TRABAJO FINAL DE CARRERA

TÍTULO DEL TFC: Banco de pruebas para redes ad-hoc.

TITULACIÓN: Ingeniería Técnica de Telecomunicación, especialidad Telemática

AUTORES: José Cavero Miranda
Francisco Sánchez Gómez

DIRECTOR: Carles Gómez Montenegro

FECHA: 20 de diciembre de 2011

Título: Banco de pruebas para redes ad-hoc.

Autores: José Cavero Miranda
Francisco Sánchez Gómez

Director: Carles Gómez Montenegro

Fecha: 20 de diciembre de 2011

Resumen

Las redes ad-hoc son sistemas autónomos de redes temporales dinámicas formadas por un conjunto de nodos móviles inalámbricos, sin la intervención de puntos de acceso centralizados o de infraestructura de ningún tipo.

En este tipo de redes, es necesario contar con un protocolo eficaz, capaz de adaptarse a cambios en la topología de la red y proporcionar posibles rutas para continuar la transmisión de datos. La capacidad de encaminar tiene efectos vitales sobre estos sistemas, o sobre cualquier red con más de un enlace.

Hay diferentes protocolos para la funcionalidad de enrutado en redes ad-hoc. Este proyecto se centrará en dos algoritmos bien conocidos: Optimized Link State Routing Protocol *(OLSR)* y Ad-hoc On-demand Distance Vector Protocol *(AODV)*.

El objetivo principal del mismo, es la creación de una maqueta real de red ad-hoc con propósitos docentes. Esta maqueta debe permitir evaluar el rendimiento de la red en función del protocolo de encaminamiento empleado (OLSR y AODV), y de su configuración.

Adicionalmente, el proyecto analizará el rendimiento de ambos protocolos, el cual será analizado y comparados en función de criterios relevantes como la latencia extremo a extremo, el ancho de banda extremo a extremo, y la latencia de cambio de ruta, considerando distintas configuraciones de parámetros.

Como resultado del proyecto, se ha construido la maqueta de redes ad-hoc y se han documentado las instrucciones y elementos necesarios para crearla (y ponerla en marcha).

Por otra parte, los experimentos realizados han permitido contrastar el comportamiento de las implementaciones de OLSR y AODV evaluadas (no siempre acorde a lo que cabría esperar) y extraer el rango de resultados de rendimiento que se pueden obtener al emplear distintas configuraciones de los dos protocolos indicados.

Title: Testbed for ad-hoc networks

Authors: José Cavero Miranda
Francisco Sánchez Gómez

Director: Carles Gómez Montenegro

Date: December, 20th 2011

Overview

Ad-hoc networks are autonomous systems of temporary dynamic networks that are formed by a set of wireless mobile nodes without the intervention of centralized access points or any infrastructure.

In such networks, it is necessary to have an effective protocol to suit be able to changes at network topology, and provide possible routes to continue the data transmission. Capacity to route has vital effects on these systems, or any other network with more than one link.

There are different protocols for routing function in ad-hoc networks. This research will focus on two well-known algorithms: Optimized Link State Routing Protocol *(OLSR)*, and ad-hoc On-demand Distance Vector Protocol *(AODV)*.

The project's main objective is the creation of a real model for ad-hoc network with teaching purposes. This model should allow assessment of network performance, based on the routing protocol used (OLSR and AODV), and its configuration.

Additionally, both protocol's performances will be analyzed and compared in this project, in terms of relevant criteria such as end to end latency, end to end bandwidth, and latency of the route's change, considering different parameter settings.

As a result of the research, has built a model's ad-hoc networks and have documented instructions and information necessary to create (and implement) it.

On the other hand, experiments have allowed to compare OLSR and AODV implementation's performance (not always according to expected) and get the range of performance results obtained by using different protocol settings.

No desaprovecharemos la ocasión de presentar nuestros agradecimientos más sinceros a los profesores que nos han ayudado a ir superando cada etapa de la carrera, muy especialmente a nuestro tutor, Carles Gómez, que con su apoyo, consejos y directrices ha sido posible la culminación de este trabajo.

A nuestras familias más cercanas, a Conchi y a Betty, por forjar nuestro núcleo con un cariño diario e incondicional.

A los amigos y compañeros, de los que aprendemos cada día.

ÍNDICE

Índice de figuras

Índice de tablas

ACRÓNIMOS

AP	Acces Point
ARQ	Automatic Repeat-Request
BER	Bit Error Rate
BSS	Basic Service Set
DSR	Dynamic Source Routing
DSSS	Direct Sequence Spread Spectrum
ETSI	European Telecomunications Standards Institute
FHSS	Frequency Hopping Spread Spectrum
GPL	General Public License
IANA	Internet Assigned Numbers Authority
IBSS	Independent Basic Service Set
IDS	Intusion Detection System
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
IrDa	Infrared Data Association
LAN	Local Area Network
LQ	Link Quality
MANET	Mobile ad hoc networks
MIMO	Multiple-in, Multiple-out
NLQ	Neighbor Link Quality
OSPF	Open Shortest Path Fisrt
P2P	Point to Point
PC	Personal Computer
PRNET	Packet Radio Network
QoS	Quality Service
RFC	Request For Comments
RIP	Routing Information Protocol
RTT	Round-Trip Time
SCTP	Stream Control Transmisson Protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
Wi Fi	Wireless Fidelity
WLAN	Wireless Local Area Network
ZRP	Zone Routing Protocol

PREÁMBULO

Introducción

Es difícil empezar un trabajo sobre redes de comunicación sin hablar del explosivo crecimiento de Internet, del auge de las tecnologías inalámbricas y de la proliferación de aparatos cada vez más pequeños susceptibles de conectarse a una red.

En los años 80, con la aparición de los primeros ordenadores personales *(PC)*, y la extensión de las redes locales, cobran especial importancia los sistemas distribuidos. Durante los 90, los ordenadores son lo bastante pequeños como para moverse: aparece la computación móvil, y como era de prever, los ordenadores también han de poder comunicarse mientras se mueven.

Por su parte, los usuarios cada vez son más numerosos y exigentes, pues dependen en mayor medida de estas tecnologías, demandan continuamente mayores y nuevas prestaciones, desde cualquier sitio donde estén. Por tanto este auge se debe considerar de manera pragmática, un aumento cualitativo y cuantitativo.

En cuanto a los dispositivos, el ordenador de hoy ya no tiene por qué ser un ordenador de sobremesa, ni tan si quiera tener el tamaño de un ordenador portátil. Puede tener dimensiones mucho menores como una agenda electrónica, puede estar integrado en un teléfono móvil, o simplemente ser un sensor encargado de almacenar o medir diferentes variables; sin obviar la capacidad de éstos de conectarse a una red, por ejemplo Internet.

Ante todo esto, han ido surgiendo nuevas tendencias que buscan que las aplicaciones informáticas actuales no se sustenten sólo en redes pre-existentes, sino que la red surja cuando la situación y el medio lo requieran, de forma autónoma, auto-organizada, distribuida, escalable e independiente de las tecnologías.

En esta nueva visión de las redes, toman la misma relevancia tanto los extremos de la comunicación, como los nodos intermedios, que pueden ser ordenadores de alto rendimiento, o simples nodos sensores con capacidad de cálculo restringida. Para el primer caso se puede usar el término *MANET* (redes ad-hoc móviles), como después se analizará, y para el segundo se habla de *redes de sensores*.

Motivaciones

Con lo poco que hasta ahora se ha visto, resulta fácil entender lo interesante que puede ser centrarse en el estudio de esta nueva tendencia en las redes de comunicación, en continua evolución a fin de mejorar la implementación de las mismas. Este interés lo despierta el hecho de tener la posibilidad de ofrecer una modesta aportación en este campo, en la que en un futuro no lejano, los autores de este trabajo vislumbran su definitiva implantación y consolidación en diferentes entornos comerciales.

Por otra parte, la asignatura *Movilidad, Redes y Servicios* del plan de estudios de grado en Ingeniería Telemática de la EETAC, prevé que los estudiantes de dicha asignatura realicen el trabajo práctico en relación con las redes ad-hoc.

Sin embargo, la EETAC no dispone actualmente de un escenario de pruebas para tal objetivo.

Objetivos

Uno de los propósitos de este trabajo es estudiar, desde un punto de vista experimental (en un escenario de pruebas de laboratorio de la universidad), las prestaciones de las implementaciones más utilizadas de los protocolos de encaminamiento para redes ad-hoc, en cuanto al retardo de la transmisión desde un nodo origen al destino, el ancho de banda que consumen, y la eficacia en el encaminamiento tras una ruptura en el enlace primario.

Por otra parte, se pretende aportar documentación con pruebas reales, manuales de instalación y configuración de este entorno, para objetivos docentes y que futuros estudiantes de la materia puedan experimentar con estos protocolos, ajustando su comportamiento mediante la modificación de sus parámetros más relevantes.

Como protocolos de encaminamiento para el proyecto, consideraremos el protocolo OLSR *(Optimized Link State Routing)*, y el AODV *(Ad-hoc On-demand Distance Vector)*. Cada uno de ellos es el más utilizado dentro de los tipos de protocolos de enrutado que engloban estas redes.

Estructura

Este trabajo, se estructura, además del preámbulo, en cuatro capítulos diferentes.

Estos cuatro capítulos son:

- **Capítulo 1. Redes ad-hoc: Conceptos y tecnologías :** Hace una descripción introductoria del conjunto de las redes inalámbricas, los estándares que rigen en la actualidad esta tecnología, y, una vez enmarcado en su contexto, se abordan las principales características de las redes ad-hoc, presentando especial interés en el encaminamiento de éstas.

- **Capítulo 2. Protocolos de encaminamiento:** Profundiza en los mecanismos sobre los que se apoyan los protocolos OLSR y AODV, ya que se tratan de los protocolos cuyas prestaciones se ponen bajo estudio.

- **Capítulo 3. Escenario, experimentos y resultados:** Donde se empieza haciendo una descripción de los diferentes escenarios, las herramientas de trabajo a nivel de hardware y de software que se harán servir en el proyecto, y una comparativa de ambos protocolos, fruto del análisis de los resultados obtenidos.

- **Capítulo 4. Conclusiones y líneas futuras:** Describe las conclusiones a la que llegan los autores, derivados de los resultados obtenidos; si se ha conseguido llevar a un escenario de laboratorio real los protocolos de encaminamiento en cuestión, y propone las líneas futuras y tendencias de estudio que hay sobre este tema.

Finalmente, se añade la **Bibliografía** de los diferentes escritos y documentación varia que se ha recopilado. Además se incluye en el apartado de **Anexos** la parte del material que se ha generado en el proyecto, como un manual de instalación y configuración de los protocolos considerados en un entorno de pruebas de laboratorio.

CAPÍTULO1. REDES AD-HOC: CONCEPTOS Y TECNOLOGIAS

1.1. Tecnologías 802.11

1.1.1. Descripción

Las tecnologías de la información ponen a disposición de los usuarios la posibilidad de conectar los equipos sin la necesidad de enlaces físicos (cables).

Una red de área local inalámbrica *(WLAN,* Wireless LAN*)*, es una red que cubre un área equivalente a la red local *(LAN,* Local Area Network), empleando medios de transmisión inalámbricos.

Esta red tiene que permitir que los nodos de la misma se puedan comunicar entre sí dentro del área de cobertura. La principal característica es la capacidad que puede tener el equipo terminal de movilidad, esto le otorga a la WLAN un grado de flexibilidad que entornos cableados convencionales no tienen.

Estas redes pueden ser usadas con dos topologías distintas:

- Modo IBSS, *Independent Basic Service Set* (modo Ad-Hoc): A grandes rasgos, son aquellas que sin la necesidad de una infraestructura fija los dispositivos se comunican directamente entre sí. Posteriormente se profundizará en este modo de operación de las redes inalámbricas, puesto que es pieza fundamental para este proyecto.

- Modo BSS *Basic Service Set* (modo Infraestructura): Aquí, los terminales de la red se comunican con un nodo principal que será el punto de acceso (*AP*, Acces Point), a través del cual se interconectarán y a su vez, accederán a redes externas.

1.1.2. Estándares WLAN

Cabe destacar los siguientes protocolos estandarizados:

- IrDa, basadas en infrarrojos. En desuso, ya que necesita visión directa.

- IEEE 802.11: Familia de estándares de la IEEE[1] (USA), adoptados por la Wireless Fidelity Alliance para sistemas inalámbricos.

[1] *IEEE.* Institute of Electrical and Electronics Engineers

- HiperLAN2: definido por la ETSI[2] (Europa) y que soporta velocidades de 54 Mbps, en la banda de 5GHz.

Mayoritariamente, las redes inalámbricas cumplen con los estándares genéricos aplicables al mundo de las LAN cableadas, pero necesitan una normativa específica adicional que defina el uso de los recursos radioeléctricos.

Estas normativas específicas definen de forma detallada los protocolos de la capa física y de la capa de enlace, que regulan la conexión vía radio.

El primer estándar de WLAN lo generó el organismo IEEE en 1997 y se denomina IEEE 802.11. Desde entonces varios organismos internacionales han desarrollado una amplia actividad en la estandarización de normativa de WLAN y han generado un abanico de nuevos estándares. En EEUU el grueso de la actividad lo mantiene el organismo IEEE con los estándares 802.11 y sus variantes (b, g, a, e, n, h, etc.) y en Europa el organismo relacionado es el ETSI con sus actividades en HiperLAN.

La Tabla 1.1 muestra las características técnicas de las tecnologías WLAN más significativas.

Estándar WLAN	**IEEE 802.11b**	**IEEE 802.11a**	**HiperLAN2**
Organismo	IEEE	IEEE	ETSI
Fidelización	1999	2002	2003
Denominación	WiFi	WiFi	
Banda de frecuencia	2,4 GHz ISM	5 GHz	5 GHz
Velocidad máxima	11 Mbps	54 Mbps	54 Mbps
Throughput medio	5,5 Mbps	36 Mbps	45 Mbps
Modulación	SS-DS	OFDM	OFDM
Disponibilidad comercial	Gran cantidad de productos disponibles	Bastantes productos disponibles	Sin previsión

Tabla 1.1 Características de los estándares WLAN más significativos

Es necesario mencionar que parte de la información transmitida en el aire es especifica de la transmisión radio (cabeceras, codificación, por ejemplo) y por tanto, no forma parte de la capacidad útil para el usuario.

Es decir, que los valores de velocidad máxima de 11 Mbps o de 54 Mbps no son equivalentes al concepto de velocidad aplicado en las redes LAN cableadas.

[2] *ETSI.* European Telecommunications Standards Institute

En la Figura 1.1 se puede observar el throughput de una WLAN, que sería equivalente al de una red cableada. Como se observa, este throughput resulta sensiblemente inferior al considerado como velocidad máxima de cada tecnología.

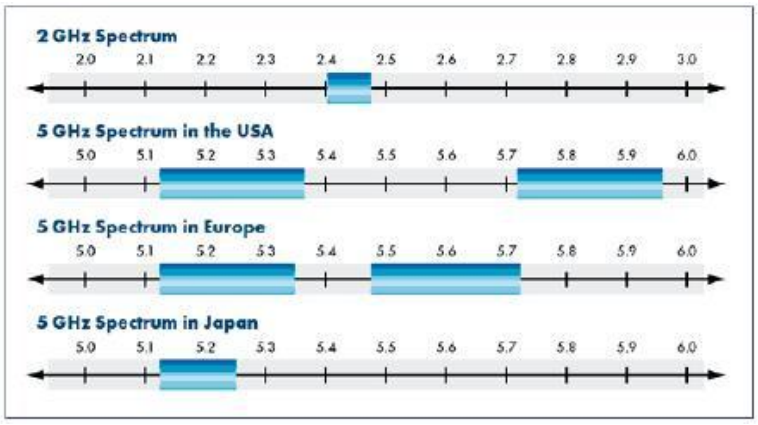

Fig. 1.1 Mapa de frecuencias WLAN

La lista de la familia de estándares más recientes promovidos por el IEEE, para redes inalámbricas es:

- IEEE 802.11g (2002): Con multiplexación OFDM permite hasta 54Mbps de capacidad máxima en la banda de 2,4 GHz. Permite interoperabilidad con el IEEE 802.11b.

- IEEE 802.11h (2003): Es una evolución del IEE 802.11a, que permite la asignación dinámica de canales y control automático de potencia para minimizar los efectos interferentes.

- IEEE 802.11i (2004): Está destinado a mejorar las prestaciones en lo que a seguridad y cifrado se refiere.

- IEEE 802.11e (2005): Diseñado para el soporte multimedia mejorado, garantizando la calidad de servicio (QoS) en comunicaciones de gran ancho de banda y tiempo real.

- IEEE 802.11n (2009): Ha sido diseñado para aumentar la capacidad efectiva de transmisión hasta los 600 Mbps teóricos en la capa física, gracias a la nueva tecnología MIMO *(Multiple Input – Multiple Output)*. Este estándar hace un uso simultáneo de las bandas 2,4 GHz y 5 GHz. Se conoce que el futuro estándar sustituto de 802.11n será IEEE 802.11ac, con tasas de transferencia superiores a 1Gbps.

- IEEE 802.11w (no concluido): Se está diseñando pensando en mejorar la capa del control de acceso al medio de IEEE 802.11, para aumentar la seguridad de los protocolos de autentificación y codificación. Las LANs

inalámbricas envía la información del sistema en tramas desprotegidos, que los hace vulnerables. Este estándar podrá proteger las redes contra la interrupción causada por los sistemas malévolos que crean peticiones desasociadas que parecen ser enviadas por el equipo válido. Se intenta extender la protección que aporta el estándar 802.11i más allá de los datos hasta las tramas de gestión, responsables de las principales operaciones de una red.

1.2. Redes ad-hoc

En la sociedad actual, a nadie se le escapa el éxito de las comunicaciones inalámbricas y de la progresiva reducción en el tamaño de los dispositivos móviles con capacidad para comunicaciones de datos.

Dentro de este tipo de redes, las más conocidas son las que cuentan con una infraestructura fija, pero se dan casos en los cuales es imposible o muy difícil disponer de dicha infraestructura.

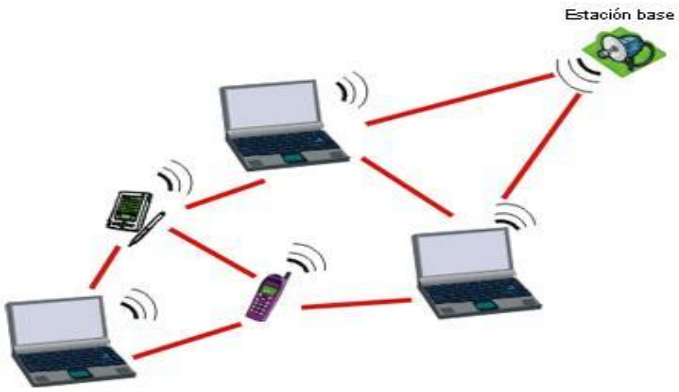

Fig. 1.2 Distintos dispositivos con capacidad computacional en un entorno inalámbrico pueden constituir una red ad-hoc

Es en este punto dónde cobran gran importancia las redes móviles ad-hoc o también llamadas MANETs *(Mobile ad-hoc NETworks)*, que definiremos en el apartado siguiente.

Las primeras redes ad-hoc comenzaron con el proyecto PRNet *(Packet Radio Network),* de la Agencia de Proyectos de Investigación Avanzada para la Defensa (Defense Advanced Research Projects Agency, *DARPA*) en 1972, el cual a su vez se inspiró en la eficiencia de la tecnología de conmutación de paquetes, con ventajas tales como la compartición del ancho de banda y el encaminamiento por almacenamiento y reenvío.

Los nodos y dispositivos de la red PRNet eran móviles, aunque tal movilidad era limitada. Esta red avanzada fue considerada buena para los años 1970.

Se puede considerar que la investigación en redes ad-hoc comenzó a partir de 1995 en una conferencia de la *IETF*[3]. Las primeras discusiones se centraron en las redes de tácticas militares, redes satelitales y *redes ponibles* (wearable networks) de computadoras, con un interés especial en que éstas (las redes ad-hoc), se desarrollasen con base en la adaptación de los protocolos existentes de soporte a lasredes IP en ambientes altamente dinámicos.

Ya para 1996 este trabajo dio nacimiento al grupo de interés en MANETs y finalmente a la creación oficial del Grupo de Trabajo sobre MANETs (MANET-WG) de la IETF en 1997, grupo que sigue activo en la actualidad.

La tarea del MANET-WG consiste en especificar interfaces y protocolos estándares para el soporte del funcionamiento de Internet basado en IP sobre redes ad hoc móviles. Es importante señalar que la mayoría de la investigación en redes ad hoc se ha realizado en el marco del MANET-WG, que actúa como el principal organismo de normalización.

1.2.1. Definición

El término “ad hoc” es una locución de origen latino que significa “para esto”, es decir, conlleva a la noción de espontaneidad, por el cual una red se crea para un propósito concreto, sin planificación o infraestructura previa.

Este tipo de redes se crean típicamente de manera dinámica y su principal singularidad es su limitación tanto temporal como espacial. Estas restricciones permiten crear y disolver redes de manera suficientemente sencilla y práctica.

Formalmente, una red ad-hoc inalámbrica se puede parametrizar considerando los siguientes calificativos:

- *Inalámbrica:* Los nodos se comunican a través de medios de transmisión no cableados (radio, infrarrojos, etc.)

- *Ad-Hoc:* La red es temporal, y se establece dinámicamente de manera arbitraria por un conjunto de nodos según se necesita.

- *Autónoma y sin infraestructura:* La red no depende de ninguna infraestructura establecida ni de ninguna administración centralizada.

[3] *IETF*: Internet Engineering Task Force, (Grupo de Tareas de Ingeniería de Internet). Es la organización que desarrolla y promueve estándares, en particular los estándares del protocolo de Internet TCP/IP.

- *Multisalto:* No necesitan encaminadores dedicados ya que cada nodo actúa como también como encaminador y reenvía paquetes hacia los otros nodos para facilitar el intercambio de información entre los integrantes de la red.

Adicionalmente, los nodos pueden estar dotados de movilidad. En este caso, las redes reciben el nombre de redes ad-hoc móviles *(MANETs)*. La topología en este tipo de redes es dinámica, debido al constante movimiento de los nodos participantes, haciendo que los patrones de comunicación entre los miembros de la red evolucione constantemente.

En definitiva, las redes ad-hoc eliminan las restricciones impuestas por las infraestructuras fijas, permitiendo a los dispositivos crear y adherirse a redes improvisadamente, haciéndolas adecuadas para adaptarse virtualmente a cualquier aplicación.

1.2.2. Aplicaciones

La principal cualidad de las redes ad-hoc es su flexibilidad. El hecho que puedan establecerse en cualquier lugar y en cualquier momento sin infraestructura, las hace muy atractivas para un amplio rango de campos de aplicación. También las podemos considerar fundamentales en escenarios de naturaleza hostil, así como en situaciones de emergencia (p.ej. desastres naturales o conflictos bélicos), ya que requieren muy poca configuración y permiten un despliegue muy rápido.

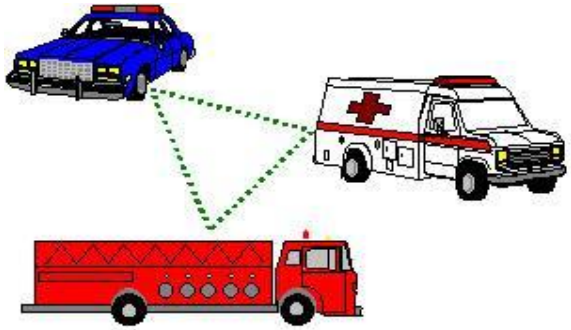

Fig. 1.3 Ejemplo de aplicación de MANETs

En la tabla siguiente se muestra una clasificación de las aplicaciones presentes y futuras de las redes ad-hoc, así como los servicios que ofrecen.

Aplicaciones	**Descripción**
Redes tácticas	Comunicaciones en operaciones militares
Redes para salvamento y emergencia	Operaciones de búsqueda y rescate Transmisión de datos y constantes vitales de los pacientes Substitución de redes con infraestructura en catástrofes naturales
Redes de sensores	Recogida de datos en tiempo real
Red urbana	Como acceso inalámbrico público en zonas urbanas, tanto para fines lúdicos de los habitantes como para servicios de localización y seguimiento relacionado con la seguridad ciudadana.
Red empresarial	Redes de área local inalámbricas *(WLAN)*para el intercambio de documentación con personal de la empresa que necesite hacer muchos desplazamientos
Redes vehiculares	Hoy en día existen diversas firmas que apuestan por la seguridad en los automóviles usando las denominadas redes *VANET*
Redes de Área Personal *(PAN)*	Para la interconexión de dispositivos inalámbricos de uso personal como PDA, móviles, ordenadores portátiles.

