Escola Politècnica Superior
de Castelldefels
UNIVERSITAT POLITÈCNICA DE CATALUNYA

# TRABAJO FINAL DE CARRERA

**TÍTULO: Estudio del consumo de baterías en dispositivos móviles IEEE802.11: técnicas de ahorro, caracterización y evaluación**

**TITULACIÓN: Ingeniería Técnica de Telecomunicaciones, especialidad Telemática.**

**AUTOR: Joshua Sanz de la Rica Mann**

**DIRECTOR: Rafael Vidal**

**FECHA: 19 de Junio de 2008**

**Título del TFC: Estudio del consumo de baterías en dispositivos móviles IEEE802.11: técnicas de ahorro, caracterización y evaluación**

**Titulación: Ingeniería Técnica de Telecomunicaciones, especialidad Telemática.**

**Autor: Joshua Sanz de la Rica Mann**

**Director: Rafael Vidal**

**Fecha: 19 de Junio de 2008**

**Resumen**

La conectividad inalámbrica crece a un ritmo vertiginoso, no obstante, aunque los móviles puedan estar en modalidad de espera y que sus baterías tarden días en agotarse, no sucede lo mismo con los dispositivos que trabajan con el estándar IEEE802.11. Su elevado consumo de batería es uno de los mayores problemas de ésta tecnología ya que limita la autonomía de los dispositivos móviles que la utilizan (teléfonos, portátiles, consolas, cámaras,…). Por ello, es necesario introducir mecanismos que nos permitan el ahorro de batería y que esto se consiga sin necesidad de sufrir los inconvenientes de la pérdida de calidad de servicio (QoS) del mecanismo *Power Save Mode* (PSM) que define el estándar IEEE802.11.

Uno de estos mecanismos, quizás el más importante, es el *Automatic Power Save Delivery* (APSD) correspondiente al estándar IEEE802.11e. Este estándar extiende la capa MAC de IEEE802.11 con capacidades de QoS y de ahorro de baterías. En este TFC se introducirá la extensión, haciendo especial énfasis en cómo consigue aunar QoS y ahorro de batería y se comparará con PSM tanto en términos teóricos como prácticos.

La recepción de tráfico broadcast/multicast afecta también al consumo. En este TFC se estudiará soluciones para reducir su impacto. Una de ellas, es el efecto que tiene sobre el consumo el parámetro DTIM *(Delivery Traffic Indication Message*. La suplantación del AP que hace de proxy para contestar los mensajes ARP Request en lugar de la STA y el descarte por parte del AP de los ARPs que no tienen ningún cliente del AP como destino, son otras de las funcionalidades estudiadas.

Con respecto a la movilidad, se ha puesto en marcha también una maqueta con soporte del protocolo Mobile IPv6 (MIPv6) que permite el traspaso de nodos IPv6/IEEE802.11 entre diferentes redes sin la necesidad de que estos se reconfiguren y pierdan la comunicación. El propósito será analizar el consumo producido por el traspaso a nivel de red (i.e. MIPv6) y el traspaso a nivel de enlace (i.e. IEEE802.11) para estudiar los factores de mayor impacto en el consumo y poder ver que partes son susceptibles a ser mejoradas. Se ha elaborado una aplicación en lenguaje Visual Basic Aplication para su análisis.

**Title: Research of battery consumpiton in mobile devices IEEE802.11: saving techniques, characterization and evaluation**

**Author: Joshua Sanz de la Rica**

**Director: Rafael Vidal**

**Date: June, 19th 2008**

## Overview

The wireless connectivity is growing very fast. Although cell phones can be in standby mode and their batteries take days to be used up, the same thing does not occur with devices working on the IEEE802.11 standard. The high battery consumption is one of the biggest problems of this technology because it limits the autonomy of wireless devices used in conjunction (cell phones, laptops, videogames, cameras…). For that reason, it is necessary to introduce mechanisms that can give us greater possibilities to save the batteries without the loss of Quality of Service (QoS) that Power Save Mode (PSM) causes in standard IEEE802.11.

One of these mechanisms, possibly the most important, is the Automatic Power Save Delivery (APSD) which corresponds to the standard IEEE802.11e. This standard extends the MAC layer of IEEE802.11 with QoS and battery saving capabilities. The uAPSD protocol will be compared to PSM through practical and theoretical means.

The reception of broadcast/multicast traffic also drains the battery. In this project, some solutions are going to be studied in order to reduce its impact. One of them is the effect that DTIM (Delivery Traffic Indication Message) has on battery consumption. An AP substitution acting as a proxy in order to acknowledge the ARP Requests instead of the STA, and the omission that the AP carries out on the ARPs that do not have any AP client as a destination, are other functionalities that have been studied.

With respect to the mobility, a test ground has also been set up with Mobile IPv6 (MIPv6) protocol support that allows a handover between IPv6/IEEE802.11 nodes within different networks without the nodes having to be reconfigured resulting in the loss of communication. The purpose will be to analyze handover consumption at an IP layer (MIPv6) and at a link layer (IEEE802.11) in order to study the factors that have a bigger impact on battery consumption and be able to see which parts are susceptible to improvement. A Visual Basic Application has been programmed to analyze these handovers

En primer lugar, me gustaría dar las gracias a mi director, Rafael Vidal, por todo el apoyo que me ha prestado durante el transcurso del proyecto. Siempre que he tenido dificultades ha sabido darme una solución o una alternativa.

En segundo lugar, me gustaría dedicar éste TFC a mis buenos compañeros y amigos que han amenizado las largas y a veces duras horas en el laboratorio. Estos son: Carlos Barrera, Raúl Giménez y José Luís Rodríguez.

# ÍNDICE

# INTRODUCCIÓN

La conectividad inalámbrica crece a un ritmo vertiginoso, no obstante, aunque los móviles puedan estar en modalidad de espera y que sus baterías tarden días en agotarse, no sucede lo mismo con los dispositivos que trabajan con el estándar IEEE802.11. Su elevado consumo de batería es uno de los mayores problemas de esta tecnología ya que limita la autonomía de los dispositivos móviles que la utilizan (teléfonos, portátiles, consolas, cámaras,…). Por ello, es necesario introducir mecanismos que nos permitan el ahorro de batería y que esto se consiga sin necesidad de sufrir los inconvenientes relativos a la pérdida de calidad de servicio (QoS) que implica el mecanismo *Power Save Mode* (PSM) que define el estándar IEEE802.11.

Uno de estos mecanismos, quizás el más importante, es el *Automatic Power Save Delivery* (APSD) correspondiente al estándar IEEE 802.11e. Este estándar extiende la capa MAC de IEEE802.11 con capacidades de QoS y de ahorro de baterías. En este TFC se introducirá la extensión, haciendo especial énfasis en cómo consigue aunar QoS y ahorro de batería y se comparará con PSM tanto en términos teóricos como prácticos. Esta extensión sin embargo no mejora otras causas de consumo de batería inherentes al funcionamiento de la red: la recepción de tráfico *broadcast* o *multicast* y la movilidad de los dispositivos.

Para el caso de la recepción del tráfico broadcast/multicast, en este TFC se estudiará soluciones para reducir su impacto. Una de ellas, es el efecto que tiene sobre el consumo el parámetro DTIM *(Delivery Traffic Indication Message)* que indica cada cuanto tiempo un dispositivo se despierta para recibir tráfico *broadcast/multicast* utilizando PSM. La suplantación del AP que haciendo de proxy para contestar los mensajes ARP *Request* en lugar de la STA y el descarte por parte del AP de los ARPs que no tienen ningún cliente del AP como destino, son otras de las funcionalidades estudiadas. Se configurarán los parámetros necesarios tanto en la tarjeta como en el AP para que por medio de distintas pruebas estudiar el ahorro en la batería que supone el uso de estas funcionalidades.

Respecto a la movilidad, se ha puesto en marcha también una maqueta con soporte del protocolo Mobile IPv6 (MIPv6), que permite el traspaso de nodos IPv6/IEEE802.11 entre diferentes redes sin la necesidad de que estos se reconfiguren y pierdan la comunicación. Este protocolo se ha diseñado para poder trasladarse entre diferentes tecnologías de forma transparente gracias a que el soporte del traspaso se hace a nivel IP. El propósito será analizar el consumo producido por el traspaso a nivel de red (i.e. MIPv6) y el traspaso a nivel de enlace (i.e. IEEE802.11) para estudiar los factores de mayor impacto en el consumo y poder ver que partes son susceptibles a ser mejoradas. Para ello, se ha creado una aplicación Excel en lenguaje VBA (*Visual Basic Aplication*) que permite mediante la introducción de los valores de consumo de una tarjeta, la extracción de los consumos producidos por los traspasos citados anteriormente de forma gráfica y numérica.

En resumen, el objetivo de este TFC ha sido realizar un estudio de los factores y funcionalidades que tienen relación directa con el consumo de batería. Por una parte, el estudio de protocolos y parámetros que contribuyen a la mejora de la duración de la batería y su completo entendimiento para poder realizar las pruebas empíricas. Por otra parte, el estudio del consumo debido a la movilidad para analizar que partes tienen mayor impacto, y por tanto son más susceptibles de ser mejoradas. Se ha tenido que entender el protocolo, y conocer la implementación necesaria para poder poner en marcha la maqueta y realizar los cálculos de consumo en el traspaso MIPv6 y IEEE802.11. Como complemento de este segundo objetivo, se ha tendido que aprender lenguaje *Visual Basic Aplication* para realizar una aplicación que automatice estos cálculos y presente los resultados de forma gráfica.

La memoria se estructura de la siguiente forma. En el primer capítulo se estudiará el funcionamiento de PSM y su problemática como paso previo a la explicación del mecanismo uAPSD (*unscheduled Automatic Power Save Delivery*), que incorpora la certificación WMM-PS de la Wi-Fi Alliance con el fin de mejorar las prestaciones de PSM.

En el segundo capítulo se introducirán brevemente las soluciones que se han comentado para el ahorro de batería y se estudiará su impacto a partir de pruebas prácticas. El tercer capítulo describe el funcionamiento del protocolo MIPv6 y de IEEE802.11 para el caso del traspaso, para luego estimar el consumo de batería derivado de la realización de traspasos IEEE802.11 y MIPv6.

Por último, en el capítulo cuarto se extraen las conclusiones a las que se ha llegado con este estudio y su impacto medioambiental. Además se propondrán algunas líneas a seguir para complementar este TFC.

Esta memoria también contiene un conjunto de anexos que complementan la información contenida en los diferentes capítulos y que oportunamente serán citados. En el Anexo VII se presenta un listado de acrónimos para consultar si es necesario.

# CAPÍTULO 1. Certificación WMM-PS: uAPSD

IEEE 802.11 está basado en CSMA/CA, que obliga a que las estaciones estén escuchando el canal continuamente provocando un gasto de batería muy grande. PSM permite ahorrar baterías a las estaciones (STA) en redes Wi-Fi. No obstante, es un gran impedimento para los servicios VoIP y *video streaming* actualmente emergentes ya que incrementa el retardo en la entrega de las tramas. En este capítulo se estudiará el funcionamiento de PSM y su problemática como paso previo a la explicación del mecanismo uAPSD (*unscheduled Automatic Power Save Delivery*), que incorpora la certificación WMM-PS de la Wi-Fi Alliance con el fin de mejorar las prestaciones de PSM. Las explicaciones teóricas se complementarán con otras prácticas, centradas en como implementa estas funcionalidades el AP Cisco 1130AGN utilizado en este TFC.

## 1.1. IEEE802.11 PSM

Con el objetivo de reducir el consumo de energía de la STA el estándar IEEE 802.11 incluye el modo *Power Save* (PSM) que hace que la STA permanezca en un estado de muy bajo consumo durante los períodos de inactividad. Los paquetes destinados a la STA durante estos periodos se almacenan en el *buffer* del AP hasta que son solicitados.

En la Figura 1.1 podemos apreciar un ejemplo de la operación. Las STAs que utilizan PSM se despiertan cada *n beacons* lo que se denomina el *Listen Interva*l (LI), para ver si el AP tiene tramas para ellas. Para indicar esta situación el AP utiliza el campo de *traffic indication map* (TIM) contenido en el *beacon*. El AP distinguirá entre estaciones utilizando el bit *association identificator* (AID) contenido en el TIM. Para que la STA pueda reclamar sus tramas, debe de enviar una trama *Power Save Poll* (PS-Poll) por cada trama almacenada en el *buffer*. La última trama entregada tendrá el campo *More Data* (MD) a 0 para indicar que ya no hay más información. No obstante, la estación no deberá esperar ningún *beacon* para poder transmitir en el canal de subida. Todo este proceso provoca un *overhead* bastante considerable y no garantiza la QoS requerida por algunas aplicaciones.

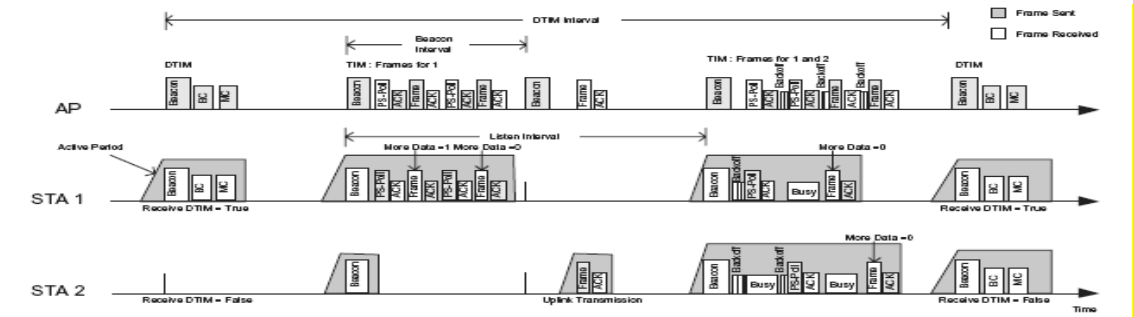


**Fig. 1.1** Ejemplo de operación de 802.11 *power save mode*

PSM ahorra mucha batería haciendo que el dispositivo de red desconecte el componente radio de la STA pasando a un estado durmiente en el que no se puede no enviar ni recibir tramas. La transición entre este estado durmiente y el activo es controlada por la capa MAC. La tecnología ha permitido llegar a valores de consumo de hasta 9mA en el estado durmiente [5] [16]. El problema es que el protocolo MAC no siempre explota el modo durmiente de una manera eficiente. Principalmente por *overhead* que introduce PSM y el retardo intrínseco que produce en la entrega de tramas (cada LI). Esta situación ha derivado en la propuesta de varios métodos para conseguir un manejo optimizado de los estados de la STA que no implican excesivos cambios en la capa MAC o la capa física [16]. A continuación se comentan algunos de ellos.

El primer estudio a comentar se centra en el aumento que produce PSM en el *TCP Round Trip Time* (RTT). Los autores proponen un mecanismo llamado *Bounded Slowdown* que evita que el RTT supere un baremo específico. El STA adapta sus cambios al estado activo independientemente del intervalo *beacon*. El método propuesto funciona con flujos TCP y necesita ciertas extensiones para aplicaciones multimedia basadas en UDP.

Otro estudio propone un mecanismo llamado *Priority Based Scheduling* (PBS) que además de ahorrar energía, proporciona QoS. Este método define un programa en el AP capaz de decidir la entrega o suspensión de un flujo de bajada (i.e. dirigido a las STAs) teniendo en cuanta un *threshold* de datos predefinido en las especificaciones QoS. Los datos que se sirvan seguirán una prioridad basada en el retardo requerido por el mecanismo. El problema de este mecanismo es la falta de robustez cuando existen transmisiones erróneas dado que los periodos en los que el STA permanece despierto no se negocian con el AP.

Otra solución es la *Adaptative Bwer Save Mode* que se basa en la estimación del tiempo entre tramas recibidas para adaptar el intervalo entre PS-Polls y de esta forma asemejar el retardo de bajada de la MAC al intervalo de llegada de tramas. El problema es que es algo conservador porque el intervalo interno de llegada de tramas es muy pequeño comparado con el retardo máximo permitido por algunas aplicaciones.

Todas las limitaciones expuestas para estas soluciones han sido superadas gracias a la introducción por parte del estándar IEEE802.11e de APSD. El estándar IEEE802.11e es una extensión del IEEE802.11 que da soporte a la capa MAC definiendo mecanismos que proporcionan diferenciación de QoS. Para ellos se definen dos métodos de acceso al canal, el *Hybrid Coordination Function* (HCF) y el *Enhanced Distributed Channel Access* (EDCA).

Paralelamente, la *Wi-Fi Aliance*, que es una asociación global de más de 200 empresas dedicadas al desarrollo de WLANs basadas en IEEE802.11 y que certifica los productos y los protocolos soportados, ha creado Wi-Fi Multimedia (WMM) como programa de certificación del mecanismo de diferenciación de QoS de 802.11e basado en EDCA. Y posteriormente la certificación WMM *power save* (WMM-PS) como una extensión de WMM que permite el ahorro de

energía gracias al protocolo uAPSD. En el siguiente apartado detallaremos los aspectos técnicos de este mecanismo destacando las mejoras sobre PSM.

## 1.2. uAPSD vs PSM

La introducción del APSD no desbanca el PSM sino que se instala encima reutilizando lo sin alterar su funcionalidad. APSD introduce el *Service Period* (SP) que puede ser "no programado" (*unscheduled*) y "programado" (*scheduled*). El primero (uAPSD) utiliza únicamente el mecanismo de acceso EDCA que opera en el *contention period* (CP), y al estar certificado por Wi-Fi WMM, es el más utilizado. El scheduled SP (sAPSD) por el contrario, puede utilizar EDCA y HCCA, que está basado en el *Hybrid Coordination Function* (HCF) operando además de en el CP, en el *contention-free period* (CFP). Este último todavía está en fases de investigación.

La idea básica de uAPSD es la utilización de las tramas de subida (STA→ AP) como indicadores (*trigger*) de los instantes en que la STA está despierta y así poder vaciar el buffer completando de manera más efectiva las comunicaciones bidireccionales. En el apartado 1.4 se explica este proceso en detalle. La diferencia principal con PSM, es que con uAPSD una STA permanece despierta durante un SP mientras que con PSM lo hace durante el transcurso de la espera del *beacon* hasta la confirmación de la recepción de la última trama mediante (PS-Polls). Como ya se ha dicho, uAPSD utiliza EDCA. Este mecanismo tiene cuatro categorías de acceso (AC), (AC_VO, AC_VI, AC_BE, AC_BK) correspondientes a voz, video, *best effort* y *background*. Cada uno de los accesos puede configurarse por separado para activar el *delibery-enabled*, estado gracias al cual el buffer puede enviar las tramas almacenadas en el transcurso de un *unscheduled* SP utilizando el mecanismo EDCA.

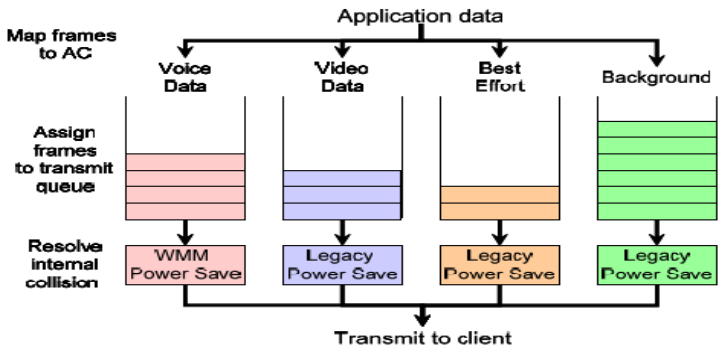


**Fig. 1.2** Categorías de acceso EDCA (ACs)

Hay dos maneras para que se active un *unscheduled* SP:

- Que una STA con configuración *delivery-enabled* envíe al AP tramas a QoS Data también llamadas *trigger* (contienen información) o QoS Null (sin información). Una vez establecido el SP, el AP podrá aprovechar

para enviar a la STA las tramas almacenadas en el buffer y así poder completar una comunicación bidireccional.

- Que un algoritmo interno de uAPSD crea conveniente que la STA se despierte para completar los requisitos de QoS en el canal de bajada.

El SP acaba cuando la STA recibe una trama *QoS Data* o *QoS Null* indicando el final del servicio. Mientras el sub-campo *More Data* (MD) siga activado, querrá decir que aún hay más datos para recibir. Además, cada una de las tramas tendrá otro sub-campo EOSP que estará a 0 mientras no se reciba la última trama del SP, esta última marcará el bit EOSP a 1. Existe un máximo de tramas permitidas en el SP correspondiente al campo *Max_SP_lenght* que se establece en la asociación con el AP. En caso de sobrepasar este máximo, si el MD sigue a 1 (más tramas por recibir), se necesitará abrir otro SP. (Ver Figura 1.3)

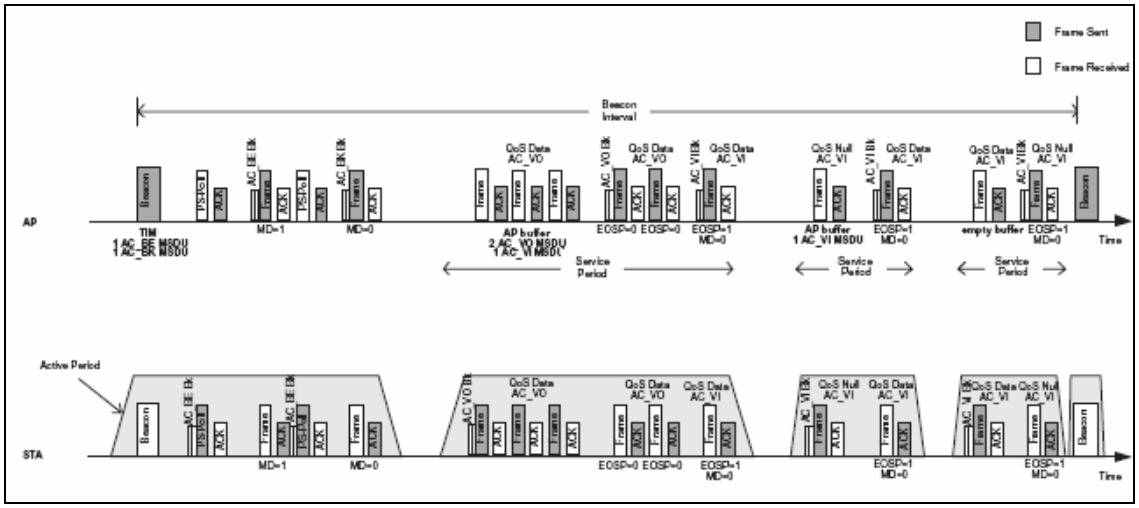


**Fig. 1.3** Ejemplo de operación uAPSD

Una vez explicado los dos mecanismos, PSM y uAPSD, se pueden apreciar tres ventajas significativas de uAPSD respecto a PSM.

1. La capacidad de proporcionar distintos tratamientos de QoS a cada AC según la prioridad de la información generando un flujo de datos en cualquier momento sin necesidad de esperar a que la STA se despierte por su cuenta.

2. La reducción del *overhead* gracias a la utilización de tramas de datos como indicadores para activar el SP, mejorando el rendimiento de las aplicaciones simétricas.

3. También reducción del *overhead* gracias a la demanda de tramas utilizando un SP y no teniendo que enviar una señalización por cada trama almacenada para poder recibirla. Esta reducción del *overhead* dependerá del tamaño del *Max_SP_lenght*.

## 1.3. Funcionamiento de WMM

Para poder entender como gestiona el mecanismo uAPSD nuestro dispositivo primero es necesario conocer la forma que tiene el AP de priorizar el tráfico. WMM proporciona diferentes servicios para aplicaciones de red que funcionan sobre IPv4 marcando (*tagging*) bits específicos de la capa 2 y 3. Al marcar los paquetes y las tramas se consigue una diferenciación sobre el resto para conseguir preferencia en las aplicaciones que se ven interferidas por tráfico de menor prioridad.

802.1p/Q es un mecanismo que sirve para soportar QoS en LANs. Este protocolo define un campo en la cabecera de capa 2 para paquetes 802.x que permite diferenciar entre 8 valores de prioridad (3 bits) mediante mecanismos de gestión de colas específicos. Al marcar las tramas se añaden 4 bytes adicionales que aumentan el tamaño de trama y pudiéndose superar el tamaño máximo permitido por algunos dispositivos (Figura 1.4). Este problema puede dar lugar al descarte de algunas tramas. Esta prioridad está limitada dentro del ámbito de la LAN. Al salir de ella a través de algún dispositivo de nivel 3, la etiqueta 802.11p/Q se elimina.

En la capa 3 se prioriza mediante un mecanismo llamado Diffserv3 que define un campo en las cabeceras de los paquetes IP denominado *diffserv codepoint* (DSCP). Las marcas que proporciona DSCP substituyen la cabecera IP sin añadir bytes adicionales. El DSCP está formado por 6 bits de prioridad (64 clases) y dos bits inutilizados. Los *routers* de las distintas redes marcarán cada paquete transmitido con un DSCP apropiado para clasificarlos según un encolamiento conocido como *per-hop behavior* (PHB) y así poder obtener una prioridad específica.

Desde la perspectiva de una red, el AP Cisco es un dispositivo de nivel 2 que actúa como un bridge. En relación con la QoS, el AP interactúa internamente con la información de nivel 2. El valor del que es informado el AP, pertenece a *class of service* (CoS). El CoS son 3 bits de información QoS contenido en el campo de prioridad de la trama t*agged* de 802.1p/Q. (Ver Figura 1.4)

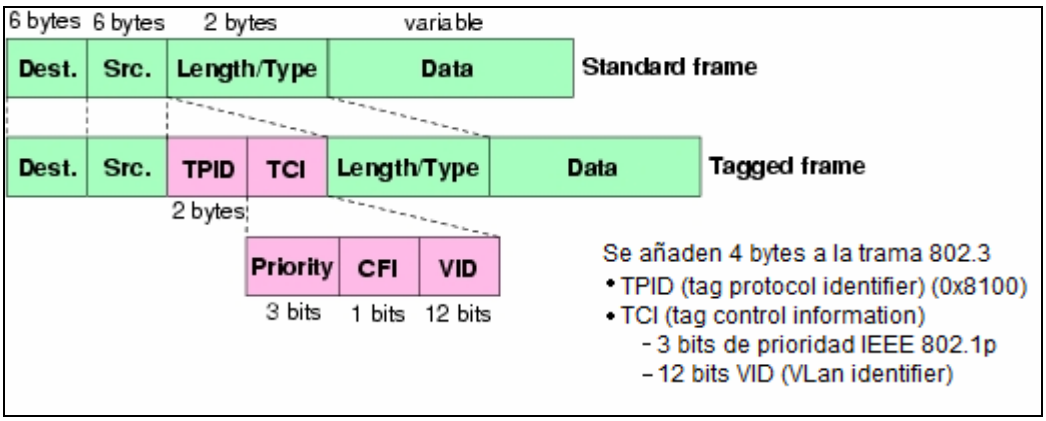


**Fig. 1.4** Ejemplo de trama 802.1p/Q

Para que el AP pueda leer el valor de 802.1p en la etiqueta 802.1q de la trama Ethernet es necesario un entendimiento entre el AP y el *switch* de acceso. Como el AP Cisco es compatible con el estándar 802.11e, los valores CoS (802.1d) se mapean a las clases 802.11e correspondientes tal y como se muestra en la Tabla 1.1.

**Tabla 1.1** 802.1d a WMM AC mapping

| Prioridad | Prioridad 802.1d (=UP) | Designación 802.11 | Access Category | Designación WME |
|---|---|---|---|---|
| Menor | 1 | BK | AC_BK | Background |
| | 2 | - | | |
| | 0 | BE | AC_BE | Best Effort |
| | 3 | EE | | |
| | 4 | CL | AC_VI | Video |
| | 5 | VI | | |
| Mayor | 6 | VO | AC_VO | Voice |

En caso de estar configurado el campo DSCP en la cabecera IP, para traducirlo a 802.1d se tomarán como valores los de la Tabla 1.2. De los 6 bits que corresponden a este campo se utilizan solamente tres (8 clases).

**Tabala 1.2** DSCP a 802.1d mapping

| DSCP P2 | DSCP P1 | DSCP P0 | DSCP - | DSCP - | DSCP - | DSCP hex | 802.1d | WMM AC |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 0 | 0 | 0x38 | 7 | AC_VO |
| 1 | 1 | 0 | 0 | 0 | 0 | 0x30 | 6 | AC_VO |
| 1 | 0 | 1 | 0 | 0 | 0 | 0x28 | 5 | AC_VI |
| 1 | 0 | 0 | 0 | 0 | 0 | 0x20 | 4 | AC_VI |
| 0 | 1 | 1 | 0 | 0 | 0 | 0x18 | 3 | AC_BE |
| 0 | 1 | 0 | 0 | 0 | 0 | 0x10 | 2 | AC_BK |
| 0 | 0 | 1 | 0 | 0 | 0 | 0x08 | 1 | AC_BK |
| 0 | 0 | 0 | 0 | 0 | 0 | 0x00 | 0 | AC_BE |

Para que puedan entenderse mejor los cambios requeridos para la implementación de QoS en el AP se consideraran tres procesos sobre la trama: el proceso Ethernet del AP, el proceso de encaminamiento del AP y el proceso en la interfaz radio del AP. Tal y como se explica a continuación, estos procesos aplican tanto para el tráfico de bajada como el de subida.

Para que el AP sea capaz de extraer el valor DSCP de nivel 3 y mapearlo a una información CoS de nivel 2 en sentido descendente se seguirá la siguiente secuencia (Figura 1.5):

**1.** El AP en la interfaz Ethernet elimina las cabeceras MAC propias de la trama Ethernet y almacena el valor DSCP.

**2.** En el proceso de encaminamiento, el AP mapea el DSCP a un valor interno de CoS perteneciente a una política de prioridad definida y después inserta tanto la etiqueta 802.1q que contiene la prioridad 802.1p como la cabecera propia de 802.11. A continuación se pasa el paquete a la interfaz radio del AP.

**3.** En la interfaz radio, el tráfico se encola basado en los valores CoS. En caso de que la trama tenga como destino otro AP que actúe como repetidor, la etiqueta 802.1q se elimina y la trama pasa a una función EDCA para su procesado final.

**4.** La trama se envía por el medio radio.

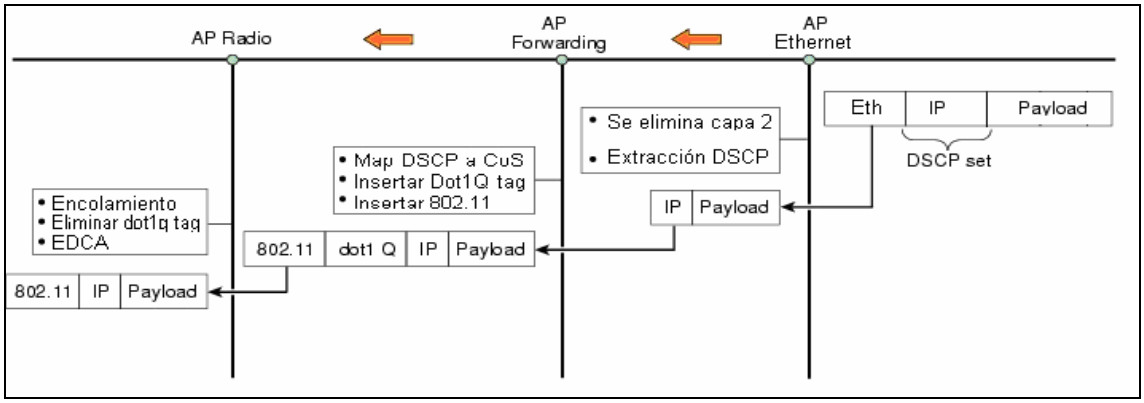


**Fig. 1.5** Procedimiento de mapeo en el canal de bajada

El *mapeo* en sentido ascendente sigue la siguiente secuencia (Figura 1.6):

**1.** El AP recibe la trama del medio radio, el AP elimina la cabecera 802.11 y aplica las políticas que han sido definidas. El resultado son valores de CoS que se almacenan en el AP.

