un importante incremento del consumo en la red (98%) y en el nodo más importante de la red (91%). Esto se puede observar en la Figura

30. En un primer vistazo a las Figuras 29 y 30 se puede considerar que la red mallada es la peor en términos de consumo. Sin embargo, si examinamos detenidamente las tablas 3, 4 y 5 se puede ver que el incremento del consumo no es mucho mayor. Por ejemplo, en el ataque por jamming, el consumo de la red mallada se dobla mientras que la res lineal y la circular apenas se ven afectadas.

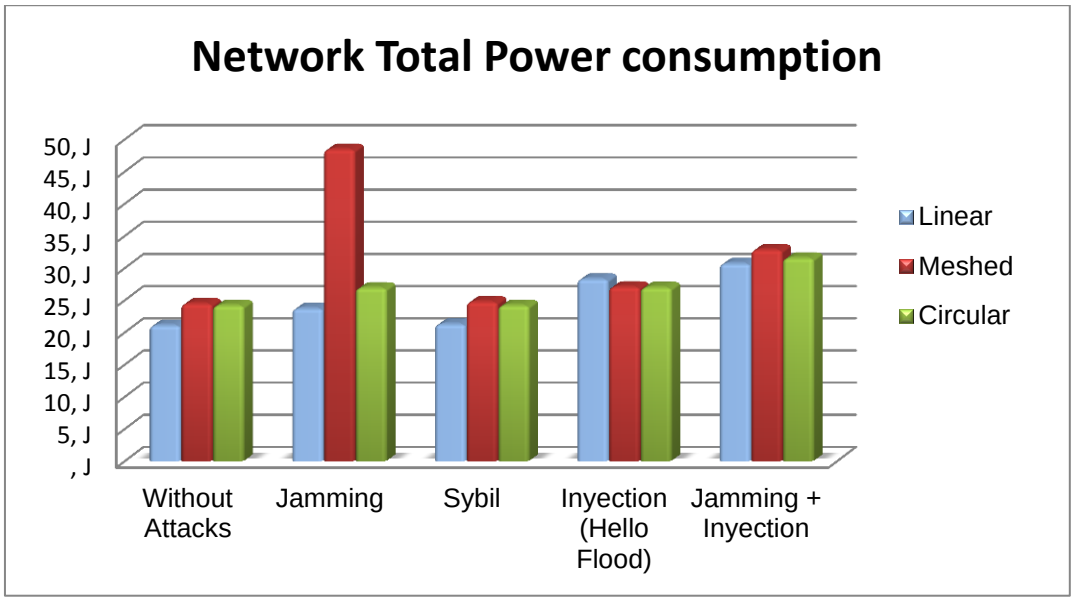

Figura 29 Consumo de las redes

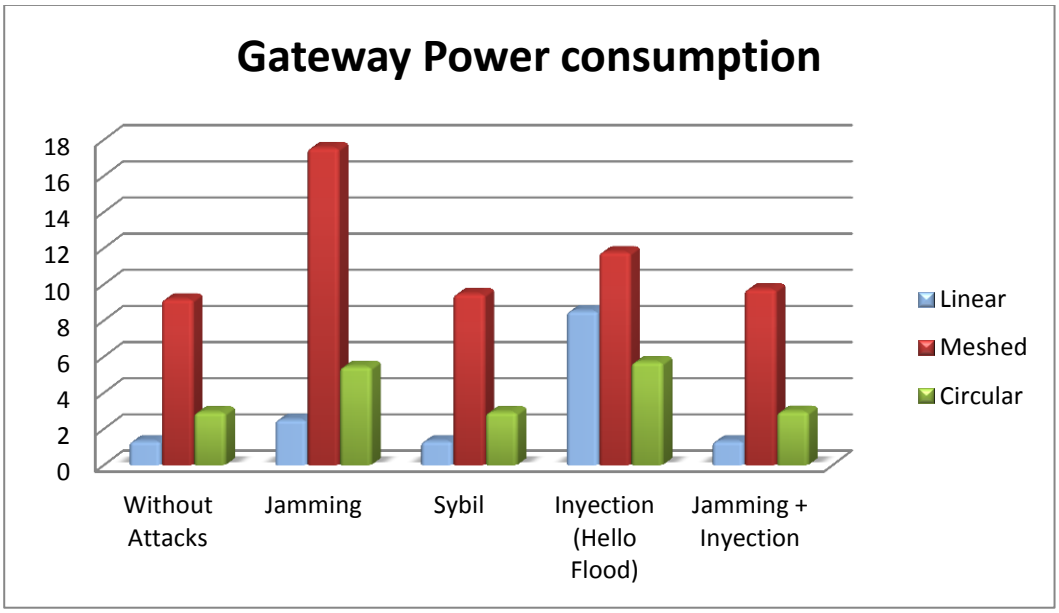

Figura 30 Consumo del Gateway de las redes

En el caso del ataque por "Sybil", ninguna red ve afectada su consumo. Esto es debido a que el número de paquetes enviados y recibidos por cada nodo es exactamente el mismo que si no estuviesen atacando, ya que el atacante no modifica el tráfico final de la red, sino únicamente el contenido de cada paquete. Estos resultados no implican que la red mallada sea menos segura que las demás

redes, porque, por ejemplo, en el caso del ataque por "Sybil", el atacante solo puede corromper la lectura de un sensor (nodo 3), sin embargo, en el caso de las redes lineales y circulares el atacante puede corromper las lecturas del nodo 3 y de los nodos anteriores al 3 al pasar los paquetes de los anteriores nodos por el nodo 3.