Tabla 1.2 Campos de aplicación de la tecnología MANET

1.2.3. Desafíos

Las redes ad-hoc, por sus características, afrontan una serie de retos que se deberían superar cumpliendo una serie de requerimientos mínimos.

Veamos a continuación en qué consisten estos inconvenientes y cómo se puede hacer frente a los desafíos existentes en el diseño e implementación de estas redes.

1.2.3.1. Enrutamiento de paquetes

La constante movilidad de los terminales supone un continuo cambio de la topología de la red e implica una constante reconfiguración de las tablas de encaminamiento de los nodos.

Posteriormente haremos énfasis en este aspecto, ya que es una característica fundamental de las MANET.

1.2.3.2. Calidad de servicio

Se ha de tener en cuenta que hay aplicaciones (como las de tiempo real en redes de sensores, por ejemplo) a las que se les debe garantizar un cierto nivel de calidad de servicio *(QoS)*. La topología dinámica modifica constantemente en las tablas de encaminamiento los nodos vecinos y el estado de los enlaces, haciendo que la provisión de QoS resulte compleja y se deba fijar unos parámetros para las diferentes líneas de estudio que existen en la actualidad. En primer lugar será necesario establecer un marco de trabajo, donde podamos identificar diferentes componentes todos ellos necesarios a la hora de proveer de calidad de servicio.

- *Modelo de QoS.* Las propuestas de QoS con mayor impacto son: Servicios Integrados *(Intserv)*, y Servicios Diferenciados *(Diffserv)*.

- *Señalización para la reserva de QoS.* Este mecanismo será el encargado de realizar la reserva y liberación de recursos en la red, así como el establecimiento de flujos. Los diversos mecanismos de señalización se dividen entre aquellos que incorporan la información de control dentro de los paquetes de datos, *in-band signalling*, y los que utilizan mensajes expresos de control, *out-of-band signalling*. Uno de los mecanismos de señalización más extendido es el protocolo RVSP, definido por el IETF, del tipo out-of-band signalling, que permite la reserva de recursos de red extremo a extremo para tráfico unicast y multicast.

- *Encaminamiento QoS.* Además del principal cometido de un protocolo de encaminamiento, que estudiaremos más exhaustivamente en próximos apartado, hay que aplicarle el carácter de calidad de servicio, con el que se debería intentar establecer una ruta que satisficiese determinados requisitos de ancho de banda, retardo, jitter, etc.

  Probablemente este sea uno de los campos en los que más trabajo se espera realizar en un presente y futuro inmediato, especialmente con la propuesta CEDAR *(Core-Extraction Distributed ad-hoc Routing)*.

  En esta, se intenta integrar el soporte de QoS mediante la definición de nodos "dominantes" a través de los cuales se hacen las computaciones de los caminos siguiendo unos parámetros que les permiten garantizar cierta calidad de servicio.

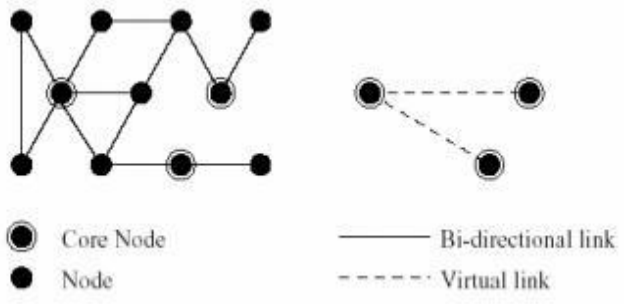

Fig. 1.4 Topología en una red CEDAR

  CEDAR propone un algoritmo para la elección de los nodos que formarán el núcleo de la red. Además, se opta por que cada nodo del

núcleo conozca el estado y la topología de los enlaces locales, así como de los enlaces más lejanos pero estables.

Finalmente establece que, cuando una fuente desee enviar tráfico a un destino, previamente tiene que mandar un mensaje indicando origen, destino, ancho de banda solicitado. Esta información se propaga por el núcleo a través de un pseudo-broadcast hasta que alcanza el destino, mientras los nodos intermedios comprueban la disponibilidad de ancho de banda en cada salto intermedio.

- *QoS en el control de acceso al medio.* Los diferentes elementos que hemos visto anteriormente, involucrados en la provisión de QoS, no tendrían ningún sentido si el control de acceso al medio *(MAC)* no nos proporcionase algún procedimiento para la reserva o priorización de recursos.

  Gran cantidad de protocolos MAC se han propuesto para entornos inalámbricos, sin embargo, la mayoría no ofrece la posibilidad de realizar una reserva de recursos.

1.2.3.3. Seguridad

Una red con cable está dotada de una seguridad inherente en cuanto a que un posible agente malicioso necesita un acceso físico al cable.

Las redes ad hoc inalámbricas, así como las redes inalámbricas en general, son vulnerables a diversos tipos de posibles ataques, ya que la transmisión de los datos es por el aire y estos podrían ser capturados “fácilmente” por algún agente o nodo malicioso.

En el contexto de las redes ad-hoc, debemos distinguir entre diferentes tipos de nodos que intentarán vulnerar la seguridad de estas redes:

- Un nodo *malicioso,* será un atacante que no puede autenticarse a sí mismo como nodo legítimo debido a la falta de información criptográfica válida.

- Un nodo *comprometido,* será un atacante interno que se comporta de forma “malintencionada”, sin embargo, ha sido autenticado por la red como nodo legítimo y obtiene la confianza de los otros nodos.

- Un nodo *selfish* (egoísta), será un nodo que tendrá tendencia a denegar la provisión de servicios en beneficio de otros nodos con el fin de conservar sus propios recursos.

1.2.3.4. Consumo de potencia

Las restricciones de potencia en algunos tipos de redes ad-hoc (las enfocadas a redes de sensores, por ejemplo), serán mucho mayores que en las redes inalámbricas convencionales, debido a que las MANET pueden operar en ambientes hostiles para el hombre, o en sitios de difícil acceso.

En ciertos casos será imposible la recarga de las baterías del nodo, por lo que será esencial el uso de protocolos muy eficientes en cuanto a la conservación de energía en los niveles de red, enlace y físico. Incluso, cuando la recarga sea una opción disponible, la vida de la red dependerá de la duración de las baterías de los dispositivos que la constituyen.

1.2.4. Arquitectura del nodo

La complejidad en las MANETs radica en que cada nodo puede actuar a su vez como emisor, receptor o nodo intermedio (enrutador).

Para la implementación exitosa de la tecnología de las redes ad-hoc, es necesario definir una pila de protocolos que se adapten a las características específicas de este tipo de redes.
.

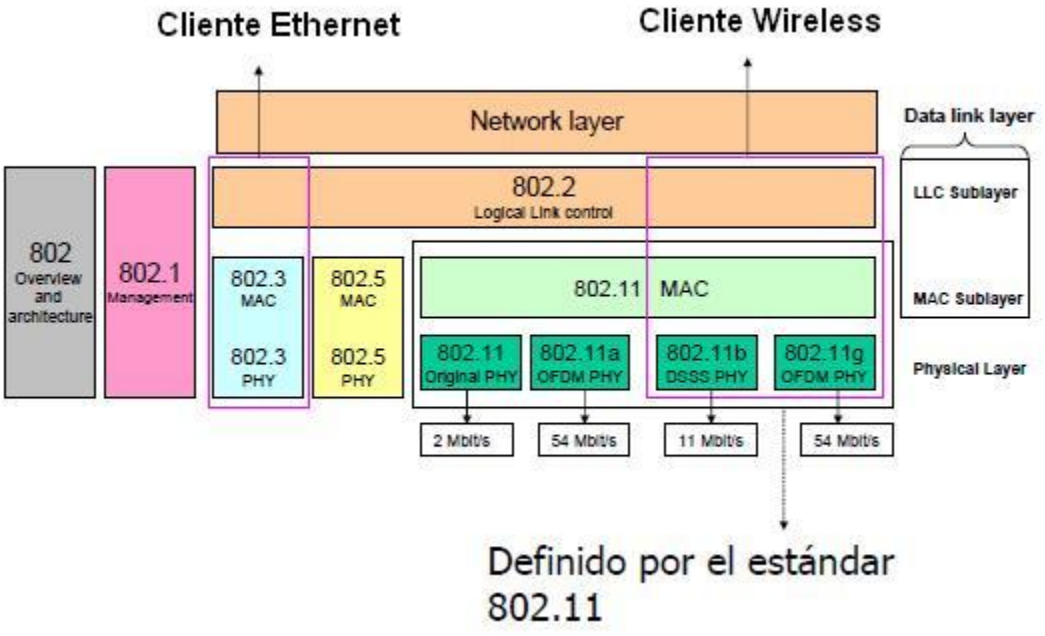

Fig. 1.5 Arquitectura de los niveles físico y enlace en 802.11

1.2.4.1. Capa física

En las redes MANET los nodos se comunican entre ellos usando tecnologías inalámbricas. Esta comunicación se realiza a través de canales de radiofrecuencia *(RF)* o a través de infrarrojos *(IR)*. Aunque realmente las redes basadas en infrarrojos se usan muy poco y sólo en aplicaciones muy concretas. Esto se debe a que los rayos infrarrojos tienen muchas desventajas respecto a los canales de radiofrecuencia, las principales serían que no pueden atravesar paredes ni otros obstáculos, y que requieren más potencia.

Las tecnologías más populares de capa física (y enlace) para construir redes ad-hoc son las basadas en IEEE 802.11.

Existen estándares de la familia IEEE 802.11 que definen interfaces para los canales de RF en las bandas de 2,4GHz y de 5GHz. La banda de 2,4GHz está más saturada ya que se utiliza para otro tipo de redes como Bluetooth, para electrodomésticos como los microondas, o para juguetes de radiocontrol. Además en ésta banda de frecuencias se permiten usos abiertos y experimentales, de manera que no hace falta tener licencias especiales para trabajar en ella, siempre que no se sobrepasen los niveles de potencia permitidos.Por lo tanto, las redes de RF que funcionan es esta banda de frecuencias están expuestas amuchas más fuentes de interferencias que las que operan en la banda de 5GHz.

Por otra parte, las señales de la banda de 2.4 GHz sufren menos pérdidas de transmisión en el espacio libre

La capa física de la especificación IEEE 802.11 ofrece dos tipos de técnicas para las transmisiones en frecuencias de radio y una especificación para transmisiones infrarrojas.

Las técnicas de radio frecuencia trabajan basadas en el concepto de “Espectro Ensanchado” o Spread Spectrum *(SS)*. Este concepto se basa en un ensanchamiento forzado del espectro de ancho de banda usando una función XOR con una secuencia pseudoaleatoria larga. Esto disminuye la densidad de potencia espectral y reduce la potencia de pico. La potencia total transmitida novaría pero la señal se hace mucho más inmune a las interferencias y al ruido.

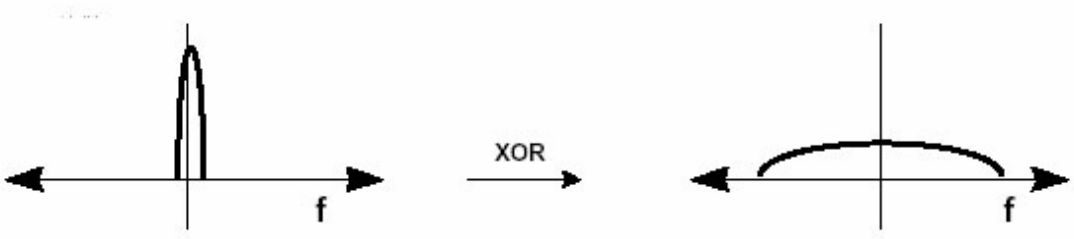

Fig. 1.6 Técnica de Spread Spectrum

Las dos técnicas especificadas en la norma 802.11 son:

- Salto de Frecuencia (Frequency Hopping Spread Spectrum, *FHSS*). Es la forma más simple de modulación de espectro ensanchado. Normalmente la mayoría de los sistemas de salto de frecuencia definen un conjunto de saltos uniformes dentro de una banda de frecuencia, aunque esto no es necesario ya que ambos extremos de la transmisión conocen de antemano el patrón de salto de frecuencias utilizado. Esta técnica consigue una alta inmunidad a interferencias y a ruido ambiente, sobre todo cuando usa patrones aleatorios de salto de frecuencia. La desventaja es que sólo se ha desarrollado para velocidades que no superen los 2Mbps. Existen 75 subcanales de 1MHz que permiten definir secuencia de saltos que no se solapan entre sí.

- Secuencia directa (Direct Sequen Spread Spectrum, *DSSS*). En ésta se usa un código de pseudo-ruido generado localmente para codificar la señal digital a transmitir. Este código se ejecuta a frecuencias varias veces más altas que la frecuencia de la señal. Si ejecutamos una función EXOR con la señal, obtenemos una señal codificada que luego será modulada usando BPSK *(Binary Phase Shift Key)* antes de ser transmitida. Esta señal, al ser recibida en el otro extremo, es decodificada usando una réplica local del código de pseudo-ruido usado en el emisor. De este modo, el receptor solo decodificará la señal que esté codificada con un código determinado, resultando en un filtro natural para las interferencias y señales adulteradas.

En cualquiera de los dos casos, las señales de Spread Spectrum se convierten en señales que tienen una baja probabilidad de interferencia con señales no deseadas.

Este tipo de modulación es exigida por la mayoría de entes reguladores para utilizar las bandas de frecuencias libres, que se encuentran entre los 2.412MHz y los 2.484MHz.

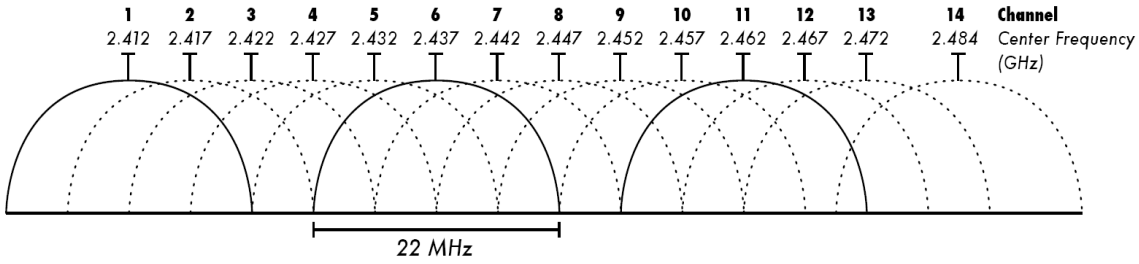

Fig. 1.7 Rango de frecuencias y canales para WLAN

En los sistemas de Secuencia Directa, es necesario compensar el ruido que se introduce en cada canal debido a su ancho de banda, para ello cada bit de datos se convierte en una serie de patrones de bits redundantes llamados “chips”. La redundancia que presenta cada chip, combinada con el

ensanchamiento de la señal a través de los 20MHz, provee un mecanismo sólido de detección y corrección de errores, minimizando las retransmisiones.

1.2.4.2. Capa de enlace

La capa de enlace en 802.11 se puede subdividir en dos subcapas:

- Control Lógico de Enlace (LLC, Logical Link Control). Es la más alta de las dos subcapas de enlace de datos definida por el IEEE, y la responsable del control de enlace lógico. La subcapa LLC maneja el control de errores, control de flujo, entramado, control de diálogo y direccionamiento de la subcapa MAC. El protocolo LLC más generalizado es el IEEE 802.2, que incluye variantes no orientado a conexión, y orientadas a conexión. En general, este estándar es común para todas las familias IEEE 802.X, es decir tanto cableadas como inalámbricas.

- Control de Acceso al Medio (MAC, Media Access Control). Existen dos familias principales de protocolos de control de acceso al medio: protocolo de acceso aleatorio (Distributed Coordination Function, DCF) y protocolo de acceso controlado (Point Coordinated Function, PCF).

  En los protocolos de *acceso aleatorio* los nodos compiten entre ellos para conseguir el canal de transmisión, mientras que en los de acceso controlado existe un dispositivo que decide cuál de los nodos puede acceder al medio en cada momento.

  De los diferentes protocolos MAC de *acceso controlado*, destacan los siguientes: FDMA (*Frequency Division Multiple Access*), TDMA (*Time Division Multiple Access*), CDMA (*Code Division Multiple Access*). Estos protocolos se aplican principalmente en redes que cuentan con una infraestructura y con un nodo maestro que se encarga de administrar el acceso al medio de transmisión por parte de los nodos de la red. En estos protocolos no existen colisiones.

  Como ya se ha explicado, las redes MANET no cuentan con una infraestructura fija por lo que se deberán usar protocolos MAC de *acceso aleatorio*.

  Concretamente, el protocolo estandarizado por el comité IEEE 802.11 fue el CSMA/CA (*Carrier Sense Multiple Access with Collision Avoidance*) debido a su inherente flexibilidad y porque resuelve el problema de los terminales ocultos a través del sencillo mecanismo de paquetes de control RTS/CTSDATA/ACK.

El estándar IEEE 802.11 define el uso de los dos niveles más bajos de la arquitectura OSI, que son la capa física y la capa de enlace, y se ha convertido en uno de los estándares más populares.

Entre las diferentes propuestas de mejora del comportamiento de los protocolos a nivel MAC para redes MANET tenemos: algoritmos para reducir el consumo de energía de los nodos móviles, que permiten que los nodos entren en un estado de bajo consumo de energía cuando éstos están ociosos; algoritmos de gestión de paquetes; reducción de los radios de transmisión y de recepción; diferentes esquemas de codificación y modulación de la señal, etc.

1.2.4.3. Capa de red

Es en esta capa es donde las redes ad-hoc se peculiarizan con respecto al resto de redes inalámbricas.

En este nivel, es necesario que los protocolos de encaminamiento para las MANET se adapten rápidamente a los cambios constantes de la topología de la red, para poder mantener una ruta para la comunicación entre los nodos, pese a que estos pueden ser móviles y su topología es cambiante.

Hay diferentes maneras de agrupar los protocolos de esta capa, pero de forma general, se pueden dividir en tres grandes tipos en función del método que utilizan para determinar las rutas:

- Protocolos reactivos
- Protocolos proactivos
- Protocolos híbridos

Además de estos tres grandes grupos, se puede incluir un cuasi-grupo que intenta dotar de QoS a los principales algoritmos de encaminamiento de los otros grupos.

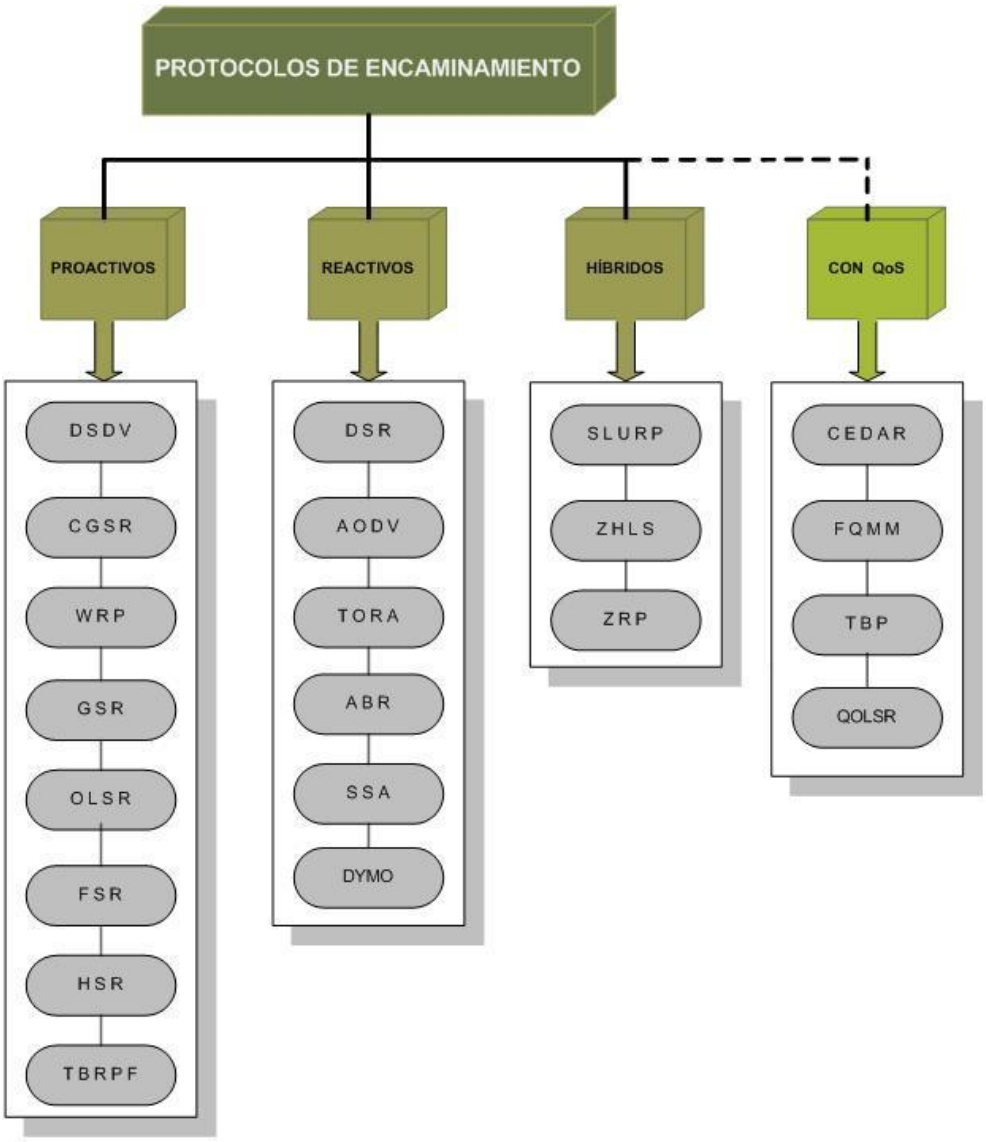

Fig. 1.8 Clasificación típica de los protocolos de encaminamiento

De los reactivos mencionar que DSR (2003) y AODV (2003) han sido publicados como RFC *(Request of Coments)* experimentales, mientras que el resto se han presentado como *draft* en el IETF. Aunque DYMO está siendo considerado por el MANET-WG para convertirse en RFC.

De igual manera, en los proactivos, TBRPF (2003) y OLSR (2003), también han sido presentados como RFC experimentales, mientras que el resto se han presentado como draft en el IETF.

Para el conjunto de los protocolos híbridos, el IETF sólo ha presentado como draft el ZRP (2002).Y finalmente, en lo que respecta a los protocolos con QoS, sólo el QOLSR (2007) ha sido presentado como draft en el IETF.

1.2.4.4. Capa de transporte

En las redes MANET se busca que sean compatibles con los protocolos utilizados por Internet, que son los más populares. Esto es porque se ve como un valor añadido de las redes MANET el que puedan interconectarse con las redes convencionales y poder ofrecer también acceso a Internet.

Por esto, las redes Ad Hoc están basadas en la pila de protocolos TCP/IP (*Transmission Control Protocol / Internet Protocol*). Es por esta razón que prácticamente todos los esfuerzos de investigación del MANET-WG están dirigidos al desarrollo de algoritmos de encaminamiento específicos para este tipo de redes, intentando que el resto de protocolos de la pila no se vean afectados. A pesar de ello, se han presentado algunas propuestas con el fin de mejorar las prestaciones de los principales protocolos de transporte de Internet, TCP y UDP (*User Datagram Protocol*), en entornos ad-hoc.

Existen propuestas de mejora de TCP para optimizar las redes Ad Hoc. Éstas utilizan mecanismos de señalización implícitos que permiten tener un TCP que no active, de manera equivocada, el mecanismo de control de congestión cuando se pierde una ruta entre el nodo fuente y el nodo destino debido a un fallo en un enlace. Un ejemplo es el TPA (*Transport Protocol for ad-hoc Networks*). El protocolo TPA incluye mecanismos que detectan cuando se pierde un enlace y se recupera la ruta, además establece mecanismos de control de congestión distintos a los de TCP.

Por otro lado, se han desarrollado otros protocolos de transporte para otras aplicaciones de tiempo real y *media-streaming* como pueden ser el SCTP (*Stream Control Transport Protocol*) y el MRTP (*Multiflow Realtime Transport Protocol*). Estos protocolos tienen como característica principal, el hecho que establecen múltiples flujos de datos entre dos nodos para una misma comunicación y funcionan junto a protocolos de encaminamiento multicamino.

1.2.4.5. Capa de aplicación y superiores

Dentro del MANET-WG uno de los objetivos principales es que las aplicaciones diseñadas para Internet se puedan soportar también en las redes MANET.