**2.** El proceso de encaminamiento del AP inserta el valor de CoS usando la cabecera 802.1q.

**3.** En la interfaz Ethernet del AP, se elimina la etiqueta 802.1q y se añade la cabecera Ethernet para luego mapear el valor CoS guardado al valor DSCP correspondiente a la Tabla 1.2 pre-configurada.

**4.** La trama se envía por el interfaz Ethernet.

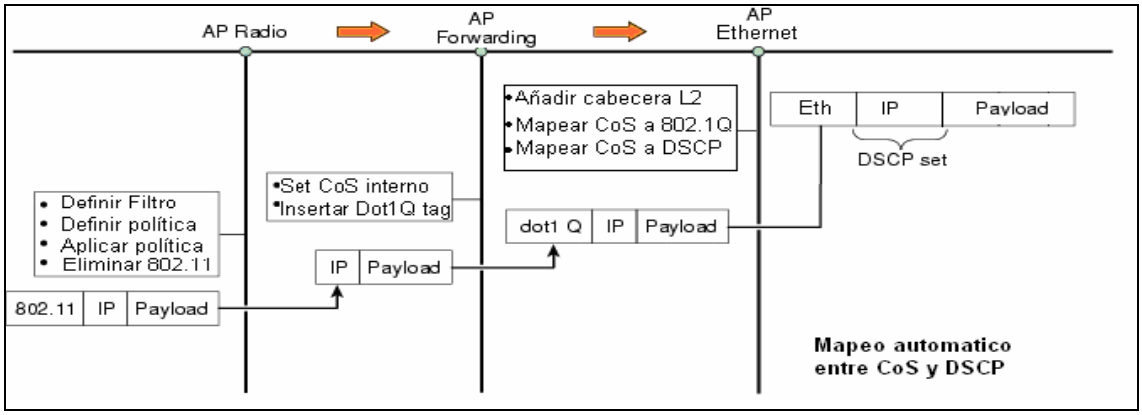


**Fig. 1.6** Procedimiento de mapeo en el canal de subida

Como complemento, el AP Cisco permite personalizar la prioridad según la tabla DSCP. Un ejemplo seria asignarle a los paquetes marcados con *Best Effort* en la tabal DSCP la prioridad que se desea, ya sea la estándar o definida a medida por el usuario. Si el paquete no viene marcado con un valor DSCP, también puede asignársele a un cliente asociado la CoS que se crea conveniente (Ver Figura 1.7). En definitiva se trata de que el administrador pueda elegir la prioridad de los datos que recibe un cliente específico o bien que la prioridad de los datos que provengan de las direcciones que se indiquen en la configuración del AP se modifiquen. Esta opción puede servir para realizar simulaciones en una red controlada o simplemente priorizar tráfico al gusto del administrador.

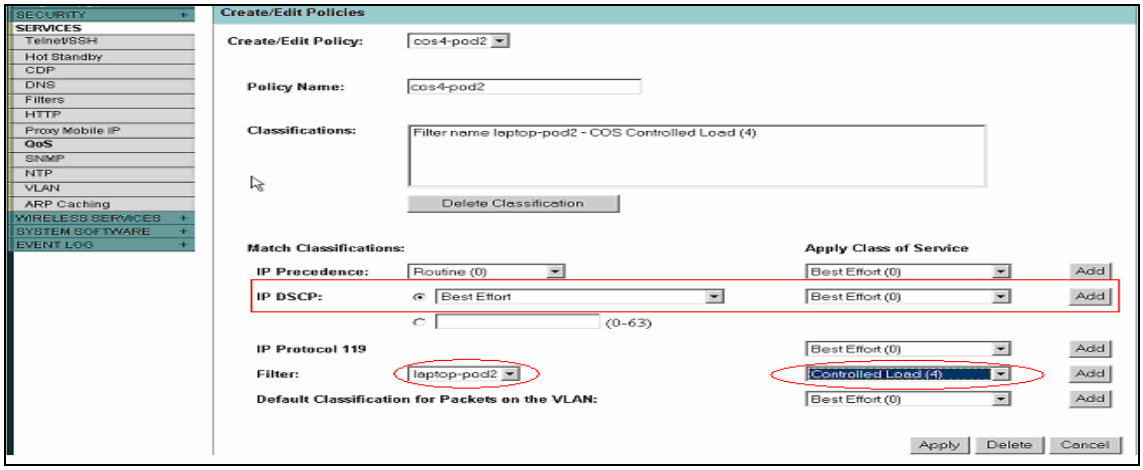


**Fig. 1.7** Creación de políticas en el menú de configuración Cisco 1130AG

## 1.4. Funcionamiento WMM-PS (uAPSD)

Un *unscheduled service period* (SP) comienza cuando el AP con funcionalidad QoS (QAP) recibe una trama *trigger* ya sea *QoS Data* o *Null* asociada a un AC que la estación tiene configurada como *trigger-enable*. Para que el QAP sepa que se trata de una trama *trigger* enviada por una estación en WMM-PS, el *Flag Untrigger* del campo *QoS Control* perteneciente a la trama MAC estará configurado a 0 (véase la Figura 1.8). De lo contrario, esa trama no tendrá la función *trigger* y no servirá para vaciar los *buffers* AC.

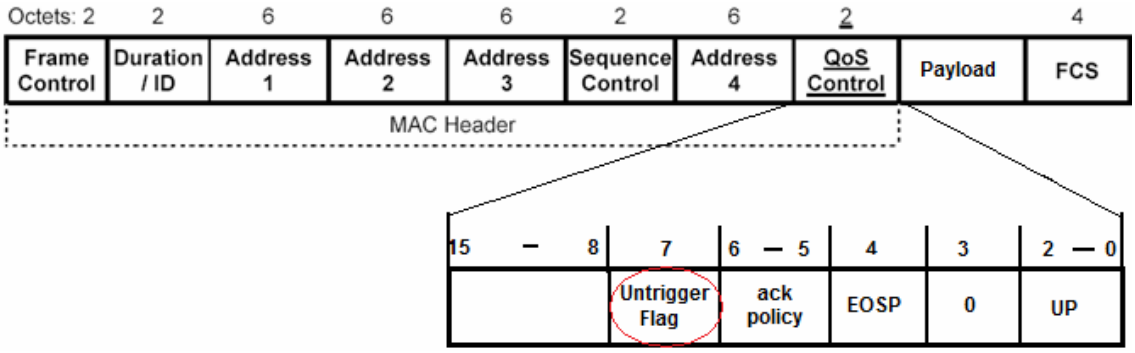


**Fig. 1.8**: Campo QoS Control de la cabecera MAC

Para que el QAP pueda entregar tramas durante un SP, el WMM STA designa sus ACs para que sean *trigger/delivery-enabled.* El WMM STA deberá configurar el QAP para que utilice el protocolo uAPSD. Para hacerlo activará el *Flag (bit=1) del AC* correspondiente definido en el subcampo *QoS Info.* Este subcampo forma parte del elemento *WMM information* también llamado *QoS Capability element* (Figura 1.9) transportado en la trama *(re) association request.*

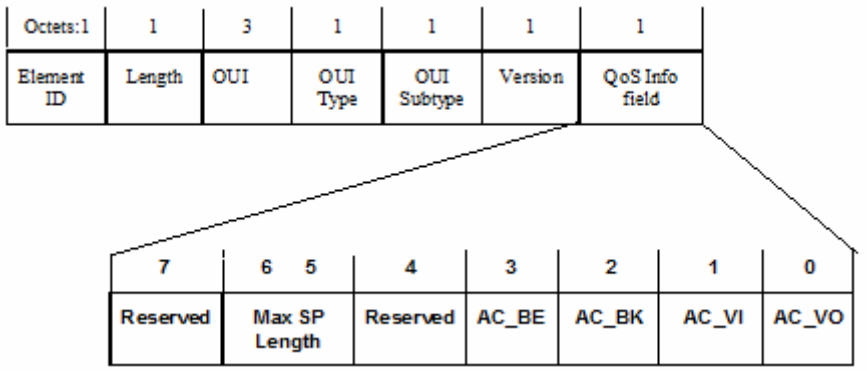


**Fig. 1.9**: Campo *QoS Info* del *QoS Capability element* en el canal de subida

Un subcampo importante es el *Max SP Lenght* que como se explicó en el apartado 1.2 permite gestionar el número máximo de tramas *unicast* que el QAP tiene que entregarle al WMM STA durante un SP. Este subcampo ocupa 2 bits (Tabla 1.3). En caso de que estén los dos a 0, el QAP entregará todas las tramas almacenadas.

.

**Tabla 1.3:** Configuración del *Max SP Lenght*

| Bit 5 | Bit 6 | USO |
|---|---|---|
| 0 | 0 | QAP tiene que entregar todas las tramas almacenadas. |
| 1 | 0 | QAP tiene que entregar un máximo de 2 tramas por SP |
| 0 | 1 | QAP tiene que entregar un máximo de 4 tramas por SP |
| 1 | 1 | QAP tiene que entregar un máximo de 6 tramas por SP |

En el canal de bajada el campo *QoS Info* tendrá los parámetros que muestran la Figura 1.10. El subcampo *Parameter Set Count*, inicialmente es arbitrario y se incrementa cada vez que un parámetro AC cambia. El otro subcampo *Reserved* configurado a 0 en el emisor, deberá ser ignorado por el receptor. Cuando el *Flag uAPSD* esté configurado a 1 en las tramas de *beacon*, indicará que el QAP permite el soporte de tramas *trigger power save*.

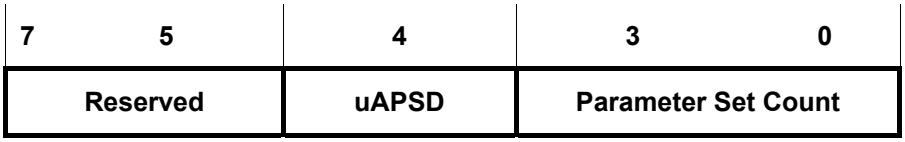


**Fig 1.10**: Subcampo *QoS Info* en el canal de bajada

En la Figura 1.11 puede verse la captura de un *beacon* procedente del AP *Cisco 1130AGN* que corresponde con los valores de WMM *information element*. En el WME (WMM) *Parameter Element* (PE) de la captura, aparecen campos como *parameter set* que corresponden al *QoS Info.* Los campos no coinciden del todo porque hay diferencias entre las distintas marcas. En las siguientes líneas de la captura se observan los parámetros del protocolo EDCA pertenecientes a los cuatro AC mencionados en el capítulo 1.2 que configuran el comportamiento de cada uno de los AC. No se explicará el funcionamiento de EDCA en profundidad pero si detallaremos resumidamente alguno de los parámetros importantes para priorizar los AC durante el acceso al canal.

Cuanto más pequeño es el campo *arbitration interframe space number* (AIFSN), mayor prioridad tiene el AC. ECWmin y ECWmax determinan el tamaño mínimo y máximo de la ventana de contención. El campo *Transmisión Oportunity* (TXOP) marca el intervalo de tiempo en el cual una QoS STA tiene derecho a transmitir. TXOP mejora la eficiencia del canal significativamente concediendo mayor tiempo de transmisión a los AC que precisan de menor retardo.

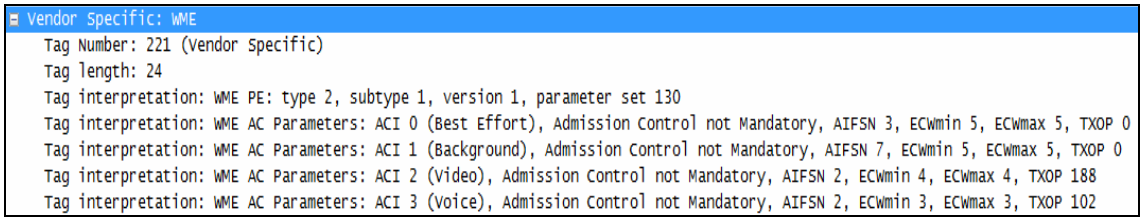


**Fig. 1.11:** Captura del campo *WME* (QoS Capability Element) del *beacon*

En resumen, la secuencia que sigue un proceso uAPSD es la siguiente:

1. Para abrir un SP, la STA envía una trama *QoS Data* al QAP con el subcampo *untrigger* = TRUE (no hace de mensaje *trigger*).

2. Esta información la responde el QAP mediante un ACK. Desde que se envía el *QoS Data* hasta que se recibe el ACK constituye el TXOP.

3. En la siguiente secuencia el subcampo *untrigger* = FALSE del *QoS Data* demanda la información almacenada en el QAP. Como el QAP contiene tramas almacenadas destinadas a la STA, responde con un ACK que contiene el campo *MD = 1.* El siguiente mensaje es un *QoS Data* enviado a la STA conteniendo la información almacenada, pero con el campo EOSP = 1 que indica el final del SP.

4. Después de un tiempo se vuelve a producir el caso 1 y 2.

5. En el último caso, la STA necesita transmitir información y el QoS Data que envía contiene el subcampo untrigger = FALSE.

6. Como no existen tramas en el QAP para ser entregadas, después de enviar el ACK, se enviará un QoS Null sin información para poder finalizar el SP mediante EOSP = 1

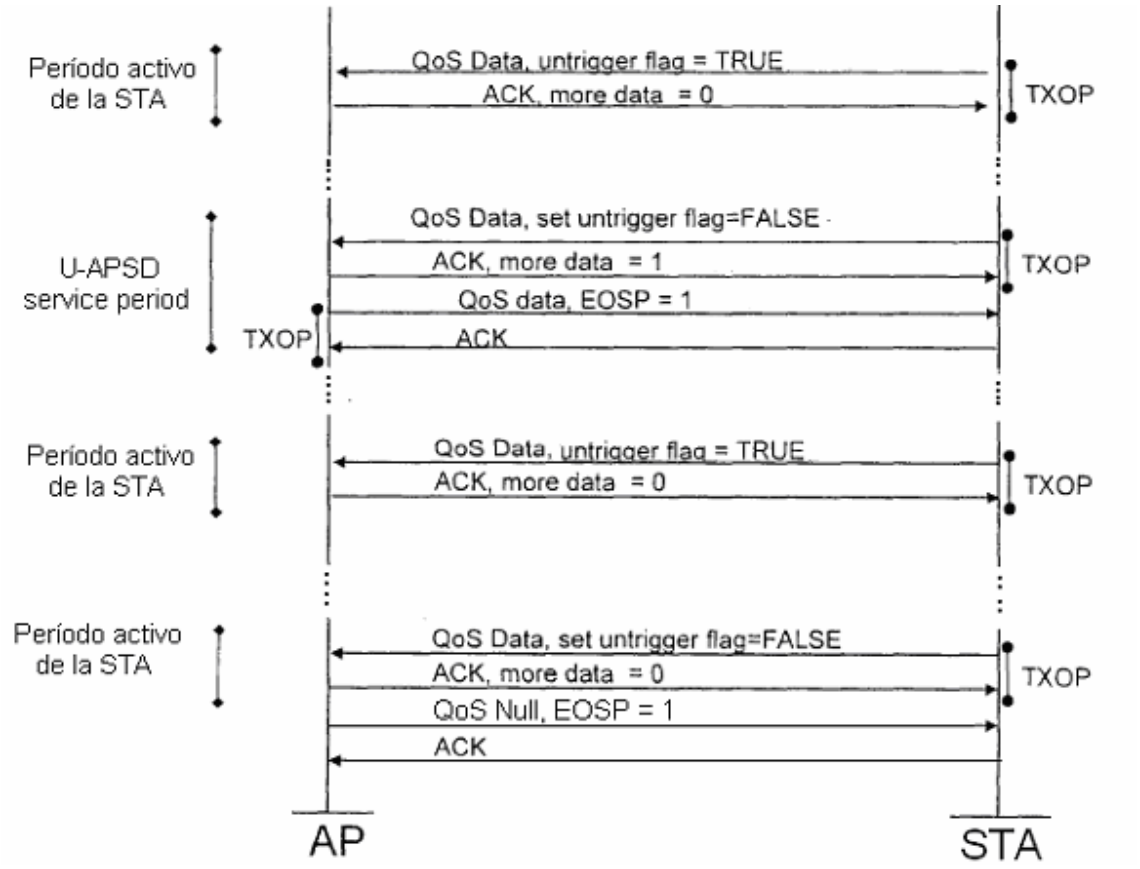


**Fig.1.12:** Secuencia uAPSD

### 1.4.1 Comportamiento del AP

Un QAP que implementa *uAPSD* deberá, si un WMM STA usa *uAPSD* y utiliza PSM guardar temporalmente en el *buffer* las tramas destinadas a ese WMM STA hasta que este envíe una trama *trigger* solicitándolas a través de la apertura de un *uncheduled* SP. En caso de que el WMM STA este en modo activo, las tramas no serán almacenadas y serán directamente transmitidas. Si por algún caso el QAP recibe una trama *trigger* durante un SP, no abrirá otro SP. En cada intervalo *beacon*, el QAP deberá enviar el *partial virtual bitmap* que contiene el estado de los *buffers* con destino la STA en el campo *TIM* del *beacon*.

Una única trama almacenada en el buffer para un STA en *PSM* se entregará al STA después de recibir un *PS-Poll* de dicho cliente. A un WMM STA que use *uAPSD* se le entregará una trama almacenada en el QAP de cualquier AC que no esté configurado como *delivery-enabled* como respuesta a un *PS-Poll*. Si todos los AC están configurados como *delibery-enabled* se entregará una trama del AC de mayor prioridad. Aunque parezca contradictorio, esta metodología se da porque aunque el QAP soporte *uAPSD*, el STA está

enviando *PS-Polls* sin la función *trigger* activada porque está funcionando en modo *PSM* y no *WMM-PS*. Por tanto no se abre un *uncheduled* SP.

Tanto para los STAs en *PSM* como los que usan *uAPSD*, el campo *More Data* estará activo para indicar que quedan más tramas almacenadas para ese cliente. En caso de que todos los AC sean *delivery-enabled*, este campo indicará que hay más *MSDU*s almacenadas o más tramas de gestión correspondientes a los AC *delivery-enabled*.

En cada *unscheduled* SP se transmitirá al menos una trama sin exceder nunca el numero indicado en el campo *Max SP Lenght* del *QoS Capability element*. Para indicar el final del SP, como ya se ha comentado en el apartado 1.2, el campo *EOSP* contenido en el *QoS Control* de la cabecera MAC se cambiará el valor a 1. Si es necesario, se enviará una trama *QoS Null* extra con este bit *EOSP* configurado a 1.

### 1.4.2 Comportamiento de la STA

Cuando una WMM STA que usa uAPSD y tiene todos sus ACs *delivery-enabled* detecta que el bit correspondiente a su AID está activo en el TIM de *beacon* que recibe, la WMM STA deberá enviar una trama *trigger* o un *PS-Poll* para recibir las tramas almacenadas en el *buffer*. En caso de enviar una trama *trigger* se abrirá un SP.

Si la WMM STA está iniciando un SP, se despertará para enviar un *trigger* al QAP. En caso de que uno o más ACs no sean *delivery-enabled*, el WMM STA tendrá que solicitar al QAP los *MPDU*s correspondientes a los ACs mediante *PS-Polls*.

El WMM STA deberá permanecer despierto hasta que reciba una trama QoS data con el subcampo *EOSP* configurado a 1.

El WMM STA deberá enviar PS-Polls adicionales si el bit *More Data* está activo en las tramas *unicast* del canal de bajada que no correspondan a ningún AC *delivery-enabled.* Si las tramas corresponden a ACs *delivery-enabled* deberá enviar tramas *trigger* adicionales.

# CAPÍTULO 2: PRUEBAS DE CONSUMO Y RESULTADOS

Para optimizar la duración de la batería, se ha realizado un estudio de diferentes técnicas, o parámetros, que pueden aplicarse en una comunicación inalámbrica para poder distinguir cuáles suponen un mayor gasto o ahorro de energía. En algunos casos, la influencia es muy leve y por ello solamente se mencionarán los que más contribuyen al agotamiento o ahorro de la batería. En este capítulo se introducirá brevemente estas técnicas o parámetros para después describir las pruebas realizadas y los resultados obtenidos. Unos resultados que en lo posible, se comparan con el comportamiento teórico esperado. Cada una de las pruebas se ha realizado en el sistema operativo Linux y se ha repetido un mínimo de 3 veces con duraciones de entre 1 y 2 horas para contrastar los resultados. La información adicional relativa a estas pruebas, por ejemplo sobre la configuración de los dispositivos o el software empleado, se encuentra en el anexo III de esta memoria.

## 2.1 uAPSD

Tal y como se ha comentado en el capítulo anterior, cabe destacar como elemento fundamental para el ahorro de batería la certificación WMM-PS de 802.11e, y concretamente uAPSD. Este protocolo permitía que la misma cantidad de datos sea enviada en menor tiempo que con PSM. Como consecuencia, la batería debe sufrir un menor desgaste.

Para poder probar uAPSD primero se debió identificar que dispositivos lo implementarán (ver Anexo II). De entre ellos se eligió el AP Cisco Aironet 1130AG y como STA la tarjeta Intel 4965AGN operando en un portátil Dell XPS con una batería de 9 celdas con capacidad de 85Wh. La configuración de estos dispositivos que van a utilizarse en ésta y en el resto de pruebas, se muestra en el Anexo III.

### 2.1.1 Pruebas uAPSD

Dado que la justificación de uAPSD es su aplicación para servicios en tiempo real, en esta prueba se ha emulado tráfico de VoIP utilizando la herramienta Mgen para generar tráfico en los 2 sentidos siguiendo una función de ráfagas de datos discontinuas (ver Anexo II.3). Para ello se ha configurado Mgen tanto en el portátil provisto de la STA como en un ordenador del laboratorio conectado directamente al AP Cisco. (Ver Figura II.1 del Anexo II)

Para emular un tráfico de voz, se ha contabilizado la cantidad de *bytes* que intervienen en una transmisión VoIP que utiliza el *códec* G729 tomando muestras cada 30ms (30 *bytes* de *payload*) y que está encapsulado en tramas IEEE802.11 (Ver Tabla 2.1).

**Tabla 2.1**: Cabeceras y *payload* de una trama VoIP.

| **Parámetros** | **Tamaño en bytes** |
|---|---|
| IP | 20 |
| UDP | 8 |
| RTP | 12 |
| Datos códec G.729 | 30 |
| **Total** | **70** |

Teniendo en cuenta que utilizando un códec G729, se envía el *payload* a 8Kbps se calculará el ancho de banda necesario según la fórmula (2.1) para luego poder aplicar un *script* con la tasa de envío utilizando la herramienta *Mgen*.

$$8Kbps \cdot (70\ bytes / 30\ bytes) = 18{,}66\ kbps \qquad \textbf{(2.1)}$$

Como el tráfico RTP se genera en los dos sentidos, el BW necesario será de 37,32 Kbps, correspondiente a la suma de los dos BW de (2.1).

Se han generado ráfagas de paquetes con una tasa de 33,33 paq/s correspondientes a que se coge una muestra de voz cada 30 ms. La duración es exponencial y sigue una distribución de Poisson. Concretamente se ha tomado un tasa de 10 llamadas por hora y un tiempo medio de duración de la llamada de 180 segundos.

Para medir el consumo de baterías, en el portátil se ha configurado el demonio *cron* para que ejecute el *script ibam* (ver Anexo I) y recoger los resultados de las mediciones de batería. En todas las demás pruebas, se recogerán los resultados del mismo modo. Además, en esta prueba, debido a las incoherencias en los resultados del primer *script ibam* se ha hecho uso también de un *script* que hemos llamado '*script bateria'* generado para recoger los datos de consumo y capacidad restante mediante comandos Linux.

## 2.1.2 Resultados

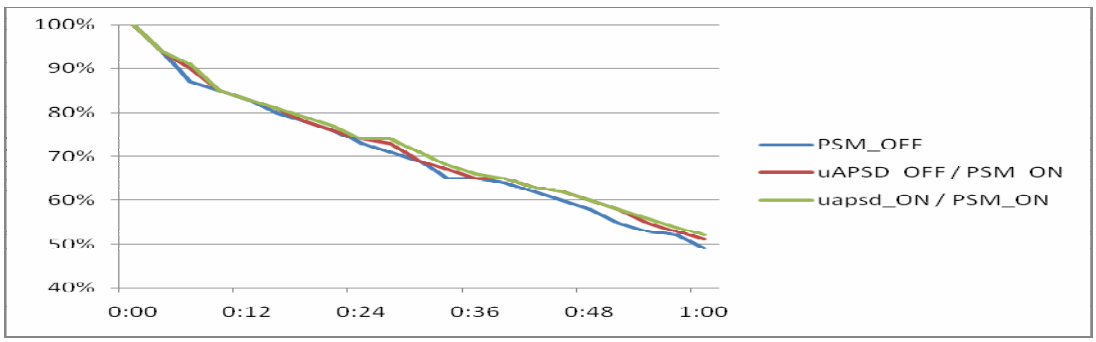


**Fig. 2.1**. Comparativa de consumo en % utilizando uAPSD y PS

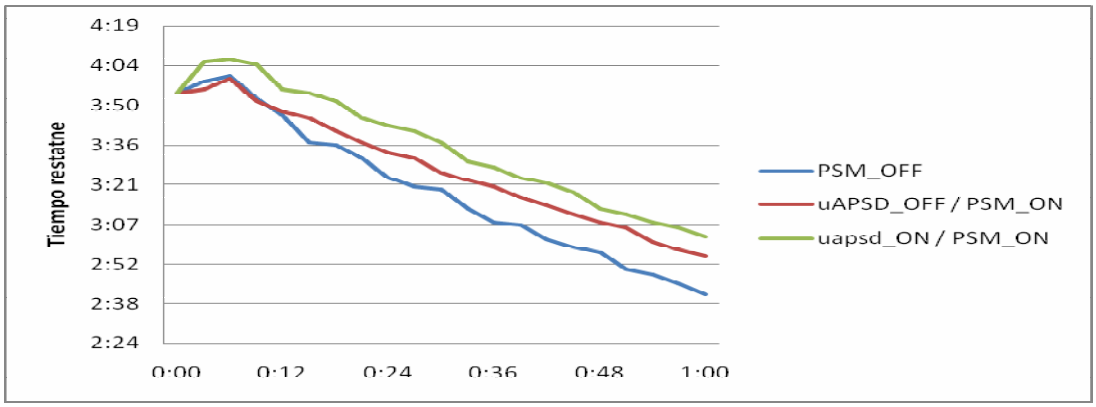


**Fig. 2.2**. Comparativa de consumo en tiempo restante utilizando uAPSD y PSM

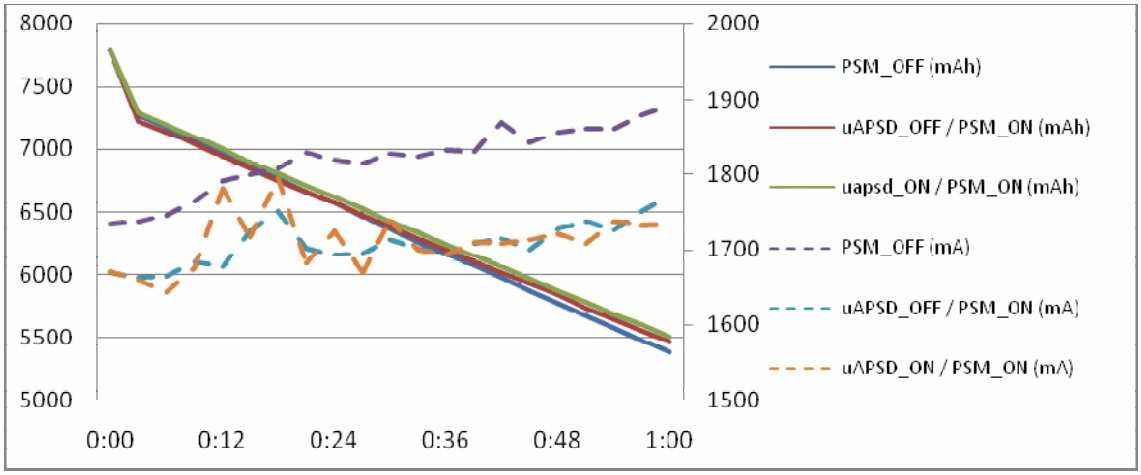


**Fig. 2.3**. Comparativa de consumo en mWh utilizando uAPSD y PSM

En las Figuras 2.1, 2.2, 2.3 se observa que la diferencia entre utilizar uAPSD o no es muy leve. Teóricamente, el hecho de que el *script* genere 10 llamadas por hora hace que se consiga abrir 10 SP en total con los cuales se van a ahorrar un máximo de 60 PS-Polls (ver Capítulo 1). Este número de mensajes es muy pequeño como para afectar al consumo de la batería por lo que, las diferencias entre PSM y uAPSD son muy pequeñas. Además como puede verse, el comportamiento de la batería es distinto para cada uno de los casos pudiendo generar imprecisiones.

No obstante, con las capacidades del *driver*, que por el momento no soporta el *untrigger flag* no se puede asegurar su correcto funcionamiento. Es este motivo, por el cual, en los resultados obtenidos, el uso de uAPSD incluso puede afectar algo más al desgaste de la batería como ha sucedido en alguna de las pruebas. Esto puede ser debido a que el hecho de modificar las tramas IEEE802.11 con WMM se añaden algunos parámetros *tagged* en los *beacon* y en los *QoS Null* (ver Anexo III) que informan al AP del estado de la tarjeta además de servir de PS-Poll. Por tanto, el aumento de estas longitudes puede ser el motivo de que el ahorro de uAPSD quede neutralizado. El AP no permite

separar ambas funcionalidades WMM y uAPSD, así que no se ha podido comprobar las diferencias de manera empírica. Por ello, a continuación, se analizarán de manera numérica.

- WMM-PS (uAPSD) desactivado:

  Tamaño *beacon* = 193 bytes
  Tamaño *IEEE802.11 Null* (PS-Poll) = 49 bytes

- WMM-PS (uAPSD) activado:

  Tamaño *beacon* = 219 bytes
  Tamaño *IEEE802.11QoS Null* = 51 bytes

- Diferencia:

  *Beacon*: 219 bytes – 193 bytes = 26 bytes
  Trama *Null*: 51 bytes – 49 bytes = 2 bytes

Como puede verse en la figura III.8 del Anexo III. Se envía 2 tramas *IEEE802.11 Null*, una para informar al AP de que la tarjeta se va a dormir y otra para que haga de PS-Poll, cada 2 segundos aproximadamente si se tiene en cuenta los momentos en que se recibe tráfico y en los que no. Por tanto:

**(2.3)**

1h de prueba: 3600 s * 2tramas / 2 segundos = 3600 tramas *IEEE802.11 Null*
3600 tramas Null * 2 bytes = 7200 bytes extras.