El simulador virtual permite detectar estos efectos funcionales como se puede ver en [30].

6 Conclusiones

En este trabajo se han analizado los problemas de seguridad que pueden presentar las redes de sensores inalámbricas, se han estudiado los ataques que pueden sufrir, y se ha propuesto una nueva metodología que permite estimar las consecuencias de estos. Dicha metodología permite simular los efectos de los ataques sobre cada red. Con el estudio de las estimaciones obtenidas a través de las simulaciones, los desarrolladores podrán tomar decisiones precisas con el objetivo de optimizar las aplicaciones software y las arquitecturas hardware de los nodos y estudiar el impacto del despliegue de las redes desde las primeras etapas del proceso de diseño. Como resultado, los desarrolladores serán capaces de proveer más seguridad a las redes de sensores inalámbricas al detectar los posibles problemas rápidamente, acortando el tiempo de llegada del producto final al mercado (time-to-market).

La técnica propuesta permite la simulación de ataques en redes de sensores inalámbricas. La plataforma virtual propuesta incluye soporte para el Hardware de los nodos, el software embebido, distintos RTOS, topologías de redes y por modelos de ataques propuestos. Para poder simular los 20 ataques detectados en el estudio realizado en la sección 2 de este documento se han propuesto tres tipos distintos de atacantes. El atacante inyector de ruido o nodo “Link-Noise”, el nodo inyector de paquetes falsos o nodo “Injector”, y el atacante directo. Estos modelos consiguen cubrir las 20 vulnerabilidades detectadas mediante la configuración y la combinación de los atacantes descritos.

Los resultados experimentales demuestran que la técnica propuesta es válida para analizar el impacto en el consumo y la funcionalidad de los ataques en las redes de sensores inalámbricas.

7 Publicaciones

Como resultado de este trabajo, se han publicado diversos artículos en conferencias internacionales:

- Wireless Sensor Network Simulation for Security and Performance Analysis, DATE 2013
- Security of Low Power Wireless Sensor Meshed Network, TRUDEVICE 2013
- Virtual platform for power and security analysis of wireless sensor network, SPIE 2013
- Modeling and Simulation of Secure Wireless Sensor Network, FDL 2012
- Virtual Platform for Wireless Sensor Networks, DSD Euromicro 2012
- Simulation of attacks in Wireless Sensor Network, DCIS 2012

También se han presentado dos posters y se ha realizado una demostración en conferencias relevantes en el área:

- Demonstration in ENF Euro Nano Forum 2013
- Poster in University Booth DATE 2012, Dresden on March 12th-16th
- Poster in Nanoelectronics forum 2011, Dublin on November 15th-16th

Además de estas publicaciones, hay dos artículos de revista con índice de impacto pendientes de revisión que esperamos que pronto puedan ser publicadas y un capítulo de un libro sobre el proyecto de investigación TOISE [31] enfocado a la seguridad en redes inalámbricas de bajo consumo.

8 Referencias

[1] Kavitha, T. and Sridharan, D. (2010). Security vulnerabilities in wireless sensor networks: A survey. Journal of Information Assurance and Security, 5:31-44.

[2] Wang, Y., Attebury, G. and Ramamurthy, B. (2006). A survey of security issues in wireless sensor networks. IEEE Communications Surveys & Tutorials, Vol. 8, No. 2

[3] Mohammad Al-Rousan, Taha Landolsi, Wafa M. Kanakri "Energy consumption considerations in dynamic wireless sensor networks with nodes and base stations mobility" Int. J. of Sensor Networks, 2010 Vol. 7 No. 4

[4] M. Mekni, B. Moulin, "A survey on sensor webs simulation tools", International Conference on Sensor Technologies and Applications, 2008, pp. 574-579.

[5] A. Stetsko et al; “Calibrating and Comparing Simulators for Wireless Sensor Networks”. MASSIEEE 2011, p. 733-738.

[6] NS-2, “The Network Simulator”, 2007

[7] OMNeT+-, "www.omnetpp.org" 2012

[8] X. Zeng, R. Bagrodia, and M. Gerla, "GloMoSim: a library for parallel simulation of large-scale wireless networks," presented at Twelfth Workshop on Parallel and Distributed Simulation, 1998.