En general, las aplicaciones de transmisión de datos que no precisen de un gran consumo de ancho de banda, ni elevados requerimientos en cuanto al tiempo de entrega de los paquetes, se pueden soportar con la filosofía *Best-Effort*.

Pero hoy en día en Internet existen una gran cantidad de aplicaciones, mayoritariamente multimedia, que son mucho muy sensibles al retardo y a la variación de este retardo. Así, para funcionar correctamente necesitan un elevado ancho de banda y otras garantías.

Por todas las limitaciones y características intrínsecas propias de MANET, aún quedan muchos retos por superar antes que éstas puedan ofrecer aplicaciones multimedia de garantía.

Existen algunas propuestas a nivel de aplicación de aplicaciones multimedia para su uso en redes MANET. Estas propuestas pasan por intentar disminuir el ancho de banda, básicamente con nuevas técnicas decodificación que sacrifiquen un poco la calidad para aumentar la compresión. Aunque también hay propuestas encaminadas a modificar la calidad de la información en función del estado en el que se encuentre la red.

1.3. Encaminamiento

En esencia, el proceso de encaminamiento se define como el acto de transmisión de información desde una fuente u origen hasta un destino, esto es, extremo a extremo.

Los protocolos clásicos de ruteo no están preparados para adaptarse a escenarios tan variantes como podrían ser las MANETs.

Un protocolo de encaminamiento para redes ad-hoc permite que la red se organice completamente por sí sola, y su objetivo principal es el establecimiento correcto y eficiente de rutas entre un par de nodos, de tal modo que los mensajes sean entregados de manera fiable y a tiempo. La construcción de rutas debe producirse con una mínima sobrecarga en las cabeceras de los paquetes de control y un mínimo consumo de ancho de banda.

Las consideraciones que han de tenerse en cuenta en el diseño y estudio de los nuevos protocolos de encaminamiento incluyen:

- Simplicidad y facilidad de implementación.
- Convergencia rápida de rutas, libres de bucles y óptimas. Incluso, puede que sea posible establecer diferentes rutas entre el mismo par de nodos origen y destino, para aumentar la robustez.
- Naturaleza distribuida y ligera, de tal forma que, ante cambios en la topología y en los patrones del tráfico, la reacción del protocolo implique una mínima sobrecarga de cabeceras.
- Eficiencia en términos de ancho de banda, potencia de transmisión y computación, con una sobrecarga de cabeceras mínima.
- Escalabilidad, de tal forma que si la red aumenta significativamente en número de nodos, no se degraden las prestaciones.

- Seguridad a nivel de confidencialidad, control de acceso y denegación de ataques de servicio producidos por nodos maliciosos.

- Capacidad de soportar requerimientos de calidad de servicio *(QoS)*.La garantía de calidad de servicio es esencial para la entrega a tiempo de tráfico multimedia.

1.3.1. Clasificación de los protocolos de encaminamiento

Con estos propósitos se han desarrollado una gran cantidad de protocolos de encaminamiento que optan por diferentes criterios de diseño. Cada uno de los mismos conlleva un conjunto de características, tanto a nivel de funcionalidad como de prestaciones.

En la mayor parte de los casos, estas peculiaridades no son completamente separables, dando lugar a distintas clasificaciones dependiendo de los criterios establecidos.

El nivel de clasificación más habitual, se basa en la estructura de los protocolos en cuanto a la homogeneidad o heterogeneidad de los roles de los nodos en el encaminamiento. Así, se distingue entre:

- Uniforme o de estructura plana: todos los nodos de la red desempeñan iguales funciones y poseen las mismas características. En este caso, no se incurre en ningún coste de mantenimiento de la estructura de la red; sin embargo, se adaptan en muy poca medida a ampliaciones conservando sus mismas prestaciones.

- No uniforme: propios de estructuras jerárquicas en las que algunos nodos desarrollan papeles especiales e incluso pueden dotarse de capacidades particulares en términos de cómputo, energía o almacenamiento, entre otros. Esto les permite soportar algoritmos más complejos, reducir la sobrecarga debida a la comunicación y ofrecer la posibilidad de balanceo de carga mientras mantienen sus características incluso ante incrementos del número de nodos en la red. Por el contrario, generan cierto coste de mantenimiento de la estructura y necesitan, en muchos casos, la disponibilidad de nodos heterogéneos.

En cada una de las categorías anteriores, los protocolos presentan una nueva peculiaridad relativa al procedimiento adoptado para el descubrimiento del camino a establecer y su mantenimiento. Bajo este punto de vista, pueden diferenciarse entre:

- Proactivos: su funcionamiento se basa en tablas, creadas a partir de una fase original de descubrimiento de ruta, y que albergan la información referente a los caminos en la red obtenidos con distintos criterios. Esta información es de ámbito global y, por tanto, todos los nodos conservan caminos posibles hacia el resto. Para difundir esta información, los nodos intercambian datos bien periódicamente o bien ante la aparición

de un cambio en ella. Los protocolos proactivos logran que el envío de datos se produzca con un retardo despreciable, ya que la información sobre la ruta a seguir está disponible previamente; no obstante, consumen recursos de la red (energía, cómputo, almacenamiento, etc.), independientemente del grado de utilización de la ruta, por lo que no resulta viable para entornos móviles.

- Reactivos o bajo demanda: las rutas se construyen únicamente en el momento en que un nodo necesita establecer una comunicación. Es en ese preciso instante cuando se desencadena una fase de descubrimiento de ruta, que concluye una vez que la fuente recibe la respuesta del destino que incluye el camino elegido para el envío de datos. El coste de mantenimiento de rutas disminuye, en gran medida acosta de introducir una latencia producida por la generación inicial del camino y un posible problema de saturación de la red fruto de la inundación de la misma con mensajes de petición de ruta.

- Híbridos: generalmente utilizado para protocolos no uniformes. Incluye los dos procedimientos anteriores en distintos niveles del encaminamiento. Así, se consigue reducir la sobrecarga de la red con mensajes de control, originada por los protocolos proactivos, mientras que se reduce la latencia de las operaciones de búsqueda (latencia presente en los protocolos reactivos).

1.3.2. Eficiencia de encaminamiento

Medir las prestaciones de un protocolo de encaminamiento para redes ad-hoc no es una tarea trivial, puesto que es necesario seleccionar aquellas variables que evalúan cualitativa y cuantitativamente sus virtudes. No obstante, el Grupo de Trabajo en Ingeniería de Internet *(IETF)* aporta una lista de métricas cuantitativas que pueden ser empleadas para valorar las prestaciones de cualquier protocolo de encaminamiento:

- Throughput y retardo. Esta métrica se basa en medir el throughput medido en recepción y/o el retardo que experimentan los datos desde el origen hasta su destino.

- Tiempo de adquisición de rutas. Esta medida calcula el tiempo necesario para establecer una ruta desde que se solicita, por lo que sólo tiene sentido en protocolos en los que el procedimiento de descubrimiento de rutas se realiza bajo demanda.

- Porcentaje de entregas desordenadas. Se trata de otra medida externa que mide el rendimiento del encaminamiento cuando algunas de las conexiones no están operativas. Es especialmente relevante desde el punto de vista de la capa de transporte.

A su vez, para evaluar estas prestaciones, no ha de olvidarse el contexto de red en el que trabaja el protocolo de encaminamiento bajo estudio.

Los parámetros esenciales que definen un contexto de red incluyen:

- Tamaño de la red. Medido en número de nodos.
- Conectividad de la red. Grado medio de un nodo.
- Tasa de cambio en la topología. Velocidad a la que varía la topología de la red.
- Capacidad de los enlaces. Velocidad efectiva del enlace medida en bits/segundo.
- Fracción de enlaces unidireccionales. Cómo responde el protocolo cuando un porcentaje de los enlaces trabaja solamente en una dirección.
- Patrones de tráfico. Cómo se adapta el protocolo cuando el patrón no es uniforme.
- Movilidad. Cuándo y bajo qué circunstancias la correlación temporal y espacial afecta al rendimiento de un protocolo de encaminamiento.
- Porcentaje y frecuencia de nodos no operativos. Cómo responde el protocolo en presencia de nodos que dejan de estar activos o vuelven a estarlo tras un periodo de inactividad.

1.3.3. Principales algoritmos de encaminamiento

A continuación se pasará a describir los principales algoritmos de encaminamiento para redes ad-hoc, obviando los dos protocolos con las que se harán las pruebas experimentales, pues en el siguiente capítulo se analizarán con mayor detalle.

- DSDV (Destination Sequenced Distance Vector): Este protocolo, basado en tablas de encaminamiento, fue diseñado por Charles E. Perkins y Pravin Bhagwat. Cada nodo de la red mantiene una tabla de encaminamiento que contiene todos los posibles destinos y el número de saltos que daría un paquete que viajara hacia el destino especificado. Cada entrada de esta tabla de encaminamiento, posee un número de secuencia asignado por el nodo destino. Los números de secuencia permiten distinguir rutas antiguas de rutas más recientes.
  Continuamente se deben enviar mensajes de actualización a través de la red para mantener la consistencia de las tablas. Las nuevas rutas contienen la dirección de destino, el número de saltos requeridos para alcanzar al destino, el número de secuencia asociado al destino y un nuevo número que identifica todo el mensaje. En caso de que haya dos rutas distintas

hacia un destino, se usará la que contenga el número de secuencia más reciente, y si ambos números coincidieran, la ruta con menor número de saltos sería la que se usaría.

- WRP (Wireless Routing Protocol): Fue creado por S. Murthy y J.J. García-Luna. Es un protocolo basado en tablas cuyo objetivo principal es mantener información actualizada de todos los nodos de la red. Cada nodo es responsable de mantener cuatro tablas:
  - Tabla de distancias.
  - Tabla de encaminamiento.
  - Tabla de coste de ruta.
  - Tabla con la lista de mensajes retransmitidos (MRL, Message Retransmission List).

  La MRL se utiliza para gestionar el envío de los paquetes de actualización de rutas. Cada entrada de la MRL contiene el número de secuencia que identifica el paquete de actualización de rutas, un contador de retransmisiones, un vector de asentimientos con una entrada por vecino y una lista de las unidades enviadas en el paquete de actualización (en ocasiones, el paquete se trocea en unidades más pequeñas). Esta MRL almacena qué unidades deben ser retransmitidas y qué vecinos deben asentir todavía los envíos.
  Los nodos se informan entre ellos de los cambios en las rutas a través de los paquetes de actualización. Estos paquetes son enviados sólo entre vecinos y contienen los elementos a actualizar en las rutas. Los nodos envían estos paquetes cuando procesan las actualizaciones recibidas de otros vecinos o cuando ellos mismos detectan un cambio en el enlace con algún vecino.
  Los nodos mantienen activo el enlace con los vecinos, siempre y cuando reciban de ellos asentimientos u otros mensajes. Si un nodo no está enviando mensajes, debería enviar un paquete HELLO cada cierto tiempo a sus vecinos, para que éstos no creyeran que el nodo se había vuelto inalcanzable. Por consiguiente, la omisión de mensajes por parte de un nodo ocasionará la ruptura de ese enlace. Cuando un nodo recibe un mensaje HELLO de un nuevo nodo, este nodo será añadido a la tabla de encaminamiento y una copia de esta tabla será enviada al nuevo nodo.

- DSR (Dynamic Source Routing): Creado por David B. Johnson y Josh Broch. Se trata de un algoritmo basado en el concepto de encaminamiento en origen. Los nodos mantienen *cachés*, cuyas entradas incluyen el destino y la lista de nodos para llegar a él. Las entradas de esta tabla son actualizadas según sea prendan rutas nuevas.
  El protocolo consta de dos mecanismos principales: descubrimiento de ruta y mantenimiento de ruta. Cuando un nodo quiere enviar un paquete a un destino, primero consulta su caché para determinar si dispone de una ruta hacia el destino. Si tiene una ruta válida, la usará para enviar el paquete. Sin embargo, si el nodo no dispone de dicha ruta, iniciará un descubrimiento de ruta enviando un paquete RREQ (Route REQuest). Este paquete contiene la dirección de destino buscada, la dirección del nodo que origina el envío y un identificador único. Cada nodo que reciba el paquete verificará si posee una ruta hacia el destino. Si no la tiene, añadirá su propia dirección

en el registro de rutas del paquete y después reenviará el paquete a través de todos sus enlaces. Para limitar la propagación excesiva de mensajes de descubrimiento de ruta, un nodo solo reenviará este mensaje si la misma petición no fue recibida con anterioridad.
Cuando un paquete RREQ alcanza su destino final, este nodo genera un paquete de respuesta de ruta (RREP). También podría contestar con un RREP un nodo intermedio que tuviera en su caché una ruta válida hacia el destino del RREQ. Si el nodo que genera la respuesta es el destino, colocará el registro de rutas contenido en el RREQ dentro del RREP. Si es un nodo intermedio el que responde, extraerá de su caché la ruta para llegar al destino, que unida al registro de rutas contenido en el RREQ, compondrá la ruta a introducir en el RREP.
El mantenimiento de rutas se completa con el uso de paquetes de error en ruta (RERR) y asentimientos. Los paquetes de error en ruta son iniciados por un nodo cuando encuentra un problema en la transmisión con algún enlace. Cuando un RERR es recibido, el nodo que provocó el error es eliminado de la caché de rutas.
También serán borradas todas las rutas en las que intervenga el enlace roto.

- ZRP (Zone Routing Protocol): Creado Nicklas Beijar. Es un protocolo híbrido a medio camino entre los algoritmos basados en tablas y los basados en encaminamiento bajo demanda. Es utilizado en una clase particular de redes ad-hoc llamadas RWNs (Reconfigurable Wireless Networks). Estas redes se caracterizan por tener gran cantidad de nodos, mucha movilidad y alto tráfico. Los protocolos anteriores no satisfacían las necesidades específicas de estas redes y los autores se decidieron a crear un nuevo protocolo. ZRP usa zonas similares a *clusters*, en las que los nodos que actúan de bordes se van seleccionando dinámicamente. Además, el radio de estas zonas se reajusta sobre la marcha según las condiciones de la red. Se pueden usar protocolos distintos para comunicarse dentro de las zonas y entre zonas distintas.

- TORA (Temporally Ordered Routing Algorithm): Creado por M. Scott Corson y Vincent Park. Es un algoritmo que se basa en mantener un grafo dirigido y sin ciclos para llegar al destino. El objetivo es minimizar la carga sobre la red. La diferencia con otros algoritmos es la imposibilidad de estimar constantemente la distancia hacia el destino o de mantener siempre la ruta más corta. Sin embargo, tiene la ventaja de que es un algoritmo muy eficiente pues no satura en exceso de tráfico la red.

1.4. Movilidad en redes ad-hoc[4]

El estudio de las redes ad hoc exige la definición de modelos que describan la topología inicial y la movilidad de los nodos que conforman la red. Esta

[4] En este proyecto, se considera la movilidad provocando cambios en la topología, haciendo caídas de nodos, pero no se contempla la movilidad de los nodos propiamente dicha.

movilidad es, en principio, mucho más crítica que en otras tecnologías de comunicaciones móviles, ya que las redes móviles e inalámbricas convencionales (entre ellas la telefonía móvil celular y las redes de área local inalámbricas), disponen de una infraestructura física preestablecida y fija.

Mientras que las redes ad-hoc carecen de infraestructura fija preestablecida a la que conectarse, lo que requerirá una definición exhaustiva de la topología de red para optimizar el proceso de encaminamiento.

Dado el gran número de elementos que intervienen en una MANET, el mecanismo de estudio y análisis en este campo pasa a veces obligatoriamente por la simulación.

En este campo, la mayor parte de los estudios emplean masivamente el modelo de movilidad denominado Random Way Point *(RWP)*.

CAPÍTULO 2. PROTOCOLOS DE ENCAMINAMIENTO

Hasta el momento, se ha explicado las MANETs en el conjunto de las redes inalámbricas, se han descrito sus principales características, los desafíos que tienen por delante, y la clasificación de las diferentes redes ad-hoc.
En este capítulo el objetivo es profundizar en los protocolos que posteriormente se implantaran en los diferentes escenarios de las pruebas de laboratorio, tratando consideraciones teóricas pero ejemplificadas con algunas capturas reales extraídas del banco de pruebas.

2.1. OLSR

OLSR *(Optimized Link State Routing Protocol)*, es un protocolo concebido como una adaptación de protocolos de redes no inalámbricas, más concretamente del RIP y del OSPF.

Se estandariza en el RFC 3626, y se clasifica dentro del conjunto de protocolos proactivos, es decir, cada nodo intercambia información regularmente con otros nodos de la red. Por tanto un nodo no solo requiere intercambiar información, si no también almacenarla, para así poder compartir la información que posee de la red con otros nodos de la misma.

2.1.1. Introducción

Fue creado por Thomas Clausen y Philippe Jacquet en el proyecto Hipercom INRIA. El tipo de enrutamiento es *hop-by-hop,* es decir, cuando un nodo recibe un paquete usa la información local que tiene para reenviarlo a otros nodos. Esta información se almacena en distintas tablas con distintas direcciones y datos útiles que la red utiliza para el enrutamiento.

Este protocolo genera transmisiones adicionales en caso de caída o de movimiento de un nodo.

Los paquetes se transmiten por la red, encapsulados en paquetes UDP, utilizando el puerto 698 asignado por la IANA.

2.1.2. Funcionamiento

Al ser un protocolo proactivo, tiene que mantener información del estado de la red. Para ello es necesario que cada cierto tiempo mande paquetes de control, bien para detectar los vecinos de alrededor de los nodos, o bien para compartir información con ellos. Pero esto conlleva a un problema o duda, y es que se

necesita configurar adecuadamente el tiempo de envío entre estos paquetes de control.

Cada nodo selecciona de sus vecinos, un conjunto de nodos que se denominan Multi Point Relays *(MPR),* que sean capaces de llegar a los nodos vecinos que se encuentren a dos nodos saltos de distancia. Estos nodos MBR, son los encargados de retransmitir y controlar el tráfico de la red de sus vecinos al resto de la red.

Un ejemplo es la Figura 2.1, donde todos los nodos a dos saltos de A, pueden ser alcanzados a través de B y D. Por tanto son elegidos nodos MPR por A. Cuando el nodo A envíe un mensaje de broadcast, solamente los nodos B y D reenviarán dicho mensaje a sus vecinos.

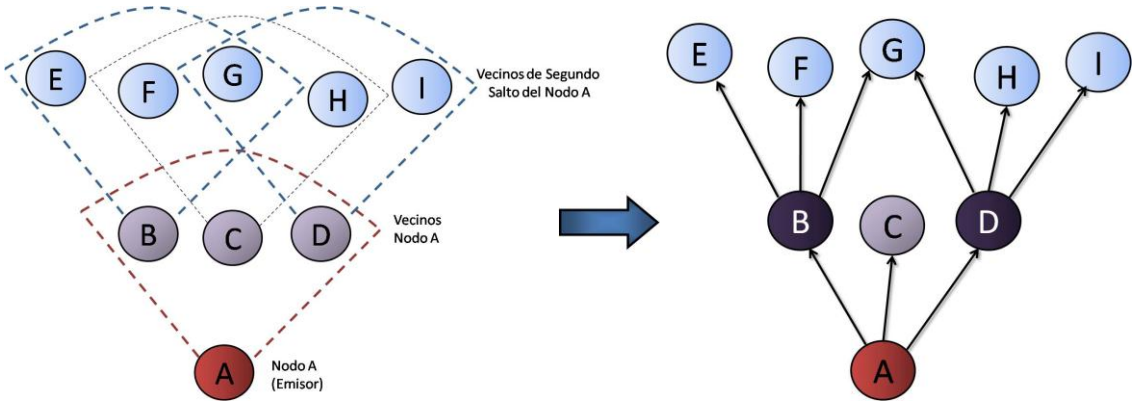

Fig. 2.1 Ejemplo de la selección del nodo MPR

Los MPR deben de ser nodos bidireccionales. Gracias a este sistema de enrutamiento, se reduce el número de retransmisiones requeridas para enviar (“inundar”) un mensaje en toda la red, esto reduce la sobrecarga de la red, también el ancho de banda consumido, y las colisiones entre los paquetes.

En la Figura 2.2, se puede apreciar un ejemplo de la transacción de mensajes desde un nodo fuente, mediante MPRs (en oscuro), y sin MPRs.

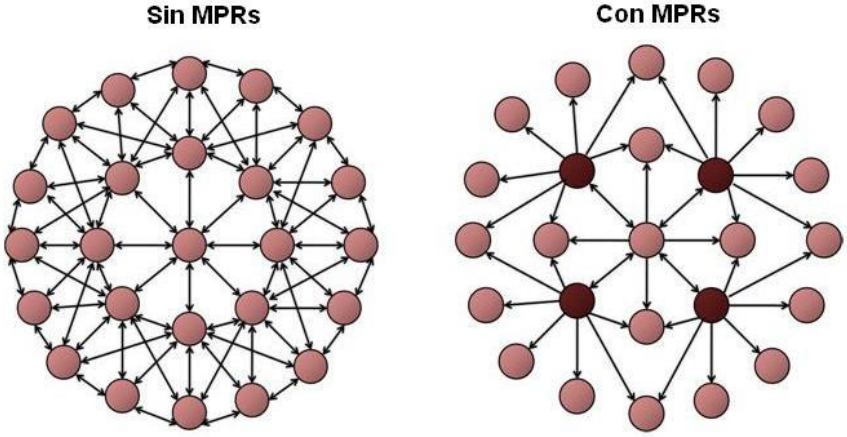

Fig. 2.2 Inundación de una MANET

Los únicos nodos capaces de realizar broadcast de paquetes al resto de la red son los MPR. Si un MPR que ha recibido un paquete, envía ese paquete a un nodo no MPR significa que está mandando el paquete al nodo destino.

No es obligatorio ningún número específico de MPRs. Realmente, el único requerimiento es que cada MPR retransmita a sus vecinos el estado del resto de vecinos alcanzables a través de dicho MPR, y que la información de control generada por un nodo alcance a todos sus vecinos a dos saltos. Periódicamente, los MPR intercambian información sobre los enlaces con sus nodos vecinos mediante el envío de mensajes de control de topología (Topology Control, *TC*). Cada nodo envía, también de forma periódica, mensajes HELLO a sus nodos vecinos para comprobar la existencia de enlace entre ellos. Este proceso lo veremos con más detalle en el siguiente apartado.

Finalmente, en los paquetes de datos generalmente se añade un número de secuencia que permite que los paquetes se reciban con desorden. Esto a priori, puede parecer poco importante, pero otros protocolos requieren que los paquetes lleguen en orden, y en caso de fallo de un paquete puede producir un retraso considerable ya que descartarían los paquetes recibidos con un número de secuencia superior al del que ha fallado. Pero todo estas explicaciones se pueden ver más claramente en el control de paquetes *ARQ*[5] y sus distintos modos (con espera, continuo selectivo, etc.) y tan solo resaltar que es útil, porque siendo más probable la colisión en un medio compartido y siendo más lento es mejor reenviar solo el paquete fallado y aceptar los paquetes en orden que cualquier otra solución que signifique perder más tiempo en los envíos.

2.1.2.1. Formato del paquete

Para tener el mayor control posible y mayor reacción ante cambios en la red, se utilizan diferentes mensajes de control que se envían para que cada nodo pueda confeccionar su tabla de enrutado para el sistema y las tablas del propio protocolo OLSR.

Cuando un nodo recibe un paquete de encaminamiento OLSR determinará el procesamiento que debe seguir basándose en sus campos.

A continuación se describen los campos más importantes de un paquete OLSR:

- Cabecera del paquete
    - Tamaño del paquete (tamaño en bytes del paquete).

[5] *ARQ. Es un protocolo utilizado para el control de errores en la transmisión de datos.*

- Numero de secuencia del paquete (PSN). Debe ser incrementado en 1 cada vez que se transmite un nuevo paquete OLSR.

- Cabecera del mensaje

  - Message Type. Indica el tipo de mensaje que se encuentra en la parte reservada para “MESSAGE” (campo de datos del paquete). En caso de encontrarse vacío lo descarta.

  - Vtime. Indica durante cuánto tiempo debe considerar la información que lleva el mensaje como válida.

  - Message Size. Tamaño en bytes del mensaje, medido desde el campo tipo de mensaje del mensaje actual, hasta el mismo campo del siguiente mensaje.

  - Time To Live. Este campo contiene el número máximo de saltos que puede realizar un paquete. Sirve para no tener paquetes perdidos por la red y al mismo tiempo descongestionarla. Cuando se recibe un paquete con valor de TTL igual a 0, se elimina de la red. Si por el contrario el TTL es mayor a 0 se decrementa su valor en una unidad y se retransmite nuevamente.