Si cada 100 ms se envía un *beacon*:

1h de prueba: 3600 s * 1/100 ms = 36000 *beacons*
36000 *beacons* * 26 bytes = 936000 bytes extras

El consumo de la longitud extra en las tramas Null enviadas a 12 Mbps (ver Anexo III) y *beacon* enviadas a 1 Mbps se calculará con los valores de la tarjeta utilizada presentada en la Tabla 2.2:

**(2.3)**

- Tramas IEEE802.11 Null:

[(7200 bytes / 12 Mbps) / 3600] horas * 1875 mW = 2,5 mWh

- *Beacons*:

[(936000 bytes / 1 Mbps) / 3600] horas * 1300 mW = 338 mWh

La suma de estos dos consumos, 340,5 mWh equivale a un 0,4 % del total de 85W que contiene la batería Dell 9 celdas con la que se ha hecho la prueba. Es decir, si se quiere ser beneficiario de una QoS con la certificación WMM, debe asumirse el coste sufrido por este *overhead*. No debe olvidarse que uAPSD tiene el propósito de integrar PSM en la certificación WMM sin sufrir pérdida de QoS. Por tanto, esta situación aquí analizada será la habitual.

Por otro lado, la desactivación de PSM afecta bastante más al desgaste de las baterías. Esto es debido a que en el tiempo entre llamadas, la STA ha permanecido en un estado de ahorro de batería (*sleep*) mientras utilizaba PSM. A partir de los valores de la Tabla 2.2 se puede calcular este ahorro debido al uso de PSM para esta prueba, utilizando la tarjeta Intel 4965AGN.

**Tabla 2.2**. Consumo tarjeta utilizada en las pruebas.

| **Chipset** | Sleep (mW) | Idle (mW) | Rx (mW) | Tx (mW) |
|---|---|---|---|---|
| Intel 4965AGN | 30 | 150 | 1300 | 1875 |

Ráfagas cada 320 segundos con duración exponencial 180s:

**(2.4)**

Aproximando: 320 -180 = 140 segundos de inactividad.
140 segundos = 43,75 % del total
0,4375 * 1 h de prueba = 0,4375 horas

- Sin PSM:
  0,4375 h * 150 mWh = 65,625 mWh

- Con PSM:
  0,4375 h * 30 mWh = 13,125 mWh

Con PSM se ahorrará hasta un 80 % más en los tiempos de espera equivalente a 0,15 % de la batería total. Comparando con los resultados de la Figura 2.1, se observa que la diferencia es mayor, un 2% aproximadamente. Esto puede ser debido a imprecisiones del *script* de medición así como la poca linealidad del comportamiento de la batería. Diferencias tan pequeñas en el consumo no pueden ser medidas de forma precisa ya que el valor mínimo de medida es un 1%.

Con los valores de consumo de la Figura 2.2 extraemos la diferencia de forma más precisa:

**(2.5)**

PSM activado al cabo de 1 hora: 5506 mAh restantes.
PSM desactivado al cabo de 1 hora: 5379 mAh restantes

5506 mAh - 5379 mAh= 127 mAh

Capacidad empírica total: 7800 mAh → 100%
Diferencia de capacidad: 127 mAh → 1,63%

Así pues, la diferencia de consumo que ofrece el monitor de batería es de 1,63%, un 1,48% más que los resultados teóricos.

## 2.2. ARP-cache y ARP-caching

La opción *ARP-cache* permite que el *AP* actúe de forma parecida a un *Proxy ARP*. Gracias a ello, el AP es capaz de contestar las peticiones ARP dirigidas a los clientes asociados a éste. Al recibir un *ARP Request*, el *AP* revisa su caché y si la IP coincide con uno de sus clientes asociados responde con la MAC de éste. En caso contrario, es decir si la IP no coincide con ningún cliente asociado, el *AP* ignora esta petición y de este modo filtra todas las peticiones *ARP* no dirigidas a clientes asociados (ver Figura 2.2).

En los *beacons* que envía un AP que utiliza *ARP-cache,* se incluye un campo que informa a las STAs que pueden ignorar de forma segura mensajes *broadcas*t. Esta característica sumada con el total de *ARP Request* que no recibe la STA y los que contesta el AP en su lugar, puede llegar a reducir el consumo de energía.

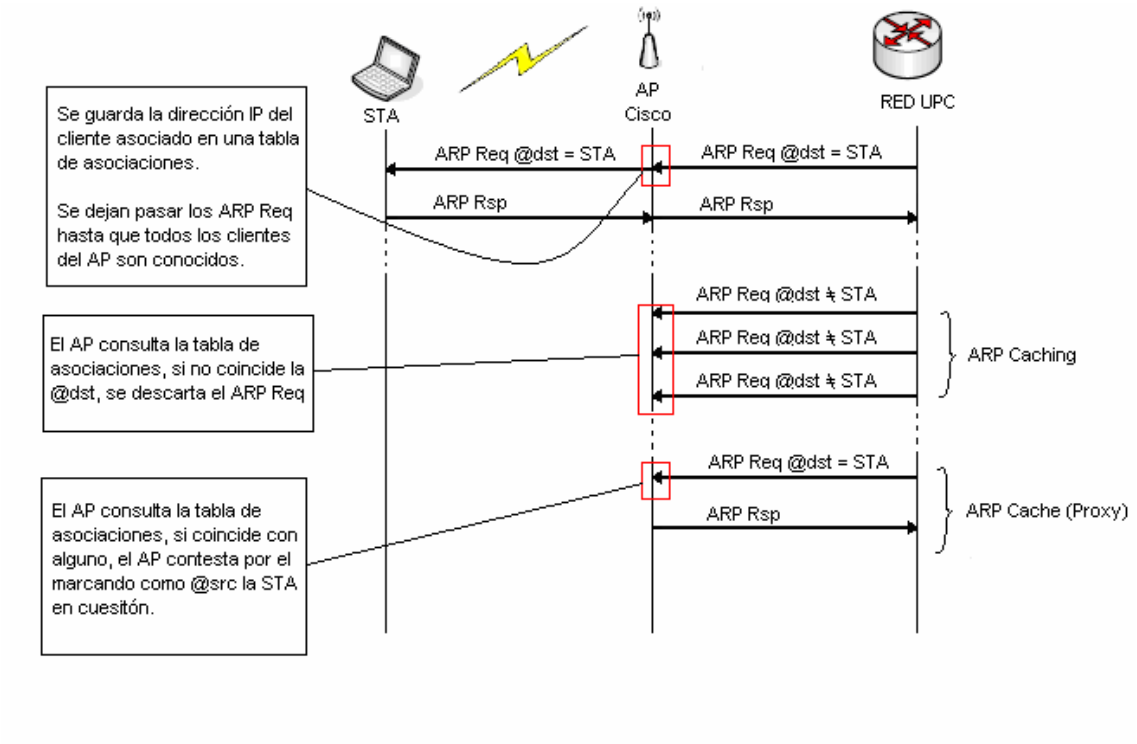


**Fig. 2.4:** Funcionamiento de ARP-Cache y ARP-Caching

### 2.2.1 Pruebas ARP-Cache y ARP-Caching

Para observar el impacto de esta funcionalidad, se realizará 2 tipos de prueba. La primera consistió en conectar el AP Cisco Aironet 1130AG a la subred de la EPSC 147.83.118.0/24 para que la STA recibiese las peticiones ARP que circulan por ella, ninguna de ellas preguntando por su IP. En esta prueba se comparan los resultados con la funcionalidad activa y sin ella para ver las diferencias.

La segunda prueba consistió en la elaboración de un *script* que borra cada dos decimas de segundo la tabla ARP de un ordenador conectado directamente al AP vía Ethernet y seguidamente envía un ping para forzar la resolución MAC con el envío de *ARP Request.* Con el script se generan estos mensajes a una tasa de 5 mensajes por segundo. Esta prueba se realizará con el ARP-cache activo, funcionalidad gracias a la cual el AP responderá las peticiones generadas por el *script*. En la siguiente prueba se desactivará el *ARP-cache* para comparar el consumo. El *script shell* utilizado es el siguiente:

```
#! /bin/sh

While [ 1 ]                 #bucle infinito
do
arp –d 147.83.118.226       #borramos dirección de la tabla ARP
sleep 0.2                   #repetimos el proceso cada decima de segundo
done
```

### 2.2.2 Resultados

2.2.2.1 ARP *Caching*

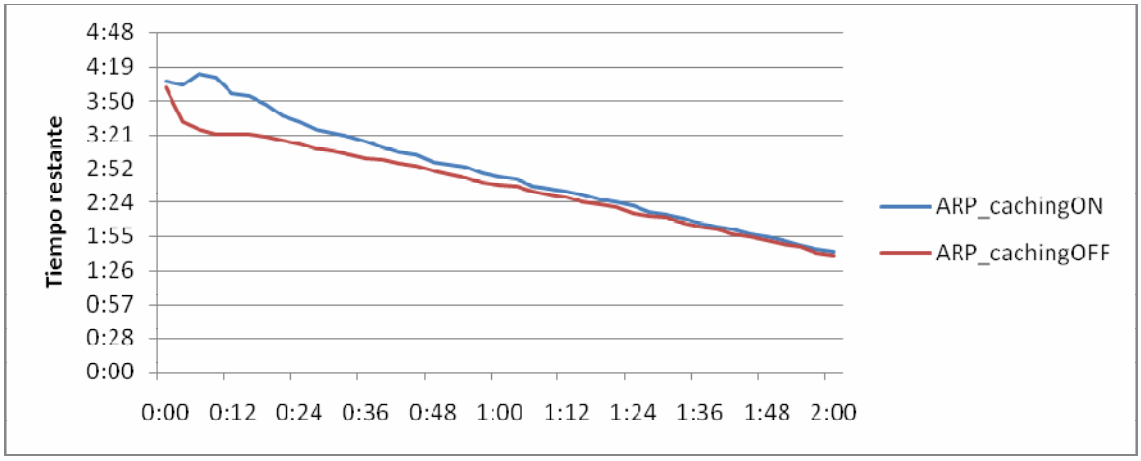


**Fig. 2.5**. Resultados de la monitorización de batería para la funcionalidad

Como muestra la Figura 2.2, aunque no se pueda apreciar del todo, el ahorro conseguido por rechazar ARPs con distinto destino al de la STA cliente es de 4 minutos según la tabla obtenida con el *script* de medición al cabo de 2 horas. La primera media hora el comportamiento de la batería es algo impreciso. A partir de este tiempo se normaliza y los resultados se aproximan más a la realidad. El ahorro se ha conseguido no solamente por el consumo producido por la multitud de mensajes ARP recibidos provenientes de la subred de la EPSC sino también porque la STA que en las dos pruebas tenía la funcionalidad PSM activada, con ARP *caching* puede permanecer en un estado de ahorro de batería mayor tiempo ya que mensajes ARP con distinto destino al del cliente del AP no despertarán inútilmente la STA. Para este caso no se ha podido estimar el consumo teórico ya que la generación de ARPs es aleatoria y variante. Así, los valores representados en la figura eran los valores medios de 3 pruebas.

### *2.2.2.2 ARP Cache (Proxy)*

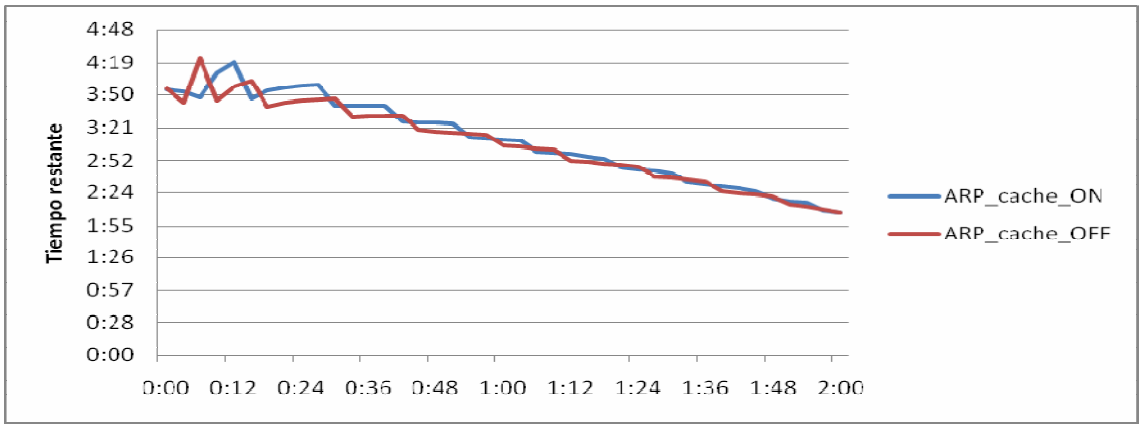


**Fig. 2.6**. Resultados de la monitorización de batería para la funcionalidad

Tal y como se puede apreciar en los resultados de la Figura 2.3, la batería restante al cabo de 2 horas, en los dos casos, tanto con el ARP *Cache* activo como desactivado, es la mismo. A continuación se analizará el ahorro teórico que supone la activación de esta funcionalidad para ver si los resultados son coherentes.

Utilizando la aplicación creada para el análisis de consumo de los traspasos (ver capítulo 3 y anexo VI) con los valores de la Tabla 2.2, se ha obtenido el consumo de un mensaje ARP. Por tanto, con estos datos y calculando el número de ARP que genera el *script* se podrá estimar el ahorro de la funcionalidad. Se procede como sigue:

**(2.6)**

- Se calcula el consumo de un mensaje ARP:

Consumo de la recepción de un ARP Req.: 5,4 e-5 mWh
Consumo de la transmisión de un ARP Rsp: 6,54 e-5 mWh

Nº ARPs = 1 ARP / 200 ms · 2 horas = 10 · 60 · 60 · 2 = 36000 ARPs

- Se calcula el consumo de todos los mensajes de la prueba:

36000 ARP Req. * 5,4 e-5 mWh = 1,944 mWh
36000 ARP Rsp. * 6,54 e-5 mWh = 2,354 mWh

- Se estima el porcentaje de batería ahorrado con el proxy (batería total=85000 mWh)

85000 mWh → 100%
1,944 + 2,354 = 4,298 mWh → 0,00506 % ahorrado

Como puede apreciarse, el ahorro es muy inferior al 1% de la batería, que es la sensibilidad que tiene el mecanismo de monitorización creado.

## 2.3. Delivery Traffic Indicator Maps (DTIM)

Los APs transmiten *beacons* a intervalos regulares. El intervalo se llama *beacon period* y se mide en *Time Units* (TU) de 1024 µs. El típico intervalo entre *beacons* suele ser de 100 TU (100 ms). El periodo DTIM indica cada cuantos *beacons* una STA en *power save mode* debería permanecer despierto para comprobar si hay tráfico *multicast y broadcast* almacenado en el buffer del AP preparado para serle transmitido. Por tanto, un periodo DTIM de 2 significará que cada 2 *beacons* se transmitirá uno con el campo DTIM activo. Este valor de DTIM es el que suele utilizarse por defecto en la mayoría de AP.

El periodo DTIM es influyente en el consumo de batería ya que cuantas más veces se despierte para recibir tráfico *multicast* y *broadcast*, más tiempo permanecerá la STA despierta consumiendo energía. Se ha realizado una serie de pruebas de consumo modificando el periodo DTIM en el AP. Teóricamente, cuanto mayor sea este intervalo, menor consumo tendrá la STA. Con un período muy grande la STA se mantiene despierta mucho tiempo para poder entregar todo el tráfico almacenado hasta el momento. Es por esto que llegados a un periodo DTIM, la reducción del consumo por el bajo número de veces que la STA se despierta, queda compensada con la cantidad de tráfico almacenado hasta el momento para ser entregado. A partir de ese periodo, las diferencias de consumo son mínimas. El coste de consumo de la recepción del tráfico es el mismo para todos los valores DTIM. No obstante, el hecho de despertarse menos veces permite un mayor ahorro.

### 2.3.1 Pruebas DTIM a realizar

La prueba a realizar consistirá en modificar el período DTIM según el Anexo III variando su valor entre 1-10. Concretamente se aplicará para los valores [1,

3,7,10,20] y se medirá el consumo durante 1 hora mientras la STA recibe tráfico *multicast* generado con Mgen. El script creado (ver Anexo III) envía mensajes UDP de longitud 42 bytes, para asemejarlo al tamaño de un ARP, a la dirección *multicast* que tienen todos los sistemas por defecto en una local LAN; la 224.0.0.1. Se ha establecido una tasa de envío de 1 paquete cada segundo para que haya un tiempo entre mensajes suficiente como para que el efecto del PSM pueda notarse.

### 2.3.2 Resultados

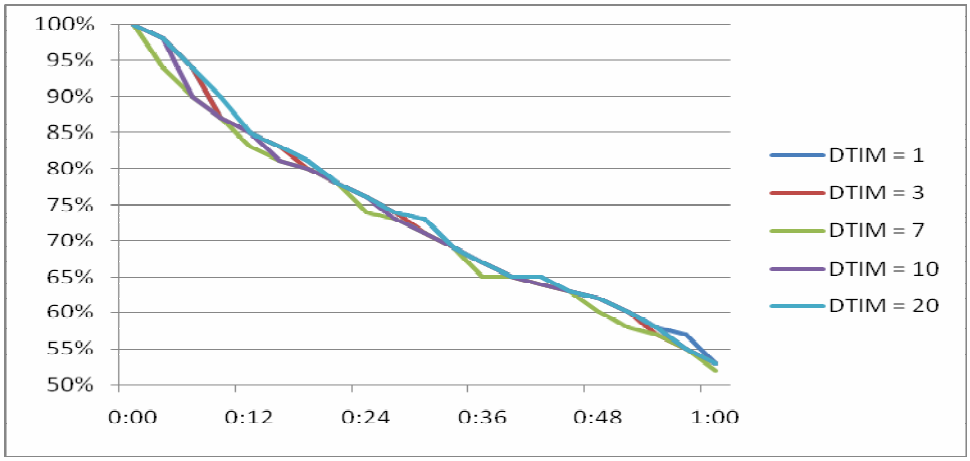


**Fig. 2.7**. Consumo en % de la variación del parámetro DTIM

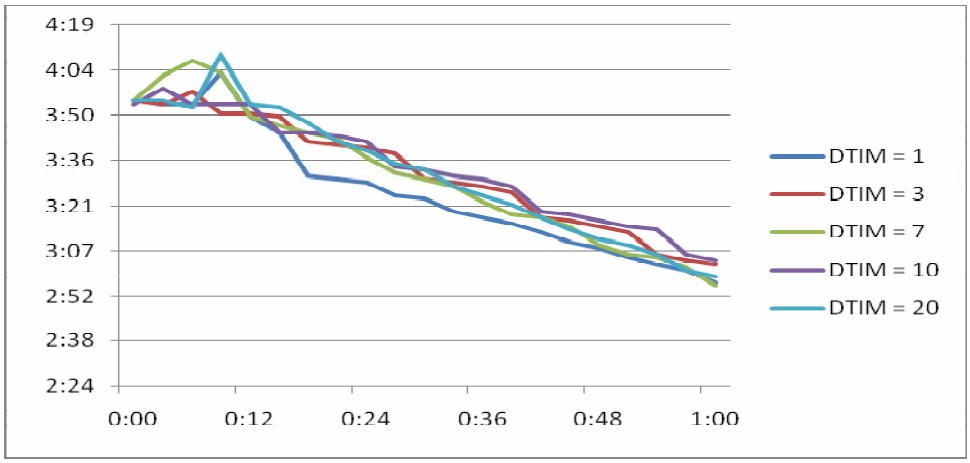


**Fig. 2.8.** Consumo en tiempo restante de la variación del parámetro DTIM

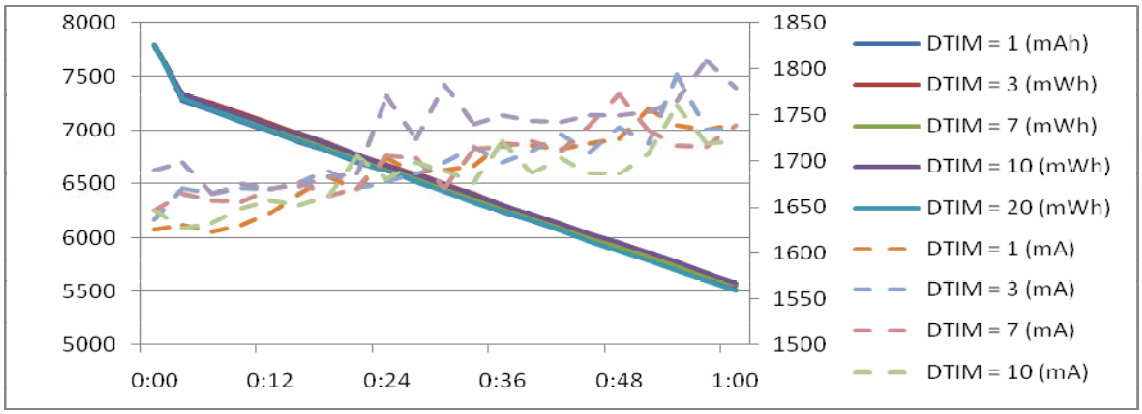


**Fig. 2.9.** Consumo en mWh de la variación del parámetro DTIM

Las figuras anteriores muestran poca variación en el consumo. No obstante, se percibe una ligera reducción hasta alcanzar el valor de máximo ahorro con un DTIM de 10. Los resultados muestran que con un valor DTIM de 10 se consigue 67mAh más de capacidad al cabo de una hora. Esto corresponde a un 0,86 % de la batería total. A partir de este valor, tal y como muestran las figuras, el consumo vuelve a elevarse ligeramente por causas que no se conocen. Sería necesario la repetición de estas pruebas cambiando por el ejemplo el tráfico inyectado, para contrastar estos resultados. También podría extenderse su duración.

# CAPÍTULO 3. Estudio del consumo de baterías derivado de los traspasos MIPv6 y IEEE802.11

En este capítulo se pretende estimar el consumo de batería derivado de la realización de traspasos IEEE802.11 y MIPv6. Se trata de estudiar qué factores influyen en estos consumos y compararlos con el fin de estudiar qué partes son las más susceptibles de ser mejoradas con el fin de ahorrar batería. Para realizar este estudio primero se va a explicar el funcionamiento del protocolo de movilidad MIPv6 y en concreto, su procedimiento de traspaso. Luego se analizará el mismo proceso para IEEE802.11. Cabe destacar que el traspaso de MIPv6 se realiza en una red IEEE802.11. Para el estudio se ha recopilado información de consumos de algunas de las tarjetas IEEE802.11b más comunes del mercado, se han capturado tramas correspondientes a la señalización de los traspasos MIPv6 e IEEE802.11 y calculado el consumo de los dos procesos. Finalmente se ha hecho una comparación de los resultados para extraer conclusiones y se han aplicado a dos productos para ver su impacto. Los resultados del consumo de los traspasos así como algunos de los gráficos presentados en este capítulo se han generado con una Aplicación Excel que se ha creado para este proyecto y que se presenta en el Anexo VI.

## 3.1 Protocolo de movilidad MIPv6

### 3.1.1 Introducción a MIPv6

Mobile IPv6 se crea para conseguir que un nodo móvil no deba rehacer sus conexiones cada vez que cambia de dirección IP debido a un cambio de red [20].

Es decir, MIPv6 permite que otros nodos en la red, los *Correspondant Nodes* (CN), puedan comunicarse con el nodo móvil (MN) sin la necesidad de saber donde se encuentra. Mientras el MN se mantiene conectado al home-link –el enlace en su red de origen-, el comportamiento será el normal en IPv6. Si por el contrario el MN cambia de red, se activa MIPv6 y un nuevo nodo situado en su *home-link* denominado *home agent* (HA) se encarga de entunelar los paquetes hacia la nueva dirección que el MN ha adquirido en la red que visita, la denominada *Care-of-address* (CoA).

Gracias a este protocolo, las conexiones con otros nodos se mantienen mientras se cambian las direcciones IP del MN. La capa de red se encarga de que las aplicaciones que se están ejecutando en los dos nodos que mantienen una conexión vean solamente la *home address* del MN. Por tanto se puede decir que la capa IP esconde a las capas superiores la movilidad o cambio de IP del nodo. Además esto se consigue independientemente de la tecnología de enlace utilizada (IEEE 802.11, UMTS, etc.).

### 3.1.2 Funcionamiento MIPv6

Los nodos conectados al *home link* tienen un comportamiento de *routing* normal. Cuando el MN cambia de red se invoca MIPv6 (ver Fig. 3.1). Al

cambiar, se detecta un nuevo *router* por defecto y se obtiene una dirección denominada *care-of address* (CoA). Esta nueva asignación se consigue utilizando el mecanismo *stateless address autoconfiguration* [RFC4862] basado en mensajes ICMPv6 *Neighbor Discovery* (ND) [RFC4861]. Primero el MN se crea una dirección link-local a partir de su dirección MAC y luego la utiliza para enviar un *Router Solicitation* (Rtsol*).* Este mensaje fuerza que el AR envíe un *Router Advertisement* (RtAdv) gracias al cual, el MN formará su nueva dirección, la *CoA* con el prefijo de la red visitada y los últimos 64 bits de la dirección link-local del MN. Para validar esta dirección y la de link-local, se comprueba su unicidad mediante el algoritmo de *Duplicate Address Detection (DAD*). Este algoritmo envía dos *Neighbor Solicitation (*NbSol), el primero para verificar la unicidad del la dirección link-local y el segundo la de la CoA. Si nadie contesta a estos mensajes significa que ambas son únicas. A continuación el MN informa al HA de su cambio de red enviando un *Binding Update* (BU). Este mensaje MIPv6 se codifica en una nueva cabecera *Mobility header,* que se añade al paquete. El BU contiene la *home address* del MN y su *CoA*. Tiene la función de informar al HA de la dirección actual del MN. El HA guarda la *CoA* junto con la de otros nodos móviles en la *binding cache* para así poder redirigir los paquetes que tengan los MNs en cuestión como destino.

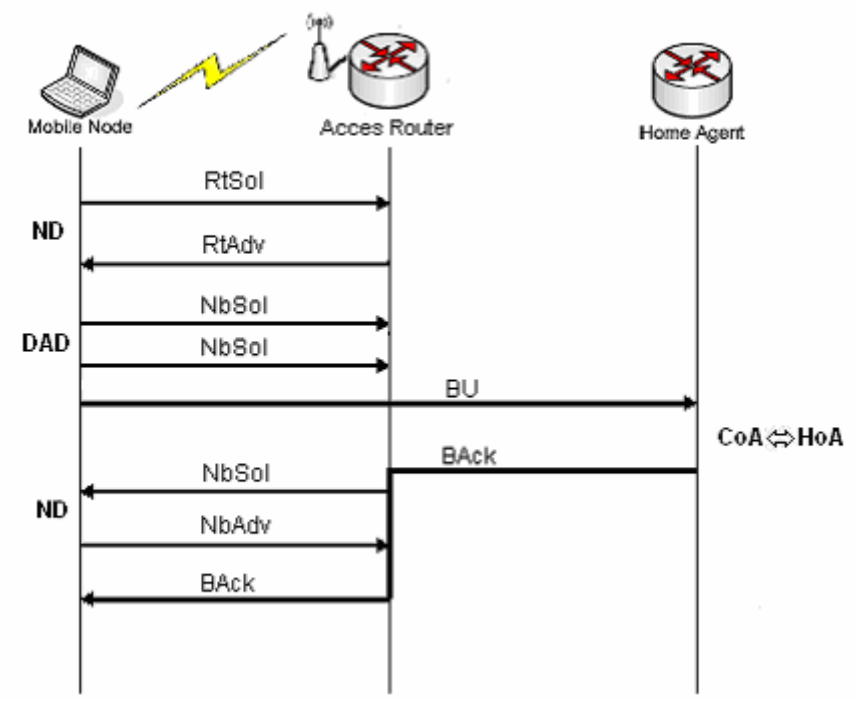


**Fig. 3.1** Traspaso MIPv6

Para que el MN pueda recibir el *Binding Ack* (Back) se aplicará de nuevo un proceso ND entre el *router* de acceso de la red visitada (AR, Access Router) y el MN. Este proceso permitirá vincular las direcciones IPv6 *link-local* con las IPv6 globales. Gracias a ello se podrá intercambiar paquetes fuera del enlace y de esta manera el *BAck* conocerá el camino hasta el MN. A partir de que el MN reciba el *Back,* el HA actuará como proxy del MN en el home link mientras el

MN permanezca en otra red. Para asegurar que los demás nodos del home link se conectan al MN a través del *Home Agent,* este les envía a todos un *proxy neighbor advertisement* con la *home address* del MN y la *home link address* utilizando la dirección *multicast*. Además el *Home Agent* defenderá la dirección del MN utilizando el mecanismo DAD para evitar que otro nodo configure la misma dirección que el MN.
Ahora el MN ya está listo para enviar o recibir paquetes de otros nodos en la red visitada (ver Fig. 3.2). El HA re-direccionará los paquetes dirigidos a la *home address* del MN utilizando un túnel. El paquete original se encapsulará en otro paquete que tendrá como dirección origen el HA y como destino la CoA del MN. Al recibir el datagrama, el MN lo des-encapsulará y obtendrá el paquete original con dirección origen el *Correspondant Node* (el nodo que quiere comunicarse con el MN) y dirección destino la *home address*. Cuando es el MN el que envía el paquete, ocurre lo mismo en sentido contrario.

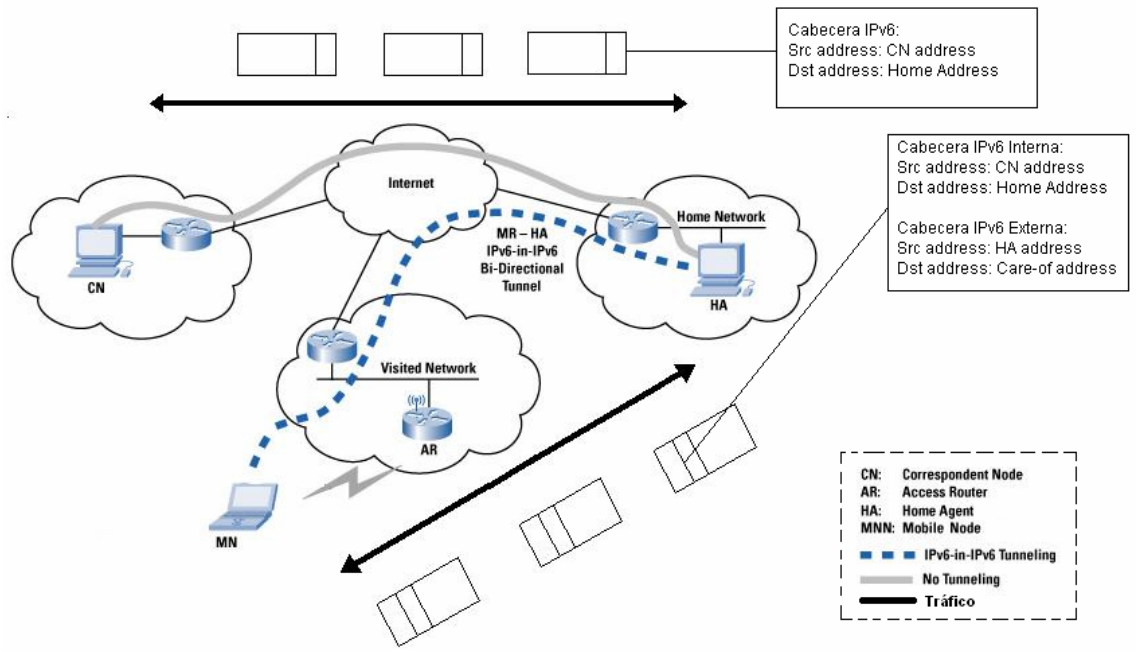


**Fig. 3.2** Esquema general del encapsulamiento en MIPv6

## 3.2 Procedimiento de traspaso para redes IEEE802.11

Un MN con una interfaz radio IEEE802.11 puede realizar un traspaso por decisión propia cuando detecta que la señal del AP en el que está asociado cae por debajo de un umbral y encuentra un AP mejor.