[9] Levis P., Lee N., "TOSSIM: A Simulator for TinyOS Networks", pal@cs.berkeley.edu, September 17, 2003

[10] Ke Deng; Bradford G. Nickerson "A fuzzy control framework for wireless sensor networks" Int. J. of Sensor Networks, 2013 Vol.13, No.1, pp.1 – 12

[11] Titzer, B. L., Palsberg, J., and Lee, D. K. Avrora: Scalable sensor network simulation with precise timing. Conference on Information Processing in Sensor Networks 2005.

[12] Mohanty, P., Panigrahi, S., Sarma N., and Satapathy S. S. (2010). Security issues in wireless sensor network data gathering protocols: A survey. Journal of Theoretical and Applied Information Technology, 14-27.

[13] Mohammadi, S., Jadidoleslamy, H. (2011). A Comparison of Link Layer Attacks on WSN. International journal on applications of graphs theory in wireless ad hoc networks and sensor networks. Vol. 3, No. 1, March 2011, 35-56.

[14] Remove for Blind Revision

[15] Mohammadi, S., Jadidoleslamy, H. (2011). A Comparison of Link Layer Attacks on WSN. International journal on applications of graphs theory in wireless ad hoc networks and sensor networks. Vol. 3, No. 1, March 2011, 35-56.

[16] Said El Abdellaoui; Mérouane Debbah; Youssef Fakhri; Driss Aboutajdine "Increasing network lifetime in an energy-constrained wireless sensor network" Int. J. of Sensor Networks, 2013 Vol.13, No.1, pp.44 – 56

[17] Mohammad Al-Rousan, Taha Landolsi, Wafa M. Kanakri "Energy consumption considerations in dynamic wireless sensor networks with nodes and base stations mobility" Int. J. of Sensor Networks, 2010 Vol. 7 No. 4

[18] Mohammadi et al A Comparison of Link Layer Attacks on WSN. International journal on applications of graph theory in wireless ad hoc networks and sensor networks. Vol. 3, No. 1, March 2011.

[19] A. Becher et al. “Tampering with Motes: Real-World Physical Attacks on Wireless Sensor Networks.” Lecture Notes in Computer Science, (3934) 104-118, Springer, 2006.

[20] P Reindl, K Nygard, D Xiaojiang, Defending malicious collision attacks in wireless sensor networks. IEEE/IFIP Conference on Embedded and Ubiquitous Computing (EUC). 2010.

[21] Abdullah M.Y, Alsharabi N; "Wireless Sensor Networks Misdirection Attacker Challenges and Solutions" IEEE International Conference of Automation ICIA, June 2008.

[22] Posadas, H., Castillo, J., Quijano, D., Fernandez, V., Villar, E., & Martinez, M. (2010). SystemC Platform Modeling for Behavioral Simulation and Performance Estimation of

Embedded Systems. In L. Gomes, & J. Fernandes (Eds.), Behavioral Modeling for Embedded Systems and Technologies: Applications for Design and Implementation (pp. 219-243). Hershey, PA: Information Science Reference. doi:10.4018/978-1-60566-750-8.ch009

[23] Torres, R.P., Valle, L., Domingo, M., Loredo, S., Diez, M.C. "CINDOOR: An engineering tool for planning and design of wireless systems in enclosed spaces" (1999) IEEE Antennas and Propagation Magazine, 41 (4), pp. 11-21

[24] .(Kees) Pronk, Verifying FreeRTOS; a feasibility study, Report TUD-SERG-2010-042, Delft University of. Technology, 2010

[25] Adi Mallikarjuna V. Reddy, A.V.U. Phani Kumar, D. Janakiram, G. Ashok Kumar "Wireless sensor network operating systems: a survey" Int. J. of Sensor Networks, 2009 Vol. 5 No. 4

[26] H. Posadas, Á. Díaz, E. Villar "SW Annotation Techniques and RTOS Modeling for Native Simulation of Heterogeneous Embedded Systems" Kiyofumi Tanaka: "Embedded Systems - Theory and Design Methodology", InTech, Croatia. 2012

[27] http://confidential.eetimes.com/analysis-reports/4230404/2011-Embedded-Market-Study

[28] A. Diaz, P. Piñil, P. Sanchez “Modeling and Simulation of Secure Wireless Sensor Network”, FDL 2012

[29] A. Diaz, P. Sanchez “Simulation of attacks in Wireless Sensor Network”, DCIS 2012

[30] A. Diaz, P. Gonzalez, J. Gonzalez-Bayon, P. Sanchez “Virtual platform for power and security analysis of wireless sensor network”, SPIE 2013

[31] www.toise.eu

[32] http://www.digi.com/products/wireless-wired-embedded-solutions/zigbee-rf-modules/point-multipoint-rfmodules/xbee-series1-module#overview

[33] http://www.arm.com/products/processors/cortex-m/cortex-m3.php