  - Hop Count. En este campo se anota el número de saltos que ha realizado el paquete. Todos los nodos cuando lo recibe incrementan su valor en una unidad. Esto permite optimizar los recursos de la red evitando utilizar los caminos más largos entre origen y destino.

  - Address. Dirección del nodo que originó el mensaje.

  - Sequence Number. Cuando se crea un mensaje, se le asigna un número de identificación. Gracias a este número se puede saber si el mensaje de ha transmitido con anterioridad y así poder evitar retransmisiones.

2.1.2.2. Procesamiento

Cuando un nodo recibe un paquete, examina la cabecera del mensaje. Basándose en el valor del campo tipo de mensaje, puede determinar el destino de dicho mensaje.

Un nodo puede recibir el mismo mensaje más de una vez, pero debe evitar el reprocesamiento de cualquier mensaje. Para ello, cada nodo mantiene un conjunto de duplicados. Este conjunto almacena información en forma de *tuplas* sobre los mensajes recibidos más recientes. Esta tupla cuenta consta de cinco campos:

- D_addr: dirección del nodo que envió el mensaje.
- D_seq_num: número de secuencia del mensaje.
- D_iface_list: lista de direcciones a las que se retransmitió el mensaje.
- D_retransmited: valor booleano que determina si el mensaje fue retransmitido o no.

  D_time: Tiempo de expiración de la tupla, es decir, tiempo durante el que la información de la tupla es válida. Cuando este tiempo se agota, se debe eliminar la tupla del conjunto de duplicados.

Por cada paquete recibido, un nodo debe realizar los siguientes pasos:

- Si no contiene mensaje, lo descarta (tamaño del mensaje ≤ cabecera)
- Eliminar el mensaje si TTL ≤ 0. El mensaje no puede dar más saltos por la red.
- Cumplir condición de procesamiento:
  - Si existe una entrada en el conjunto de duplicados, donde D_addr = Dirección del nodo que creó el mensaje y D_seq_num = número de secuencia del mensaje que se procesa, entonces se supone que el mensaje ya ha sido procesado y no debe procesarse de nuevo.
  - Si el nodo es capaz de implementar lo que pide el campo Message Type, éste se procesa.
- Cumplir condición de reenvio:
  - Si existe una entrada en el conjunto de duplicados, donde D_addr = Dirección del nodo que creó el mensaje y D_seq_num = número de secuencia del mensaje que se procesa, no reenviar.
  - Si el nodo puede procesar el tipo de mensaje, el mensaje debe ser considerado para ser reenviado.
  - Si el nodo no puede procesar el mensaje, el mensaje debería ser considerado para ser reenviado.

2.1.2.3. Algoritmo de reenvío

- Si no se puede detectar al emisor del mensaje a sólo un salto del nodo fuente del reenvío, el algoritmo se detiene y el mensaje no es reenviado.

- Si en el conjunto de duplicados del nodo que quiere reenviar el mensaje existe una entrada donde D_addr=Dirección del nodo que creó el mensaje, y D_seq_num =número de secuencia del mensaje, entonces el mensaje sólo será considerado para el reenvío si en esa misma tupla, D_retransmited es falso y la dirección del nodo que recibirá el mensaje no está entre las direcciones de D_iface_list.

- En otro caso, si no existen entradas en el conjunto de duplicados como la descrita anteriormente, el mensaje se podrá reenviar.

Si después de estos pasos, el mensaje no puede ser reenviado, el algoritmo habrá parado. Si no, el protocolo actúa de ésta forma:

- Si la dirección del emisor, es la dirección de un selector MPR, y el TTL del mensaje es mayor o igual que 1, el mensaje debe ser transmitido.

- Si en el conjunto de duplicados, existe una entrada con la misma dirección del nodo que originó el mensaje, y el mismo número de secuencia del mensaje, la entrada en el conjunto se actualiza como sigue:
  - D_time: tiempo actual.
  - La dirección del receptor se añade a D_iface_list
  - D_retransmitted toma valor True sí el mensaje se retransmite cumpliendo quela dirección del emisor sea la dirección de un selector MPR, y que el TTL del mensaje sea mayor o igual que 1.
  - De acuerdo con esto último, el mensaje debe ser retransmitido así:
    1) El TTL se decrementa.
    2) La cuenta de los saltos del mensaje se incrementa.
    3) Se hace un broadcast del mensaje.

2.1.3. Mensajes

2.1.3.1. Mensajes HELLO

Este mensaje es el que necesitan los nodos para obtener información local sobre el estado de los enlaces. Los HELLO se transmiten solamente a un salto ya que tiene el campo TTL es igual a 1, evitando su retransmisión más allá de un solo salto desde el nodo que genera el HELLO.

Se transmiten periódicamente según el tiempo configurado en cada nodo. La emisión es a la dirección broadcast de la red.

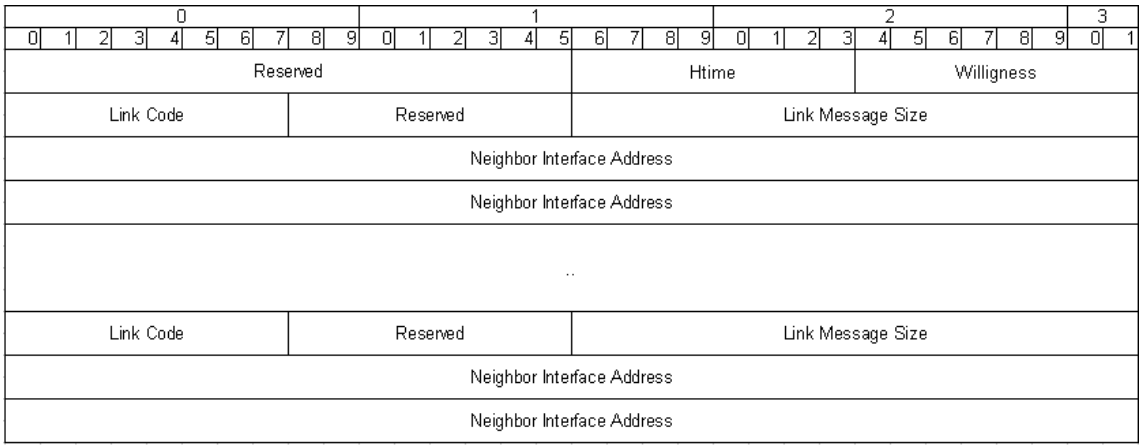

Fig. 2.3 Formato del paquete HELLO de OLSR

Los mensajes HELLO son enviados periódicamente por cada nodo de la red a sus nodos vecinos. Estos mensajes contienen la lista de vecinos conocidos por el nodo emisor así como la identidad de los *MPR* seleccionados por transmisor. Su intercambio permite a cada nodo de la red conocer los nodos situados a uno y dos saltos de distancia (es decir, aquellos a los que se puede hacer llegar un mensaje con una transmisión directa o con una transmisión y una retransmisión), y saber si ha sido seleccionado como MPR por alguno de sus vecinos.

En la Figura 2.4, podemos ver un ejemplo del mensaje HELLO capturado en el escenario de la parte experimental.

```
Optimized Link State Routing Protocol
    Packet Length: 20
    Packet Sequence Number: 5882
    Message: HELLO (LQ, olsr.org) (201)
        Message Type: HELLO (LQ, olsr.org) (201)
        Validity Time: 12,000 (in seconds)
        Message: 16
        Originator Address: 192.168.1.11 (192.168.1.11)
        TTL: 1
        Hop Count: 0
        Message Sequence Number: 8997
        Hello Emission Interval: 4,000 (in seconds)
        Willingness to forward messages: always (7)
```

Fig. 2.4 Captura de un mensaje HELLO

2.1.3.2. Mensajes TC

Son generados por todos los nodos, pero los mensajes TC *(Topology Change)* solamente son reenviados por los nodos designados como MPR.

Para que esto sea posible, el campo TTL es igual a 255, y se irá restando una unidad por cada nodo que lo reenvía. Contienen información de qué vecinos tiene el emisor de ese mensaje, y la calidad del enlace correspondiente a cada uno.

En la Figura 2.5 se muestra el formato de un mensaje TC.

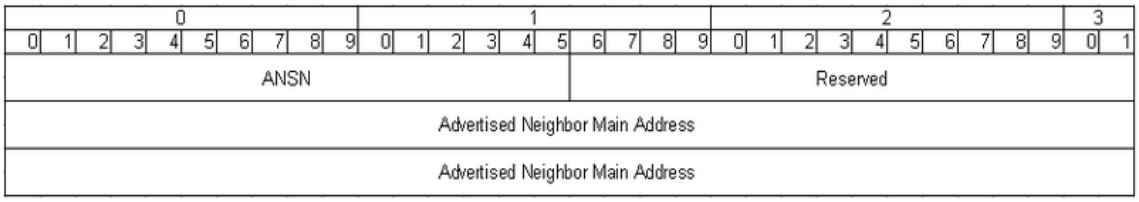

Fig. 2.5 Formato del mensaje TC

Los parámetros que se distinguen son:

- Número de secuencia de un vecino anunciado. (ANSN). Se asocia un número de secuencia a cada conjunto de vecinos anunciados. Cuándo el conjunto de vecinos anunciados de un nodo varía, incrementa su ANSN. Este valor se envía en el campo ANSN del formato de paquete TC, para seguir la pista de la información más reciente. Cuando un nodo recibe un mensaje TC, puede decidir, basándose en su ANSN si la información recibida en el mensaje TC es o no más reciente que la que él tenía.

- Dirección principal de un vecino anunciado. Este campo contiene la dirección principal de un nodo vecino. En los mensajes TC se deben incluir todas las direcciones de nodos vecinos anunciados. Si la longitud máxima de un mensaje (impuesta por la red) no lo permite, se mandarán tantos mensajes TC como sean necesarios para, hasta que se envíen todas las direcciones de nodos vecinos.

- Reservado: Este campo es reservado y siempre tiene valor “0000000000000000”.

En la Fig. 2.6 se muestra la captura de un mensaje TC de OLSR, de la parte experimental de este proyecto.

```
Optimized Link State Routing Protocol
    Packet Length: 48
    Packet Sequence Number: 5887
    Message: TC (LQ, olsr.org) (202)
        Message Type: TC (LQ, olsr.org) (202)
        Validity Time: 30,000 (in seconds)
        Message: 24
        Originator Address: 192.168.1.11 (192.168.1.11)
        TTL: 255
        Hop Count: 0
        Message Sequence Number: 9002
        Advertised Neighbor Sequence Number (ANSN): 2
        Neighbor Address: 192.168.1.12 (112/112)
            Neighbor Address: 192.168.1.12 (192.168.1.12)
            LQ: 112
NLQ: 112
```

Fig. 2.6 Captura de un mensaje TC

Aquí, además de las direcciones del vecino, y el número de secuencia del vecino anunciado, se observan los parámetros LQ y NLQ que posteriormente servirán para el cálculo del ETX (fórmula 2.4), para definir la calidad del enlace.

En general, si se establece un intervalo pequeño de envíos se generan muchos paquetes y por tanto aumenta la sobrecarga y las colisiones de la red. Sin embargo, si el tiempo es excesivamente grande puede que la información contenida en cada nodo en el momento de retransmisión de un paquete sea obsoleta. Sería óptimo que este tiempo fuera ajustable en tiempo real, o incluso adaptativo a las características de la red.

Por ejemplo en una red con alta movilidad será mejor un tiempo entre envíos de mensajes de control bajo para tener la máxima información en todo momento.

2.1.4. Calidad del enlace

A partir de la implementación (del demonio de OLSR) 0.4.8, se añadió un sistema para poder hacer distinciones de la calidad de los enlaces. Este sistema se basa en la medición de los mensajes de HELLO perdidos y recibidos, que un nodo recibe constantemente de sus vecinos.

El protocolo, gracias a la información contenida en cada nodo es capaz de mandar un paquete de un nodo a otro por el camino más corto. Ahora bien, puede darse el caso de que existan dos caminos alternativos desde el mismo origen al mismo destino, uno de ellos pasando por un nodo o enlace defectuoso, y la otra alternativa pasando por un nodo en excelentes condiciones.

Por nodo o enlace defectuoso entendemos un nodo o un enlace que conlleva una alta probabilidad de pérdidas de paquetes. OLSR está implementado de

forma que distingue los nodos defectuosos de los nodos en buenas condiciones.

Para ello definimos el concepto de calidad del enlace (Link Quality*, LQ*), es la probabilidad de paquetes recibidos de ese nodo. Si perdemos 2 mensajes de 10 (20%) que nos han enviado, nuestra *LQ* será el tanto por ciento que si ha llegado 8 de 10 (80%). De esta manera determinamos la probabilidad que tenemos de enviar un paquete con éxito hacia ese nodo.

$$\mathrm{LQ} = {}^{\text{Nº paquetes recibidos}}/_{\text{Nº paquetes totales}} \quad \textbf{(2.1)}$$

Aun así, también es importante tener en cuenta el porcentaje de paquetes enviados por un nodo que han llegado bien a su destino. Para ello, un nodo lleva la cuenta de cuantos paquetes de control ha enviado y cuantas confirmaciones ha recibido. Definimos este porcentaje como Neighboor Link Quality *(NLQ).*

$$\mathrm{NLQ} = {}^{\text{Nº ACK recibidos}}/_{\text{Nº paquetes enviados}} \quad \textbf{(2.2)}$$

Este último concepto es algo menos estricto que el anterior, ya que no recibir una confirmación de un paquete no significa haber perdido dicho paquete, si no que se podría haber perdido la propia confirmación. Por tanto, para definir la calidad de un enlace, OLSR usa una aproximación en base a los dos conceptos anteriores:

$$\text{Calidad} = \mathrm{LQ} \times \mathrm{NLQ} \quad \textbf{(2.3)}$$

Cuanto más cerca de uno (o 100), menor será la probabilidad de pérdida de un paquete por parte de un nodo y, por tanto, más fiable será la retransmisión de paquetes a través de dicho nodo.

Otro parámetro que relaciona los anteriores es el ETX. Éste se define como el número esperado de transmisiones (y retransmisiones) requeridas para entregar con éxito un paquete a lo largo de un enlace. El rango de este parámetro se encuentra entre 1 e infinito, siendo $ETX = 1$ una transmisión perfecta, y $ETX = \infty$ una transmisión nula.

Su valor se obtiene:

$$\mathrm{ETX} = {}^{1}/_{(\mathrm{LQ} \times \mathrm{NLQ})} \quad \textbf{(2.4)}$$

2.1.5. Tabla OLSR

A continuación se aprecia una captura de la tabla OLSR una vez establecida la conexión con sus vecinos.

```
---    23:02:44.045889    ------------------------------------------------
LINKS
IP address       hyst         LQ       ETX
192.168.1.12     0.000  0.965/0.965    1.074

--- 23:02:44.045930 ---------------------- TWO-HOP NEIGHBORS
IP addr (2-hop)  IPaddr (1-hop)  Total cost
192.168.1.13     192.168.1.12     2.394

---     23:02:44.045954     ------------------------------------------------
TOPOLOGY
Source IP addrDest IP addr          LQ        ETX
192.168.1.11    192.168.1.12    0.965/0.965    1.074
192.168.1.12    192.168.1.11    0.929/0.906    1.188
192.168.1.12    192.168.1.13    0.765/0.765    1.710
192.168.1.13    192.168.1.12    0.824/0.765    1.588
```

Fig. 2.7 Captura de la tabla OLSR con único camino para llegar a 2 saltos

La tabla OLSR consta de varias partes bien diferenciadas:

- LINKS. Muestra el coste que tienen los links directos con sus vecinos.
- TWO_HOP NEIGHBORS. Muestra el coste total hasta los nodos a 2 saltos de diferencia. El parámetro *Total cost* es la suma de los costes ETX de cada enlace hasta el nodo final.
- TOPOLOGY. Muestra una tabla desde la cual se puede reproducir la topología actual de la red y conocer el coste de cada enlace.

2.2. AODV

El protocolo de encaminamiento AODV (*Ad-hoc On-demand Distance Vector),* es el protocolo desarrollado en 1999 por Charles E. Perkins, del Grupo de Desarrollo Avanzado (Sun Microsystems), y Elizabeth M. Royer, del departamento de Ingeniería Electrónica y Computacional de la Universidad de California.

Su propósito fue diseñar un protocolo para redes ad-hoc formadas por nodos móviles, tomando como punto de partida el protocolo DSDV, pero solventando sus mayores deficiencias: el alto número de envíos en modo broadcast, y la latencia en la transmisión.

EL documento oficial vigente, que describe las especificaciones del protocolo AODV se especifica en el RFC3651, publicado en 2004 con la categoría de experimental.

2.2.1. Introducción

AODV es un protocolo reactivo, ya que el proceso de búsqueda de rutas se inicia sólo cuando un nodo necesita enviar información a otro nodo, y desconoce como acceder a él. AODV está basado en la familia de algoritmos de vector de distancias y puede transmitir en modo unicast y multicast.

El protocolo AODV combina técnicas extraídas de los protocolos DSDV y DSR, dando lugar a un algoritmo que usa el ancho de banda de manera eficiente y que responde con rapidez a los cambios en la red al tiempo que garantiza la ausencia de bucles.

Con el fin de mantener sólo la información de ruteo más reciente, el protocolo AODV toma de su predecesor, el concepto de número de secuencia. Es decir, cada nodo se encarga de mantener su propio contador o número de secuencia. Este número es un valor entero que cada nodo incrementa antes de generar un mensaje de control para copiarlo en éste antes de enviarlo.

De manera complementaria al número de secuencia, cada nodo se distingue por un identificador único dentro de la red. De este modo, con la pareja de valores formada por el identificador del nodo y el número de secuencia, es posible distinguir la información válida de la anticuada.

Si un nodo recibe dos paquetes con el mismo identificador de nodo pero con diferentes números de secuencia, la información más reciente será la incluida en el paquete de mayor número de secuencia.

El uso de estos números de secuencia garantiza la ausencia de bucles en todo momento y evita problemas como el de la *cuenta al infinito*[6].

2.2.2. Funcionamiento

El protocolo AODV emplea un mecanismo de descubrimiento de rutas en modo broadcast que también emplea el protocolo DSR con ciertas modificaciones.

En el protocolo DSR, es el nodo origen quien se encarga de calcular la ruta completa hasta el nodo destino. Esto puede llegar a degradar la prestación de la red sobrecargándola y por otra parte, limitaba la duración de las baterías en los terminales, ya que cada paquete incluye en su cabecera la secuencia de nodos por los que debe pasar desde el origen hasta el destino, aumentando así la potencia de transmisión necesaria.

Por el contrario, con AODV el camino se forma gracias a la información mantenida en las tablas de ruta de los nodos intermedios, siendo apropiado para redes ad-hoc, ya que éstos intercambian mensajes únicamente cuando necesitan establecer una comunicación, es decir, envía mensajes a sus vecinos para definir cada ruta. Esto evita la problemática de DSDV, pero introduce una latencia cada vez que se busca una ruta, por lo que debe ser asumible cierto retardo para el primer paquete de la transmisión.

2.2.2.1. Información de encaminamiento

Como ya hemos comentado, AODV almacena la información de encaminamiento en los nodos intermedios en forma de tablas de rutas. Cada uno de estos nodos tiene en su tabla tantas entradas como destinos conoce.

Cada registro contienen los siguientes campos:

- Direcciones IP. Se trata de las direcciones IP de la fuente y del destino para saber en todo momento de donde vienen y hacia dónde van los paquetes.

- Número de secuencia. Número de secuencia del nodo destino, cuyo valor se obtiene de los mensajes de control. Este valor sirve para distinguir información nueva de la información obsoleta, de modo que evitamos bucles y transmisiones de rutas antiguas.

- Indicador de validez del número de secuencia del nodo destino. Si se pretende alcanzar un nodo destino y ha fallado uno de los enlaces implicados, o la ruta ha expirado, el número de secuencia asociado a ese nodo destino se marca como inválido.

[6] La *cuenta a infinito,* hace que los costes o distancias se incrementen indefinidamente sin que el algoritmo llegue a converger nunca.

- Otros indicadores sobre estado y rutas. Por ejemplo, indicadores sobre si la ruta es o no válida, y en este último caso si es reparable, no es reparable y se debe buscar un camino alternativo, o bien, si está siendo reparada.

- Interfaz de red.

- Número de saltos (*hop count*). Número de saltos necesarios para alcanzar el destino desde este nodo. Así, discernimos en caso de tener múltiples rutas, cuál de ellas es la más corta y muy probablemente la que tenga que seleccionarse para hacer el envío de información.

- Siguiente salto. Nodo adyacente al que se debe enviar el paquete para llegar al destino deseado.

- Lista de precursores. Lista de nodos que forman el camino resultante del proceso de descubrimiento de rutas.

- Tiempo de vida de la ruta (lifetime). Tiempo en el que la ruta caduca o debe ser borrada, de forma que se evita que viajen paquetes perdidos por la red y utilizar enlaces de los que no se conoce su estado desde hace mucho tiempo.

Algunos de estos parámetros los encontramos en un paquete AODV, del tipo Route Reply (ver siguiente apartado). Un ejemplo de este mensaje lo encontramos en la Figura 2.8.

```
Ad hoc On-demand Distance Vector Routing Protocol, Route Reply, Dest IP:
192.168.1.11, Orig IP: 192.168.1.11, Lifetime=2000
    Type: Route Reply (2)
    Flags:
        0... .... = RREP Repair: Not set
        .0.. .... = RREP Acknowledgement: Not set
    Prefix Size: 0
    Hop Count: 0
    Destination IP: 192.168.1.11 (192.168.1.11)
    Destination Sequence Number: 7
Originator IP: 192.168.1.11 (192.168.1.11)
Lifetime: 2000
```

Fig. 2.8 Captura de un paquete AODV - RREP

2.2.2.2. Descubrimiento de rutas

Cada vez que un nodo origen pretende comunicarse con un destino, se inicia un proceso de descubrimiento de ruta hacia él, que finaliza cuando la fuente recibe un paquete con la ruta calculada.

Además, existe otra secuencia conocida como mantenimiento de la ruta, que sirve para actuar en caso de caída de un enlace a lo largo de una ruta, pero profundizaremos en esta en un apartado posterior.

Con lo visto hasta el momento, podemos decir que en AODV el descubrimiento de ruta es siempre bajo demanda y sigue un ciclo de petición/respuesta de ruta. Las peticiones son enviadas usando un paquete especial denominado RREQ (Route REQuest). A su vez, las respuestas son enviadas en un paquete RREP (Route REPly).

La secuencia de pasos para el descubrimiento de una ruta podría resumirse de esta manera:

- Cuando un nodo desea conocer una ruta hacia un nodo destino, envía un mensaje RREQ en broadcast.
- Cualquier nodo que conozca una ruta hacia el destino solicitado (incluido el propio destino) puede contestar enviando un mensaje RREP en unicast.
- Esta información viaja de vuelta hasta el nodo que originó el RREQ y sirve para actualizar las rutas de los nodos que lo necesiten.
- La información recibida por el nodo destino del RREP se almacena en su tabla de enrutamiento.

A modo de ejemplo, en la Figura 2.9 se quiere iniciar una comunicación entre el nodo S y el nodo H. Para ello, el nodo S inicia un descubrimiento de ruta enviando un mensaje (RREQ) en modo *broadcast* a sus nodos vecinos, los cuales irán reenviando el mensaje hasta llegar al destino. Durante el proceso de búsqueda todos los nodos van actualizando sus tablas de encaminamiento.

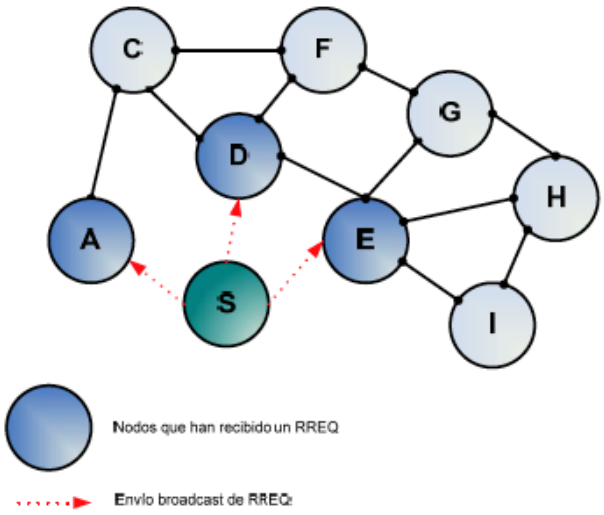

Fig. 2.9 Descubrimiento de ruta, hacia H, iniciado por S

Uno de los campos más relevantes es el identificador RREQ, que se va modificando cada vez que se genera un reenvío. Este identificador sirve para que los nodos que lo vayan recibiendo (nodos intermedios) sepan si el paquete ya ha sido recibido (tiene el mismo identificador) y deben descartarlo o, por el contrario, sí deben retransmitirlo (porque el identificador de paquete es distinto).