Para completar un traspaso existen varias fases que siguen la estación y el AP o APs en cuestión (ver Fig. 3.3). Primero hay un escaneo del medio radio, activo o pasivo, para encontrar AP candidatos a una asociación. A continuación es necesaria una asociación entre el cliente y el nuevo AP. En caso de estar habilitado un mecanismo de seguridad, el cliente deberá seguir un proceso de

autenticación. Si no lo está, el proceso se realizará igualmente pero con sistema abierto (sin intercambio de claves).

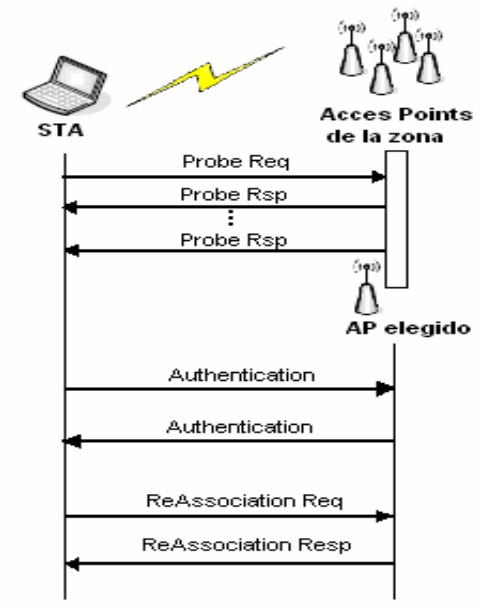


**Fig. 3.3:** Proceso 802.11 en un traspaso

En los siguientes apartados se explicarán cada una de estas fases por separado para entenderlas mejor.

### 3.2.1 Escaneo

Antes de poder utilizar un AP hay que encontrarlo. Para ello el usuario tendrá que especificar varios parámetros que a continuación detallaremos. Algunos parámetros ya están pre-configurados en el driver como es el caso del MinChannelTime y el MaxChannelTime.

- *BSSType*: especifica si el tipo de conexión es ad hoc o normal.
- *BSSID:* permite un escaneo *broadcast* para captar cualquier BSS.
- *SSID:* string de bits que identifica el nombre de la red.
- *ScanType*: especifica si el escaneo es pasivo o activo.
- *ChannelList*: lista de canales accesibles en la red 802.11 para escaneo pasivo.
- *ProbeDelay:* retardo en microsegundos antes de que se inicie un escaneo activo para evitar que se bloque el canal si no se reciben tramas.
- *MinChannelTime and MaxChannelTime*: tiempo mínimo y máximo expresado en *time units* (TU=1024μs) que se escanea un canal en particular.

A partir de estos parámetros se pueden realizar los dos tipos de escaneos que se explican a continuación.

#### *3.2.1.1 Escaneo pasivo*

El escaneo pasivo ahorra energía porque no requiere transmitir nada para realizarlo. La estación hace un barrido por cada canal del *channel list* y espera a recibir tramas *beacon*. Estos *beacons* se almacenan y se extrae información sobre el BSS que lo envió de cara a encontrar un AP perfectamente compatible al que asociarse. El problema es que este mecanismo es algo más lento ya que depende del *Beacon Interval*, típicamente unos 100ms y por tanto no permite forzar un traspaso de manera rápida.

#### *3.2.1.2 Escaneo Activo*

En el escaneo activo la estación envía tramas *Probe Request* para solicitar *Probe Responses* de una red identificada por su nombre. Este método consume más energía pero permite un traspaso más rápido y personalizado gracias a que se tiene la opción de elegir el AP al cual conectarse especificando el ESSID en el gestor de red. El procedimiento de un escaneo activo es el siguiente:

1. Elegir un canal y esperar a recibir alguna trama para saber si está en uso. También puede esperarse a que expire el *timer ProbeDelay,* tiempo mínimo antes de enviar un *Probe Request* en el cual se asegura que un canal vacío o ligeramente cargado no bloquea el escaneo.

2. Ganar el acceso al medio mediante DCF y enviar una trama *Probe Request*.

3. Esperar el tiempo *MinChannelTime* para ver si en ese intervalo se ocupa el canal. Si así eso querrá decir que el canal está siendo utilizado y se esperará al *MaxChannelTime* para procesar los *Probe Response* recibidos. Si antes de que termine el período *MinChannelTime* no se ve el canal ocupado, se cambia de canal.

### 3.2.2 Autenticación

El proceso de autenticación nos permitirá evitar intrusiones en nuestra red ya que para acceder a ella la estación deberá autenticarse con una clave. Principalmente destacaremos dos tipos de autenticación, *open-system* y *shared-key* (WEP).

La primera opción realmente no es una autenticación aunque exista el proceso de *Authentication Request* y Response. Cualquier estación podrá conectarse al AP sin que se verifique su identidad. En el *Authentication Request* hay dos elementos de información, *Authentication Algorithm Identification* que estará configurado a 0 para indicar que es *open-system* y el campo *Authentication Transaction Sequence* que valdrá 1 para indicar que es la primera trama de la secuencia.

El mecanismo *shared-key* se basa en una clave de 128 bits que existe en los dos extremos, AP y estación. Este proceso se completa con 3 tramas, la primera igual que en *open-system* pero con el bit *Authentication Algorithm Identificador* a 1. El AP contesta a esta primera trama con un *Response* que tendrá el campo Status a 0 si autoriza la autenticación. Si la autenticación se autoriza, además se añade el campo *Challenge Text* que contiene la clave de 128 bits generada por el mecanismo WEP. La autenticación se completa con una última trama que también contiene la clave WEP para que el AP la desencripte y se pueda establecer una comunicación.

La mayoría de redes actuales utilizan un sistema abierto. No obstante, suelen utilizar una encriptación WPA (Wi-Fi Protected Access, parte del IEEE802.11i) que no va a tenerse en este estudio, motivo por el cual no va ser explicada.

### 3.2.3 (Re) Asociación

Una vez completada la autenticación, las estaciones pueden asociarse o re-asociarse a un AP. Una vez asociado el AP deberá registrar la estación en la red para que las tramas destinadas a la estación se entreguen en ese AP. Un método para el registro es el envío de un ARP para que la MAC de la estación se asocie con el puerto del AP. Una estación no puede estar asociada a más de un AP. Seguidamente se explican los pasos a seguir para completar una asociación.

1. Una vez autenticada, la estación enviará un *Association Request*. Las estaciones que no se hayan autenticado recibirán una trama *Deauthentication* del AP como respuesta.

2. El AP procesa el *Association Request* y determina si concede la asociación o no. En caso afirmativo, el AP responde con un código estatus igual a 0 (exitoso) y un *Association ID* (AID). El AID es un identificador numérico usado para identificar de forma lógica la estación. Si por el contrario la asociación no es exitosa, el AP responde solamente con el código estatus y el final del procedimiento.

3. El AP que recibe tramas destinadas a una estación asociada hace de puente entre Ethernet y el medio radio o bien las almacena en un buffer si la estación se encuentra en PSM o WMM-PS (ver Capítulo 1).

La re asociación es prácticamente igual excepto que en la trama *Association Request* se añaden 6 bytes correspondientes a la dirección del antiguo AP. La re asociación se produce normalmente cuando la estación conectada a un AP percibe que la potencia de su señal es muy baja y que la de otro AP por el contrario es más alta y comparten el mismo ESSID.

## 3.3 Impacto del traspaso MIPv6 en la batería

Para poder caracterizar el consumo de los mensajes que intervienen en el traspaso MIPv6, se ha capturado sobre una maqueta con soporte de este protocolo (ver anexo III). La suma de los parámetros de cada mensaje capturado permite calcular el tamaño total del mensaje en aquellos casos en los que el protocolo no estipula un tamaño genérico. Para obtener el tamaño total de las tramas que transportan estos mensajes, a los paquetes que se muestran a continuación (Fig. 3.5 - 3.10) se les sumará las cabeceras IEEE802.11 descritas en (3.1).

1. *Router Solicitation*

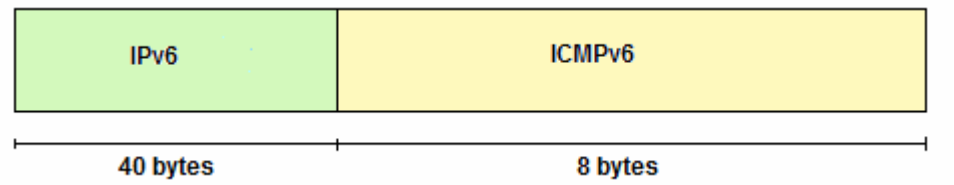


**Fig. 3.4:** Router Solicitation

2. *Router Advertisement*

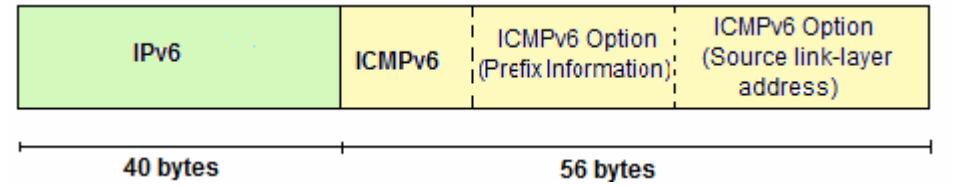


**Fig. 3.5:** Router Advertisement

3. *Neighbour Solicitation*

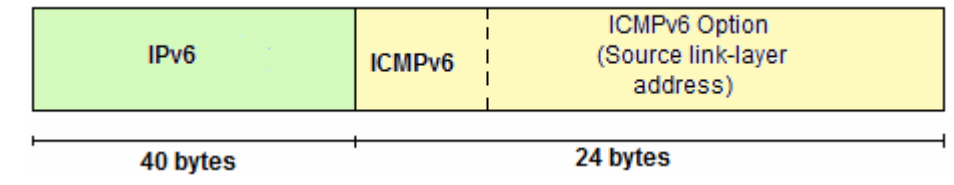


**Fig. 3.6:** Neighbour Solicitation

4. *Neighbour Advertisement*

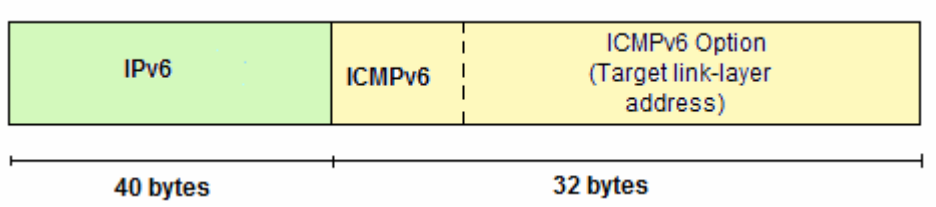


**Fig. 3.7:** Neighbor Advertisemnt

5. *Binding Update*

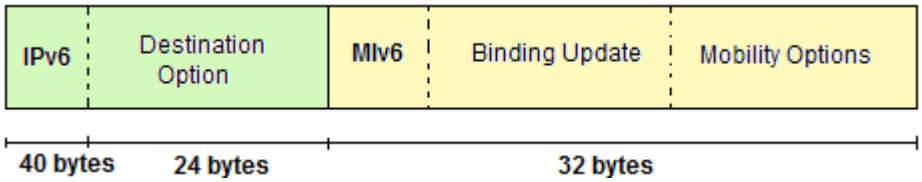


**Fig. 3.8:** Binding Update

6. *Binding Ack*

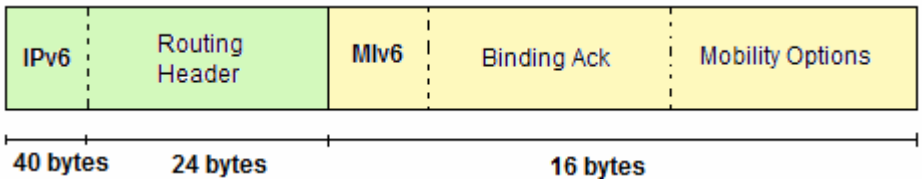


**Fig. 3.9:** Binding Ack

En el Anexo III se ha detallado la estructura de estos mensajes así como sus diferentes campos. Hay que tener en cuenta que estos mensajes son paquetes de nivel 3. Por tanto, en el análisis del impacto sobre la batería deberemos tener en cuenta tanto la cabecera MAC como el preámbulo que introduce IEEE802.11 al enviar las tramas (véase la Figura 3.11).

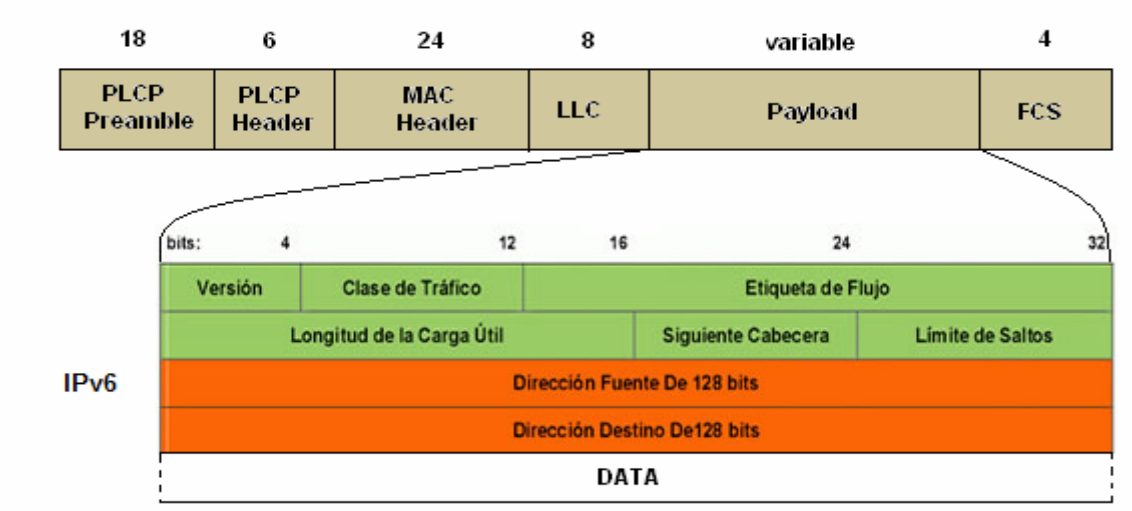


**Fig. 3.10:** IEEE802.11 transportando un paquete IPv6

### 3.3.1 Análisis general

En este apartado se a plantea un análisis del tiempo y consumo necesario para la transmisión y recepción de los distintos mensajes MIPv6 sin tener en cuenta el tiempo de espera entre ellos y utilizando valores correspondientes al protocolo IEEE802.11b como son la velocidad a la que se transmite el *Physical Layer Coverage Protocol Header* (PLCP Header), 1Mbps y la utilización del preámbulo largo por defecto. En el capítulo 3.4 se analizará con más detalle estos aspectos.

En los cálculos presentados en este capítulo, se ha utilizado el preámbulo largo de 144 bits (24bytes) aunque IEEE802.11b también puede soportar el preámbulo corto de 72 bits (9 bytes) de manera opcional. Para realizar el cálculo del tiempo en el que la STA está activa se tendrá en cuenta que el preámbulo se envía a 1 Mbps con modulación DBPSC, el cual permite adquirir la señal y sincronizar el demodulador. No obstante, en caso de utilizar el protocolo IEEE802.11g, además de utilizar preámbulo corto obligatoriamente, el PLCP H. se enviaría a una velocidad de 2 Mbps gracias a los cuatro niveles de codificación que permite DQPSK utilizado para modular la señal. Así pues, la trama puede expresarse de la siguiente forma:

**(3.1)**

***Trama 802.11*** *= Long Preamble + PLCP H. + MAC H. + LLC + FCS + paquete IPv6*
*= 18 + 6 + 24 + 8 + 4 + paquete IPv6 =* ***60 bytes + paquete IPv6***
**=** 24 bytes (**1Mbps**) + 36 bytes (**11Mbps**) + paquete IPv6 (**11Mbps**)

***Paquet IP*** *= Header + Payload =* ***40 bytes + Payload*** **(3.2)**

Donde el Payload es un parámetro variable correspondiente a los valores detallados en las Figuras 3.4 – 3.9.

Utilizando los tamaños de trama obtenidos en (3.1) el cálculo para medir el consumo de los paquetes IP en MIPv6 será el siguiente:

- Se obtiene el tiempo de transmisión o recepción de los paquetes aplicando:

*Tiempo = Longitud / Vel. Transmisión* ***(3.3)***
*Tiempo (s) = (24 bytes/1Mbps) + ((N-24 bytes) /11Mbps)*
*Tiempo (s) =((24 ·8)/1e6) + (((N-24)·8)/11e6)*
*Tiempo (ms) = ((24 ·8)/1e3) + (((N-24)·8)/11e3)*
***Tiempo (ms) = (24/125) + ((N-24)/1375)***

Donde N es el número total de bytes por trama IEEE 802.11

- Una vez obtenido el tiempo en el que la tarjeta está activa en modo *Recepción* (Rx) o *Transmisión* (Tx) se aplican los consumos indicados en la Tabal 3.1 transformado en mW los valores de corriente según *P=V·I*.

***Consumo= T·mW*** (Rx o Tx según se envíe o reciba el paquete. Ver Fig. 3.1)

- Se Divide el tiempo obtenido en ms entre 3600000 para obtenerlo en horas y así poder presentar los resultados en *mWh*.

*T(h) = T(ms)/3600000*
**Consumo (mWh)** = (Tms·mW)/3600000

Como puede observarse en la Tabla 3.1, el consumo está condicionado por la PCMCIA utilizada. Se trata de datos obtenidos a partir de un estudio empírico de consumo [5].

**Tabla 3.1.** Corriente consumida por dos tarjetas IEEE 802.11b operando a 11Mbps. *Vcc = 5.0 V*. La potencia se calcula con: *P = i·V*

| Chipset PCMCIA | Sleep (mA) | Idle (mA) | Rx (mA) | Tx (mA) |
|---|---|---|---|---|
| **ORiNOCO PC Gold** | 12 | 161 | 190 | 280 |
| **Cisco Air-PCM350** | 9 | 216 | 260 | 375 |

La diferencia de consumo a nivel de paquete o de trama que se observa en la Figura 3.11 se debe principalmente al hecho de que el preámbulo se envíe a 1 Mbps y no a 11 Mbps. La diferencia que hay entre consumo de transmisión y consumo de recepción, además de por las diferencias que hay en los valores de la tarjeta para cada uno de los estados, se debe a que el MN envía más

mensajes de los que recibe en el proceso de traspaso (ver Figura 3.1). A continuación, en la Figura 3.12 se presenta el tiempo necesario para transmitir o recibir los distintos mensajes IP que intervienen en MIPv6. Nótese que se expresa en tiempo para que la diferencia entre el consumo en modo Rx y modo Tx no enmascare los resultados. Se puede apreciar que la diferencia de tiempo de transmisión entre los distintos paquetes que intervienen en el traspaso MIPv6 no es muy significativa si la comparamos con la diferencia que hay entre el tiempo que se emplea en transmitir la información de un paquete IP y el que se emplea en transmitir la trama IEEE802.11 que contiene el paquete.

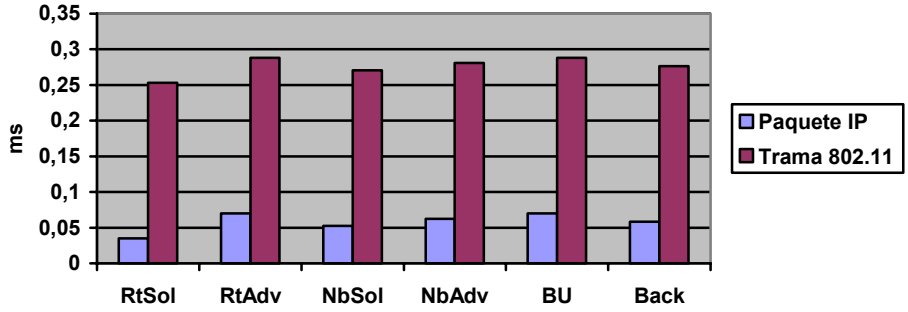


**Fig. 3.11**: Tiempo de transmisión de las tramas/paquetes IP de MIPv6

En la Figura 3.12 se presenta por separado el consumo producido por el envío o recepción de los paquetes IP con los mensajes MIPv6 y el mismo consumo cuando se tiene en cuenta la trama IEEE802.11 que los transporta. Para realizar estos cálculos y todos los próximos se ha tenido en cuenta los valores de consumo de la tarjeta *Cisco Air-PCM350* y se ha asumido que la STA se encuentra en modo activo durante todo el traspaso IEEE802.11 y MIPv6.

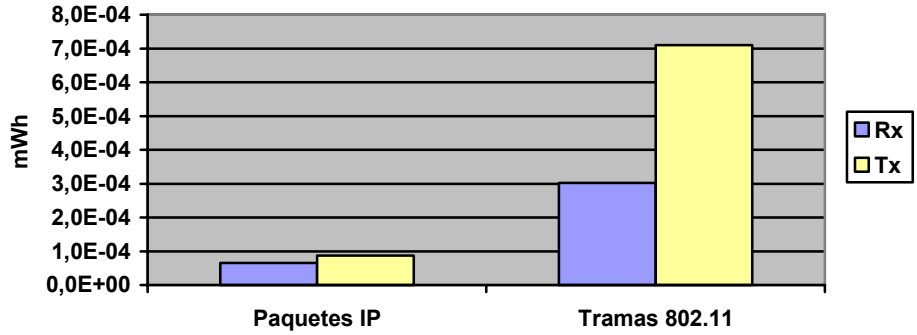


**Fig. 3.12:** Consumo producido por MIPv6

### 3.3.2 Consumo de un traspaso MIPv6 teniendo en cuenta los tiempos de espera entre mensajes

En este apartado se estudiará el impacto de los mensajes MIPv6 teniendo en cuenta el tiempo que permanece la STA esperando a recibir los mensajes. Este tiempo difiere según si utilizamos el mecanismo de unicidad DAD convencional, DAD optimista [RFC 4429], o si se deshabilita este mecanismo para no retrasar el traspaso.

En cualquier caso, se utilice DAD o DAD optimista, habrá un tiempo de espera todavía superior. Este tiempo es el que emplea el protocolo *Multicast Listener Discovery* (MLD) [RFC 3810]. Este protocolo es utilizado por los routers para descubrir la presencia de nodos que desean recibir paquetes multicast. Un nodo utilizando MLD envía un mensaje *Multicast Listener Report* para suscribirse y anunciar su dirección multicast al AR. Como el *Router Advertisement* que se ha recibido al cambiar de AR es un mensaje *multicast*, el *Multicast Listener Report Message* sufre un retardo de hasta 1 segundo antes de ser enviado para desincronizarlo de otros nodos que puedan estar respondiendo al mismo *Router Advertisement* [20]. Después de ser enviado, el MN aplica, si está implementado, el mecanismo DAD. No obstante, si eliminamos el DAD, también se eliminará el tiempo de espera que emplea el mecanismo MLD para desincronizar los nodos ya que el siguiente mensaje a enviar después de recibir el Router Advertisement es un BU, que no es *multicast.*

#### *3.3.2.1 DAD convencional*

La utilización del mecanismo DAD ralentiza mucho un traspaso provocando un mayor consumo de la batería. Esto es debido a que después que el MN aplique el ND (DAD, mensajes NbSol) no se enviará el BU hasta un tiempo de espera de algunos segundos). Con este tiempo de espera se da un margen para recibir un posible *Neighbor Advertisement* en caso de duplicación de la dirección solicitada.

En la Figura 3.13 se ha plasmado en una escala logarítmica el tiempo empleado en transmitir y recibir los paquetes y las tramas comparándolo con el tiempo de espera entre los distintos mensajes involucrados en el traspaso MIPv6 (ver Anexo IV). Los valores del tiempo de espera por desincronización y por el mecanismo DAD son variables en cada traspaso. Cabe destacar que en todos los cálculos se asume que la STA está permanentemente despierta.

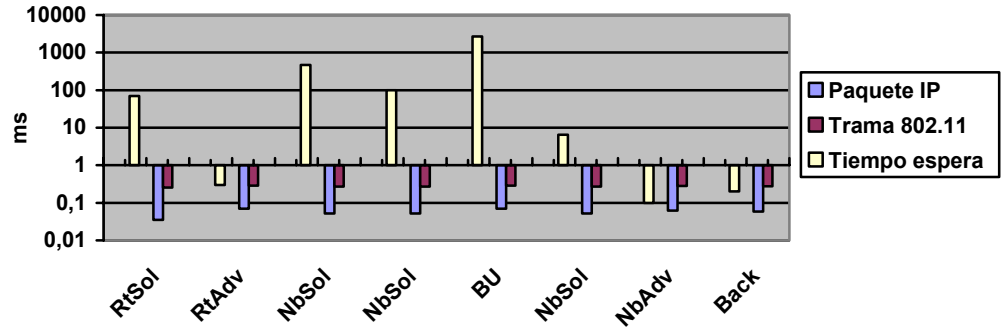


**Fig. 3.13**. Tiempos de espera entre mensajes con DAD

Como muestra la Figura 3.13 los tiempos de espera son mucho mayores que los de transmisión o recepción llegando a suponer un 99,99% del total. Por tanto, no influye solamente el tamaño de los paquetes que se envían en un traspaso MIPv6 sino el tiempo total que se emplea en completar el traspaso contando los tiempos de espera ya que en todo momento la tarjeta consume una cantidad considerable de energía.

### *3.3.2.2 DAD optimista*

Para combatir el retardo producido por el mecanismo DAD, se crea DAD optimista (ODAD), que reduce por completo el tiempo de espera después de enviar el NbSol. Esto se consigue considerando la dirección auto configurada como única mientras se está verificando su unicidad con el envío de un NbSol. Mientras no se haya verificado, esta dirección será una *Optimistic address.* Una vez comprobada su unicidad pasará a ser una dirección *Preferred* o *Deprecated* según el RFC 4429.

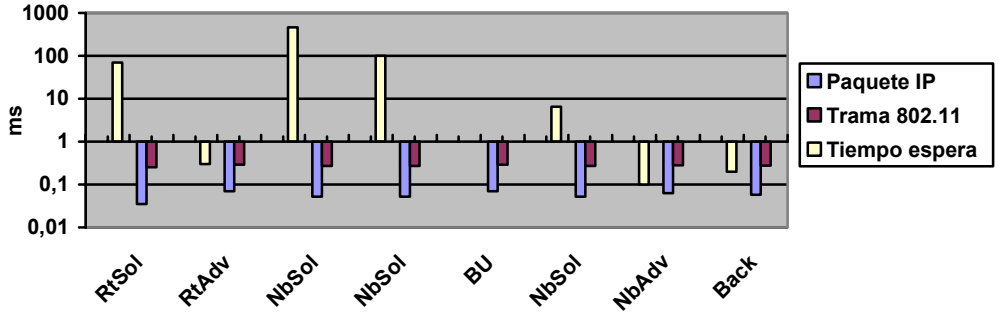


**Fig. 3.14**. Tiempos de espera entre mensajes con ODAD

### *3.3.2.3 No DAD*

El uso de DAD podría sacrificarse para mejorar el retraso. Se ha demostrado que las desventajas son mayores que las ventajas. Particularmente en una red controlada, donde es muy poco probable que exista una duplicidad de dirección. Es por ello que se analizará el consumo de un traspaso MIPv6 sin este protocolo.

Como ya se ha dicho al comienzo de este apartado, al eliminar DAD, se elimina el tiempo de espera que introducía el protocolo MLD para la desincronización del canal *multicast* . En definitiva se eliminan tanto los tiempos de espera como los mensajes ND. La reducción de consumo que deriva de esta aproximación se debe sobre todo al ahorro en el tiempo de espera (ver Fig. 3.15).

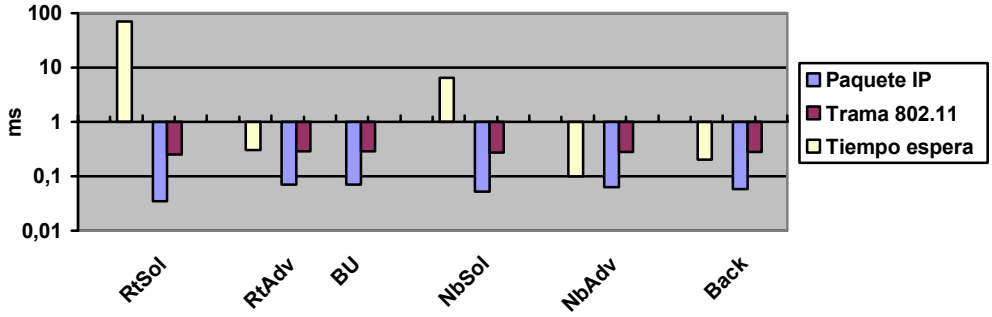


**Fig. 3.15**. Tiempos de espera entre mensajes sin DAD

### *3.3.2.4 Comparativa del consumo según mecanismo de DAD utilizado*

En la Figura 3.16 se muestra el consumo producido por un traspaso MIPv6 según el protocolo DAD aplicado.

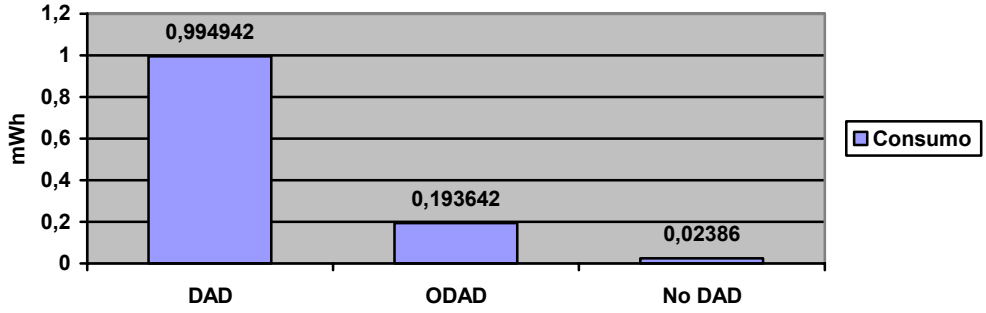


**Fig. 3.16**. Consumo traspaso MIPv6 con DAD, ODAD y No DAD

## 3.4 Impacto del traspaso IEEE802.11 en la batería

Una vez realizado el estudio del consumo de un traspaso de nivel 3, vamos a analizar el consumo producido por los menajes que intervienen en el traspaso de nivel físico que hemos explicado en el apartado 3.1.3.

En primer lugar, para tener una idea de cómo está formado un PPDU (*Presentation Protocol Data Unit*) que es lo que la STA recibe y transmite en cualquier transmisión IEEE 802.11, en la Figura 3.17 hemos dividido cada uno de los campos que lo forman.