En el formato del paquete RREQ, nos encontramos los siguientes campos:

- Dirección IP del origen
- Número de secuencia del origen
- Dirección IP del destino
- Número de secuencia del destino
- Identificador RREQ
- Contador de saltos (*hop count*)

2.2.2.3. Formación del camino de vuelta

Cuando un nodo origen desea alcanzar un nodo destino y desconoce cómo acceder a él, genera un mensaje RREQ. En él se incluyen las direcciones IP y los números de secuencia de los nodos origen y destino. Antes de enviar esta solicitud, el nodo origen incrementa su número de secuencia a fin de evitar conflictos con peticiones anteriores. En el campo correspondiente al número de secuencia de destino, el nodo incluye el último valor aprendido (en caso de que ya hubiese solicitado esa ruta con anterioridad), o bien indica que es desconocido. El mensaje RREQ se difunde por inundación.

La inundación es una técnica de envío de paquetes por la que cuando un nodo tiene información dirigida a un destino concreto, la transmite a sus vecinos.

Si el nodo que la recibe no es el destinatario de esta información, la reenvía de nuevo. Este proceso continúa sucesivamente hasta alcanzar el nodo destino. En complemento a la técnica de inundación y con el objetivo de evitar un consumo excesivo del ancho de banda, el nodo origen emplea el algoritmo de búsqueda expansiva en anillo.

De acuerdo a este algoritmo, inicialmente el mensaje RREQ tiene asociado un valor pequeño de su tiempo de vida TTL (Time To Live), de tal manera que el mensaje se descarta cuando este tiempo expira. Si no se encuentra el destino antes de un plazo determinado, este valor se incrementa progresivamente en el envío de las posteriores solicitudes de rutas. Con el fin de que un nodo no permanezca eternamente intentando alcanzar un destino inaccesible, se tiene un número máximo de intentos.

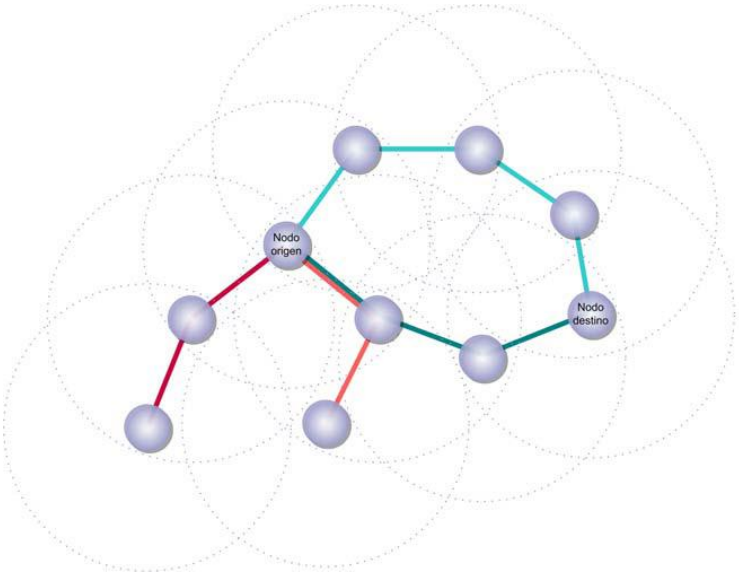

Fig. 2.10 Ejemplo de formación del camino de vuelta

2.2.2.4. Formación del camino de ida

Si el nodo que recibe la solicitud es el propio nodo destino o es un nodo intermedio que tiene una ruta activa hacia el destino, se genera un mensaje de respuesta de ruta.

Se considera que un nodo intermedio tiene una ruta activa hacia el destino cuando el número de secuencia del nodo destino almacenado en la tabla de rutas es mayor o igual al número de secuencia del nodo destino de la solicitud.

Cuando es el nodo destino quien genera la respuesta, incluye en el mensaje RREP como número de secuencia el valor máximo entre su propio número de secuencia y los números de secuencia de destino incluidos en los mensajes de solicitud de rutas.

A diferencia de la solicitud, el mensaje RREP se reenvía de vuelta al origen de forma unicast. Un mensaje RREP siempre sigue el camino inverso de su mensaje RREQ correspondiente, por lo que los nodos típicamente asumen que los enlaces son bidireccionales.

Las respuestas para completar la fase de formación del camino de ida, en este caso, puesto que se supone que ninguno de los nodos intermedios conoce la ruta, es el nodo destino quien genera la respuesta, por lo que los dos mensajes RREP que llegan al nodo origen tienen el mismo número de secuencia de destino.

Para escoger una de las dos rutas posibles, se atiende al menor número de saltos, y así se selecciona el camino trazado en verde azulado, de tres saltos, en lugar del de color turquesa, que consta de cuatro.

Cuando los nodos intermedios por los que pasó previamente la solicitud reciben la respuesta de rutas, pueden verse en la necesidad de actualizar su tabla de rutas.

En la Figura 2.11 se muestra el método que siguen los nodos intermedios para decidir si actualiza o no la entrada en la tabla correspondiente al nodo destino.

Un nodo intermedio procede a la actualización de rutas en dos casos. En primer lugar, refresca su ruta si el nuevo número de secuencia asociado al nodo destino que se incluye en el mensaje RREP es mayor que el que figura en su tabla para ese destino. En segundo lugar, cuando ambos números de secuencia coinciden, se procede a la actualización cuando el número de saltos indicado en la respuesta es inferior al indicado en su tabla.

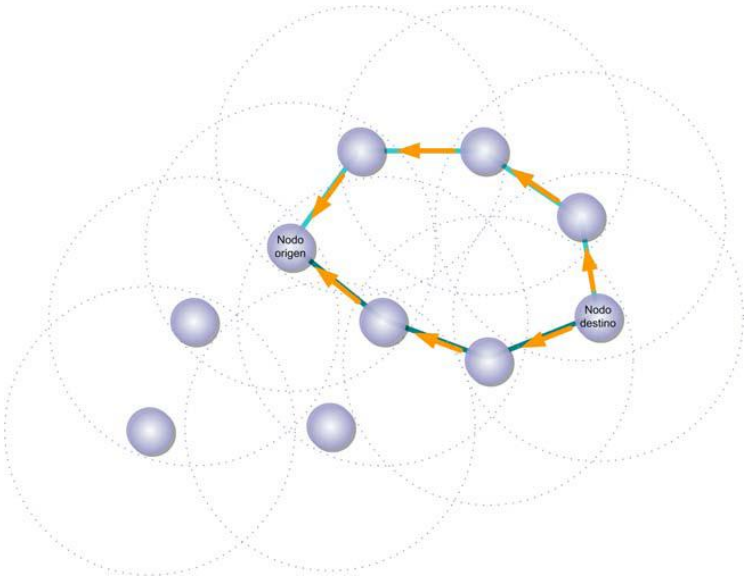

Fig. 2.11 Reenvío de mensajes RREP

Para computar el número real en función del número de saltos que aparece en la respuesta ha de sumarse una unidad para incluir al propio nodo.

Los mensajes RREP redundantes o con un número de secuencia de destino menor se descartan automáticamente. Cuando finalmente el nodo origen recibe el mensaje de respuesta, guarda la ruta hacia el destino y puede comenzar a transmitir paquetes de datos.

2.2.2.5. Mantenimiento de los caminos

El protocolo AODV, al igual que otros protocolos de encaminamiento, emplea mensajes HELLO (o beacons) para que los nodos anuncien a sus vecinos su pertenencia a la red y, de esa manera, se pueda monitorizar en una ruta activa el estado del enlace hacia el siguiente salto.

Los HELLO se envían de manera periódica, lo que permite detectar fallos de enlace. Cuando un nodo deja de recibir estos mensajes por parte de alguno de sus vecinos, puede concluir que el enlace ha dejado de estar operativo. En el momento en el que un nodo advierte un fallo en un enlace, difunde por broadcast un mensaje de error de ruta (RERR, *Route Error*) a sus vecinos, que a su vez lo propagan hacia nodos cuyas rutas podrían verse afectadas por esta eventualidad. Puesto que el mensaje RERR se propaga hacia el nodo origen, cada nodo intermedio marca como inválida la ruta cuando recibe el mensaje de error. No obstante, el nodo origen perjudicado puede reiniciar su operación de descubrimiento de rutas en caso de que aun necesite alcanzar ese nodo destino.

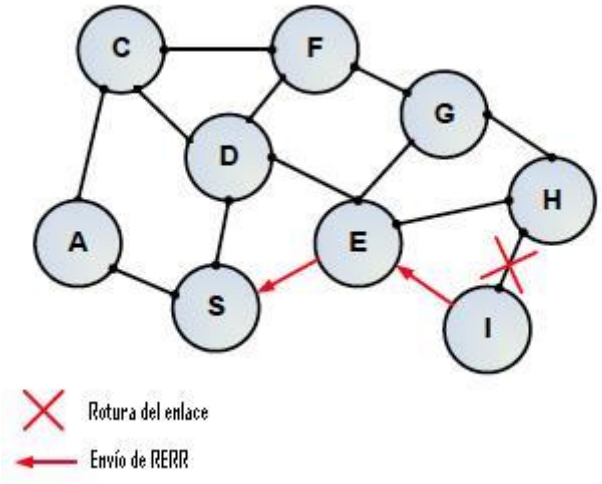

Fig. 2.12 Envío de REER en el mantenimiento de la ruta

2.2.3. Alcance de los mensajes de control del protocolo AODV

Los mensajes RREQ y RREP, junto con los mensajes HELLO y RERR, son los mensajes de control que se intercambian en el protocolo AODV.

Adicionalmente, los nodos también pueden intercambiar otros mensajes denominados mensajes de confirmación de respuesta de rutas, (RREP-ACK, Route Reply ACKnowledgment). Se trata de mensajes mucho más infrecuentes ya que sólo se emplean cuando la red está formada por enlaces unidireccionales.

2.2.4. Parámetros

La implementación del protocolo AODV goza de ciertos grados de libertad para adaptarse a las peculiaridades de la red real a la que se aplica. Previo a la instalación del protocolo en los nodos de la red, es posible modificar el valor numérico de las constantes indicadas en el documento RFC3561.

Se presentan a continuación algunos de los parámetros cuyo valor puede interesar ajustar:

- NET DIAMETER: Número máximo de asaltos para establecer un camino entre un nodo origen y un nodo destino. Por defecto su valor es 35.

- ACTIVE ROUTE TIMEOUT: Tiempo de validez de una ruta activa. Por defecto su valor es de 3 segundos establecidos en el RFC del protocolo.

- MY ROUTE TIMEOUT: Cada nodo elige el tiempo de validez de las rutas en las que él es el nodo destino. Este valor se copia en el campo del tiempo de vida en el mensaje RREP que se genera previamente a la formación del camino de ida. Por defecto su valor es el doble de ACTIVE ROUTE TIMEOUT.

- ALLOWED HELLO LOSS: Número máximo de mensajes HELLO no recibidos antes de considerar que un enlace no está operativo. Por defecto su valor es 2.

- HELLO INTERVAL: Es el tiempo (en milisengundos), que transcurre entre los envíos de HELLO de un nodo.

- RREQ RETRIES: Número máximo de intentos de descubrimiento de ruta antes de considerar un destino inalcanzable. Por defecto su valor es 2.

- TTL START: Tiempo de vida inicial del mensaje RREQ empleado con el algoritmo de búsqueda expansiva en anillo, computado en número de saltos. Por defecto su valor es de 1.

- TTL INCREMENT: Incremento del valor de TTL START empleado con el algoritmo de búsqueda expansiva en anillo. Por defecto su valor es de 2.

2.2.5. Proceso de Local Repair

El proceso de Local Repair se inicia cuando un enlace, de una ruta activa, se rompe. Si esto ocurre, AODV genera un mensaje de RREQ desde el nodo intermedio de la ruta, de manera que la reconstrucción de la ruta no se realiza desde el nodo origen sino desde el intermedio.

El proceso que se sigue es similar al del establecimiento de rutas. Si este procedimiento no consigue reparar la ruta se procede a la generación de un RERR tal y como se ha comentado en el apartado de mantenimiento de rutas.

2.2.6. Algunas particularidades

Como ya se indicó en el comienzo de este capítulo, el protocolo de encaminamiento AODV está diseñado para trabajar con redes móviles ad-hoc extensas, con una población de decenas. Los nodos que implementan el protocolo AODV pueden soportar tasas de movilidad baja, moderada o relativamente alta, así como un amplio rango de niveles de tráfico de datos.

En algunas configuraciones, las redes ad hoc son capaces de proporcionar conectividad entre dominios externos de encaminamiento que no emplean este protocolo. En este caso, se puede emplear una red que trabaje con el protocolo AODV como una simple red de tránsito. En cuanto a las perspectivas de futuro, AODV está preparado para convivir con IPv6. El único cambio necesario para la migración es incrementar la longitud de campos que contienen direcciones IP de 32 a 128 bits.

Actualmente, el protocolo AODV no proporciona ningún mecanismo de seguridad, pese a que los protocolos de encaminamiento son uno de los objetivos principales en los ataques de suplantación.

Por el contrario, está diseñado para el uso en redes en las que los nodos pueden confiar en el resto de los integrantes de la red, ya sea por el uso de claves preconfiguradas o porque se conoce que no hay nodos maliciosos.

Para su uso en redes en las que la pertenencia de los nodos a la misma no está supervisada, se hace necesario añadir algún mecanismo de autenticación, entre los cuales se recomienda IPSec AH (IP Security Authentication Header)

CAPÍTULO 3. ESCENARIO, EXPERIMENTOS Y RESULTADOS

Una vez descritos los protocolos OLSR y AODV, en este capítulo se describen los escenarios de pruebas que se han construido, con el fin de evaluar las prestaciones de dos implementaciones populares de ambos protocolos de encaminamiento, y se presentan los resultados de los experimentos realizados.

En particular, se estudiarán las principales características como son la latencia extremo a extremo, el ancho de banda que consume cada protocolo para trabajar, y el tiempo que tarda en recuperarse la red ante la desaparición de un nodo de ella. Para poder relacionar mejor los diferentes valores obtenidos, se enfrentan resultados de los protocolos a los obtenidos con ambos protocolos.

3.1 Escenarios

3.1.1 Descripción del escenario cálculo Latencia y BW

En el primer escenario, donde los nodos se disponen en forma de cadena, se realizarán las mediciones del retardo que experimenta un paquete que tiene que atravesar varios saltos hasta su destino, así como del ancho de banda disponible extremo a extremo. También se realizarán pruebas de rendimiento de la red con diferentes valores en los tiempos entre envío de los mensajes periódicos de OLSR y AODV.

Compararemos el rendimiento cuando se emplean los protocolos de enrutado AODV y OLSR, con el obtenido en una configuración con rutas estáticas en los nodos.

El origen de los paquetes será siempre el mismo nodo. El destino irá alejándose cada vez más en el escenario de la fuente que inyecta tráfico. Se utilizarán *iptables* para emular las coberturas y lograr que los 4 equipos estén en línea. Esto se explica en el siguiente apartado. El escenario final tendrá la disposición de la Figura 3.1.

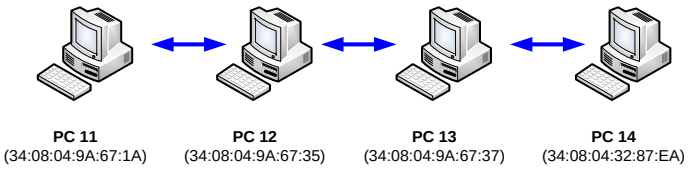

Fig. 3.1 Escenario 1

3.2.2 Descripción del escenario cálculo RCL

Para evaluar la RCL (Route Change Latency, o latencia de cambio de ruta), utilizamos un escenario en el que situamos los 4 equipos donde todos tienen dos vecinos (ver Fig. 3.2). La transmisión tiene como fuente el PC11, y como destino el PC13. Con la configuración de este escenario en marcha, y arrancado el protocolo correspondiente, podremos observar el comportamiento de los protocolos y cuál será el nodo que encaminará los paquetes hacia el destino.

Para la ilustración de esta topología, cogemos a modo de ejemplo el caso particular en el que el PC 12 es el vecino del PC 11 para enlazarlo con el destino de la comunicación.

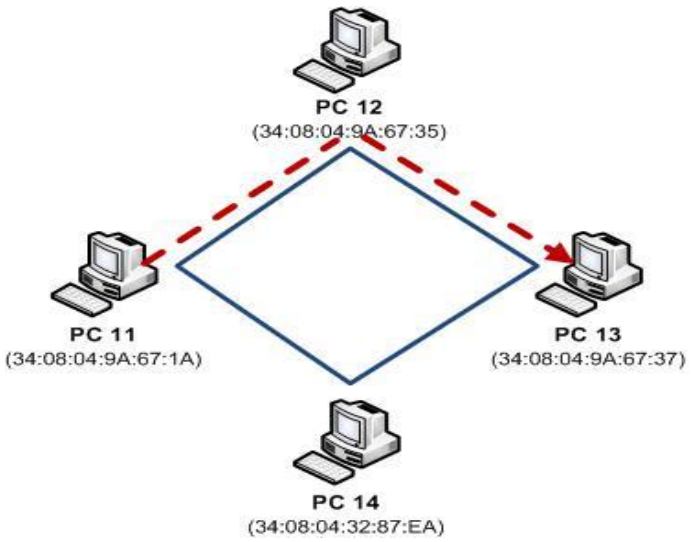

Fig. 3.2 Escenario 2

Forzaremos el recalculo de la ruta por el que se transmitirá la información, al desactivar la interface de red en el equipo vecino de la fuente que redirige el tráfico hacia el destino. Esto lo hacemos con el comando *ifconfig wlan down* en el terminal de Linux sobre la interficie en cuestión.

3.2 Material utilizado

3.2.1 Hardware

3.2.1.1 Desktop

Los hosts que utilizamos son ordenadores de sobremesa. El laboratorio donde se realizará las pruebas, tiene 12 ordenadores. Todos son el mismo modelo, un HP Compaq Intel Core 2 Duo 3.00 Ghz con 2 Gb de RAM. Son equipos que vienen con la instalación predefinida de Windows Vista, aunque el equipo de mantenimiento de la universidad ha optado inteligentemente por instalar una distribución Linux. El hardware utilizado es muy holgado para preocuparse por los gastos de recursos. Es un equipo que está conectado a la red eléctrica, así que el consumo de baterías no cabe en nuestro estudio. La Tabla 3.1 muestra las especificaciones técnicas que ofrece el fabricante.

Fabricante del sistema	**Hewlett-Packard**
Modelo del sistema	HP Compaq dc5800 Microtower
Tipo de sistema	Equipo basado en X86
Procesador	x86 Family 6 Model 23 Stepping 6 GenuineIntel ~2992 MHz
BIOS Versión/Fecha	Hewlett-Packard 786F2 v01.04, 31/01/2008
Versión de SMBIOS	2.5
Directorio del sistema	C:\WINDOWS\system32
Dispostivo de inicio	\Device\HarddiskVolume1
Configuración regional	España
Memoria física total	2.052,00 MB
Memoria física disponible	1,13 GB
Memoria virtual total	2,00 GB
Memoria virtual disponible	1,96 GB
Espacio de archivo de paginación	3,83 GB

Tabla 3.1 Características estación trabajo

3.2.1.2 Tarjeta wireless

Las tarjetas inalámbricas que se utilizarán para montar este escenario corresponden al modelo *D - link "N" DWA - 125*. Esta tarjeta está soportada por múltiples sistemas operativos. En Ubuntu está soportada para trabajar por

defecto en modo infraestructura. Pero para hacerla trabajar en modo ad-hoc es necesaria la instalación de unos drivers que no vienen soportados en el núcleo del actual Ubuntu. La Tabla 3.2 muestra las características técnicas de la tarjeta inalámbrica descrita.

Standards	**IEEE 802.11b** **IEEE 802.11g**
Bus Type	USB 2.0 (1.1 compatible)
Security	WPA/WPA2 - Wi-Fi Protected Access (TKIP, AES, MIC, IV Expansion, Shared Key Authentication) WPS (PIN and PBC
Media Access Control	CSMA/CA with ACK
chipset: ralink	2870/3070 (rt3070)
Frequency Range	2.4GHz to 2.483GHz
Power Consumption (802.11n)	Tx: 220 mA Rx: 160 mA
Modulation Technology	Orthogonal Frequency
Division Multiplexing (OFDM)	Complementary Code Keying (CCK)
Transmitter Power Output	17 dBm (802.11b/g)
Operating Voltage	5 VDC +/- 10%
Operating Temperature	32°F to 104°F (0°C to 40°C)
Operating Humidity	10% to 90% maximum (non-condensing)
Certifications	CE Wi-Fi WPS Certified

Tabla 3.2 Características tarjeta wireless

3.2.2 Software

3.2.2.1 Sistema operativo

El sistema operativo que se utiliza es Ubuntu 10.04 LTS - Lucid Lynx. Basado en Debian, su código es completamente abierto. Regularmente, se lanzan actualizaciones y nuevas versiones ofreciendo apoyo después de su lanzamiento. El núcleo que utiliza es la subversión 2.6.32.

El núcleo es la parte que se encarga de gestionar la comunicación entre el software y el hardware. Un dispositivo sólo podrá ser usado si el núcleo del sistema operativo lo soporta, o si hay un controlador capaz de controlarlo y si se configura apropiadamente para hacerlo.

Para poder utilizar las tarjetas inalámbricas se ha utilizado la versión de núcleo 2.6.32-wl, que tiene pre-compilados los controladores necesarios que permiten configurarlas en modo ad-hoc, ya que los drivers que trae por defecto el núcleo 2.6.32 no lo permiten.

Para poder realizar la instalación del núcleo 2.6.32-wl, se deben instalar dos paquetes:

- *linux-headers-2.6.32-wl_rt3070_i386.deb :*

Si los headers no están presentes en el momento de la actualización de núcleo, los módulos que deban actualizarse no se lo harán, y en consecuencia al reiniciar el equipo los dispositivos que dependen de esos módulos no van a funcionar.

- *linux-image-2.6.32-wl_rt3070_i386.deb :*

Esta es una imagen pre-compilada con los módulos para las tarjetas wireless.

3.2.2.2 aodv-uu-0.9.6

La implementación escogida es aodv-uu-0.9.6, desarrollada por Uppsala University. El código está bajo licencia GPL. Esto permite que los usuarios podamos descargar el paquete con la última versión del protocolo. La instalación y puesta en marcha se describe en los Anexos.

3.2.2.3 0.5.6-r7-1

Esta implementación está incluida dentro de los repositorios de la versión de Linux utilizada. Es configurable sin tener que volver a recompilar el código y dispone de un manual dentro del sistema operativo para poder consultar sus opciones. Todo esto hace que esté bien implementado para utilizarlo a lo largo del proyecto. La instalación y puesta en marcha se describe en los Anexos.

3.2.2.4 Iptables

Puesto que el escenario real es con ordenadores de sobremesa dentro de la misma habitación, es necesario utilizar herramientas de filtrado de paquetes para emular las coberturas de los nodos en los distintos escenarios de pruebas. En Linux disponemos de la herramienta *iptables* para definir las políticas del filtrado del tráfico que circula por la red y pasan por el nodo.

Las reglas que permite definir están estructuradas dentro de cadenas. El siguiente esquema es la representación de las diferentes cadenas en iptables.

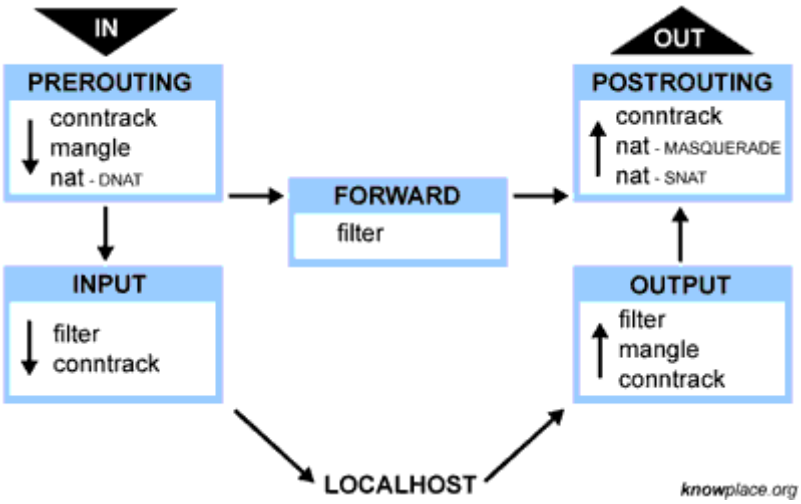

Fig. 3.3 Esquema de iptables

Todos los nodos reciben los paquetes que se emiten en la red. Filtraremos por la dirección MAC que tienen los dispositivos con la siguiente regla:

$> iptables –t mangle –A PREROUTING –m mac --mac-source 00:00:00:00:00:00 –j DROP

Donde pondremos la MAC del nodo que deba estar a fuera del alcance del nodo al que aplicamos la regla, hasta acabar definiendo el dibujo del escenario.