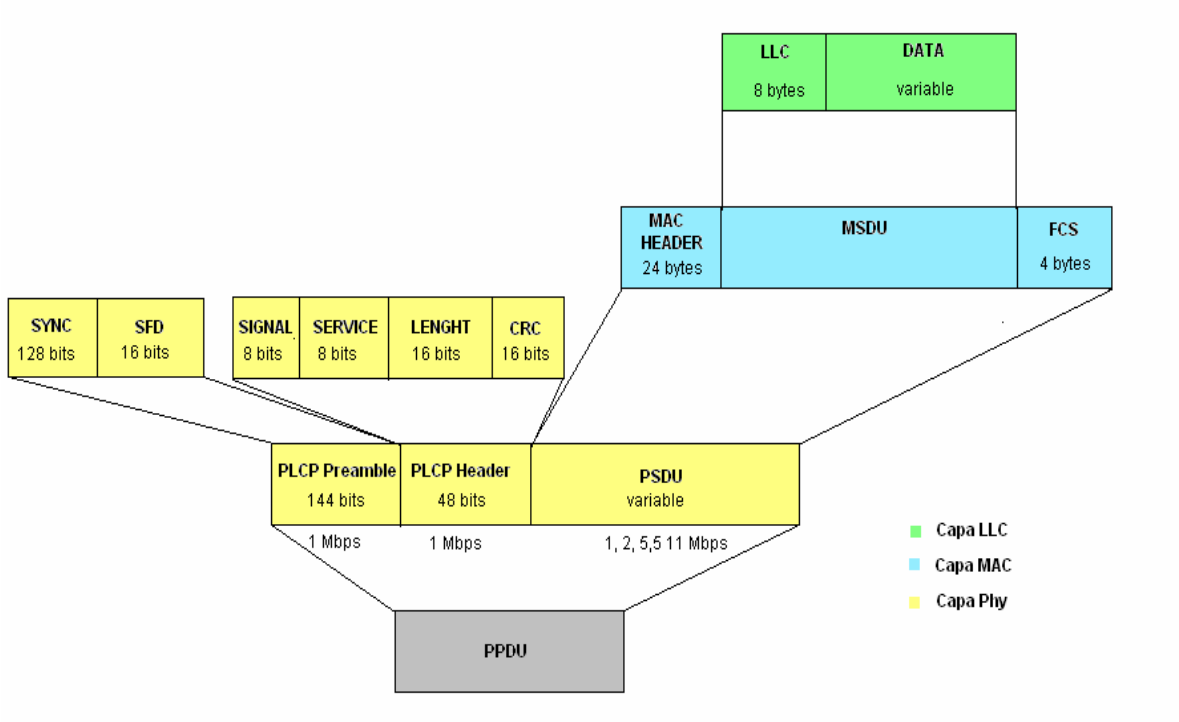


**Fig. 3.17**: PPDU 802.11

Es el campo *MSDU* (*Mac Service Data Unit*) del *PSDU* (*Physical layer Data Unit*) el que variará según el mensaje que se reciba o envié durante el traspaso. El tamaño del PLCP *Preamble* corresponde al preámbulo largo que se ha utilizado para realizar los cálculos.

### 3.4.1 Análisis general

A diferencia del apartado 3.23, en este, se ha realizado el cálculo necesario para obtener el tamaño de una trama 802.11 sin tener en cuenta la cabecera LLC. Esto es debido a que las tramas de escaneo y asociación no contienen paquetes de niveles superiores.

***Trama 802.11 =*** *Long Preamble + PLCP H. + MAC H. + FCS + Tagged parameters = 18 + 6 + 24 + 4 + Tagged parameters =* ***(3.4)***
***52 bytes + Tagged parameters***

***Trama MAC*** *= MAC H. + FCS + Tagged parameters = 24 + 4 + variable =*
***28 bytes + Tagged parameters***

Los *Tagged parameters* son campos que contienen las tramas IEEE802.11 y difieren según la tarjeta y AP utilizados y las opciones de las que disponen. Una de las posibles opciones sería si soporta uAPSD o no y todos los parámetros necesarios para implementarlo. La Figura 3.18 y 3.19 permiten tener una idea de la estructura de cada uno de los mensajes necesarios para la asociación descritos en el apartado 3.1.3.

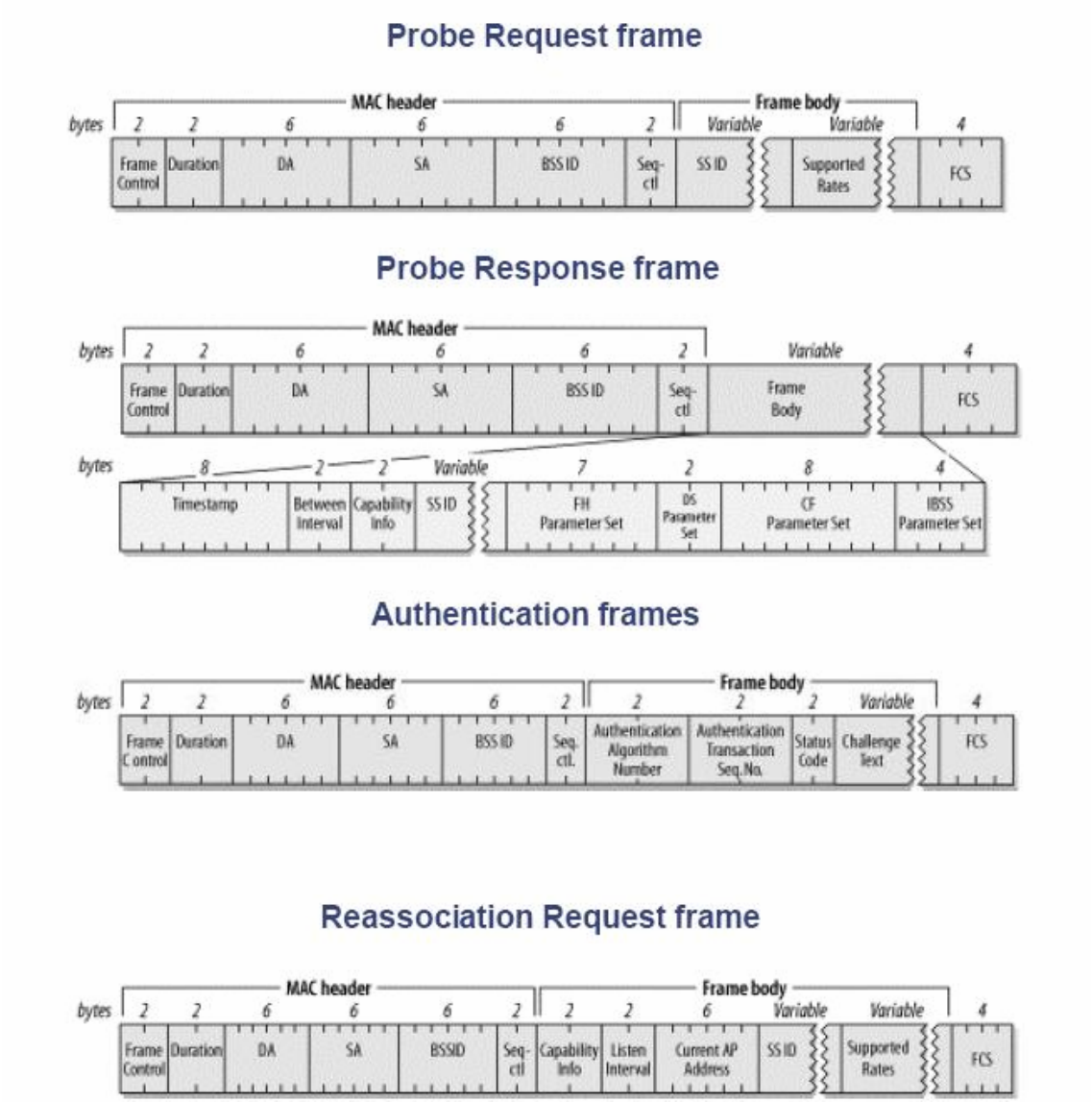


**Fig. 3.18:** Estructura de los mensajes de asociación a un AP en 802.11

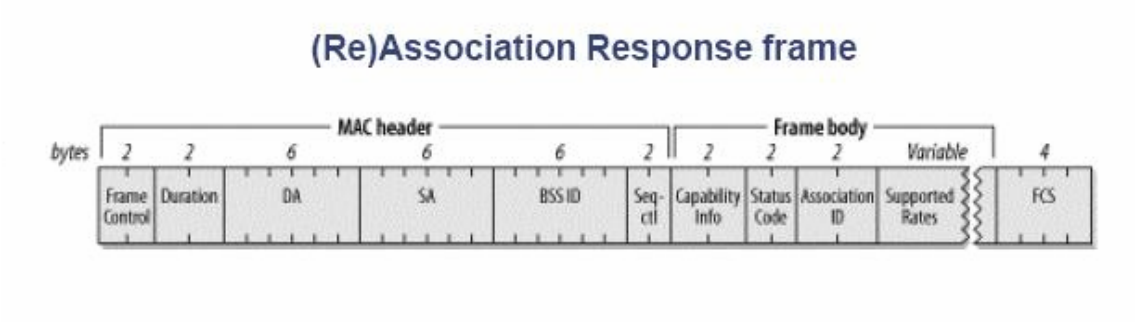


**Fig. 3.18b:** Continuación de la estructura de los mensajes de asociación a un AP en 802.11

La única diferencia respecto con el cálculo realizado para las tramas que transportan paquetes IPv6 (ver fórmula 3.2) es que en estas, tanto el preámbulo como el resto de los datos se envía a 1Mbps. El motivo por el cual el proceso de asociación o de traspaso se hace a una tasa de 1Mbps es para minimizar la probabilidad de pérdida de estos mensajes. Así, utilizando los tamaños de trama obtenidos en 3.4 y con la ayuda de las capturas realizadas con *kismet* y *ethereal* sobre un demostrador (ver Anexo III) se aplicarán los siguientes cálculos para obtener el consumo del traspaso de nivel MAC:

*Tiempo (s) = N bytes/1Mbps* **(3.5)**
*Tiempo (s) = (N ·8)/1e6*
*Tiempo (ms) = (N · 8)/ 1e3*
***Tiempo (ms) = N / 125***

Y el consumo se calculará del mismo modo que en (3.2)

En la Figura 3.19 se representa el consumo de los mensajes en un traspaso IEEE802.11. Esta vez, las dos barras distinguen las tramas IEEE 802.11 con y sin preámbulo para mostrar la importancia de este factor en cuanto al consumo de energía. A continuación en la Figura 3.20 se observa la duración de los mensajes por separado como ya se hizo en 3.3.1

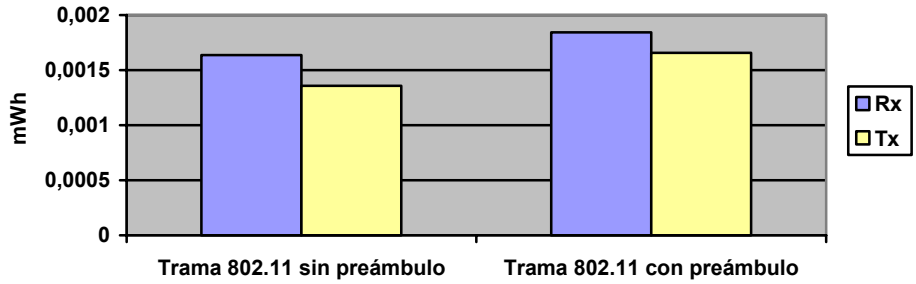


**Fig. 3.19**. Consumo de mensajes en un traspaso IEEE 802.11

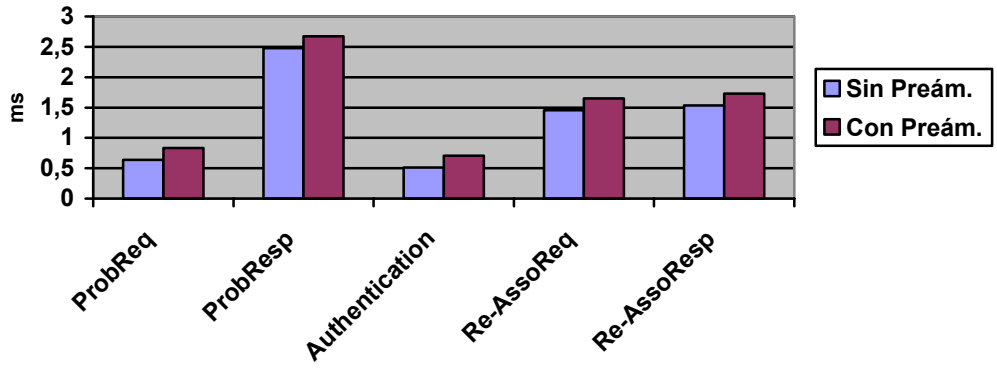


**Fig. 3.20.** Duración de las tramas IEEE 802.11 con y sin preámbulo

En la Figura 3.20 se puede ver que el preámbulo ahora afecta muy poco a la duración del tiempo en que la STA tiene que estar despierta para transmitir la trama. Es lógico, ya que ahora toda la trama se envía a 1 Mbps. Puede observarse también que el *Probe Response* es el mensaje de mayor longitud ya que contiene todos los parámetros necesarios para que el cliente se conecte a la red.

### 3.4.2 Consumo de un traspaso IEEE802.11 teniendo en cuenta los tiempos de espera entre mensajes

El tiempo de espera entre los mensajes durante el traspaso también afecta al consumo de la batería. En la Figura 3.21 se muestra el consumo producido por los mensajes que intervienen en el traspaso teniendo en cuenta estos tiempos de espera.

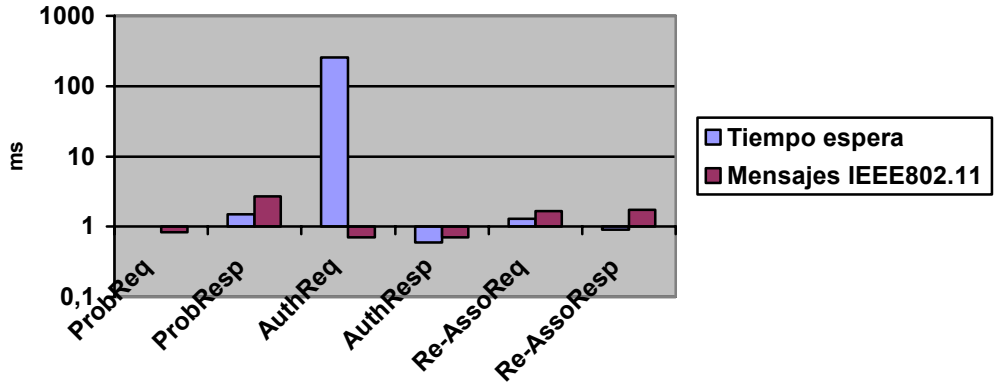


**Fig. 3.22:** Tiempo entre tramas IEEE 802.11 en un traspaso

Finalmente notar que el consumo representado en la Figura 3.19 no es del todo real ya que parte de la base que sólo se escanea un canal en el que solamente hay un AP candidato con el que asociarse. Además, no se tienen en cuenta el *MinChannelTime* y el *MaxChannelTime* que son los tiempos durante los cuales

la STA está tanto en modo Tx/Rx como en modo idle (activo pero sin recibir ni transmitir) según se reciban o no *ProbeResponses*. En el siguiente apartado se va a analizar más en profundidad este proceso de escaneo para caracterizar su consumo de cara a estimar su influencia en el consumo total de energía respecto a los demás procesos.

### 3.4.3 Consumo producido por el escaneo

Para caracterizar el consumo producido por el escaneo activo de la STA deberá tenerse en cuenta las distintas fases que componen este procedimiento. En primer lugar se envía un *Probe Request* en el primero de los canales que utiliza IEEE802.11 para forzar una respuesta de los posibles APs que operen en ese canal. Durante el *MinChannelTime* la STA permanecerá en estado *idle*. Si hay actividad en ese intervalo de tiempo, se esperará hasta llegar a un tiempo *MaxChannelTime* para que los APs puedan competir por el canal y enviar el *Probe Response*. En caso de no recibir nada en este período de tiempo, se cambiará de canal y repetirá el mismo proceso hasta llegar al 13[1].

En la Figura 3.22 se representa este proceso para N canales. El parámetro *CS&T* es la suma del tiempo que emplea la STA en cambiar de canal y el *overhead* de la transmisión. Este valor depende del hardware y en cualquier caso es pequeño en relación a los otros, por lo cual no se ha tendido en cuenta para realizar los cálculos que se presentan a continuación.

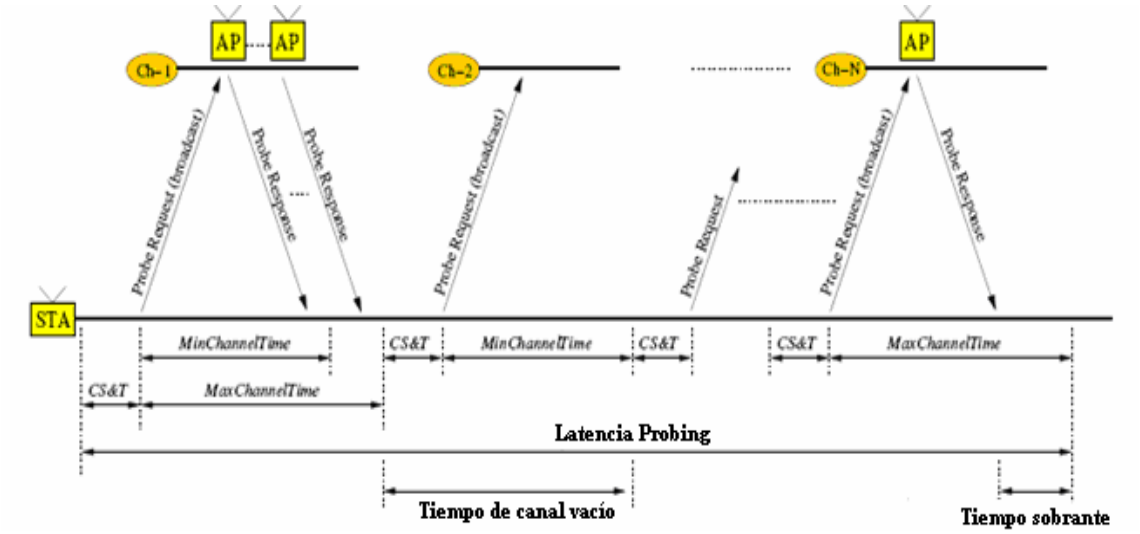


**Fig. 3.22:** Proceso de escaneo de N canales

[1] En España el número de canales disponibles para el uso de IEEE802.11 en la banda de 2.4Ghz es de 13

El *MinChannelTime* y *MaxChannelTime* vienen definidos por el fabricante y se estiman de forma que se consiga un mínimo tiempo de escaneo pero con el suficiente tiempo para que los APs puedan competir por el canal. Si descuidamos el tiempo de propagación y el de generación del *Probe response* obtenemos que el tiempo máximo de acceso al canal sea el siguiente:

*MinChannelTime = DIFS + (aCWmin · aSlotTime)*

Donde DIFS es el *Distributed InterFrame Space*, *aCWmin* es el máximo número de slots en la mínima ventana de contención y *aSlotTime* es el tamaño del slot. Si se substituyen estos valores por los de la Tabla 3.2 pertenecientes a 802.11b obtenemos 670µs. Tanto el *MinChannelTime* como el *MaxChannelTiem* son expresados en *Time Units* (TU) equivalentes a 1024µs. Por tanto el *MinChannelTime* valdrá 1 TU como mínimo. Los valores obtenidos empíricamente para la tarjeta *Cisco Aironet 350* en [13] son de unos 7 TU (7ms).

**Tabal 3.2:** Valores para el acceso al canal en 802.11b

| | |
|---|---|
| **aSlotTime** | 20 µs |
| **aCWmin** | 31slots |
| **DIFS** | 50 µs |

La obtención del *MaxChannelTime* idóneo es algo más complicado ya que depende del número de estaciones conectadas al AP. Los valores típicos del *MaxChannelTime* suelen ser de 10 TU [13]. Para hacer un análisis más detallado del impacto que tiene la fase del escaneo se realizan una serie de cálculos basándose en el número de canales ocupados y el número de APs al alcance operando en dichos canales. Como valores de referencia se utilizarán los obtenidos en los análisis empíricos de [15], que son 7 ms para el *MinChannelTime* y 11 ms para el *MaxChannelTime.*

Se puede caracterizar el tiempo total del escaneo de N canales según la siguiente fórmula:

**(3.5)**

$$T_N = \sum_{i=1}^{N} [\,(MinCT + f(x)\cdot(MaxCT-MinCT)\,] \qquad N\cdot MinCT \le T_N \le N\cdot MaxCT$$

$$f(x)\begin{cases} 1 & \text{Canal ocupado} \\ 0 \neq & \text{Canal ocuapdo} \end{cases}$$

Donde *MinCT = MinChannelTime* y *MaxCT = MaxChannelTime.* Durante estos tiempos el consumo dependerá de si se reciben *Probe Responses* y de cuantos

se reciben. El tiempo durante el cual no se reciben *Probe Responses,* la STA permanecerá en un estado *idle*. Por el contrario, durante el tiempo de recepción de los *Probe Responses* la STA consumirá más energía por cambiar a un estado de recepción. En la formula (3.6) se expresa el consumo de un escaneo de *N* canales.

**(3.6)**

$$C_N = \sum_{i=1}^{N} [\, CPRq + (MinCT - y(x)) \cdot Idle + (f(x) \cdot CPRp) + f(x)\,(\,(\,(MaxCT - MinCT) - ((n-1) \cdot y(x))\,) \cdot Idle + (n-1) \cdot CPRp)\,]$$

$$f(x) \begin{cases} 1 & \text{Canal ocupado} \\ 0 \neq & \text{Canal ocuapdo} \end{cases}$$

$$y(x) \begin{cases} TPRp & \text{Canal ocupado} \\ 0 \neq & \text{Canal ocuapdo} \end{cases}$$

MinCT = 7ms
MaxCT = 11ms
CPRp = 0,000896 mWh
CPRq = 0,000333 mWh
TPRp = 2,48 ms
n = nº AP del canal

Donde *CPRp* es el consumo de un *Probe Response, CPRq* es el consumo de un Probe Request y *TPRp* es el tiempo de transmisión de un *Probe Response*, Estos valores se han obtenido de los cálculos en (3.2). La variable *n* cambiará según el canal que se esté escaneando ya que representa el número de AP conectados a un canal específico. Para simplificar la sumatoria $C_N$, la hemos transformado en un cálculo (*Cs*) en el cual tenemos 13 canales en total (según la legislación española) y unos parámetros variables según el número de canales activos y APs por canal (3.7).

**(3.7)**

- *Si AP = 1* (*n* en la fórmula 3.6)

$$Cs = 13 \cdot CPRq + (((MinCT \cdot (14 - \mathbf{c}) + [(\mathbf{c} \cdot (MinCT - TPRp))) \cdot Idle] / 3600000 + n \cdot CPRp$$

- *Si APs > 1*

$$Cs = 13 \cdot CPRq + (((MinCT \cdot (14 - \mathbf{c}) + [(\mathbf{c} \cdot (MinCT - TPRp))) \cdot Idle] / 3600000 + \mathbf{n} \cdot CPRp + [(\mathbf{c} \cdot (MaxCT - MinCT - ((\mathbf{a} - 1) \cdot TPRp))) \cdot Idle] / 3600000$$

***c**=canales activos*
***a**=APs por canal*
***n**=número de Probe Requests= c·a*

Se ha fijado un número máximo de APs a los que puede acceder un cliente. Un número de 3 APs por canal nos parece un valor propio para una zona con una densidad alta de AP. Por otra parte, hemos fijado un número máximo de canales ocupados de 3 ya que debido a la interferencia de canal adyacente, en la mayoría de casos, se utilizan los canales 1, 6 y 11. Por tanto, según estos valores, en el escaneo tendremos acceso hasta 9 APs. En la Figura 3.23 se presenta el consumo en relación al número de canales utilizados y el número

de AP por canal. Nótese que se ha generalizado a un número equivalente de AP por canal aunque podría darse el caso de que el número de AP en cada canal fuera diferente. Se ha obviado esta posibilidad por la multitud de casos posibles. Se puede ver que a partir de un canal ocupado, al pasar de uno a dos AP, el consumo crece de forma exponencial. Esto es debido a que con más de un AP el tiempo empleado en esperar *Probe Responses* pasa de valer MinCT a MaxCT.

La Figura 3.23 muestra una estimación del consumo del proceso de escaneo según el número de canales ocupados y el número de APs por canal.

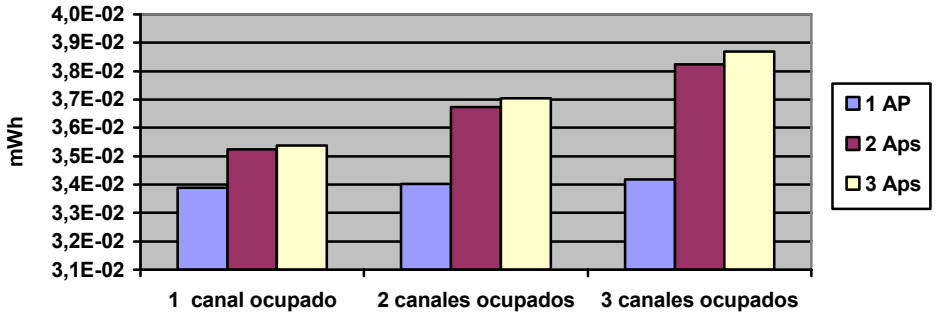


**Fig. 3.23**: Consumo del escaneo en función de los canales ocupados y el número de APs por canal

Para poder apreciar mejor el consumo de las diferentes fases del traspaso MAC de IEEE802.11 y la importancia del escaneo, en la Figura 3.24 se han representado los consumos de cada una de las fases (escaneo, autenticación y re-asociación) sobre el total. En la primera figura se muestra el porcentaje del consumo de cada fase sin tener en cuenta los tiempos de espera entra cada una de ellas. En la segunda figura sí se han tenido en cuenta estos tiempos de espera asignándole a cada fase el tiempo de espera anterior a su inicio. Para el escaneo se ha tomado el caso en el que hay un solo canal ocupado con un único AP candidato.

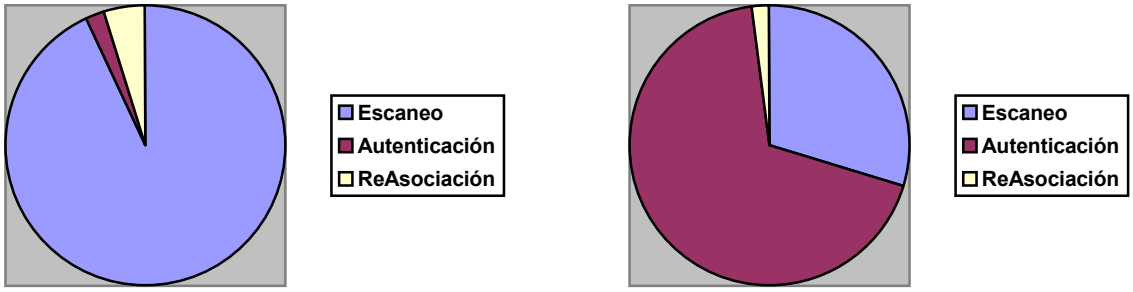


**Fig. 3.24**. Consumo sobre el total de las diferentes fases de un traspaso IEEE802.11 sin tiempos de espera y con ellos.

En la primera de las figuras vemos como el escaneo, sin contar los tiempos de espera, es el proceso que más recursos de energía consume. En la segunda figura se aprecia un importante consumo en la fase de autenticación. No obstante, este consumo es debido al tiempo de espera que hay antes de enviar un *Authentication Request* y forma parte del estado de Autenticación porque se ha decidido así. Ese tiempo de espera podría asignarse al proceso de escaneo si se decidiera que el tiempo de espera después de una fase forma parte de esta misma.

## 3.5 Conclusiones y caso práctico

A lo largo de este capítulo se han analizado distintos aspectos del consumo producido por un traspaso utilizando el protocolo de MIPv6 en una red IEEE802.11. Después de observar los resultados tanto del traspaso a nivel de red como a nivel de enlace, en la Figura 3.25 se presenta una comparativa del consumo producido por estos traspasos. En ella se representan el mejor y el peor caso para el traspaso IEEE802.11 tomando los valores de consumo producidos por el traspaso a nivel de enlace con 1 AP en solo un canal ocupado y 3 APs candidatos para una asociación en cada uno de los canales 1, 6 y 11. Además, se ha comparado con un traspaso IP aplicando DAD, ODAD y sin aplicar DAD.

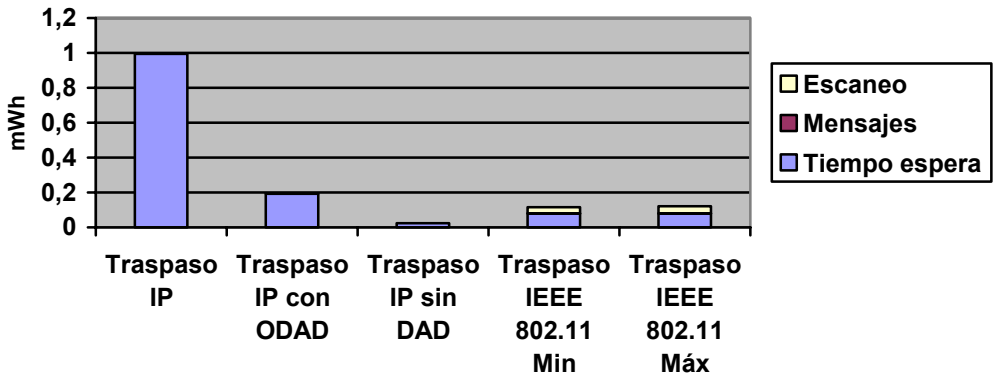


**Fig. 3.25**: Comparativa de consumo de traspaso IEEE802.11 e IP

En la figura se observa una diferencia significativa del consumo producido por un traspaso a nivel de enlace y uno de red en el mejor de los casos, i.e. sin DAD. Independientemente de si hay una situación de muchos APs candidatos repartidos en varios canales, el escaneo producirá un consumo de la STA en modo *Idle* durante 98 ms como mínimo, correspondientes al *MinChannelTime* de 13 canales. Además, el estado *Idle* del STA con la tarjeta utilizada para el análisis (Tabla 3.3) consigue tan solo una reducción de un 16 % del consumo respecto al estado de Recepción. Gracias a protocolos como el IEEE802.11k que permiten que un STA conozca los AP vecinos se puede reducir mucho el consumo del escaneo o incluso eliminarlo.

No obstante, el factor que más contribuye al consumo de batería son los tiempos de espera entre mensajes. Este consumo podría reducirse en el traspaso MIPv6 prescindiendo de DAD si se asume un traspaso en un entorno controlado. Además, si se aplicase el mecanismo PSM durante los tiempos de espera, también se mejoraría la duración de la batería. El problema de PSM es que, como se ha visto en el Capítulo 1, PSM puede ralentizar el traspaso de red. Sin embargo, para el traspaso IEEE802.11, PSM no puede utilizarse ya que debe realizarse en modo activo.

Cuando se realiza un traspaso MIPv6, primero ha tenido que realizarse un traspaso IEEE802.11. Por tanto, el consumo real producido por un traspaso MIPv6 será la suma de los dos traspasos, el traspaso MAC y el IP. Para ver el impacto que supone el consumo de energía de dicho traspaso se ha aproximado el número de traspasos que podría realizarse para gastar el 1% de la batería de un teléfono WiFi SMC WSKP100 (1200mAh) y de un portátil Dell XPS de 9 celdas (7200 mAh). Se ha asumido que no se utiliza 802.11k, que el escaneo consume su valor mínimo y que el mecanismo DAD se suprime. Para contrastar los resultados, también se asumirá la utilización de DAD convencional y un escaneo de consumo máximo.

Como es evidente, el teléfono WiFi no soporta la PCMCIA Cisco Aironet 350, utilizada para realizar los cálculos de este TFC. Es por eso que se han cogido los valores de consumo correspondientes a una miniPCI Atheros AR6002 [21]. Gracias a la herramienta programada que se presenta en el Anexo VI se ha podido obtener fácilmente el valor de un traspaso realizado con dicha miniPCI.

Primero se calcula el consumo en mAh para un traspaso y para cada una de las tarjetas, obteniendo los siguientes resultados:

- Cisco Aironet 350:

-Consumo Mínimo:

Consumo traspaso MIPv6 sin DAD + Consumo traspaso IEEE802.11 con escaneo mínimo = 0,138735 mWh

0,13874 mWh / 5 V = **0,027748 mAh**

-Consumo Máximo:

Consumo traspaso MIPv6 con DAD + Consumo traspaso IEEE802.11 con escaneo máximo = 1,11472mWh

1,11472 mWh/ 5 V = **0,222944 mAh**

- Atheros AR6002:

-Consumo Mínimo:

Consumo traspaso MIPv6 sin DAD + Consumo traspaso IEEE802.11 con escaneo mínimo = 0,00055 mWh

0,00055 mWh / 5 V = **0,00011 mAh**

-Consumo Máximo:

Consumo traspaso MIPv6 con DAD + Consumo traspaso IEEE802.11 con escaneo máximo = 0,00225 mWh

0,00225 mWh / 5 V = **0,00045 mAh**

Y a partir de estos valores ya pueden calcularse el número de traspasos:

- Dell XPS 9 celdas: 1% → (7200 mAh) = 72 mAh

  Consumo Mínimo: 72 mAh / 0,027747 mAh = 2594 traspasos
  Consumo Máximo: 72 mAh / 0,222944 mAh = 322 traspasos

- SMC WSKP100: 1% → (1200mAh) = 12 mAh

  Consumo Mínimo: 12 mAh / 0,00011 mAh = 109090 traspasos
  Consumo Máximo: 12 mAh / 0,00045 mAh = 26666 traspasos

El número de traspasos obtenidos para el teléfono Wi-Fi es muy superior, aun teniendo una capacidad de batería inferior a la de un portátil. El motivo es la elección de la tarjeta Atheros AR6002 que además de consumir la duodécima parte en transmisión y la octava parte en recepción que la tarjeta Cisco, tiene un consumo en estado Idle inferior al que tiene la tarjeta Cisco Arionet 350 en estado *sleep* de PSM (ver Tabla 3.3). Por este motivo, los tiempos de espera entre mensajes no tienen el impacto en el consumo de energía que puedan tener otras tarjetas. También se destaca la reducción de traspasos posibles que sufren los terminales con las características de consumo máximo. Un 87,5 % menos para el portátil Dell XPS de 9 celdas con tarjeta Cisco y un 76,5 % menos para el teléfono con tarjeta Atheros.

**Tabla 3.3**. Consumos de tarjeta Cisco Aironet 350 y Atheros AR6002

| **Chipset** | Sleep (mW) | Idle (mW) | Rx (mW) | Tx (mW) |
|---|---|---|---|---|
| Cisco Air-PCM350 | 35 | 1080 | 1300 | 1875 |
| Atheros AR6002 | 0.06 | 1.42 | 140 | 140 |

Para concluir, con los datos obtenidos en este capítulo, se podría afirmar que dado el bajo impacto del consumo de los traspasos en las baterías de los terminales y la diferencia entre el consumo de un traspaso MIPv6 sin DAD respecto a un traspaso IEEE802.11, la movilidad no debería ser un factor clave en la duración de las baterías.

# CAPÍTULO 4. Conclusiones

## 4.1. Fases del proyecto y resultados obtenidos

En este apartado se va a explicar las fases por las que ha pasado el proyecto así como los principales resultados obtenidos.

En la primera fase, se estudió la certificación WMM-PS para luego investigar sobre los dispositivos que soportan uAPSD. Se hizo un listado detallado sobre sus características que puede encontrarse en el Anexo II. Posteriormente se escogieron los dispositivos que más convenían por prestaciones y facilidades de configuración. En paralelo, se realizaron varios *scripts* que permitían la monitorización de la batería. Estos *scripts* pueden encontrarse en el Anexo I junto con el análisis de otros programas de monitorización de la batería.

En la segunda fase, mientras se esperaba la llegada de los dispositivos, se puso en marcha la maqueta MIPv6 configurando de nuevo todos los parámetros de red necesarios. Poner la maqueta en marcha fue una tarea complicada debido a problemas de compilación del *kernel*. Con la maqueta ya en marcha se realizaron pruebas para verificar el correcto funcionamiento del protocolo. Una vez realizadas, se procedió al análisis de un traspaso a nivel de red mediante capturas durante el proceso. Estas capturas se encuentran en el Anexo IV.

La tercera fase constó del análisis de las tramas del traspaso de red y del traspaso de enlace para estudiar su impacto en la batería. Para ellos se hizo un estudio centrado en la fase de escaneo en el traspaso IEEE802.11 para luego calcular una serie de fórmulas que permitieran extraer el consumo en función de los canales ocupados y los AP presentes en cada canal. Todo ello se plasmó en una tabla Excel que puede verse en el anexo V, mientras que el resumen de los resultados más importantes se presenta en el capítulo 3.

En la cuarta fase del proyecto se estudió en profundidad el protocolo uAPSD para entender su funcionamiento, así como otras funcionalidades que podrían contribuir al ahorro de la batería. Como resultado de esta fase, en el capítulo 1 se explica el funcionamiento teórico de uAPSD y las configuraciones necesarias en el anexo III.

La quinta fase fue la más larga. A partir del estudio realizado en la fase anterior se prepararon y realizaron las pruebas de medición de batería, cuyos resultados se reflejan en el capítulo 2. Mientras se realizaban estas mediciones, se programó una aplicación en VBA que a partir de de las tablas Excel mencionadas anteriormente permite medir el impacto en la batería de los traspasos de enlace y de red. Esta aplicación se presenta en el Anexo VI.

El trabajo realizado en estas fases ha permitido cumplir todos los objetivos planteados en la introducción de este TFC. Se han adquirido los conocimientos

teóricos de los protocolos a estudiar, y el impacto de éstos en la batería se ha estudiado de manera teórica y práctica. Respecto a los resultados empíricos cabe decir que su exactitud está limitada por el mecanismo de medida empleado. Además, estos resultados deben asociarse a las pruebas realizadas, que han tratado de ser lo más representativas posibles, y para el caso particular de uAPSD al *hardware* empleado, con un soporte para Linux todavía limitado. No obstante, después de contrastar los resultados con los consumos calculados teóricamente, pueden darse como válidos, como mínimo en las tendencias u órdenes de magnitud que indican.

Los resultados más significativos son los siguientes. Se ha comprobado que *ARP cache* reduce el impacto en la batería, pero tan solo un 0,00506%. ARP *caching* reduce el impacto dependiendo de la densidad de tráfico en la red, se ha conseguido hasta un 1% de ahorro según los resultados empíricos. Éramos conocedores del ahorro que puede suponer el incremento del DTIM a partir de algunos artículos publicados. Se ha comprobado empíricamente que es así. Cuando DTIM = 10, el ahorro es máximo, un 0,87 % más de capacidad que cuando DTIM =1. Respecto a uAPSD se esperaba algo más de ahorro en su comprobación empírica. El ahorro mayor medido ha sido de algo menos de un 1%. Una hipótesis factible para justificar este resultad es que su ahorro queda compensado por el *overhead* que introduce su uso conjunto con WMM. Por último, a diferencia de lo que se pensaba cuando se planteó este TFC, se ha comprobado que el impacto de los traspasos en la red es pequeño en el consumo de las baterías. Tanto es así que con un teléfono WiFi de última generación puede llegar a realizarse 109090 traspasos gastando tan solo el 1% de su batería.

## 4.2. Líneas futuras

Aunque se ha intentado abarcar los parámetros más influyentes en el consumo de energía de la tarjeta inalámbrica, se podría profundizar algo más.

En primer lugar, debido a que uAPSD es relativamente nuevo, y que está presente en pocos dispositivos solo se ha podido analizar su impacto para un determinado modelo de tarjeta y AP. Y para el caso de la tarjeta con unos *drivers* con escasas posibilidades de configuración. En un futuro, cuando el protocolo haya sido sometido a mayor número de pruebas, los controladores sean completos y se disponga de un mayor número de dispositivos que lo soportaran, podría realizarse un estudio de mayor profundidad. También, se podría realizar un estudio del comportamiento de una misma batería para distintos dispositivos con características similares de consumo. Dependiendo de la marca de la tarjeta, los campos que permiten la activación y configuración de uAPSD varían. Esto se ha comprobado porque algunos campos QoS dependen de la pestaña *tagged flags* que son propios de cada dispositivo.

Otro análisis más detallado podría ser realizado comprobando el funcionamiento del protocolo en terminales VoIP reales que generaran tráfico RTP. En este proyecto se ha simulado este tipo de flujo mediante generadores de tráfico y aplicando mapeos específicos en el AP para marcar el CoS.

En cuanto a la parte relativa al consumo debida a la movilidad, sería conveniente realizar medidas de carácter empírico para ver si se corresponden con el consumo teórico aquí calculado. De ser así, propuestas como la de [2], de utilización de IP *paging*, no serían justificadas des del punto de vista de consumo de batería.

Para terminar, la aplicación creada con VBA servirá en un futuro para analizar el comportamiento de distintas tarjetas. No obstante, si las tarjetas soportan el protocolo IEEE802.11n debería actualizarse la aplicación para, por ejemplo calcular los tiempos de trama con velocidades superiores (hasta 300 Mbps) y lo mismo con el preámbulo. La aplicación también podría ampliarse con nuevas opciones, como por ejemplo la de comparar consumos para diferentes pruebas de dos tarjetas diferentes.

Finalmente respecto a las medidas empíricas, ya se ha comentado las limitaciones de las herramientas utilizadas. Sería interesante utilizar alguna técnica de medición directa mediante el uso de un *hardware* específico de medición para disponer de medidas de consumo más precisas. También el número de medidas realizadas por cada prueba podría aumentarse para observar si sus valores son estables, Y lo mismo con el número y tipo de pruebas. Aquí, por limitaciones de tiempo, debidas a la propia duración del TFC, se ha realizado las que se han considerado más relevantes.

## 4.3. Impacto medioambiental del proyecto

Este proyecto ha tenido un impacto en el medioambiente mínimo. La maqueta de MIPv6 ha sido reutilizada de un proyecto anterior y por tanto no ha habido que invertir en nuevos ordenadores. Para la maqueta de ahorro de batería mediante funcionalidades IEEE802.11 se ha empleado un ordenador personal con una tarjeta PCI capaz de soportar las necesidades exigidas por las pruebas a realizar. El único material que se ha comprado nuevo han sido 2 AP Cisco cuyas características técnicas eran necesarias para el desarrollo del proyecto.

No obstante, el objetivo de este proyecto es el ahorro de baterías gracias a la implementación de protocolos capaces de gestionar un mejor uso de la red inalámbrica. Es por este motivo que el análisis de dichos protocolos servirá para que en un futuro el consumo de energía sea menor.

# Bibliografía y Referencias

[1] Pérez-Costa, X., Campus-Mur, D., Vidal, A., "*On distributed power saving mechanisms of wireless LANs 802.11e U-APSD vs 802.11 power save mode*", ELSEVIER, Germany, February, 2007

[2] Rubio Albaladejo, V., "*Xarxa cel·lular de 4G basada en IPv6: desenvolupament d'un demostrador*", TFC, EPSC, Spain, July, 2006

[3] Blanco Sastre, I., "*Veu sobre IP sobre WiFi: desenvolupament de millores per a reduir el consum de bateries*", TFC, EPSC, Spain, March, 2007

[4] Gast, M., *"802.11® Wireless Networks: The Definitive Guide",* O'Reilly, April, 2002

[5] Shih, E., Bahl, P., Sinclair, M., "*Wake on Wireless: An Event Driven Energy Saving Strategy for Battery Operated Devices",* pp. 162, MOBICOM, September 2002

[6] Jakab, L., Cabellos-Aparicio, A., Serral-García, R., Domingo-Pascual, J. "*Software Tool for Time Duration Measurements of Handovers in IPv6 Wireless Networks", May 24, 2004*

[7] *Jokela, J.* "*WLAN Standardisation and QoS* " 83180 Wireless LANs, Nokia, March, 2005

[8] Gupta, A., Mohapatr, P., *"Power Consumption and Conservation in WiFi Based Phones: A Measurement-Based Study* "

[9] "*WMM Power Save for Mobile and Portable WiFi CERTIFIED Devices*". WiFi Alliance, December 2005. Available: http://www.wi-fi.org

[10] "*Intel(R) Wireless WiFi Link 4965AGN User Guide"*, Intel(R) Corporation

[11] "*Cisco IOS Software Configuration Guid for Cisco Aironet Access Points"* Cisco IOS Release 12.3(8)JA, February 2006

[12] Brown, Len., Karasyov, K., Lebedev, V., Starikovskiy, A., "*Linux Laptop Battery Life,Measurement Tools, Techniques, and Results",* Intel Open Source Technology Center, May, 2005

[13] Velayos, H., Karlsson, G., "*Techniques to reduce the IEEE 802.11b handoff time*", (2-3) KTH, Royal Institute of Technology, Sweeden, June 2004

[14] Mishra, A., Shin, M., Arbaugh, W., "An Empirical Analysis of the IEEE 802.11 MAC Layer Handoff Process", ACM Computer Communications Review, 2003.

[15] Mishra, A., "Supporting secure and transparent mobility in wireless local-area networks", 2005

[16] Ciccarese, G., Convertino, G., De Blasi, M., " *A Novell APSD Scheduler for WLAN IEEE 802.11e*" pp. 1-3, University of Leche, Via Monteroni, LSMicroelecronics, Italy.

[17] "*IEEE 802.11e/D4.4 Draft Supplement to Part 11: Wireless Medium Access Control (MAC) and physical layer (PHY) specifications: Medium Access Control (MAC) Enhancements for Quality of Service (QoS*)", June 2003.

[18] Mangold, S., Choi, S., Hiertz, G.R., Klein O. and Walke, B., "*Analysis of IEEE 802.11e for QoS Support in Wireless LANs*", pp. 40-50 IEEE Wireless Communications Mag. Diciembre 2003

[19] "*Intel(R) Wireless WiFi Link 4965AGN User Guide*", Intel Corporation, 5200 N.E. Elam Young Parkway, Hillsboro, OR 97124-6497 USA, 2004–2007

[20] Vogt, C., Zitterbart, M., "*Efficient and Scalable, End-to-End Mobility Support for Reactive and Proactive Handoffs in IPv6",* Institute of Telematics, Universität Karlsruhe (TH), Germany, June 2006

[21] *Atheros AR6002 chipset*
*http://www.atheros.com/pt/bulletins/AR6002Bulletin.pdf*
*http://www.en-genius.net/site/zones/networkZONE/product_reviews/netp_11120*


# ANEXOS

**TÍTULO: Estudio del consumo de baterías en dispositivos móviles IEEE802.11: técnicas de ahorro, caracterización y evaluación**

**TITULACIÓN: Ingeniería Técnica de Telecomunicaciones, especialidad Telemática.**

**AUTOR: Joshua Sanz de la Rica Mann**

**DIRECTOR: Rafael Vidal Ferré**

**FECHA: 19 de Junio de 2008**

# ÍNDICE

# ANEXO I: Monitores de batería

Para poder estudiar el impacto de determinados parámetros del estándar IEEE 802.11 en el consumo de las baterías, es necesario disponer de alguna herramienta que lo mida. En este Anexo se ha realizado un estudio sobre distintos programas que permiten la monitorización del consumo de la batería tanto en el sistema operativo Windows Vista como en el sistema operativo Linux. Dichos programas tienen la misma finalidad pero difieren en sus características y funcionalidades que a continuación se describen.

## I.1 Windows Vista

En Windows Vista encontramos muy poca variedad de monitores de batería. No obstante, uno de ellos, el *Battery Mon 1.0* parece ser uno de los más fiables ya que después de realizar una medición, las previsiones del tiempo restante que muestra la aplicación se aproximan mucho a la realidad. Por el contrario el *gadged* que proporciona Windows Vista cambia constantemente el valor de sus estimaciones siendo muy impreciso. Además, el error de estas estimaciones es significativo.

### I.1.1 Battery Mon 1.0

*Precio Licencia*: 24$ (1,33 Mb) http://es.software.emule.com/batterymon-2-1/

BatteryMon es un programa de Windows fácil de utilizar que permite la monitorización de las baterías del ordenador portátil y suministros de energía ininterruptible (SAI). Se puede observar gráficamente la carga y velocidad de descarga y diagnosticar problemas en las celdas de las baterías. Proporciona más de 20 estadísticas incluyendo niveles de voltaje, química de la batería, nivel completo de capacidad, capacidad actual, nivel de ajustes, etc. Se mantiene un archivo histórico (log) con el cual podemos analizar las estadísticas y utilizar los datos para generar un gráfico más personalizado.

Aunque el tiempo estimado de descarga o aproximación del tiempo que tardará la batería en agotarse, no sea demasiado fiable, ya que depende del consumo en cada instante de tiempo, la capacidad actual de la batería expresada en % sí que lo es. Esto nos permitirá poder realizar un análisis objetivo en un intervalo de tiempo sin tener que realizar una prueba de descarga completa de la batería. La Figura I.1 nos muestra un gráfico estadístico de *BatteryMon.*

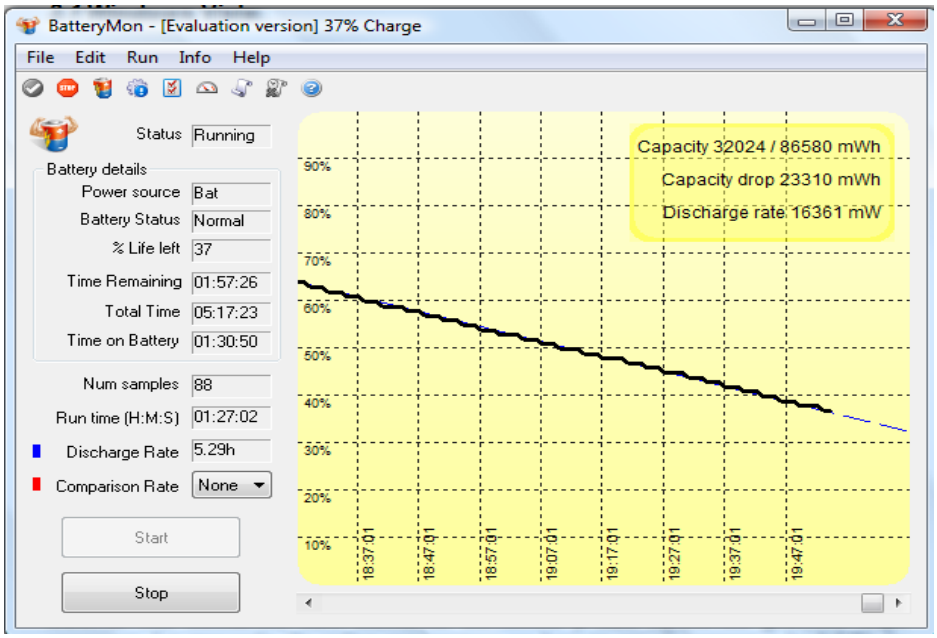


**Fig. I.1** Interfaz gráfica de BatteryMon 1.0

## I.2 Linux

Para el sistema operativo Linux existe una gran variedad de monitores de batería. Sin embargo, son bastante sencillos y sus opciones gráficas algo escasas. La mayoría de estos programas se basan en el acceso a la ACPI (Advanced Configuration and Power Interface) de Linux.

### I.2.1 Forma manual

Mediante el acceso a `/proc/acpi/battery/*/` donde "*" es el nombre de la batería, ya sea BAT0 o BAT1, podemos mostrar por pantalla el estado e información de la batería expresada en mWh (ver Figura I.2).

```
joshua@ubuntu:~$ cat /proc/acpi/battery/*/state
present:                 yes
capacity state:          ok
charging state:          charging
present rate:            2580 mA
remaining capacity:      6321 mAh
present voltage:         14800 mV
```

**Fig. I.2**: Comando *state* del acpi de Linux

Con esta información podemos obtener el resultado en minutos gracias al siguiente comando (I.1):

```
$ echo '60 * 6321 / 2580' | bc
```
**(I.1)**

Para poder simplificar la obtención del tiempo restante, se ha creado un script (ver Figura I.3) que permite modificar estos valores e imprimir por pantalla resultados más intuitivos.

```
#!/usr/bin/perl

chop (@info = `cat /proc/acpi/battery/BAT0/state`);
chop (@capacidad = `cat /proc/acpi/battery/BAT0/info`);

chop ($left = @info[4]);
chop ($left = `echo \"$left\" | awk '{ print \$3 }'`);

$state = @info[2];
chop ($state = `echo \"$state\" | awk '{ print \$3 }'`);

chop ($lleno = @capacidad[1]);
chop ($lleno = `echo \"$lleno\" | awk '{ print \$3 }'`);

$restante = $left / $lleno * 100;
$restante = int ($restante);
$date = `date`;
$str = substr($date,11,5);

chop (@time = `cat /proc/acpi/battery/BAT0/state`);
chop (@time2 = `cat /proc/acpi/battery/BAT0/info`);

$rcap = @time[4];
chop ($rcap = `echo \"$rcap\" | awk '{ print \$3 }'`);

$prate = @time[3];
chop ($prate = `echo \"$prate\" | awk '{ print \$3 }'`);

if ($prate > 0)
{
      $segundos = $rcap / $prate * 60 * 60;

      $minutos = $segundos / 60;
      $minutos = sprintf("%.2f", $minutos);

      $horas = $segundos / 60 / 60;
      $horas = sprintf("%.2f", $horas);

      chop ($singhora = $horas);
      chop ($singhora);
      chop ($singhora);

      $horamin = $horas;
      $horamin =~ s/^.//;
      $horamin =~ s/^.//;

      $horamin = $horamin * .6;
      $horamin = sprintf("%.0f", $horamin);
}
else
{
      $segundos = 0;
}

print " $str";
if ($singhora >= 1)
```

```
{
      print " $singhora";
}

if ($horamin >= 1)
{
      if ($singhora < 1)
      {
      print " 0";
      }
      if ($horamin < 10)
      {
      print ":0$horamin";
      }
      else
      {
      print ":$horamin";
      }
}
else
{
      print ":00";
}

print " $restante%\n";
```

**Fig. I.3**: Script para la obtención del tiempo restante de batería.

Para poder obtener un resultado de dichos valores con un intervalo de tiempo específico, hemos creado un acceso a *crontab*. *Cron* es un demonio que ejecuta programas a intervalos regulares (cada minuto, día, semana o mes). El *crontab* es el archivo donde se especifican los procesos a ejecutar y la hora a la que debe hacerlo.

Para editarlo utilizaremos:

*$ /etc/crontab –e* **(I.2)**

En la siguiente leyenda mostrada en la Figura I.4 se explica el formato a seguir del *corontab*.

```
*     *   *   *    *  "comando a ejecutar"
-     -    -    -    -
|     |     |     |     |
|     |     |     |     +----- día de la semana (0 - 6) (Domingo=0)
|     |     |     +------- mes (1 - 12)
|     |     +--------- día del mes (1 - 31)
|     +----------- hora (0 - 23)
+------------- min (0 - 59)
```

**Fig 3.4:** leyenda *crontab –e* de linux

Especificamos que cada 10 minutos se escriba en el fichero *log_bateria.txt* los parámetros de la ejecución del *srcipt_bateria* como se muestra en la Figura I.5.

```
#SHELL=/bin/sh

*/10     *     *     *     *     /usr/bin/script_bateria     >>
/home/joshua/Escritorio/log_bateria.txt
```

**Fig. I.5**: configuración *crontab –e*

El resultado obtenido en el fichero log_bateria.txt es el siguiente:

```
19:58 1:28 34%          20:01 1:25 33%
19:59 1:29 34%          20:03 1:25 32%
20:00 1:27 33%          20:04 1:23 32%
```

## I.2.2 Gestión de energía 2.20.0

Este software viene integrado en el sistema operativo Ubuntu Gusty Gibbon. Es una herramienta muy potente de monitorización de batería con muchas posibilidades. Detalla los cambios de estado que sufre la máquina según esté en estado de ahorro de energía, modo activo, inactivo, hibernando, con la tapa cerrada o abierta. Puede medirse desde el consumo de energía hasta el voltaje utilizado. A continuación en la Figura I.6 presentamos algunas muestras de esta herramienta.

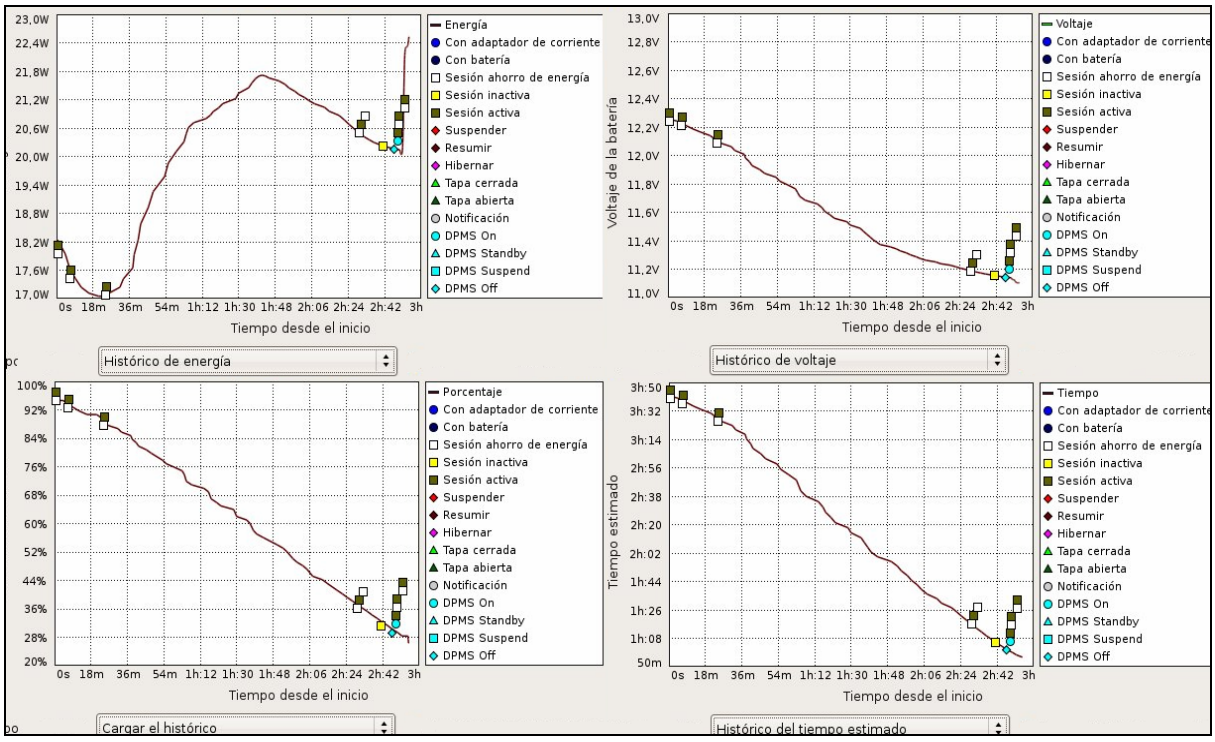


**Fig. I.6:** Herramienta de gestión de energía 2.20.0 de Ubuntu

### I.2.3 Ibam-0.4 (código libre GNU)

(0,019 Mb) http://linux.softpedia.com/get/System/Loggin/IBAM_7153.shtml

Otra herramienta muy completa para Linux es Ibam-0.4. Este monitor de batería proporciona unas estimaciones muy cercanas a la realidad gracias a que usa métodos estadísticos linealmente adaptados. Estos métodos consisten en monitorizar el desgaste y la recarga completa de la batería sucesivas veces para optimizar los resultados del comportamiento de ésta. Lo conseguimos gracias a un comando interno del programa. Cuantas más veces lo utilicemos mayor será la exactitud de los resultados.

La mayoría de los monitores de batería se basan en la escala de tiempo que proporciona la BIOS que no es del todo correcta. Según el siguiente gráfico (ver Figura I.7) obtenido con el comando de *ibam*, " *--plot* " se puede comprobar que entre los 200 minutos y los 60 minutos de vida de una batería, cada minuto-BIOS corresponden a 50 segundos reales. Mientras que entre los 40 y 20 minutos restantes de la batería, cada minuto-BIOS tiene 10 segundos de duración. Como consecuencia, cuando a uno le parece que le queda una hora de batería puede ver que esta hora se consume en tan solo 20 minutos.

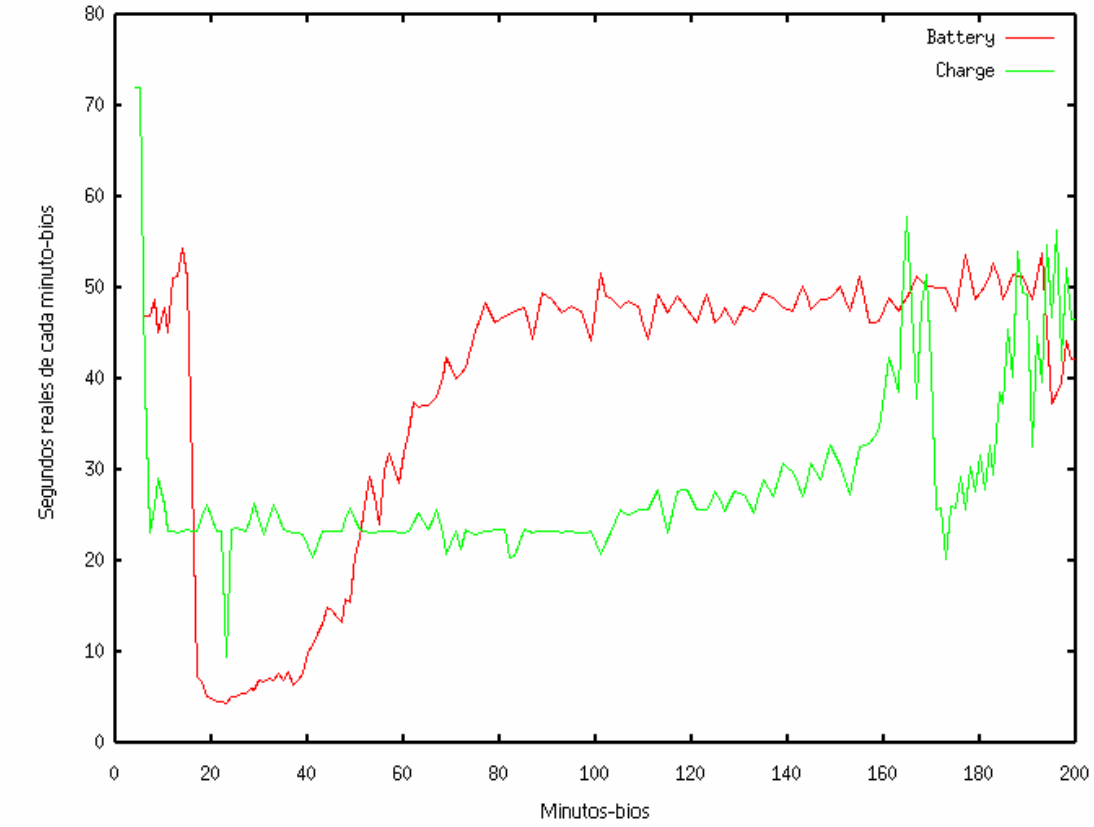


**Fig. I.7** Comportamiento del reloj BIOS

Teniendo en cuenta lo anterior, *ibam* genera una estimación del tiempo restante, de carga y total muy semejante a la realidad con los datos que

obtiene de hacer un análisis de la BIOS. Estos valores más ajustados al comportamiento real son los que están referenciados como Adapted en la Figura I.8.

```
joshua@joshua-laptop:~$ ibam -a
Bios:                        16 %
Battery percentage:          15 %
Charge percentage:           16 %
Bios:                         0:36:08
Battery time left:            0:27:24
Adapted battery time left:    0:45:15
Charge time left:             6:00:30
Adapted charge time left:     3:38:19
Total battery time:           3:07:10
Adapted total battery time:   5:09:03
Total charge time:            7:09:10
Adapted total charge time:    4:19:55
Profile logging enabled.
Current file: /home/joshua/.ibam/profile-003-battery
```

**Fig. I.8:** Todos los parámetros que permite visualizar ibam

Al contrario de lo que sucedía en la Figura 2.7 en lo que respecta al comportamiento de la BIOS, en el ordenador con el que realizaremos las pruebas de descarga de la batería, el *Dell XPS-9celdas,* observamos que el reloj de la BIOS se comporta a la inversa. Es decir, cada minuto-BIOS corresponde aproximadamente a unos 200 segundos en algunos rangos de la descarga. Estos valores dependen de la BIOS de cada ordenador y del estado de la batería. Otro factor influyente es el número de celdas de ésta (ver Figura 2.9).