En el caso de tener que deshacer una regla, se debe cambiar el siguiente flag en el comando:

$> iptables –t mangle –D PREROUTING –m mac --mac-source 00:00:00:00:00:00 –j DROP

3.2.2.5 Ping

La herramienta ping se utiliza normalmente para el diagnóstico de conectividad en redes. Ping utiliza el protocolo *ICMP*, el cual envía mensajes a direcciones IP esperando confirmación de que el destino los ha recibido.

Se envían mensajes de solicitud ICMP *ECHO_PING_REQUEST*, encapsulado en un paquete IP hacia una dirección IP. El host con destino responderá con un *ECHO_PING_REPLY* con dirección destino el origen del mensaje de solicitud.

EL encapsulamiento que hace el protocolo IP sobre el mensaje ICMP añade 20 bytes. El mensaje ICMP de solicitud añade una cabecera de 8 bytes que incluye, además del tipo de mensaje y el código del mismo, un número identificador y una secuencia de números, de 32 bits, que deberán coincidir con el mensaje ICMP de respuesta; además de un espacio opcional para datos, por defecto tiene un tamaño de 56 bytes. Enviando tramas de 98 bytes.

Fig. 3.4 Trama ICMP

Al terminar la ejecución de ping, el resultado ofrece el tiempo de ida y vuelta (RTT) máximo y mínimo que ha tardado un paquete, la desviación media y el parámetro que más nos interesa, el RTT medio de un paquete.

La herramienta ping permite configurar el número de paquetes *ICMP request* que se enviarán. Para todos, se espera la recepción de un paquete *ICMP reply.* En el comando a ejecutar en el terminal, se indica la dirección IP destino del ping y con *–c* el número de pings que se envían:

$> ping 192.168.1.14 –c 400

También permite configurar el intervalo entre paquetes enviados. El flag *–i* en el comando seguido del número de segundos permite la configuración del tiempo:

$> ping -i 0.001 -c 1000 -s 1472 192.168.1.14

Esto junto al número de paquetes enviados, permite poder inyectar un tráfico que podemos calcular:

$$(1514 \times 8) \,/\, 0{,}001 = 12.112 \text{ Kbps} \qquad \textbf{(3.1)}$$

En caso de pérdida de paquetes, el protocolo *ICMP* reacciona enviando paquetes duplicados hasta que comienza a recibir los paquetes de respuesta, reduciendo así el número de paquetes perdidos. De todo esto se hace un resumen en la finalización del ping.

3.2.2.6 Herramientas configuración tarjeta

ifconfig

El comando *ifconfig* se usa para verificar y configurar las interficies de red de que disponga el equipo.

Esta herramienta permite tanto activar como desactivar las interficies de red.

Durante las pruebas del cálculo del tiempo RCL, será el comando que utilicemos para simular la caída del nodo.

Los comandos que se utilizarán son los siguientes:

$> ifconfig wlanX Y.Y.Y.Y

- Configura la IP Y*.Y.Y.Y* de la interficie *wanX*

$> ifconfig wlanX up

- Levantamos interficie.

$> ifconfig wlanX down

- Bajamos interficie.

iwconfig

Herramienta que permite configurar características las interficies inalámbricas del host. En este proyecto se utiliza con los siguientes parámetros para configurar por igual cada host.

$> iwconfig wlanX mode ad-hoc channel Y essid NOM_XARXA rate 11M

- mode ad-hoc : modo de trabajo de la tarjeta.
- channel Y : elección del canal por donde transmitir.
- essid NOM_XARXA : nombre de la red a la que conectar
- rate 11M : velocidad de transmisión de la tarjeta, en Mbps. (en el nostre cas, 11Mbps)

iwlist

Linux proporciona una serie de herramientas que permiten, una vez configurada la tarjeta, analizar el medio aéreo para tener información de las redes que actualmente estén transmitiendo.

$> iwlist wlanX scan

- Escaneo del medio aéreo buscando datos de las redes que en ese momento estén emitiendo.

3.2.2.7 Wireshark

Wireshark es un programa que trabaja analizando los paquetes que pasen por la interficie de red del equipo. Este tipo de programas esnifadores de tráfico, se utilizan para la resolución de problemas en administraciones de red. Tiene la posibilidad de utilizar filtros en base a diferentes parámetros que le hacen muy potente y el uso de la interfaz es sencillo y detallado, es posible desglosar por capas de protocolos los paquetes capturados.
La versión que se utiliza es Wireshark Version 1.2.7 para Ubuntu. Esta versión está dentro de los repositorios del sistema operativo, es fácilmente instalable y se mantiene actualizada junto al resto de programas que figuran en los repositorios del equipo.

Su uso en este tipo de estudios es muy recomendado, ya que proporciona una visión del comportamiento de los protocolos frente a cambios en la estructura de la red, cómo y cuándo anuncia los cambios, con qué frecuencia, etc.

La ejecución del programa la realizaremos desde la consola del sistema y bajo los permisos de superusuario (root). Esto permite capturar paquetes desde la interfaz seleccionada.

Esto sería un riesgo para la seguridad del sistema, porque por ejemplo, podría permitir la ejecución de código externo. Esto que en administración de redes podría ser un agujero de seguridad, aquí podemos permitírnoslo, ya que tenemos la certeza de que el tráfico que capturamos es el que se genera de uno de los host el escenario.

3.3 Experimentos

3.3.1 Condiciones de propagación de señal

Las mediciones se realizan en un escenario cerrado. El mayor problema que nos encontramos son la propagación multicamino (causada por reflexiones de la señal) y señales que puedan estar emitiendo en el mismo canal de frecuencia. La emisión de estas antenas wireless es omnidireccional. Esto hará que antes de emitir desde el origen debemos revisar qué canal es el que está más libre.

Las mediciones se han realizado en un entorno donde los nodos tienen plena cobertura unos con otros, por lo que el tiempo de propagación resulta despreciable, y los posibles problemas para la transmisión de señal indicados anteriormente son cuantitativamente tendrán poca influencia en las pruebas.

3.3.2 Latencia

Vamos a realizar las mediciones del tiempo que tarda un paquete en ir y volver para diferentes destinos atravesando diferentes nodos intermedios. En cada salto se añade un tiempo que requiere el host, para buscar si la ruta de destino está registrada dentro de la tabla de enrutado de sistema para poder reenviar el paquete recibido y reenviar el paquete. En este paso intervienen diferentes elementos propios del host, como el hardware descrito como en la eficiencia del software utilizado. Todos estos tiempos extras al tiempo de transmisión, definen la latencia que sufre la transmisión de datos.
Este parámetro es importante para ver el tiempo que añaden los diferentes protocolos utilizados. Esto se ve reflejado en la eficiencia de enrutado de cada protocolo y ver el tiempo de descubrimiento de ruta, los retardos se acumulan en cada salto provocados por retardos de acceso al medio, tiempo en colas y de transmisión.

Para la prueba se mandarán 400 mensajes ICMP con un intervalo de 1 segundo y con el tamaño de paquete por defecto de 56 bytes de datos. El número de paquetes transmitido es lo suficiente como para que si hay algún tipo de incidencia en el canal aéreo, pase lo más desapercibido posible.

3.3.2.1 Rutas estáticas

El host tiene definida una tabla en su sistema, que le permite al kernel poder enviar los paquetes por la interfaz adecuada.

La manipulación de esta tabla puede ser controlada por la ejecución de algún protocolo o se pueden añadir entradas manualmente a ella.

Si la hacemos manualmente, las entradas deberán introducirse siempre de esta manera, perdiendo la ventaja de que se pueda actualizar automáticamente.

Esto es posible gracias a el comando de sistema *route*. Con él podemos consultar y modificar las entradas para que los paquetes añadan como dirección IP destino la que se necesaria para llevar a cabo la transmisión.

Para añadir la dirección a la tabla de rutas se utiliza el siguiente comando:

$> route add -host [IP host] gw [IP gateway]

- *[IP host]* - IP del host vecino
- *[IP gateway]* - IP del host a dos saltos o más

La tabla de rutas incrementará su número de entradas. Cuanto más completa esté esta tabla, mayor capacidad de enrutado tendrá. Un ejemplo de la configuración de la tabla para el escenario que nos interesa sería el de la Tabla 3.3.

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.13    192.168.1.12    255.255.255.255 UGH   0      0        0 wlan0
192.168.1.14    192.168.1.12    255.255.255.255 UGH   0      0        0 wlan0
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 wlan0
```

Tabla 3.3 Tabla de rutas estáticas

Aquí se muestran los enlaces que tiene el host, en este caso el PC con IP 192.168.1.11 con el resto de hosts de la red. El PC tiene como siguiente nodo al que le entregará los paquetes para que los encamine al PC 12. En nuestro escenario vemos que no tenemos visión directa con los PCs 13 y 14. Todas las direcciones IP están asociadas a la interficie wireless wlan0.

La configuración de un host no permite por defecto poder encaminar paquetes de la red al que pertenece. Esto se debe activar manualmente en el archivo que indica al kernel del sistema que lea las cabeceras de los paquetes que recibe, y si no van dirigidos a él, busque en su tabla de rutas si puede llegar hasta el destino del paquete original:

$> echo 1 > /proc/sys/net/ipv4/ip_forward

En las mediciones que se han realizado, se han ido incrementando el número de saltos hasta 3. El origen de los pings será la dirección IP 192.168.1.11 y el destino serán las direcciones IP 192.168.1.12 (1-hop), 192.168.1.13 (2-hops) y 192.168.1.14 (3-hops). Los host donde debe estar activada la opción de routing serán los PCs que tengan más de un vecino en las entradas de la tablas de ruta.

De la realización de las pruebas obtenemos unos valores en la ejecución del ping, que nos indican la latencia en la transmisión a través de hasta tres saltos.

	RTT ms				
Saltos	REmin	Reavg	Remax	Remdev	% loss
1	0,282	0,427	2,274	0,13	0
2	0,462	0,748	7,5	0,691	0
3	0,87	1,224	7,748	0,663	0

**Resultados dados en milisegundos.*

Tabla 3.4 Resultados de rutas estáticas.

La tabla muestra los valores que tomaremos como referencia, para calcular el añadido de los protocolos de enrutado, ya que el sistema no espera el resultado del proceso de elección de ruta por parte de un demonio que está ejecutándose en el sistema, si no ya tiene las entradas necesarias en su tabla para poder encaminar el paquete. Esta incapacidad para borrar y añadir rutas, no le permite al host adaptarse por si solo a una reconfiguración de la red, debido por ejemplo a la caída o pérdida de un nodo.

3.3.2.2 OLSR

Los paquetes que se transmitan ahora, contarán con el añadido en la transmisión de paquetes de control que envía el demonio que está corriendo en los hosts de la red. Por cada uno de ellos se genera un tráfico extra que se añade el que nosotros inyectamos. Para cada prueba se ha cambiado el tiempo de configuración del protocolo OLSR de sus diferentes mensajes de control.

Para estas pruebas se doblan los valores del HELLO_INTERVAL y del TC_INTERVAL. Este cambio afecta también a los tiempos que validan la entrada de los mensajes en el sistema, HELLO_VALIDITY_TIME y TC_VALIDITY_TIME. EL protocolo OLSR tiene influencia en el consumo de ancho de banda.
En cambio, en cuanto a la ocupación del canal se refiere, el trabajo que realiza el protocolo para mantener actualizada la tabla de rutas, hace que sea transparente a la latencia que sufre el canal.

Parámetro	***Tiempo (s)***			
	H 05	*H 1*	*H 2*	*H 4*
HELLO_INTERVAL	0.5	1	2	4
HELLO_VALIDITY_TIME	1.5	3	6	12
TC_INTERVAL	1.25	2.5	5	10
TC_VALIDITY_TIME	3.75	7.5	15	30

Tabla 3.5 Configuraciones de parámetros OLSR

El valor del mensaje es el que aparece junto a la H. Los valores que se han recogido de la ejecución de los diferentes pings son los siguientes:

- *RTT - Tiempos mínimos*

En los resultados de los tiempos mínimos se aprecia el incremento de tiempo a medida que se atraviesan los hosts intermedios en la comunicación. Es poco probable que las medidas de mínimos estén influenciadas por factores aleatorios.

	min H05	min H1	min H2	min H4
1 salto	1,067	1,032	1,033	0,953
2 saltos	2,322	2,318	2,318	2,28
3 saltos	3,226	3,759	3,7005	2,861

**Resultados dados en milisegundos.*

Tabla 3.6 RTTs mínimos

- *RTT – Tiempos medios*

Las medias de los tiempos, para cada salto, no exhiben diferencias significativas entre los diferentes valores de intervalos de los HELLO. La frecuencia con la que se envían los mensajes de OLSR no afecta a los tiempos latencia. Los tiempos medios de ida y vuelta aumentan de forma lineal con el número de nodos intermedios que reenvían la información.

	avg H05	avg H1	avg H2	avg H4
1 salto	1,272	1,182	1,269	1,524
2 saltos	2,685	2,681	2,681	2,913
3 saltos	3,651	4,148	4,148	4,453

** Resultados dados en milisegundos.*

Tabla 3.8 RTTs medios

- *Porcentaje de paquetes perdidos*

El resumen que proporciona la herramienta ping ofrece también un resultado de los paquetes perdidos en la transmisión. En la realización de estas pruebas se han detectado que la configuración del protocolo OLSR con el tiempo de HELLO igual a 4 segundos, obtuvo peores resultados que el resto de configuraciones, aunque no se valoran como problemas graves.

	% loss H0,5	% loss H1	% loss H2	% loss H4
1 salto	1,67	2	0,5	2,33
2 saltos	1,33	2,5	1,5	3
3 saltos	2	3,5	3	4,33

Tabla 3.9 Porcentaje de paquetes perdidos

3.3.2.3 AODV

Las pruebas de latencia realizadas con este protocolo son diferentes. Debido a su comportamiento, se debe tener en cuenta el primer mensaje que se envía al iniciar la comunicación. Por esta razón, se han hecho más series de pings, de menos duración, así se aprecia mejor la influencia del descubrimiento de ruta que se lleva a cabo.

Después de cada *ping* se ha vaciado la tabla *arp* de cada host. Esto provoca que el host que inicia la comunicación tenga que hacer todas las peticiones necesarias en cada prueba.

Inicio del establecimiento de ruta

Para ver el inicio del establecimiento de ruta en este escenario, tomaremos como ejemplo un ping mandado desde el PC 11 atravesando un host hasta el PC 13, teniendo los todos los hosts en línea. Estas pruebas han servido para ver el comportamiento real de la distribución instalada del protocolo AODV de *Upsala University.*

El inicio de la comunicación capturado con el *wireshark* es el siguiente:

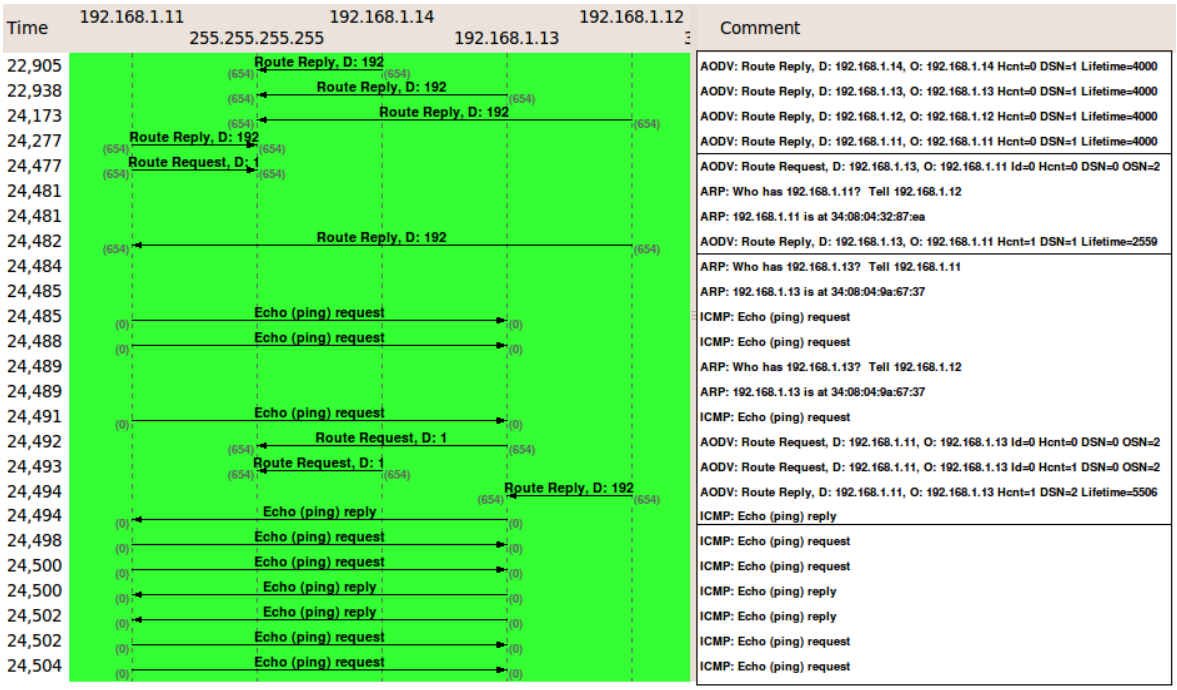

Fig. 3.5 Flujo de peticiones para establecer la conexión

La captura que vemos empieza con los mensajes *ROUTE_REPLY* que se envían cada HELLO_INTERVAL. Cada uno de los host de la red los envía con destino *broadcast*, actualizando así el tiempo de vida de la ruta con sus vecinos.

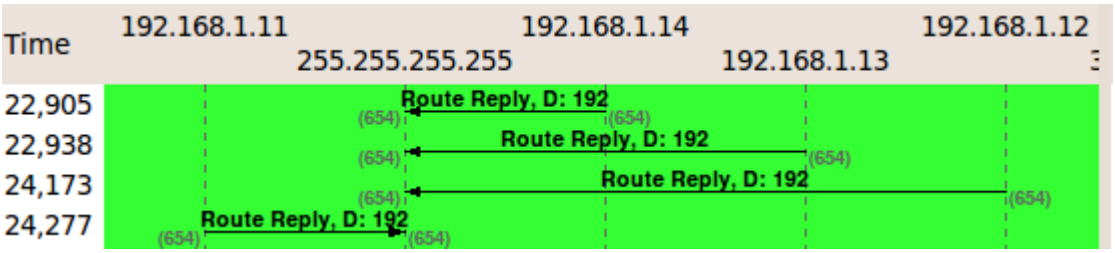

Fig. 3.6 Detalle de los mensajes HELLO AODV

Antes de enviar el primer paquete ICMP se debe establecer los enlaces necesarios. El protocolo AODV del PC 11 que inicia la comunicación en el segundo 24,477 de la captura, manda un mensaje *RREQ* preguntando a sus vecinos una ruta para el destino del mensaje ICMP. El único host que responde es el PC 12, pero antes de enviar el mensaje *RREP,* debe conocer la dirección MAC del destino, por lo que aparecen mensajes del protocolo ARP para solucionar esta petición. Estos mensajes no aparecen representados en el gráfico ya que sólo mostramos los protocolos a nivel de red. Una vez recibe el mensaje con la respuesta ARP, registra la nueva entrada en la tabla ARP del sistema, y ahora sí, el PC 12 puede enviar un paquete directo al host que envió el mensaje *RREQ,* estableciendo así el primer enlace entre hosts para establecer la comunicación.

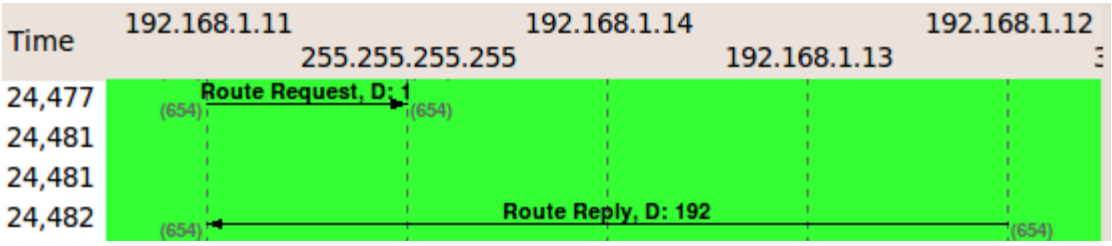

Fig. 3.7 Detalle de la petición de ruta

Es aquí donde encontramos un punto donde la distribución del AODV *aodv-uu-0.9.6* no se comporta como se esperaba.

El descubrimiento de ruta en AODV sufre siempre la pérdida de los dos primeros paquetes. En el momento que el host origen recibe el *ROUTE_REPLY,* el host manda un mensaje ARP preguntando la dirección MAC de la tarjeta que tiene la IP 13, para tratar de enviar el mensaje directamente a ella. El host responde informando de su MAC, ya que aunque estén dropados por iptables entre ellos, las reglas de descarte de paquetes se aplican para paquetes IP. Esto permite que el PC 11 tenga la entrada de la MAC del destino en la tabla arp del sistema, y haga un intento de enviar un paquete directamente en el tiempo 24,485. El *ECHO_PING_REQUEST* que se manda en 24,488 lo hace a la IP 13 con la MAC del host vecino.

Esto provoca el intercambio de mensajes ARP entre los hosts con IP 12 y 13. Estableciendo así el enlace que faltaba para poder establecer el camino entre los PCs 11 y 13.

El ping enviado en el tiempo 24,491 es el que reenvía el host intermedio (IP 12) con la IP origen del 11. Ahora el PC 13 debe responder con un *ECHO_PING_REPLY* al PC 11.

Para ello envía el mensaje de descubrimiento de ruta *ROUTE_REQUEST,* a la dirección broadcast, a la cual responden los dos vecinos que tiene, el host con IP 12 y el host con IP 14. La IP 14 retransmite a dirección broadcast el mensaje de la IP 13.

En cambio, el host con IP 12 si tiene una entrada en su tabla de rutas para llegar a la IP 11, por lo que responde a la solicitud con un *ROUTE_REPLY* directamente a la IP 13. El último paquete es del host con IP 13 que hace un envío directamente a la IP 11 contestando al último ping que recibió.

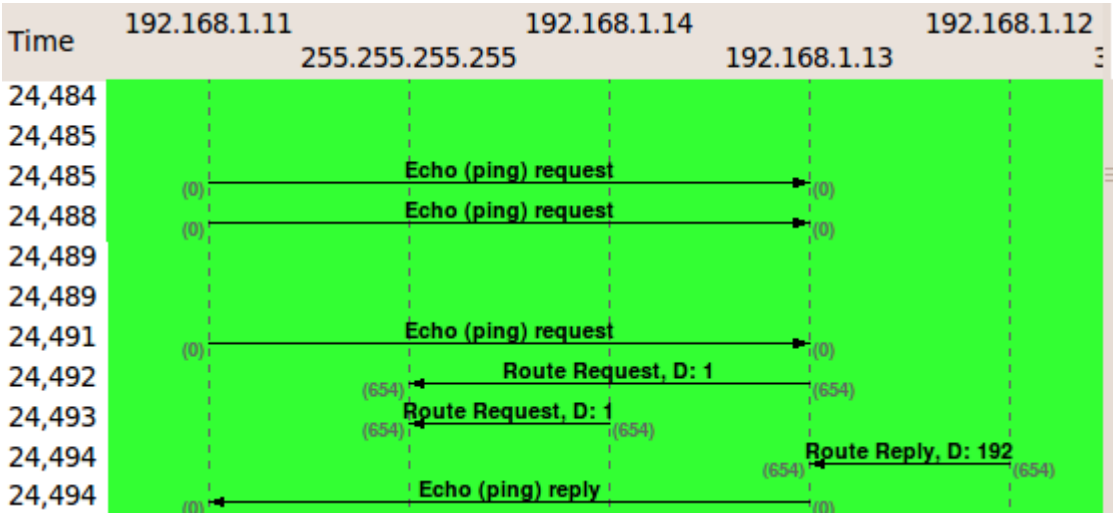

Fig. 3.8 Detalle del establecimiento ruta

A partir de aquí se establece la comunicación entre los host con IP 11 y 13, pasando por un host intermedio, encargado de encaminar los paquetes por la ruta establecida.

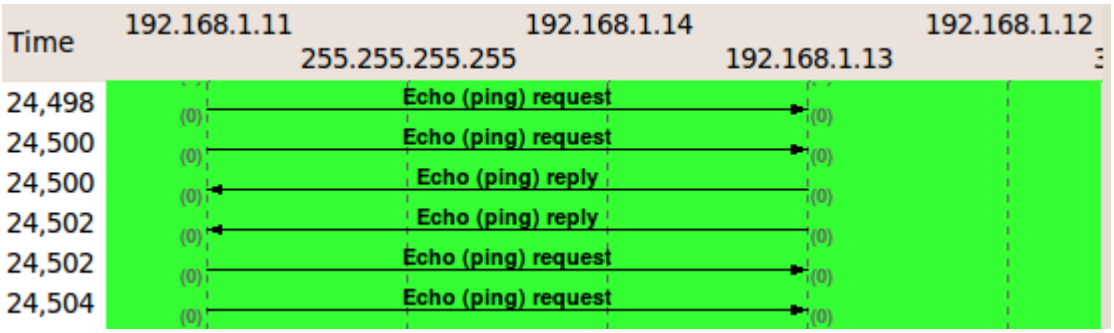

Fig. 3.9 Detalle del inicio comunicación ICMP

Se han realizado pruebas con una única configuración de AODV ya que el objeto de estudio son las pérdidas al inicio de la transmisión, y estas se producen por igual en todas las configuraciones del protocolo.

Teniendo en cuenta que se mandan diferentes series de 20 pings y siempre se pierden dos paquetes cuando se atraviesa un host intermedio, el mínimo de paquetes perdidos será del 10 %.

	Min	avg	max	mdev	% loss
1 salto	1,231	1,332	3,184	0,497	10
2 saltos	2,344	2,862	11,647	1,312	10
3 saltos	3,663	4,222	13,892	1,132	15

** Resultados dados en milisegundos.*

Tabla 3.10 Tabla de los resultados de RTTs en AODV

3.3.2.4 Comparación de resultados

Para las comparaciones hemos cogido las configuraciones que proponen los RFCs para cada protocolo. En estos resultados queda claro que es mejor utilizar rutas estáticas. Utilizaremos esas medidas para compararlas con los protocolos, ya que el uso de los protocolos por el beneficio que aportan, conlleva un sacrificio de tiempo extra.

La diferencia que se aprecian entre los dos protocolos y las rutas estáticas, se debe al tiempo de procesado de los mismos.

Por eso los tiempos son algo superiores con AODV. En la Fig. 3.10 que representado la diferencia entre tiempos.

	RE avg	OLSR avg	AODV avg
1 salto	0,427	1,269	1,231
2 saltos	0,748	2,647	2,863
3 saltos	1,224	4,111	4,222

** Resultados expresado en milisegundos.*

Tabla 3.11 Comparación de los diferentes RTTs

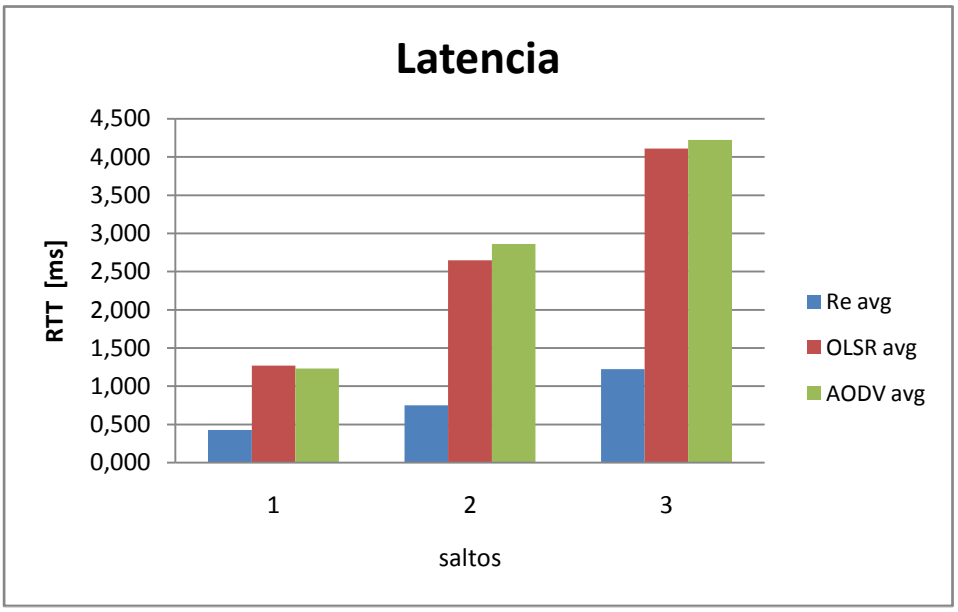

Fig. 3.10 Gráfica de comparación de los diferentes RTTs

Estas gráficas muestran como el protocolo AODV despunta un poco más que el OLSR, pero podemos afirmar que una red que pudiera funcionar con rutas estáticas, con la implantación cualquiera de los dos protocolos estudiados, pueden llegar a sufrir una latencia superior al 30% de lo que sufrirían con rutas estáticas.

3.3.3 Ancho de banda

Una vez realizadas las pruebas de latencia, para seguir evaluando las prestaciones de los protocolos, se realizarán las pruebas para ver qué ancho de banda consume cada uno. El escenario sigue siendo el mismo que para la latencia (topología de nodos en cadena).

Se medirá el ancho de banda disponible entre origen y destino, variando el número de nodos intermedios, y se analizará cómo afecta la transmisión de mensajes de los protocolos de encaminamiento al ancho de banda disponible.
Para ello, se utiliza la herramienta ping pero configurando un tamaño de datos, para que la longitud de la trama resultante sea el tamaño máximo de MTU. Se enviarán mil paquetes en todas las pruebas con un intervalo entre paquetes de 1 milisegundo. A partir de los datos recogidos se calculará el ancho de banda. Para cada prueba se espera a que se vacíen las tablas ARP para que en todas tarden lo mismo en el descubrimiento de ruta.

El tráfico generado es algo mayor que el que se configuró por defecto en las tarjetas inalámbricas, 12 Mbps del ping y 11 Mbps, respectivamente. De este modo se podrá medir el máximo ancho de banda disponible en cada caso.

Las medidas se hacen cambiando los tiempos de los intervalos de los mensajes de control (y sus respectivos tiempos de validez) igual que con el cálculo de la latencia. Para las pruebas se han tomado las medias de los valores obtenidos.

3.3.3.1 OLSR

Para OLSR y protocolos proactivos, hay mecanismos para tener una relación de la calidad de los enlaces que tiene el host.

Esto que es útil para redes móviles, aquí añade un retardo que no es necesario. Los saltos que deben hacer los paquetes penalizan mucho el ancho de banda en OLSR.

	OLSR				**BW Rutas Estat.**
	BW H05	*BW H1*	*BW H2*	*BW H4*	
1 salto	730,278	821,720	978,348	1113,408	1571,898
2 saltos	252,346	277,498	274,201	362,731	411,584
3 saltos	49,906	47,674	63,120	133,723	177,353

** Resultados dados en Kbps.*

Tabla 3.12 Throughput OLSR

Con 0,5 segundos de valor de HELLO_INTERVAL (H05), el ancho de banda se ve afectado de forma considerable. Con 4 segundos de valor HELLO_INTERVAL (H4), se tienen los valores más cercanos a los obtenidos con rutas estáticas. El sacrificio de ancho de banda también afecta al número de paquetes perdidos, lo que hace que cuanto menos tiempo entre mensajes de HELLO mayor número de paquetes se pierdan. Las redes que tengan esta configuración de protocolo, deben ser redes que transporten poca información.

La Fig. 3.11 muestra el ancho de banda disponible obtenido según el número de saltos, y para las distintas configuraciones de OLSR.

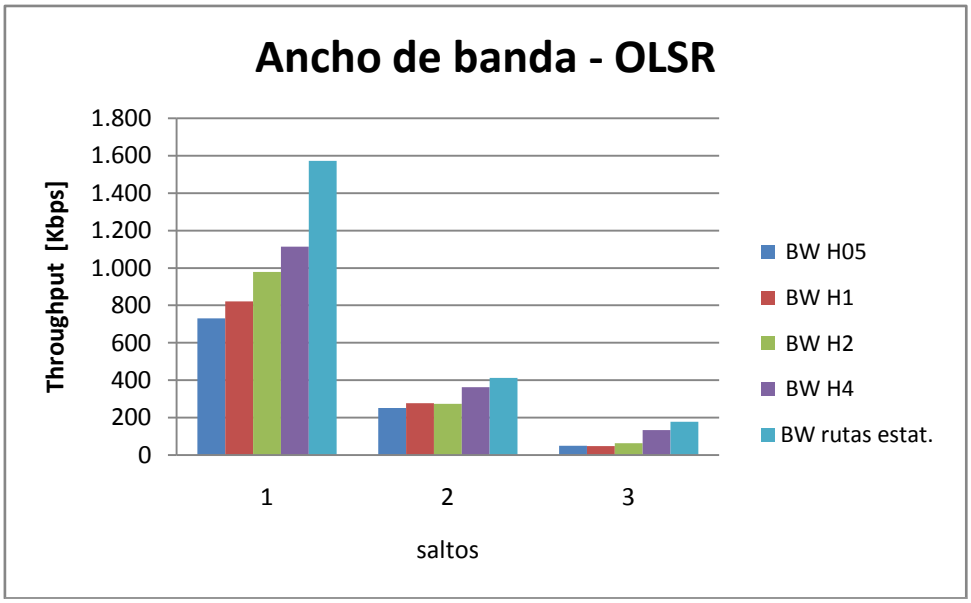

Fig. 3.11 Gráfica del throughput en OLSR

3.3.3.2 AODV

En estas pruebas se han modificado los intervalos entre mensajes de HELLO y se han utilizado los mismos valores que con OLSR. El inconveniente que tenía AODV sobre el descubrimiento de ruta no influye significativamente las medidas de ancho de banda disponible. En cambio se destaca por unos valores de ancho de banda tan altos como los obtenidos con rutas estáticas.

	AODV				**BW Rutas Estat.**
	BW H0,25	*BW H0,5*	*BW H1*	*BW H2*	
1 salto	1451,256	1489,871	1518,419	1570,529	1571,898
2 saltos	378,920	344,885	408,395	304,423	411,584
3 saltos	45,298	81,337	47,628	85,387	177,353

** Resultados dados en Kbps.*

Tabla 3.13 Throughput en AODV

El hecho de que el protocolo guarde las entradas en la tabla de rutas del sistema de sus vecinos, y el menor tamaño de sus paquetes, hace que los valores obtenidos con el primer salto representen mejor como AODV deja más ancho de banda disponible. Podemos ver como en el mejor de los casos obtenidos con AODV, el ancho de banda casi llega a igualarse con los obtenidos con rutas estáticas.

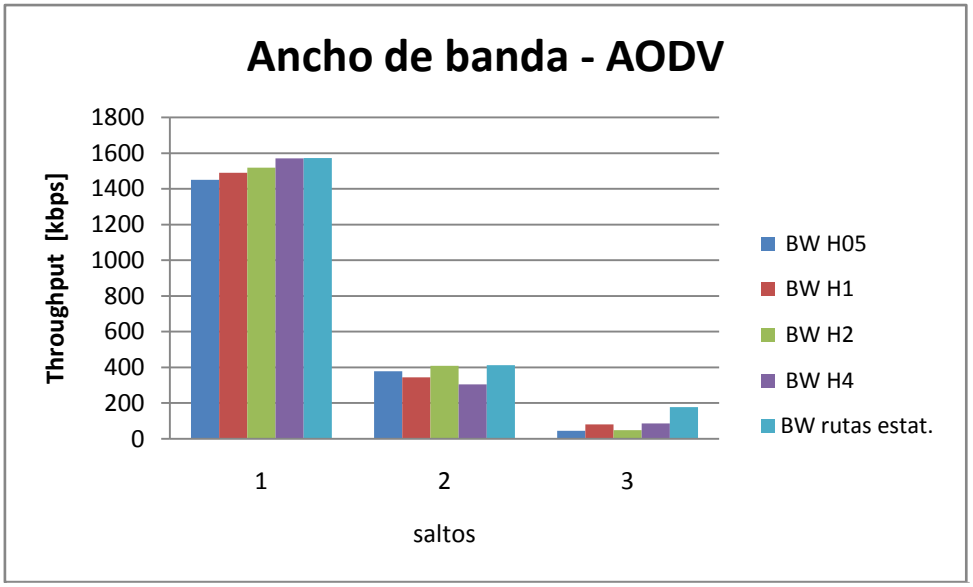

Fig. 3.12 Gráfica del throughput en AODV

3.3.3.3 Comparación de resultados

La Figura 3.13 muestra una captura de ejemplo donde hay el intervalo de tamaños que pueden tener los diferentes paquetes que envían los protocolos estudiados. Los tamaños de los paquetes de HELLO y TC de OLSR son mayores, de aquí que el consumo se ancho de banda sea mayor que con AODV.

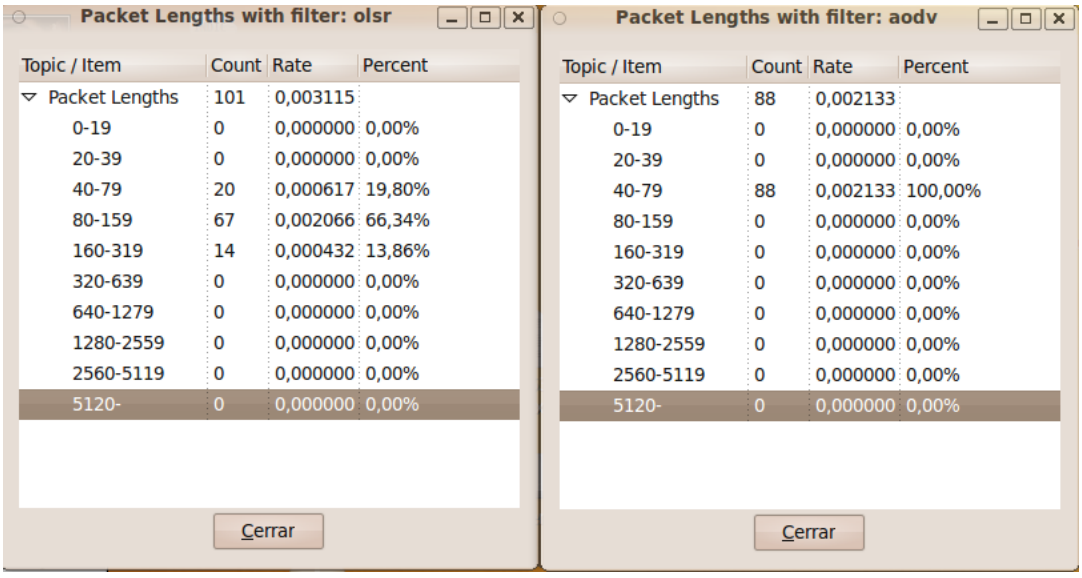

Fig. 3.13 Tamaño paquetes OLSR y AODV

AODV en cambio, manda en sus mensajes de HELLO el tiempo de validez de ruta, sin mandar más información de la red puesto que no ha recibido ninguna solicitud.

La tabla 3.14 muestra los resultados obtenidos con las configuraciones recomendadas por sus respectivos RFC.

	OLSR BW H2	AODV BW H1	BW rutas estat.
1 salto	978,347	1489,873	1571,898
2 saltos	274,2	344,885	411,584
3 saltos	63,12	81,336	177,352

** Resultados dados en milisegundos.*

Tabla 3.14 Comparación del cálculo de los throughputs

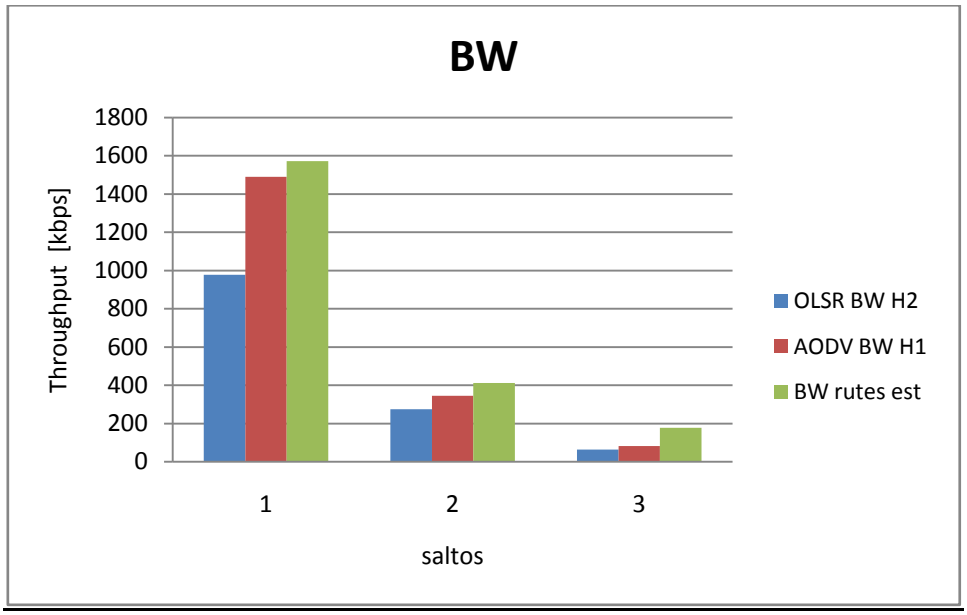

Fig. 3.14 Comparación throughput

Dado los anchos de banda obtenidos, los protocolos reactivos son mejores para comunicaciones de mayor tráfico de datos y estáticos, siempre cumpliendo que el tiempo entre paquetes transmitidos sea mayor que dos veces el tiempo de HELLO configurado para que no se produzca la pérdida de la ruta.

OLSR utiliza más ancho de banda para el control del estado de la red lo que hace que facilita los trabajos de monitorización, ya que todos conocen la topología de la red. Esto permite que haya un nodo con más capacidad de recursos que el resto de nodos en la red ad-hoc, y pueda dedicarse a múltiples tareas como registrar información sobre el estado de las conexiones o permitir el acceso a internet.

3.3.4 Latencia de cambio de ruta (RCL)

Para un protocolo de enrutado es importante solventar lo antes posible la caída de un nodo y evitar en lo mayor posible la pérdida de paquetes. El tiempo o *gap* que se tarda en recuperar la comunicación debe ser el menor posible y es aquí donde entra se encuentra un conflicto con el ancho de banda consumido.

El protocolo OLSR mantiene la tabla de enrutado llena con los caminos posibles que tiene el host para enviar sus paquetes, pero también mantiene una relación de la calidad de cada enlace, por lo que mientras está transmitiendo paquetes, se ocupa de controlar esa característica y en el transcurso de la comunicación, tiene la capacidad de cambiar el camino si lo encuentra necesario, sin esperar a la caída del host.

Esta es una diferencia importante frente al protocolo AODV, el cual mantiene siempre fijo el camino por el que empezó a transmitir. El único motivo por el que cambiaría el destino de salida de sus paquetes sería por la caída de algún host de la red el cual hiciera modificar el camino.

A continuación se mide el RCL con OLSR y AODV, con distintas configuraciones de estos protocolos. Los valores serán los mismos que para el cálculo del ancho de banda. Para ello se emplea la herramienta ping para generar tráfico entre origen y destino y se procede como se ha descrito en el apartado 3.2.2.5

Para tener precisión razonable en la medida, el intervalo entre paquetes enviados es de 10 ms.

3.4.4.1 OLSR

EL protocolo OLSR descubre la ruptura del enlace cuando expira el tiempo configurado 3 * HELLO_INTERVAL. Para poder ver un ejemplo vemos las capturas de cómo se repone el OLSR tras la caída de un nodo intermedio. El valor de HELLO es de 4 segundos y se envían pings cada 10 ms. El tráfico se envía desde el PC 11, el nodo que reenvía los paquetes es el PC 14 y el destino el PC 13. EL PC 12 escucha el canal y envía sus mensajes de HELLO pero no interviene en la comunicación.

El último *ECHO_PING_REPLY* de la Figura 3.15 en el instante *60,003* corresponde al momento en que el PC 14 cae de la comunicación, por lo que el PC 11 sigue mandando paquetes con dirección IP destino 13 pero con la MAC del nodo caído. A partir de este último paquete comienza el gap de conectividad.

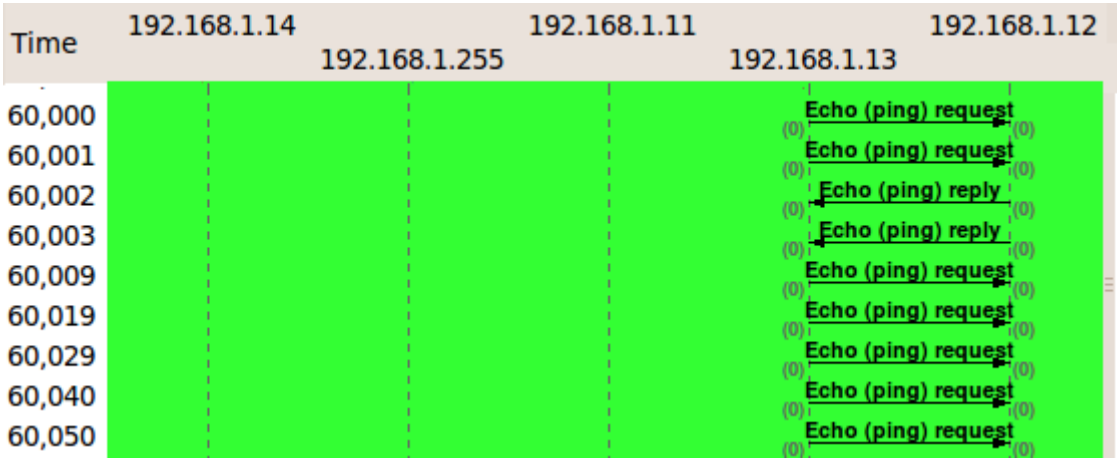

Fig. 3.15 Inicio gap OLSR

El final del gap es en el tiempo *70,014* donde el primer paquete que envía el nodo que ahora reenvía los paquetes por el nuevo camino, en este caso envía una petición de ping a la que el destino responde. Este paquete de respuesta, lleva en su cabecera IP como dirección destino la IP 11, pero la MAC es la del host que ha sustituido al nodo caído.

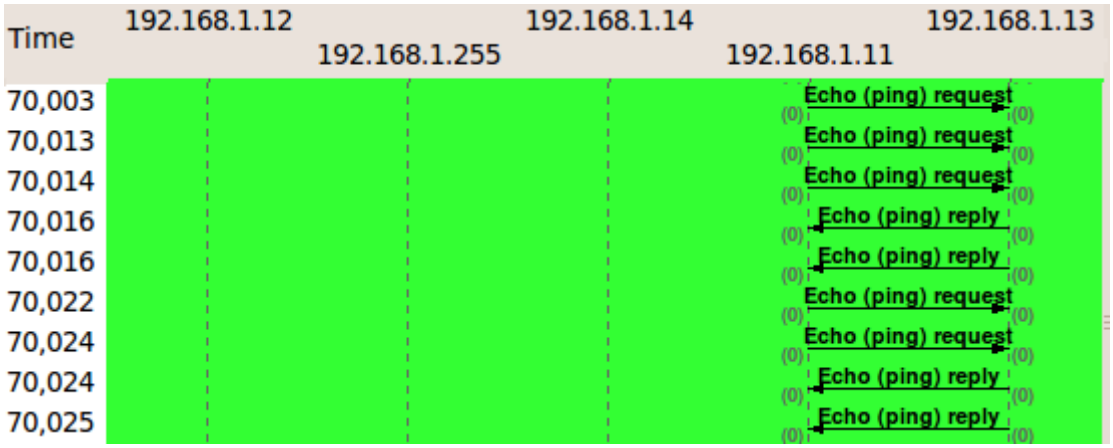

Fig. 3.16 Final gap OLSR

Los resultados son los esperados para este escenario. El gap está dentro de los márgenes que se pueden dar dadas las configuraciones. El gap debe ser mayor a 2 * *HELLO_INTERVAL* y no debe ser mayor a *3 * HELLO_INTERVAL.* Sin embargo, retardos adicionales de proceso pueden contribuir a los resultados obtenidos.

H0,5	H1	H2	H4
1,508	2,972	4,665	6,644

** Resultados dados en segundos.*

Tabla 3.15 Resultados de RCL en OLSR

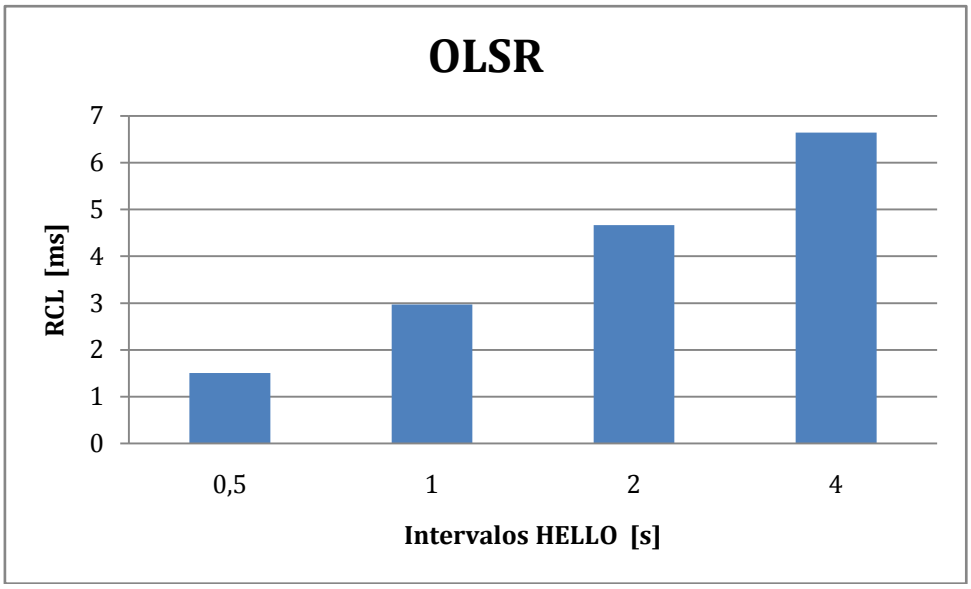

Fig. 3.17 Gráficas de RCL en OLSR

3.4.4.2 AODV

Para la medida de la duración del gap se debe buscar el momento en que el nodo que inicia la transmisión con IP 11, pierde el enlace y solamente envía ECHO_PING_REQUEST con dirección IP 13 durante lo que dura el gap. La finalización del gap se produce cuando el nodo deja de enviar paquetes a la dirección del host caído, y manda un *RREQ* para solicitar el descubrimiento de una nueva ruta y seguir con la transmisión. El tiempo de intervalo de HELLO es de 2 segundos.

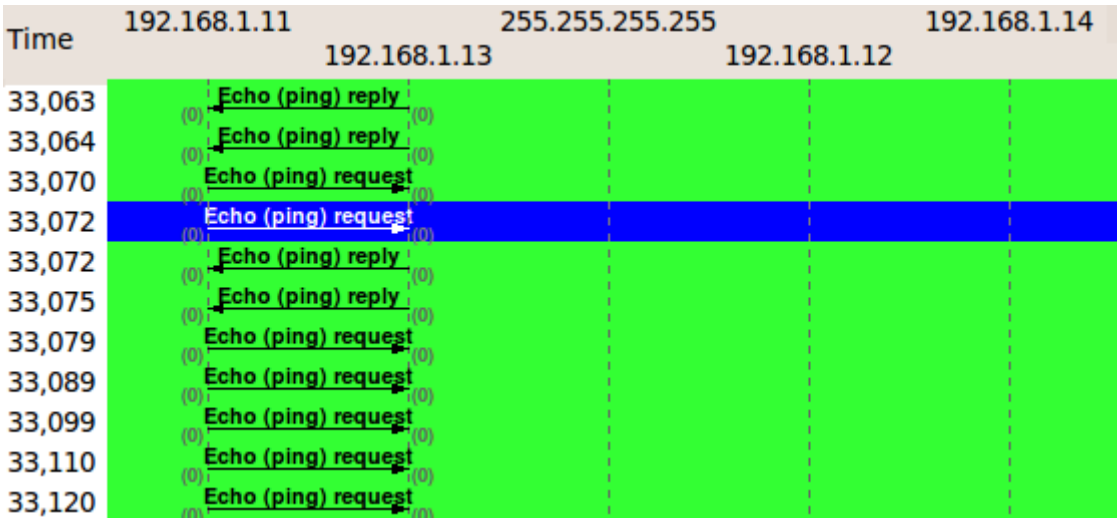

Fig. 3.18 Inicio del gap en AODV

Al acabar el contador que lleva el tiempo de validez de ruta en los nodos vecinos al nodo caído, estos mandan un *RREQ* a la dirección broadcast preguntando que nodo tiene en su tabla de ruta la dirección IP 13 destino del

ping. En este caso, el nodo con IP 11 hace la pregunta en el tiempo 35,218, justo después hay una transmisión por parte del nodo con IP 13 de un HELLO, debido que pasó el tiempo configurado en el HELLO_INTERVAL. En el instante 35,222, el PC 12 envía un *RREQ* ya que no puede responder la petición hecha por el nodo con IP 11 por que no tiene la entrada, y pregunta por ella a la dirección broadcast así podrán responder vecinos de la IP 12 que no sean vecinos de la IP 11. Inmediatamente después, la IP 13, que sí tiene visión directa con el nodo con IP 12, responde con un *ROUTE_REPLY* directamente al PC 12 que él es el destino por el que preguntaba, así que el nodo de IP 12, llena su tabla de rutas del sistema. Ahora sí que conoce el camino de cómo llegar al destino del ping y responder así la primera petición de la IP 11. El nodo con IP 11 llena la tabla de rutas del sistema con la entrada que indica a qué nodo hay que enviar el tráfico dirigido a la IP 13, aquí será la IP 12. A partir de este momento, vemos que la transmisión sigue normalmente.

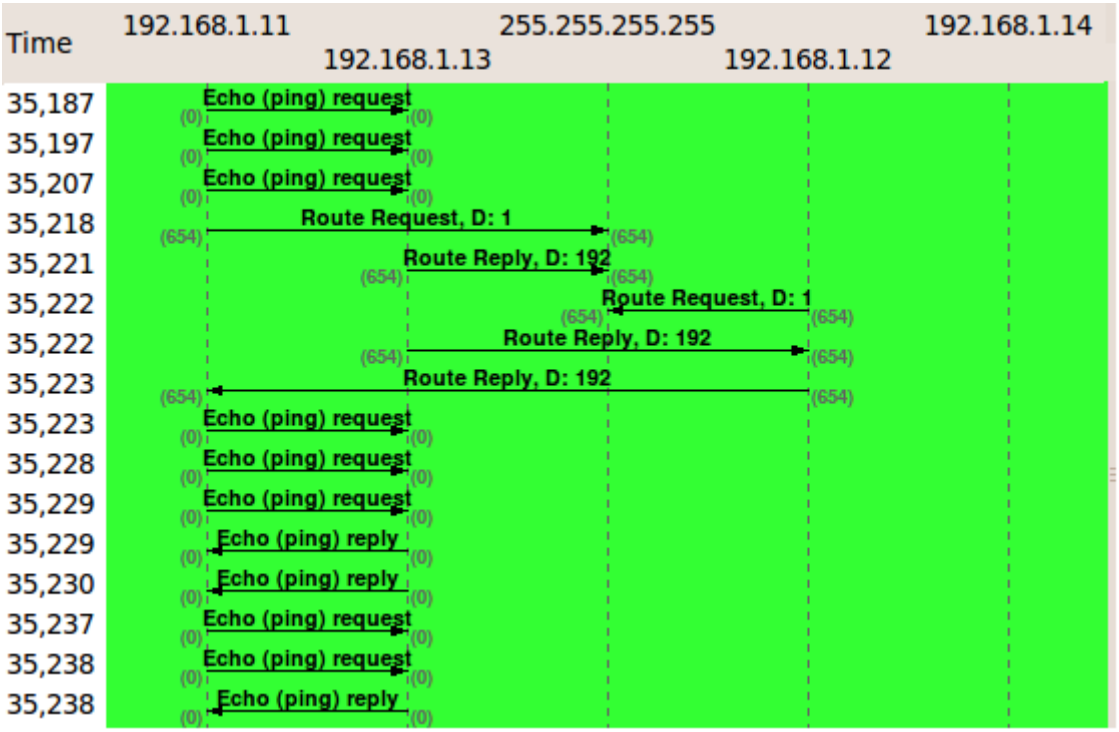

Fig. 3.19 Final gap en AODV

Pero para observar mejor el comportamiento del protocolo, primero tendremos en cuenta cuándo se recibió el último refresco de ruta del nodo que caído. Este paquete se envía en el segundo 31,158 de la captura de la IP 14 a broadcast, para que la reciban todos los vecinos

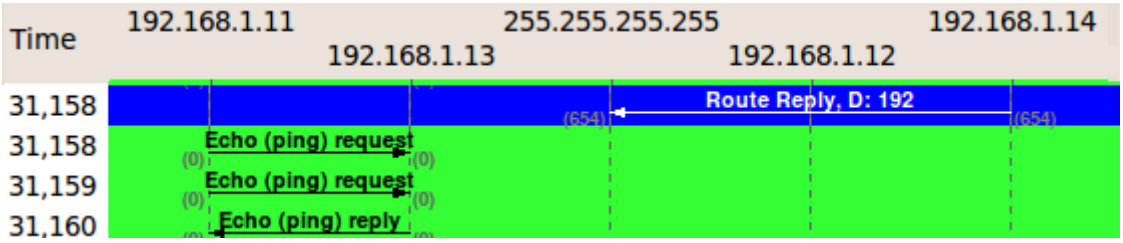

Fig. 3.20 Último paquete HELLO nodo caído en AODV

La diferencia entre el momento en que se envía el último mensaje de HELLO del nodo caído y se envía la petición para encontrar la nueva ruta, debe ser el tiempo configurado en el AODV.

El tiempo máximo de gap en AODV sería:

$$35{,}218 - 31{,}58 = 4{,}06 \text{ s} = 2 * \textit{HELLO\_INTERVAL} \quad \textbf{(3.2)}$$

Los tiempos resultantes son menores que el doble del intervalo de mensajes de HELLO así que son válidos.

H0,25	H0,5	H1	H2
0,460	0,771	1,742	2,253

** Resultados dados en segundos.*

Tabla 3.16 Resultados de RCL en AODV

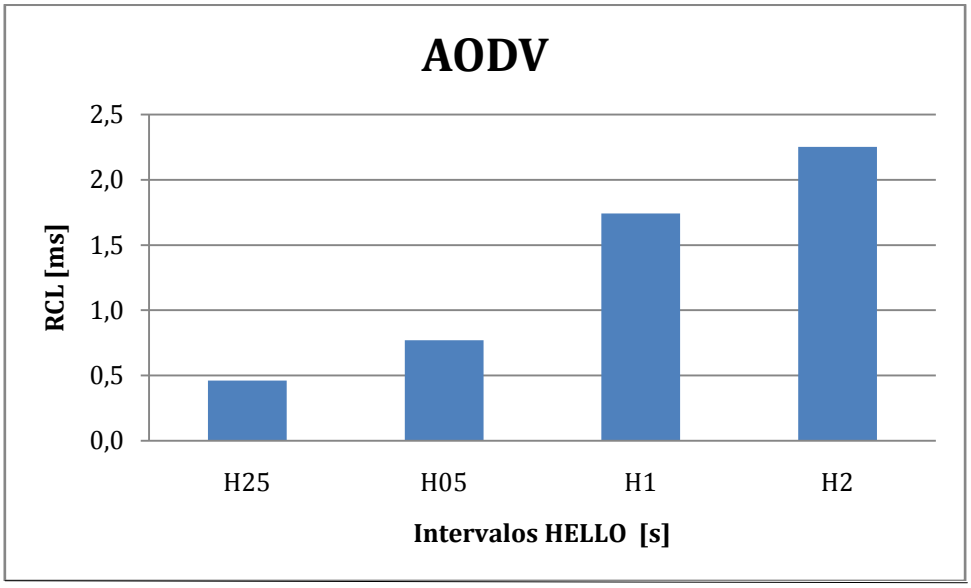

Fig. 3.21 Gráfica RCL AODV

Comparación de resultados

Las difirencias de RCLs entre AODV y OLSR hace valer la principal característica de los protocolos reactivos, y es que el tener menos control sobre la topología de la red, proporciona tiempos mejores dado un escenario estático como este.

Los paquetes que envia el AODV son de menor tamaño que los mensajes de que envia OLSR.

AODV				OLSR			
H0,25	*H,05*	*H1*	*H2*	*H0,5*	*H1*	*H2*	*H4*
0,460	0,771	1,742	2,253	1,575	2,972	4,665	6,644

** Resultados dados en segundos*

Tabla 3.17 Comparación del gap AODV-OLSR

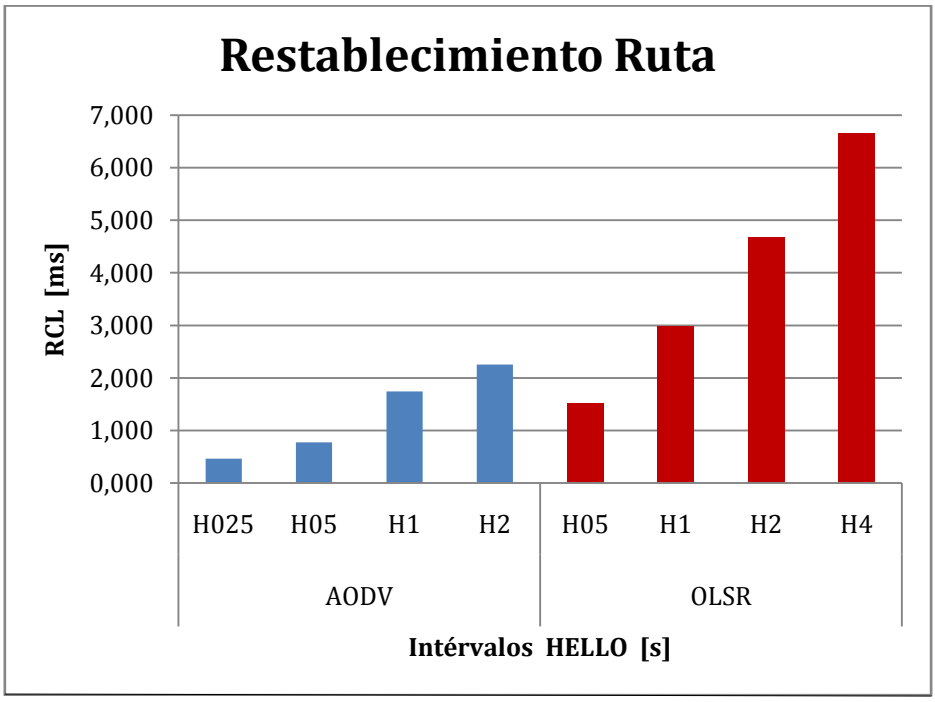

Fig. 3.22 Gráfica gap AODV-OLSR

Si comparamos los valores obtenidos de cada protocolo con el mismo intervalo de HELLO, vemos que la duración del gap es menor. Así que con las mismas configuraciones OLSR será menos eficiente que AODV.

La configuración del tiempo de validez es diferente en cada protocolo. Mientras AODV da por perdida una ruta cuando deja de recibir dos mensajes de HELLO, el protocolo OLSR tiene un tiempo de validez de tres mensajes.

CAPÍTULO 4. CONCLUSIONES Y LINEAS FUTURAS

4.1 Conclusiones

Las redes ad-hoc posibilitan que un conjunto de nodos móviles inalámbricos puedan comunicarse sin la necesidad de que haya cobertura entre todos ellos. Con las redes ad-hoc, se puede crear una red sin infraestructura. Si en estas redes fuera necesario ampliar servicios, como acceso a internet, basta con que un componente de la red tenga acceso a Internet para que el resto de nodos dispongan también de este servicio. Los protocolos de encaminamiento deben proporcionar a la red la capacidad de ser auto-reparable, dada su dinamicidad, y deben detectar problemas de enrutamiento como pérdidas de comunicación con nodos vecinos o caída de calidad de los enlaces, y actuar para corregir el error y que la transmisión se vea lo menos afectada posible.

En este proyecto se ha desarrollado una maqueta de pruebas de redes ad-hoc con finalidades docentes. Se han realizado experimentos con implementaciones reales de los protocolos de encaminamiento para este tipo de redes más populares: OLSR y AODV. Estos protocolos ofrecen diferentes prestaciones. Esto hace que el uso de cada uno venga determinado por las características del escenario en el que se implemente y del volumen de tráfico que curse la red..

En ambos protocolos, el mecanismo de detección de caídas de enlaces es el elemento que determina el tiempo que se tarda en restablecer una ruta. Por defecto, OLSR requiere mayor tiempo para detectar la caída de un enlace, y genera más tráfico de control (consumiendo mayor ancho de banda). Sin embargo, los parámetros de este protocolo (y de AODV) pueden configurarse de forma distinta a la propuesta por defecto en la especificación correspondiente.

Por otra parte, se ha observado que la implementación de AODV utilizada no se comporta de forma acorde a lo que cabría esperar. En particular, el nodo origen tarda un tiempo en hacer efectiva la información de encaminamiento obtenida mediante el proceso de descubrimiento de rutas, lo cual resulta en la pérdida de los paquetes iniciales de la comunicación.

La preparación de los escenarios para poder realizar el estudio de los protocolos, ha pasado por diferentes fases.

El hardware del que se dispone son los que dispone la universidad en el laboratorio de telemática, con sistema operativo estaba instalado. Solo se tuvo que visitar diferentes proveedores de material los cuales ofrecían diferentes soluciones. La elección estaba ligada a una instalación de drivers nuevos. Aquí hay poca variedad ya que son drivers específicos. Gracias al software libre, se instaló sin problemas una imagen de Ubuntu que contenía los drivers. El usuario que inicie el ordenador debe seleccionar el arranque de la imagen y

configurar la tarjeta desde el terminal. Las comandas para realizar la práctica no son complejas ni vulneran la seguridad del sistema.

Por otra parte, se ha creado la documentación necesaria para construir la maqueta y utilizarla.

4.2 Líneas futuras

Actualmente existen muchas implementaciones de protocolos de encaminamiento y continúan apareciendo nuevos protocolos, o mejoras de las implementaciones existentes. Esto demuestra que se está apostando por esta rama de las comunicaciones en redes ad-hoc.

Esta tecnología es aplicable en muchos escenarios y para cubrir diferentes necesidades. Como gran ejemplo de expansión de una red mesh tenemos el proyecto *Guifi.net*. Red neutral y abierta creada por los propios usuarios.

Las principales líneas de trabajo futuro consisten en evaluar otras implementaciones de protocolos de encaminamiento, tanto de OLSR y AODV, como de otros protocolos, p.ej. DYMO.

BIBLIOGRAFÍA

[1] C. Perkins, E.M.Belding-Royer, S. Das ”Ad hoc On-Demand Distance Vector (AODV) Routing” RFC 3561, 2003

[2] E. Belding-Royer, C. Perkins, “Evolution and future directions of the ad hoc on-demand distance-vector routing protocol”, 2003.

[3] C. Gómez, J. Paradells “Redes ad-hoc: el próximo reto”, 2003

[4] J. Liu, S. Singh “ATCP: TCP for Mobile Ad-Hoc Networks”, 2003

[5] L, Klein-Berndt “A quick guide to AODV routing”, 2003

[6] T. Clausen, P.Jacquet “Optimized Link State Routing Protocol (OLSR)” RFC 3626, 2003

[7] P. Sarolahti, A. Kuznetsov “Congestion control in Linux TCP”, 2002

[8] A. Kesselman, Y. Mansour “Optimizing TCP retransmission timeout”, 2004

[9] C. K. Toh, “Ad Hoc Mobile Wireless Networks: protocols and systems”, Prentice-Hall, 2002

[10] P. Misra “Routing Protocols for Ad Hoc Mobile Wireless Networks”, 2000

ANEXOS

Manuals de configuració

Prerequisits

Inici

1. Abans d'arrancar el PC, hem de desconnectar el cable de xarxa de la interfície per evitar que arranqui el gestor REMBO.
2. Arranquem el sistema.
3. En el menú del GRUB hem de seleccionar la imatge 2.6.32-wl.
4. En el menú d'inici entrarem amb l'usuari *ubuntu*
5. Obrirem un terminal

Comandes prèvies

Abans de res, i per evitar-nos problemes de permisos, introduïm el password de root.

$> sudo -s

Ens assegurem que som a la imatge d' *Ubuntu 2.6.32-wl* on hi han els drivers de la tarja.

$> uname -r

Si som en un altra imatge haurem de reiniciar i arrencar en la imatge correcta.

Llistem les interfícies que té configurat l' equip. Hem de trobar una wlanX, on X és el número que li dona el sistema a la interfície wireless.

$> ifconfig -a

Configuració

Primer configurem la tarja de l'equip per a que treballi en mode ad-hoc i estigui en la mateixa xarxa que la resta.

L'ordre de configuració/creació dels equips dins de la xarxa no importa.

Comprovem els canals lliures que tenim per la interfície wireless

$> iwlist wlanX scan

Parem el gestor de connexions que té per defecte l' ubuntu:

$> service network-manager stop

Configurem el mode de treball, el nom que tindrà la xarxa (essid), el número del canal i la velocitat de treball de la tarja.

$> iwconfig wlanX mode ad-hoc essid NOM_XARXA channel NUM_CANAL rate 11M

Configurem la adreça IP de la tarja.

$> ifconfig wlanX ADREÇA_IP/MASCARA

OLSR

Configuració

El dimoni té un arxiu on podem configurar el temps de HELLO, el seu temps de validesa, el temps de TC i també el seu temps de validesa. Això es troba en l'arxiu /etc/olsrd.conf.

Accedim a la carpeta on es troba l'arxiu de configuració del dimoni

$> cd /etc

Editem l'arxiu per canviar els paràmetres. Hem de tenir en comte que el número que posem seran els segons que configuren per aquest paràmetre (sempre han d'anar amb una decimal sinó quan arranqui el dimoni donarà error).

$> nano olsrd.conf

Extracte de l'arxiu