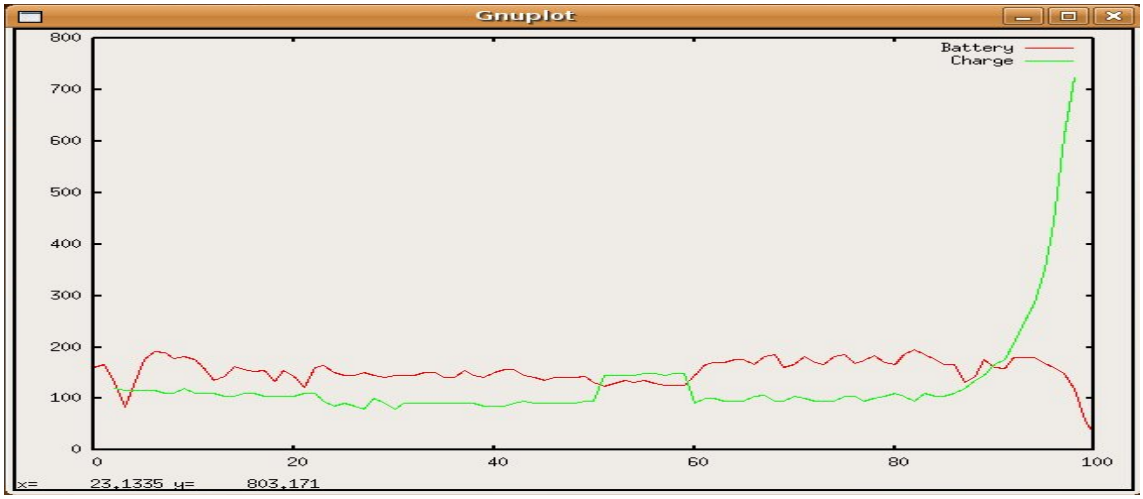


**Fig. I.9:** Comportamiento del reloj BIOS del *DELL XPS-9celdas*

Para poder recoger los datos más relevantes de *ibam* y posteriormente hacer un análisis del consumo, se ha hecho el *script_ibam* que como en el anterior *script* representa la batería restante, el porcentaje actual y la salida del reloj interno del *kernel.* A continuación en la Figura I.10 detallamos las funcionalidades del *script*.

```
#!/usr/bin/perl
#

@info = `ibam -a`;
$porcentaje = @info[2];
$tiempo = @info[5];

$date = `date`;

$strr = substr($date,11,5);
$substr($porcentaje,28,2);
$str2 = substr($tiempo,28,5);
$print " $strr $str2 $str1%\n";
```

```
#Guardamos en un array la salida del ejecutable ibam –a.
#La posición 2 del array
#pertence al valor % restante.
#La posición 5 pertenece al tiempo restante adaptado.
#Se guarda la parte que nos interesa del string ya que el comando
#"date" nos muestra más valores de los que nos interesan.
#Sucede lo mismo con los valores guardados anteriormente.
#Cojemos los datos que nos interesan
```

**Fig. I.10:** Script ibam para con tiempo real, tiempo adaptado y % restante.

## I.2.4 Kpowersave Y Klaptopdeamon

Estas dos herramientas propias de Linux son algo más sencillas. Por el contrario, además de medir la batería permiten gestionarla para conseguir su ahorro. Sólo pueden ser utilizados por entornos provistos de un escritorio *KDE* (K Desktop Environment*).*

Para gestionar el ahorro de energía Kpowersave es capaz de configurar la frecuencia de la CPU para que opere en modo ahorro, dinámico o alto rendimiento. Así mismo pueden configurarse perfiles de rendimiento ajustando el brillo de la pantalla, gestionando la alimentación de la pantalla, suspensión del disco o de la RAM según un tiempo especificado etc. En cambio, Klaptopdeamon simplemente nos proporciona el % y tiempo restante de la batería además de algunas opciones de alerta. Podemos ver un ejemplo de estas dos herramientas en la Figura I.11.

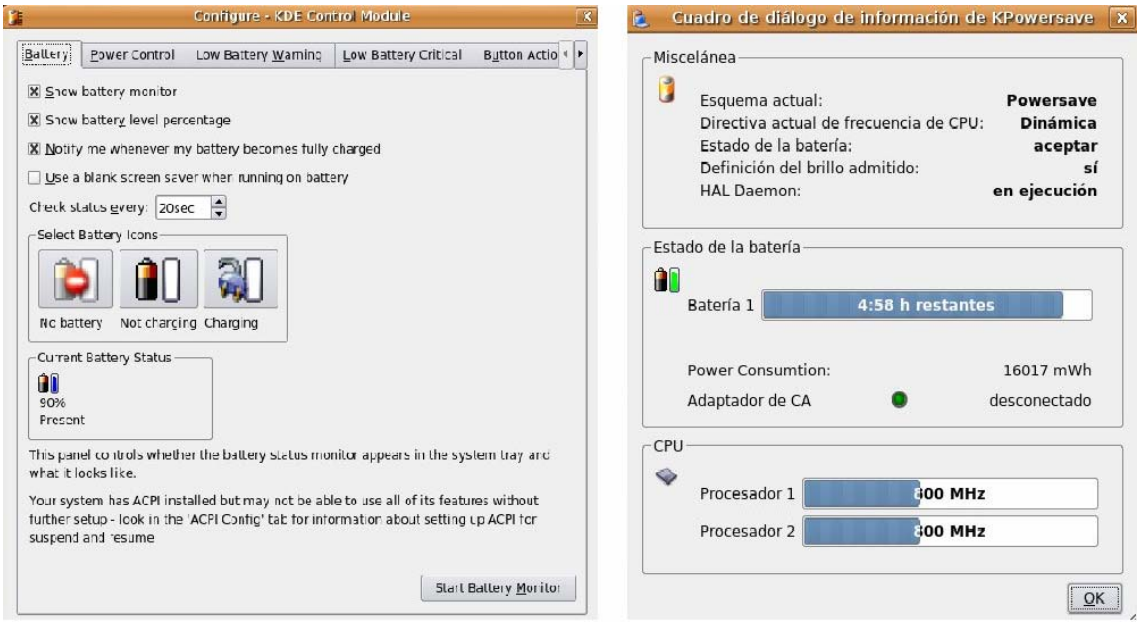


**Fig. I.11**: Interfaz gráfica de *klaptopdeamon* y *kpowersave*

A continuación en la Tabla I.1 haremos una comparativa de estas herramientas anteriormente descritas destacando las características que cada una puede proporcionar.

**Tabla I.1**: Comparativa entre los distintos monitores de batería.

| Herramienta monitor de batería. | Características a destacar | Genera logs | Precisión |
|---|---|---|---|
| BatteryMon 1.0 | Parámetros de medida:<br>-Voltaje<br>- % de batería<br>-Capacidad restante y total en mWh.<br>-Tiempo de consumo, restante y total.<br><br>Genera gráficos.<br><br>Detecta problemas en las celdas<br><br>Detalles de la batería como modelo, química, numero de serie etc. | SI | Buena precisión global en términos de %.<br><br>Mala precisión instantánea.<br>Los valores de tiempo restante varían mucho en cada muestra. |
| Script LINUX | Parámetros de medida:<br>- Tiempo restante | NO | + 14,7% error |

| | | | |
|---|---|---|---|
| | - % de batería | | |
| Acceso a la ACPI | Parámetros de medida:<br>-Tasa de consumo actual en mA y restante en mAh.<br>-Voltaje actual. | NO | Valores de consumo (mAh), bastante precisos. |
| Gestión de energía 2.20.0 de Ubuntu | Parámetros de medida:<br>- Energía (W)<br>- Voltaje (V)<br>- Tiempo<br>- % de batería<br><br>Genera gráficos detallados con cambio de estados. | | Proporciona un valor máximo de 3 horas restantes.<br><br>Mala precisión Global. |
| Ibam-0.4 | Parámetros de medida:<br>- Tiempo<br>- Tiempo adaptado<br>- Tiempo total<br>- Tiempo según BIOS<br>- Tiempo de recarga<br>- % de batería<br><br>Permite adaptar los tiempos de la BIOS con el tiempo real Esto nos permite precisar el tiempo restante con mayor exactitud independientemente del estado de la batería o las características del reloj interno de la BIOS | NO | + 3% error |
| Kpowersave | Parámetros de medida:<br>- Tiempo<br>- % de batería<br><br>Ajustar parámetros de ahorro:<br>- Frecuencia de CPU<br>- Suspensión de RAM<br>- Suspensión de disco<br>- Brillo de pantalla | NO | -7% error |
| Klaptopdeamon | Parámetros de medida:<br>- Tiempo<br>- % de batería<br>Configuración de alertas. | NO | -16 % error |

La precisión la hemos calculado tomando la diferencia entre los minutos totales durante los cuales se ha estado usando la batería y los minutos que estima el monitor de batería. Para el caso de Ibam, hemos separado la precisión de cada

hora de consumo para encontrar el momento más óptimo durante el cual realizar las pruebas

A partir de estos resultados, de todas estas herramientas de monitorización, hemos elegido *ibam-0.4* para realizar las pruebas. Esta herramienta nos proporciona mayor precisión que las demás y la capacidad de generar un *log* gracias a la interacción con el *crontab* de Linux. El *log* lo utilizaremos para extraer las estadísticas y plasmarlas en un gráfico. Aunque el resto de herramientas sea capaz de proporcionar un mayor número de características, la precisión es la principal prioridad. Puede parecer que la precisión es bastante mala, pero se debe a que en la primera hora tarda en estabilizarse el consumo y en la última, la batería tiene un comportamiento impreciso. Por tanto, las pruebas de consumo las realizaremos en el primer sector de la batería dejando que se estabilice ejecutando el mínimo de operaciones posibles antes de comenzar el test.

# ANEXO II: Dispositivos WMM-PS

## II.1 Dispositivos que cumplen con WMM-PS Wi-Fi Certified

Para la realización de este trabajo ha sido necesaria la adquisición de hardware con soporte de la certificación WMM-PS PM. Como paso previo a esta adquisición se realizó una búsqueda que dio como resultados la lista de dispositivos que aquí se presenta, actualizada al 22/11/07

**Tabla II.1:** Dispositivos AP / Wi-Fi Router

| Nombre | Link | Certificaciones | Otras certificaciones | Precio |
|---|---|---|---|---|
| **Broadcom** BCM94704AGR | Broadcom | 802.11 a/b/g | MAC filtering WPA/WPA2 | _____ |
| **Broadcom** BCM4705 | Broadcom2 | 802.11 a/b/g/n/h/d | WPA/WPA2 | _____ |
| **Linksys** Wireless-G VPN **Router** with Range Booster (WRV200) | Linksys | 802.11 b/g/i | MAC filtering ACLs (acces control lists) WPA/WPA2 | 60€ buy |
| **Linksys** WRT54GL | Linksys2 | 802.11 b/g | MAC filtering WPA/WPA2 | 56,95€ buy |
| **HP ProCurve** Wireless Edge Services xl Module (J9001A) | HP1 | 802.11 b/g/h/d | WPA/WPA2 ACLs u-APSD | 3500€ buy |
| **HP ProCurve** Radio Port 210,220 y 230 | HP2 | 210 → 802.11 b/g/d | WPA/WPA2 u-APSD | $250buy |
| | | 220 → 802.11 a/b/g/h/d | | $350buy |
| | | 230 → 802.11 a/b/g/h/d | | $360buy |
| **Marvell** Libertas AP-52 | Marvell | 802.11 a/b/g | WPA | _____ |
| **Motorola** Wi-Fi Router (RSG 2500) | Motorola | 802.11 b/g/d/i | WPA/WPA2 *Stateful packet inspection Firewall* u-APSD | 120€ buy |
| **NetGear** RangeMax TM Next Wireless-N Router (WNR834B) | NetGear | 802.11 b/g | NAT/SPI WPA/WPA2 | $100 buy |
| **Thomson** SpeedTouch 585 v6, 780 WL, 706, 620 s, 608 WL | Todos los modelos Thomson | **585v6** 802.11b/g | WPA/WPA2 | £58 buy |
| | | **780wl** 802.11b/g | | £69 buy |
| | | **706** 802.11b/g | | £72 buy |
| | | **620s** 802.11b/g | | £359buy |
| | | **608wl** 802.11b/g | | £139buy |

| | | | | |
|---|---|---|---|---|
| **Alcatel** OAW AP41 | Alcatel_41 | 802.11 a/b/g/h/d | WPA/WPA2 | 195$ buy |
| **Alcatel** OAW AP60, 61, 70 | Alcatel_60_61_70 | 802.11 a/b/g/h/d | WPA/WPA2 | 295$ buy<br>295$ buy<br>595$ buy |
| **Alcatel** OAW AP65 | Alcatel_65 | 802.11 a/b/g/h/d | WPA/WPA2 | 495$ buy |
| **Alcatel** OAW AP80M | Alcatel_80 | 802.11 a/b/g/h/d | WPA/WPA2 | 1,795$ buy |
| **Aruba** AP41 | Aruba_41 | 802.11 a/b/g/h/d | WPA/WPA2 | 156$ buy |
| **Aruba** AP60-61 | Aruba_60_61 | 802.11 a/b/g/h/d | WPA/WPA2 | 272$ buy |
| **Aruba** AP65 | Aruba_65 | 802.11 a/b/g/h/d | WPA/WPA2 | 396$ buy |
| **Aruba** AP70 | Aruba_70 | 802.11 a/b/g/h/d | WPA/WPA2 | 476$ buy |
| **Aruba** AP80M | Aruba_80M | 802.11 a/b/g/h/d | WPA/WPA2 | 1,436$ buy |
| **Bluesocket** Bluesecure AP1540 | Bluesocket | 802.11 a/b/g/h/d/i | WPA/WPA2<br>ACLs | 409$ buy |
| **Cisco** Aironet 1240AG Series 802.11A/B/G Access Point | Cisco_1240AG Model # AIR-WLC2006-K9 and AIR-LAP1242AG/AIR-WLC2006-K9 and AIR-LAP1242AG | 802.11 a/b/g/h/d/i | WPA/WPA2 | 493$ buy |
| **Cisco** Aironet 1130AG Series 802.11A/B/G Access Point | Cisco_1130AG Mode l# AIR-WLC2006-K9 and AIR-LAP1131AG/AIR-WLC2006-K9 and AIR-LAP1131AG | 802.11 a/b/g/h/d/i | WPA/WPA2 | 388$ buy |
| **Cisco** Aironet 1230AG Series 802.11A/B/G Access Point | Cisco 1230AG Model # AIR-WLC2006-K9 and AIR-LAP1232AG/AIR-WLC2006-K9 and AIR-LAP1232AG | 802.11 a/b/g/h/d/i | WPA/WPA2 | 513$ buy |
| **Linksys** WAP200 | Linksys WAP200 | 802.11 b/g | WPA/WPA2 | 108$ buy |
| **HP ProCurve** Wireless Services zl Module (J9051A) | HP Wireless edge | 802.11 a/b/g/h/i | WPA/WPA2<br>ACLs (IP filtering)<br>u-APSD | 4180$ buy |
| **Motorola** AP-5131 | Motorola AP-5131 | 802.11 a/b/g/h/d | WPA/WPA2 | 617$ buy |
| Motorola AP-5181 Outdor | Motorola AP-5181 | 802.11 a/b/g/h/d | WPA/WPA2 uAPSD | 1452$ buy |

**Tabla II.2:** Tarjetas externas (PCMCIA/USB)

| Nombre | Link | Certificaciones | Chipset | Driver Linux | Consumo | Precio |
|---|---|---|---|---|---|---|
| **Broadcom** BCM94309 CB | Broadcom | 802.11 a/b/g | BCM4309 | Si linux | Tx=990mW Rx=792mW PSM < 33mW | ____ |
| **Zyxel** 802.11b/g Wireless USB Adapter (G-210H) | Zyxel | 802.11 b/g | _____ | NO | _____ | $48 buy |
| **Realtek** RTL8187B USB 2.0 | Realtek | 802.11 b/g | RTL8187B | Si linux | _____ | _____ |
| **TRENDnet** TEW-424UB **USB** Adapter | TEW 424UB | 802.11 b/g | Realtek RTL8187B | Si linux | Tx=1650mW Rx=800mW | $23buy |
| **Motorola** Netopia 3-D Reach Wireless Adapters **PCMCIA/USB** | Motorola | 802.11 b/g | Ralink RT2500 | Si linux | _____ | $33buy PCMCIA $42buy USB |

**Tabla II.3:** Tarjetas PCI (última actualización 22/11/07)

| Nombre | Link | Certificaciones | Chipset | Driver Linux | Consumo | Precio |
|---|---|---|---|---|---|---|
| **Intel®** Wireless Wi-Fi Link 4965AGN | Intel | 802.11a/b/g/h/n/i WPA/WPA2 | Intel 4965AGN | Si linux driver | _____ | $61 buy |
| **Intel®** PRO/Wireless 3945ABG | Intel2 | 802.11a/b/g/h/d/i WPA/WPA2 | Intel 3945ABG | Si linux driver (sección 3.9) | _____ | $22 buy |
| **Ralink** RT2800 PCI/mPCI/CB | Ralink | 802.11 a/b/g/h/d/n/j/i WPA/WPA2 | RT2800PD | Si linux | _____ | ____ |
| **Ralink** RT5201 dual-band **USB** | Ralink2 | 802.11 a/b/g/d/h WPA/WPA2 | RT5201U (RT5226 + RT2571W) | Si linux | _____ | ____ |
| **Philips** BGW211 Low-power WLAN SiP | NXP-Philips | 802.11 b/g ***Teléfonos,PDAs*** … | Philips BGW211 | Si linux | Rx= 300 mW **b** 400 mW **g** Tx= 550mW **b** 600mW **g** | $12 |

# ANEXO III: Escenario de pruebas WMM-PS (u-APSD)

En este anexo explicaremos los pasos a seguir para configurar el AP y la tarjeta PCI WiFi del ordenador portátil para que puedan implementar las funcionalidades que analizamos durante las pruebas del capítulo 2. Las herramientas utilizadas para generar tráfico simulando VoIP o comprobar la conectividad también están descritas en este documento.

## III.1 Topología inalámbrica

En este primer escenario configuraremos una red que permita la comunicación inalámbrica a través de un AP entre un ordenador portátil y otro nodo.

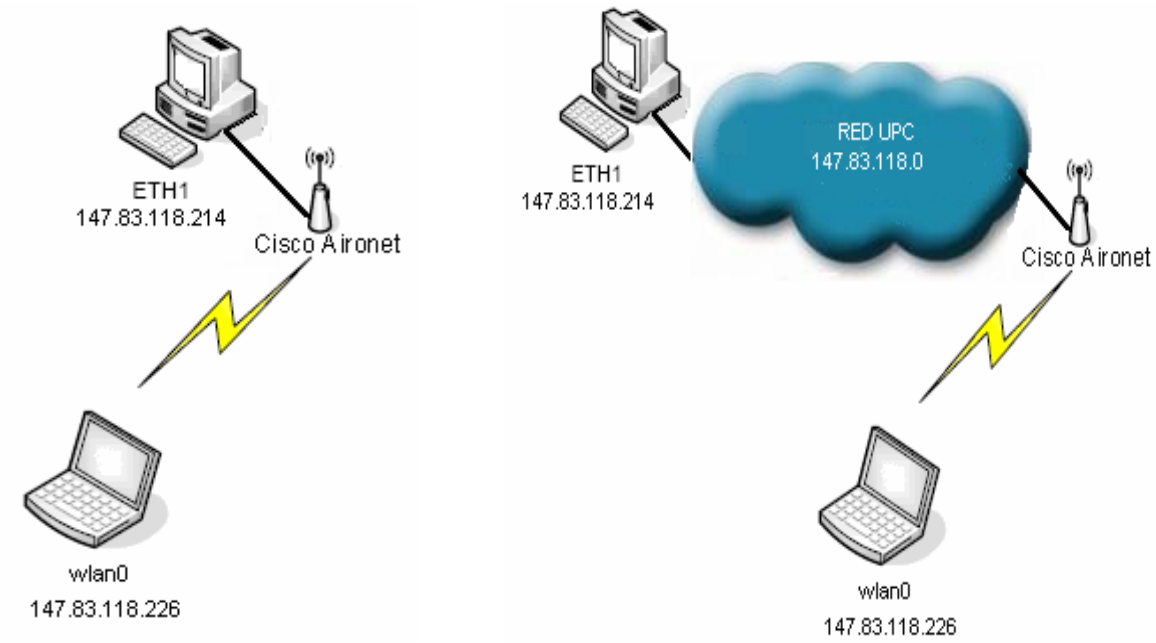


**Fig. III.2: Escenario de pruebas de ahorro de batería**

Utilizaremos un ordenador portátil provisto de una tarjeta inalámbrica de última generación capaz de implementar la tecnología uAPSD y un ordenador conectado a un AP que configuraremos con diversas características para realizar pruebas de ahorro de batería mediante herramientas activas como MGEN, IPERF o PING. Para ello utilizaremos un AP Cisco. Únicamente para las pruebas de *ARP Caching* conectaremos el AP a la red de la universidad para que lleguen al nodo móvil todas la peticiones ARP que circulan por esta red. Esto nos permitirá comprobar el ahorro de este mecanismo en una red real. En la Tabla III.1 están detallados los aspectos técnicos de los dispositivos del escenario de pruebas.

**Tabla III.1**: Características técnicas de los dispositivos utilizados

| **NODO A** | |
|---|---|
| **CPU** | Intel® Core Duo® 2.00GHz |
| **SO 1** | Windows Vista Home Premium 32 bits |
| **SO 2** | Ubuntu Gusty Gibon 7.10 patched for Dell XPS 1330M |
| **WIFI PCI** | Intel® Wireless Wi-Fi Link 4965AGN |
| **Certificaciones** | 802.11a/b/g/h/n/i WPA/WPA2 |
| **Chipset** | Intel 4965AGN |
| **Driver Windows** | Intel® Wireless WiFi Link 4965AGN: 11.5.0.32 |
| **Driver Linux** | iwlwifi-1.0.0-1 |
| **NODO B** | |
| **CPU** | Intel(R) Pentium(R) 4 CPU 1.60GHz |
| **TARJETA ETH0** | Broadcom Corporation: device 8401 |
| **SO** | Fedora Core 4 kernel 2.6.15 |
| **DRIVER ETH0** | Broadcom Corporation BCM4401 100Base-T (rev 01) |
| **AP** | |
| **Modelo** | Cisco Aironet 1130AG IEEE 802.11 A/B/G Access Point |
| **Certificaciones** | 802.11 a/b/g/h/d/i WPA/WPA2 |

## III.2 Configuración de los dispositivos

Para poder activar las distintas funcionalidades que permiten el ahorro de batería, habrá que realizar las configuraciones necesarias en el AP y la STA. La tarjeta *pci Intel 4965AGN* solamente se tendrá que configurar para activar el PSM y el soporte QoS en Linux y uAPSD en Windows. Por el contrario, las demás funcionalidades incluyendo ésta, se configurarán en el AP *Cisco Aironet 1130AG* mediante la interfaz web y/o comandos Cisco utilizando el Terminal de consola.

### III.2.1 Activación de uAPSD

*III.2.1.1 Tarjeta PCI Intel 4965AGN*

**Windows:**

Para poder activar la funcionalidad uAPSD en Windows los Registros tendrán que ser modificados de la siguiente manera según [19]:

-Editar la Key siguiente:

- *HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Class\ {4D36E972-E325-11CE-BFC1-08002bE10318\00XX\ApsdTriggerQueues = c o f*

(Donde c = VI y VO están activos, f = todos las AC están activas)

-Añadir la Key siguiente:

- HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-11CE-BFC1-08002bE10318\00XX\ApsdFillCmd = 1

- HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-11CE-BFC1-08002bE10318\00XX\ApsdTriggerMode = 3

Además, el mecanismo PSM deberá estar activado. Se activará de la siguiente manera:

1. *Inicio*
2. *Panel de control.*
3. *Sistema y mantenimiento.*
4. *Opciones de energía.*
5. *Seleccionar Recomendado.*
6. *Cambiar opciones avanzadas de energía.*
7. *Configuración de adaptador inalámbrico*
8. *Modo de ahorro de energía.*
9. *Seleccionar Máximo ahorro de energía*

**Linux:**

Para utilizar esta funcionalidad en Linux, habrá que tener instalado los drivers iwl4965, mac80211 y cfg80211 en la versión del kernel 2.6.24 o superior http://linuxwireless.org/en/users/Download

Una vez descargados se activarán mediante *modprobe* seguido del nombre del driver a cargar.

Una vez instalados según los pasos referenciados en el link anterior habrá que configurar el driver para que actúe en PSM y para que tenga soportados suplementos QoS previamente compilados en el kernel.
http://www.lesswatts.org/tips/wireless.php

- PSM:

```
echo 5 > /sys/bus/pci/drivers/iwl4965/*/power_level
```

- QoS support:

```
Echo 1 > /sys/bus/pci/drivers/iwl4965/module/parameters/qos_enable
```

### *III.2.1.1 AP Cisco Airones 1130AG*

Para configurar el AP vía interfaz web marcaremos la casilla correspondiente a nuestra comunicación radio del apartado WMM de QoS en el menú *Other services*. Esto además de permitirnos aplicar el servicio de multimedia, activará el mecanismo de ahorro de batería WMM-PS.

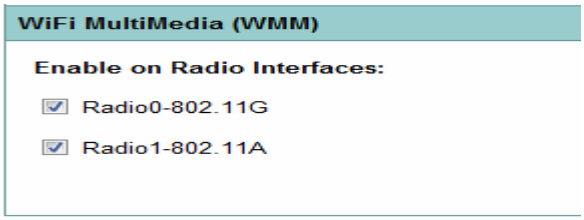


**Fig. III.2**: Configuración uAPSD en AP Cisco Aironet 1130AG

Para hacerlo desde el comando de consolas aplicaremos:

- *AP> enable* (para acceder a la configuración mediante el password)
- *AP# configure Terminal* (para entrar en modo configuración)
- AP(config)# *dot11 qos mode wmm* (activar el wmm y el soporte uAPSD)

## III.2.2 Mapear DSCP

Para que cuando utilicemos tráfico UDP podamos darle prioridades de VoIP o video entre otras, deberemos mapear el tráfico entrante de prioridad DSCP *Best Efford 0x00* a una CoS de Voz o video (ver Tabla 1.2). Esto lo haremos utilizando la interfaz web tal como muestra la Figura 1.7 del primer capítulo o mediante los siguientes comandos por consola:

- *AP(config)#* ***class-map miclase*** (creamos una política llamada miclase)
- *AP(config-cmap)#* ***match ip dscp X*** (a esta política le añadimos la prioridad X dependiendo de cual queramos siguiendo la tabla 1.2)
  *AP(config-cmap)#* ***exit***

## III.2.3 Modificar el DTIM

La modificación del DTIM se realizará en el AP mediante interfaz web (Figura II.3) o utilizando comandos Cisco.

| | | | | | |
|---|---|---|---|---|---|
| Beacon Period: | 100 | (20-4000 Kusec) | Data Beacon Rate (DTIM): | 2 | (1-100) |
| Max. Data Retries: | 64 | (1-128) | RTS Max. Retries: | 64 | (1-128) |
| Fragmentation Threshold: | 2346 | (256-2346) | RTS Threshold: | 2347 | (0-2347) |

**Fig. III.3**: Configuración DTIM en AP Cisco Aironet 1130AG

Mediante comandos Cisco del Terminal de consola:

- *AP(config)#* mbssid [guest-mode] [dtim-period *period*]

Donde *guest-mode* especifica la inclusión del SSID en los *beacons* y el *dtim-period* especifica cada cuantos *beacons*, parámetro *period*, el dispositivo envía uno con el campo DTIM activo. Este parámetro puede oscilar entre 1 y 100.

### III.2.4 Activar ARP-cache y ARP-caching

Esta funcionalidad, se representa en la interfaz web del AP como ARP-caching y como ARP-cache en los comandos Cisco.

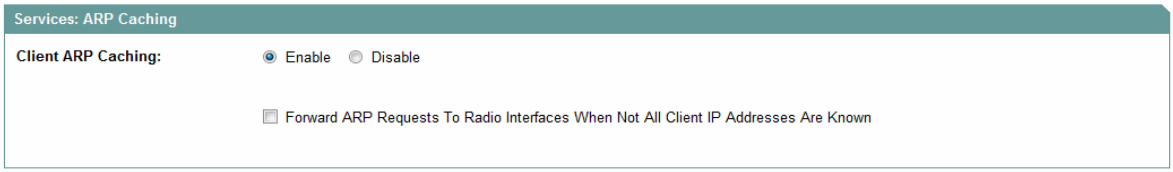


**Fig. III.4**: Configuración web ARP-Caching en AP Cisco Airones 1130AG

- *AP(conifg)# dot11 arp-cache [optional]*

Si se añade *optional* en el comando es como marcar la casilla *Forward ARP* de la Figura III.4. Esta funcionalidad permite dejar pasar los ARPs que no son conocidos, i.e no se conoce la relación dirección MAC⇔IP, hasta que todos los clientes asociados son conocidos. A partir de este momento, no deja pasar ARPs con dirección distinta a los clientes asociados y contesta a las peticiones ARP en lugar del cliente asociado.
.

## III.3 Herramientas activas y pasivas para el análisis de redes

Para realizar las distintas pruebas de consumo de batería aplicando distintos parámetros, hemos utilizado distintas herramientas adaptadas a cada prueba.

### III.3.1 Herramientas Activas

*III.3.1.1 Ping*

Un *ping* (*Packet Internet Grouper*) es una utilidad diseñada para comprobar el estado de una conexión con uno o varios equipos remotos por medio de

paquetes ICMP de solicitud de eco y de respuesta de eco para determinar si un sistema IP es accesible en una red.

Hemos utilizado esta herramienta para comprobar la accesibilidad a nuestro AP y al Nodo B. Además, hemos utilizado la versión *ping6* para comprobar el la correcta configuración de los direccionamientos en la maqueta MIPv6.

### *III.3.1.2 MGen*

El Multi-Generator (MGen) es un software *Open-Source* desarrollado por el *Naval Research Laboratory* (NRL) en su grupo de trabajo *PROTocol Advanced Networkin*g (PROTEAN) *Research Group*. MGEN proporciona la capacidad de desarrollar pruebas de prestaciones en redes basadas en IP mediante el uso de tráfico UDP/IP. MGen es una aplicación de libre distribución que genera genera patrones de tráfico IP en tiempo real para cargar a la red en distintas formas.