```
DebugLevel    1
...
Interface "wlanX"

{ ...
    # Emission intervals.
   # If not defined, RFC proposed values will
   # be used in most cases.

   # Hello interval in seconds(float)
   HelloInterval                 4.0
```

```
    # HELLO validity time
    HelloValidityTime           12.0

    # TC interval in seconds(float)
    TcInterval                  10.0


    # TC validity time
    TcValidityTime              30.0

    # MID interval in seconds(float)
    MidInterval                 10.0

    # MID validity time
    MidValidityTime             30.0

    # HNA interval in seconds(float)
    HnaInterval                 10.0

    # HNA validity time
    HnaValidityTime             30.0

...}

...

Hna4

{
#   Internet gateway:
#   0.0.0.0      0.0.0.0
#   more entries can be added:
   192.168.1.0  255.255.255.0
}
```

. Interface "wlan0" : *-- Especificació de l'interfície on enviarà missatges el protocol.*
. HelloInterval *-- Temps en segons del HELLO_INTERVAL*
. HelloValidityTime *-- Temps en segons de validesa del HELLO_INTERVAL.*
. TcInterval *-- Temps en segons del TC_INTERVAL*
. TcValidityTime *-- Temps en segons de validesa del TC_INTERVAL*
. Hna4 *-- Rang d'IP's que formaran part de la xarxa ad-hoc*

Desplegament del dimoni

Finalment, aixequem el dimoni olsrd per a fer córrer el protocol.

$> olsrd

AODV

Configuració

Entrem en el directori on tenim el paquet d'instal·lació d' AODV (en el nostre cas, en la home d' Ubuntu):

$> cd /home/ubuntu/aodv

Descomprimim el paquet

$> tar –zxvf aodv-uu-0.9.6.tar

Accedim a la carpeta d' AODV que s'hi genera

$> cd aodv-uu-0.9.6

Configurem els paràmetres necessaris de l' AODV

$> nano params.h

Extracte de l'arxiu

```
/* Settings for Link Layer Feedback */
#define ACTIVE_ROUTE_TIMEOUT_LLF    10000
#define TTL_START_LLF               1
#define DELETE_PERIOD_LLF           ACTIVE_ROUTE_TIMEOUT_LLF

/* Settings for HELLO messages */
#define ACTIVE_ROUTE_TIMEOUT_HELLO  3000
#define TTL_START_HELLO             2
#define DELETE_PERIOD_HELLO                                    K *
max(ACTIVE_ROUTE_TIMEOUT_HELLO,          ALLOWED_HELLO_LOSS        *
HELLO_INTERVAL)
{...}
```

Compilem el paquet AODV

$> make
$> make install

Desplegament del dimoni

Amb la següent comanda tindrem ja el daemon d' AODV *(aodvd)* executant-se a la màquina, i ja comença a enviar missatges per la seva interfície wireless.

$> aodvd –l –r 1

On distingim els atributs –l per guardar els logs del dimoni en l'arxiu *aodvd.log,* i el –r servirà per guardar la taula de route del dimoni en l'arxiu *aodvd.rtlog.*