El tráfico generado puede utilizarse para calcular estadísticas de como se está comportando la red ante una determinada carga analizando distintos parámetros de calidad de servicio, como el *troughput*, la cantidad de paquetes perdidos, el retardo experimentado por los paquetes. MGen es una aplicación que actualmente sólo corre sobre un sistema operativo basado en UNIX y/o WIN32. En este proyecto se ha utilizado la versión 4.4 para poder implementar una secuencia de envío de mensajes UDP siguiendo una función de Poisson que nos permitirá asemejar al máximo una conversación de VoIP.

- Script para generar tráfico UDP y simular llamadas VoIP (apartado 2.1):

  ON 1 UDP SRC 5005 DST 147.83.118.226/5005 BURST [RANDOM 320.0 POISSON [34.0 94] EXP 180.0]

  *-BURST: ráfagas*
  *-RANDOM: cada 320 segundos (10 llamadas por hora)*
  *-POISSON: envío de 34 paquetes por segundo con longitud 94 bytes con distribución de poisson.*
  *-EXP: 180 segundos distribuidos de forma exponencial. Duración de las ráfagas*

- Script para generar tráfico UDP con destino multicast (apartado 2.3):

  ON 1 UDP SRC 5005 DST 224.0.0.1/5005 PERIODIC [10.0 28]

  *-PERIODIC: tráfico periódico. 10 paquetes por segundo (1 paq cada 100 ms) con tamaño 28 bytes equivalentes a un ARP.*

### *III.3.1.3 Iperf*

Esta otra herramienta también puede utilizarse para generar tráfico UDP o TCP. Se utiliza para medir el máximo ancho de banda de una comunicación UDP o TCP. Además se puede obtener el *jitter*, el porcentaje de pérdidas de paquetes etc. Para poder utilizar esta herramienta hay que configurar un extremo en modo servidor para la recepción de tramas de otro extremo en modo cliente.

## III.3.2 Herramientas pasivas

Son las herramientas encargadas de analizar el tráfico sin inyectar ningún tipo de mensaje en la red.

### *III.3.2.1 Wireshark*

Wireshark es una evolución del clásico Ethereal. Es un analizador de protocolos que se utiliza para solucionar problemas en redes de comunicaciones para desarrollo de software y protocolos. Tiene una funcionalidad similar a la de tcpdump pero con una interfaz gráfica y muchas opciones de organización y filtrado de información. Con este programa podremos ver todo el tráfico que pasa a través de la red en modo promiscuo. Además, podremos capturar tramas IEEE802.11 configurando la tarjeta en modo monitor.

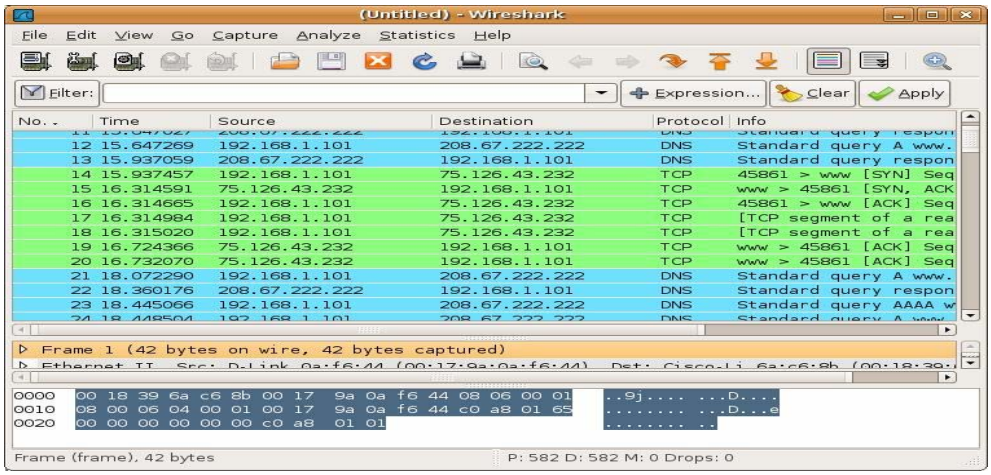


**Fig. III.5:** Ejemplo de wireshark en el sistema operativo Ubuntu

### *III.3.2.2 Kismet*

Kismet es un programa para Linux que permite detectar redes inalámbricas (WLANs) mediante la utilización de tarjetas de red de los estándar 802.11a, 802.11b y 802.11g. Puede utilizarse para verificar el correcto funcionamiento de nuestra STA en modo monitor, detectar otras redes que puedan estar

causando interferencias en la nuestra y detectar los APs que están a nuestro alcance.
Gracias a que kismet guarda los paquetes capturados, podremos analizarlos posteriormente. De todas formas, los paquetes IEEE802.11 los analizaremos con wireshark. Kismet nos servirá para que al ejecutarlo configure automáticamente nuestra tarjeta inalámbrica en modo monitor ya que el driver del que disponemos no nos lo permite hacer manualmente por incompatibilidades con el kernel. Por tanto, activando kismet, nuestra tarjeta pasará al estado monitor, y con wireshark podremos escoger la opción de análisis de tramas IEEE802.11. En la Figura 2.3 se muestra un ejemplo de la interfaz gráfica de *kismet.*

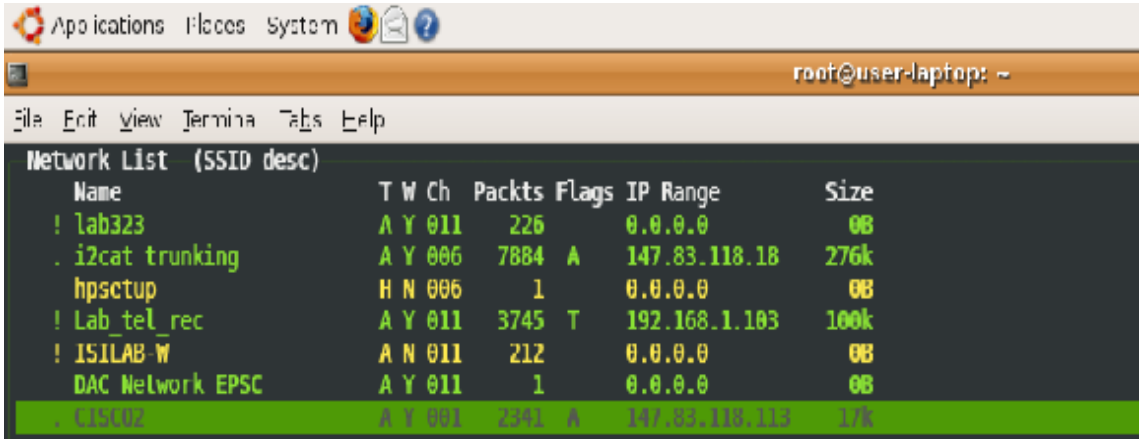


**Fig. III.6:** Ejemplo de kismet en el laboratorio

## III.4 Captura de tramas IEEE802.11 y beacons con y sin QoS según la certificación WMM

### III.4.1 Capturas con certificación WMM desactivada

Las siguientes figuras muestran una captura de un beacon y un IEEE802.11 Null sin la funcionalidad WMM activa en el AP que permite la calidad de servicio mediante ACs (ver Capítulo 1).

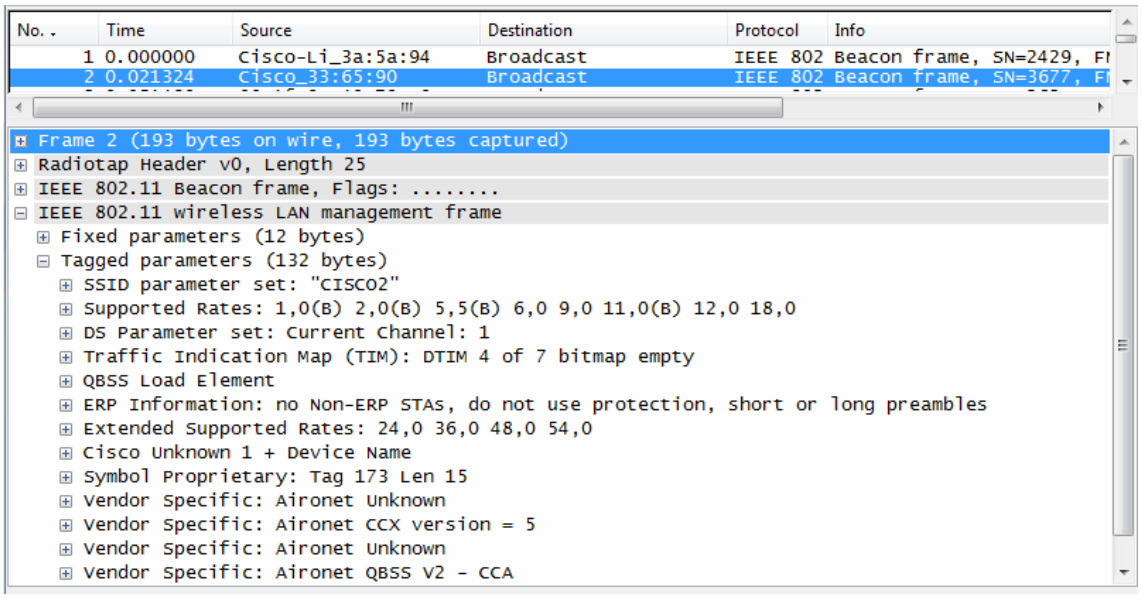


**Fig. III.7**. Captura de un *beacon* sin la funcionalidad WMM

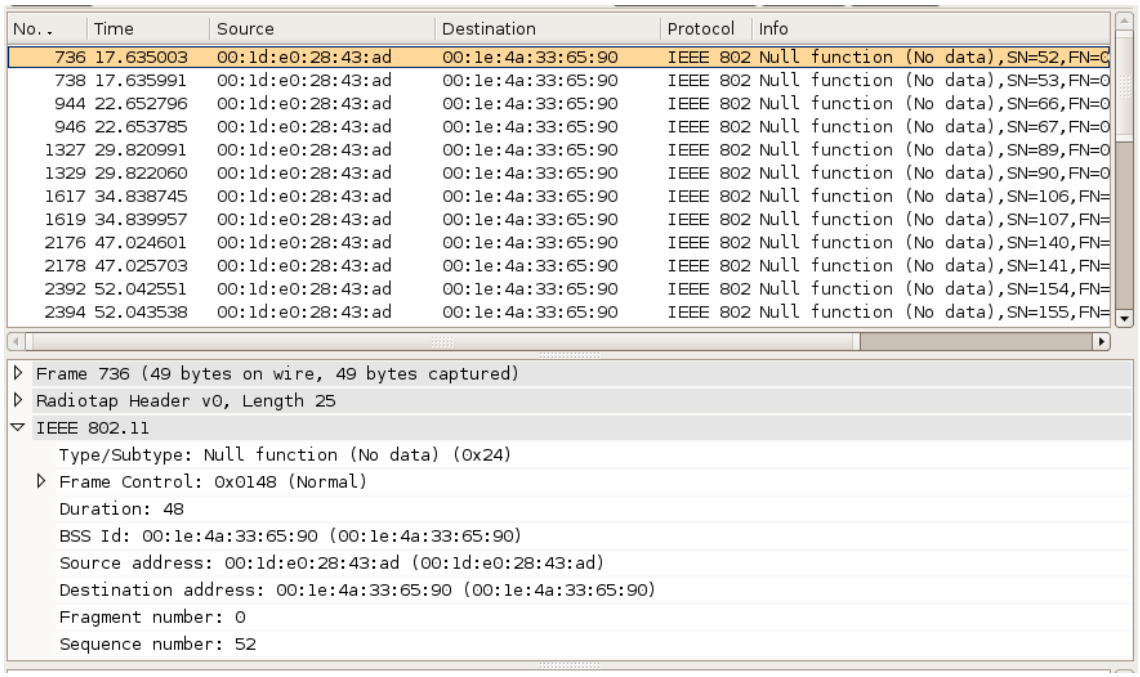


**Fig. III.8**. Captura de un *IEEE802 Null* sin la funcionalidad WMM

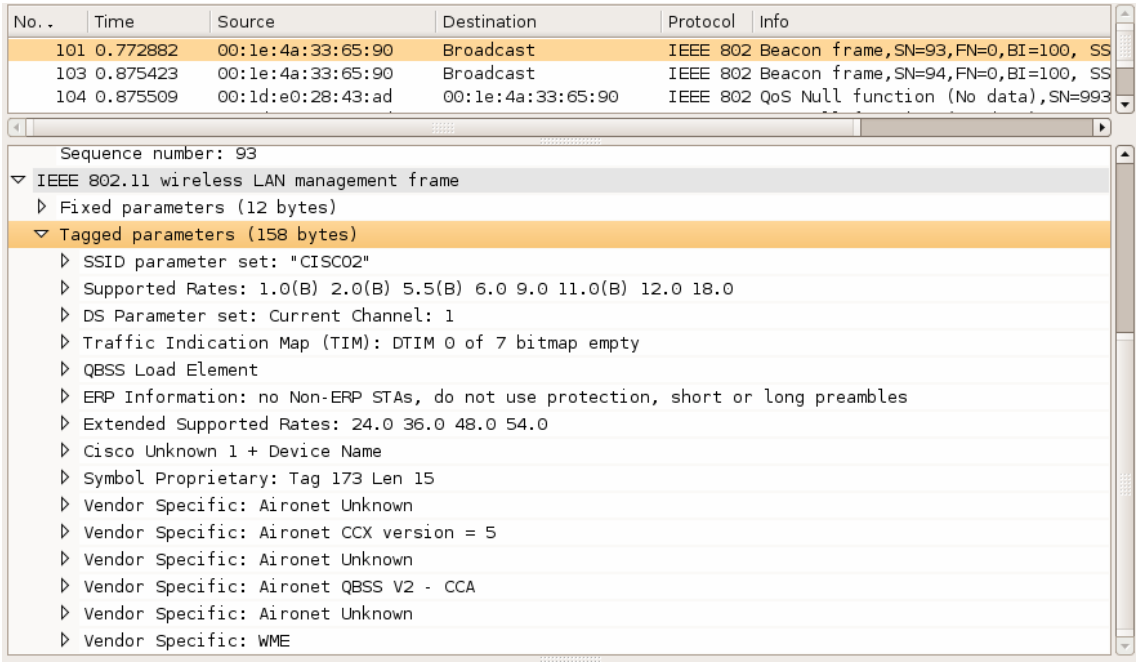


**Fig. III.9**. Captura de un *beacon* con la funcionalidad WMM

Los *Tagged parameters* de un beacon que procede de un QAP son más extensos, principalmente por la pestaña *Vendor Specific: WME* desglosada en la figura 1.11 del primer capítulo del TFC. Este parámetro supone un incremento de 26 bytes en el beacon.

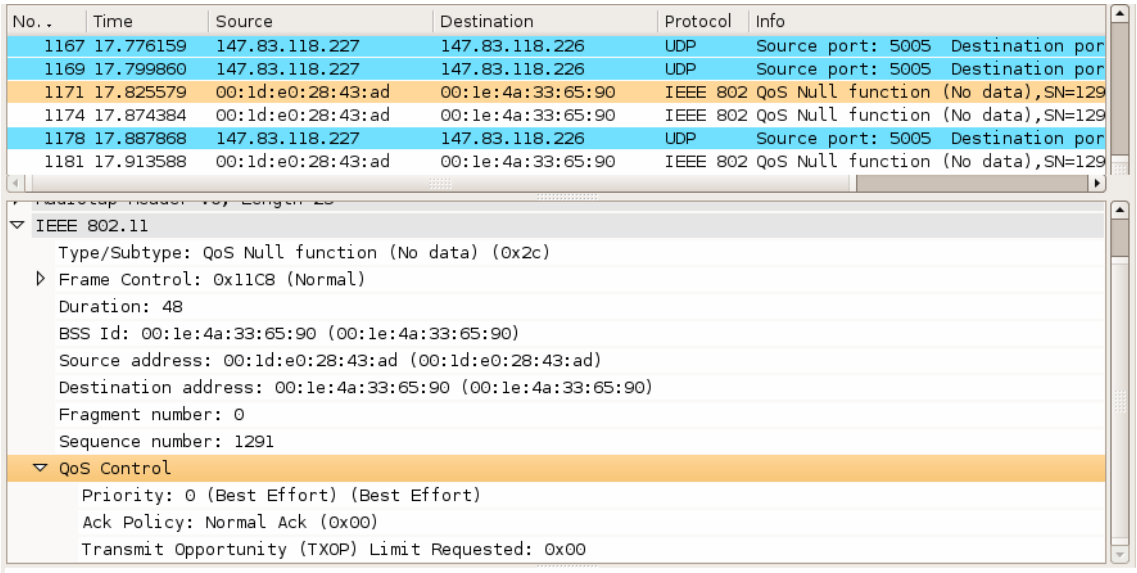


**Fig. III.10**. Captura de un *IEEE802 QoS Null* con la funcionalidad WMM

Los *IEEE802 Null* que son QoS, tienen 2 bytes más que los que no lo son, correspondientes al campo QoS Control. Este campo se explica en el capítulo 1.

# ANEXO IV: Implantar MIPv6 en una red IPv6

## IV.1 Topología de red MIPv6

Se ha utilizado el demostrador ya existente [2] para utilizar el protocolo MIPv6. Este demostrador ha requerido de ciertos cambios del original. Así, las subredes servidas por el *Access Router* 1 y el *Access Router* 2 están ahora formadas por un acceso inalámbrico que ha requerido la re-configuración de los routers.

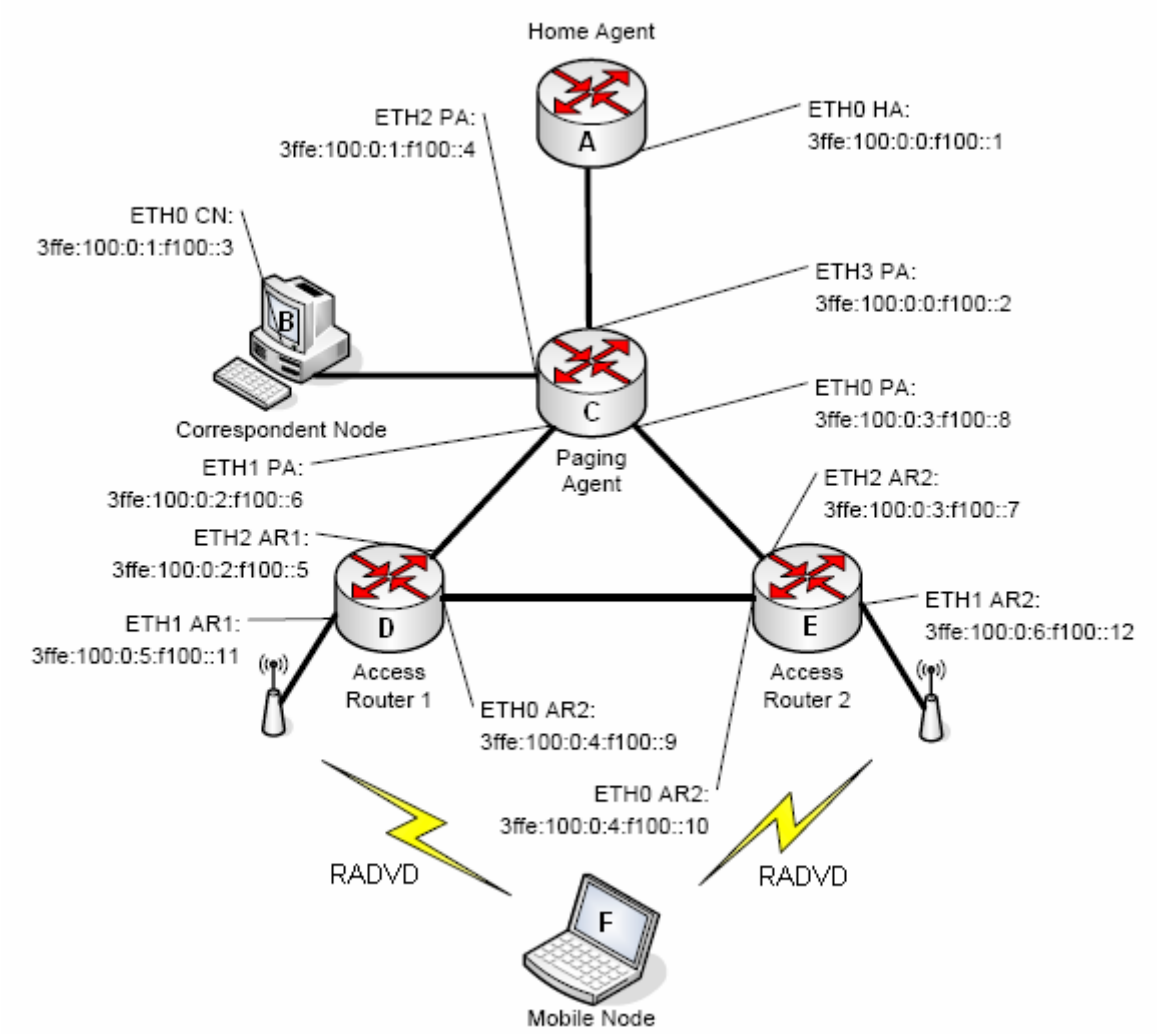


**Fig. IV.1:** Direccionamiento y topología de la maqueta IPv6 de MIPv6

### IV.1.1 Características hardware i software

Nuestra red la forman 5 PCs que tendrán la función de nodos del demostrador MIPv6 y un ordenador portátil que se utilizará como MN con conexión inalámbrica. A continuación detallaremos cada una de las características hardware de estos dispositivos tales como CPU, memoria RAM y tarjetas de red. La utilización de según qué tipo de hardware podrá influir de algún modo en el comportamiento del demostrador condicionado los resultados obtenidos. Otro aspecto a tener en cuenta es el software utilizado en los dispositivos de la maqueta como es el SO de las máquinas o los "drivers" de las tarjetas de red.

Además, la tabla también tendrá la información y características referentes a los AP y las tarjetas elegidas finalmente para realizar las pruebas.

**Tabla 3.1:** Características técnicas de los PCs utilizados

| **NODO A** | |
|---|---|
| **CPU** | Intel(R) Pentium(R) 4 CPU 3.00GHz |
| **TARJETA ETH0** | Realtek Semiconductor Co., Ltd. RT8139 |
| **SO** | Fedora Core 4 kernel 2.6.15 |
| **DRIVER ETH0** | Realtek Semiconductor Co., Ltd. RTL-8139/8139C/8139C+ (rev 10 |
| **NODO B** | |
| **CPU** | Intel(R) Pentium(R) 4 CPU 1.60GHz |
| **TARJETA ETH0** | Broadcom Corporation: device 8401 |
| **SO** | Fedora Core 4 kernel 2.6.15 |
| **DRIVER ETH0** | Broadcom Corporation BCM4401 100Base-T (rev 01) |
| **NODO C** | |
| **CPU** | Intel(R) Pentium(R) 4 CPU 1.80GHz |
| **Tarjeta ETH0** | 3Com Corporation 3C905C-TX Fast Etherlink for PC Management NIC |
| **Tarjeta ETH1** | 3Com Corporation 3C905C-TX Fast Etherlink for PC Management NIC |
| **Tarjeta ETH2** | 3Com Corporation 3C905B Fast Etherlink XL 10/100 |
| **Tarjeta ETH3** | 3Com Corporation 3C905C-TX Fast Etherlink for PC Management NIC |
| **SO** | Fedora Core 4 kernel 2.6.15 |
| **Driver ETH0** | 3Com Corporation 3c905C-TX/TX-M [Tornado] (rev 78) |
| **Driver ETH1** | 3Com Corporation 3c905C-TX/TX-M [Tornado] (rev 78) |
| **Driver ETH2** | 3Com Corporation 3c905B 100BaseTX [Cyclone] (rev 64) |
| **Driver R ETH3** | 3Com Corporation 3c905C-TX/TX-M [Tornado] (rev 78) |
| **NODO D** | |
| **CPU** | Intel(R) Pentium(R) 4 CPU 3.00GHz |
| **Tarjeta ETH0** | D-Link System Inc DFE-538TX 10/100 Ethernet Adapter |
| **Tarjeta ETH2** | ASUSTeK Computer Inc.: device 811d |

| | |
|---|---|
| **SO** | Fedora Core 4 kernel 2.6.15 |
| **Driver ETH0** | D-Link System Inc RTL8139 Ethernet (rev 10) |
| **Driver ETH2** | Intel Corporation 82541GI/PI Gigabit Ethernet Controller (rev 05) |
| **NODO E** | |
| **CPU** | Intel(R) Pentium(R) 4 CPU 3.00GHz |
| **Tarjeta ETH0** | D-Link System Inc DFE-538TX 10/100 Ethernet Adapter |
| **Tarjeta ETH2** | ASUSTeK Computer Inc.: device 811d |
| **SO** | Fedora Core 4 kernel 2.6.15 |
| **Driver ETH0** | D-Link System Inc RTL8139 Ethernet (rev 10) |
| **Driver ETH2** | Intel Corporation 82541GI/PI Gigabit Ethernet Controller (rev 05) |
| **NODO F** | |
| **CPU** | Intel Mobile Pentium 4-M 1.4GHz |
| **Tarjeta WLAN2** | LUCENT WaveLan SILVER (chipset Orinoco) |
| **SO** | UBUNTU 6.06 kernel 2.6.15 |
| **Driver WLAN2** | Wavelan (orinoco_cs/wavelan_cs) |

**Tabla 3.2:** Características técnicas de los dispositivos Wi-Fi utilizados

| | |
|---|---|
| **AP1** | |
| **Modelo** | Cisco Aironet 1130AG IEEE 802.11 A/B/G Access Point |
| **Certificaciones** | 802.11 a/b/g/h/d/i WPA/WPA2 |
| **AP2** | |
| **Modelo** | Cisco Aironet 1130AG IEEE 802.11 A/B/G Access Point |
| **Certificaciones** | 802.11 a/b/g/h/d/i WPA/WPA2 |
| **PCI con uAPSD** | |
| **Modelo** | Intel® Wireless Wi-Fi Link 4965AGN |
| **Certificaciones** | 802.11a/b/g/h/n/i WPA/WPA2 |
| **Chipset** | Intel 4965AGN |
| **Driver Windows** | Intel® Wireless WiFi Link 4965AGN: 11.5.0.32 |
| **Driver Linux** | iwlwifi-1.0.0-1 |
| **PCMCIA con PSM** | |
| **Modelo** | Cisco Airones 350 |
| **Certificaciones** | 802.11 a/b |
| **Chipset** | Cisco Air-PCM350 |
| **Driver** | Aironet driver (airo_cs) |

## IV.2 Instalación de MIPv6

La implementación utilizada durante el proyecto ha sido desarrollada por la organización Mobile-IPv6 de Linux. La versión utilizada ha sido la 2.0.1 compatible con el kernel 2.6.15. Primeramente habrá que parchear (hacer un *patch*) el kernel para modificar la configuración de IPv6 que incluye y así

hacerlo compatible con MIPv6. Para ello se utilizará el *patch mipv6-2.01-linux-2.6.15.patch.gz*. A continuación se instalará el demonio *mipv6-2.0.1.tar.gz*

## IV.2.1. Modificación del kernel

Extraemos el *patch* descomprimiéndolo para luego copiar mipv6-2.0.1-linux-2.6.15.patch en el directorio donde está el código fuente del kernel (/usr/src/Linux).

***# gunzip mipv6-2.0.1-linux-2.6.15.patch.gz***

Se aplica el *patch*.

***# patch –p1 < mipv6-2.0.1-linux-2.6.15.patch***

Una vez aplicado el *patch*, mediante *make menuconfig* se configura los siguientes parámetros para que mip6d sea compatible con el kernel. Si este método nos resultase costoso, se podrá guardar el archivo .*conf* del kernel para luego abrirlo y configurarlo con un procesador de textos.

**CONFIG_IPV6**=y //Activa la implementación del protocolo IPv6.
//el valor m (módulo) predeterminado.

**CONFIG_IPV6_MIP6**=y //Activa la implementación del protocolo IPv6.
//Parámetro nuevo.

**CONFIG_IPV6_TUNNEL**=y //Activa la opción de encapsular paquetes //IPv6 en paquetes IPv6. Es nuevo.

**CONFIG_IPV6_ADVANCED_ROUTER**=y //Activa el soporte de router IPv6

**CONFIG_IPV6_MULTIPLE_TABLES**=y //Activa el suporte a diferentes
//tablas de información sobre IPv6
//(BUL, rutes...). Es nuevo.

**CONFIG_IPV6_SUBTREES**=y //Activa les sub-ramas de IPv6. Es //nuevo.

**CONFIG_INET6_ESP**=y //Activa el soporte a IPsec en IPv6.
//Tiene el valor m predeterminado.

**CONFIG_XFRM_ENHANCEMENT**=y //Activa la opción de modificar
//la arquitectura de red de linux. Es
//nuevo.

**CONFIG_NET_KEY**=y //Activa el cifrado de red. M por //defecto.

**CONFIG_NET_KEY_MIGRATE**=y //Activa la opción de cambio de //claves cifradas en red

Para comprobar que todo está correctamente configurado ejecutamos el siguiente script antes de compilar el kernel. Este script nos informa de los parámetros mal configurados. Dicho script se encuentra en mipv6-2.0.1 y hay que pasarle como parámetro la ruta del código fuente que utilizaremos.

***# ./chkconf_kernel.sh /usr/src/linux/***

Ahora ya se puede compilar el kernel.

***# make clean*** //Borra los archivos de compilaciones anteriores.
***# make bzImage*** //Compila el kernel.
***# make modules*** //Compila los módulos del kernel.
***# make modules_install*** //Instala los módulos del kernel.
***# make install*** //Instala el nuevo kernel compilado y modifica el //gestor de arranque creando la opción de hincar el nuevo //kernel.

A partir de aquí ya podemos reiniciar el kernel y seguir con la instalación de mipv6.

Este procedimiento se hizo en el anterior proyecto y nosotros solo hemos tenido que repetirlo para el nuevo MN. Debido a los numerosos problemas irresolubles para aplicar dicho procedimiento al nuevo MN, hemos optado por modificarlo para que se pudiera adaptar a las características de el nuevo portátil ya que se producían diversas incompatibilidades.

### *IV.2.1.1 Modificación del kernel en el portátil.*

Para poder realizar los pasos anteriores con éxito en el ordenador portátil ha sido fundamental que se reunieran una serie de requisitos.

Se ha buscado un sistema operativo basado en *debian* y con el kernel 2.6.15 por defecto, Ubuntu 6.06. Una vez instalado el sistema operativo se procede a descargar las fuentes del kernel en la página http://www.eu.kernel.org/pub/linux/kernel/v2.6/ (importante no hacerlo mediante *apt-get install*). Y finalmente hay que compilarlo e instalarlo de la manera Ubuntu.

Descomprimimos el archivo de las fuentes en la carpeta /usr/src/ y creamos una carpeta virtual “Linux”.

**#tar xjf linux-2.6.15.tar.bz2**
**#ln -s linux-2.6.18.1 linux**
**#cd /usr/src/linux**

Copiamos la configuración actual de nuestro kernel por defecto.

**#cp /boot/config-`uname -r` ./.config**

Después de haber descomprimido el *patch* de mipv6 (apartado anterior IV.2.1) pasamos a parchear el kernel:

**#/usr/src/ *mipv6-2.0.1-linux-2.6.15.patch* | patch -p1 --dry-run**
**#/usr/src/ *mipv6-2.0.1-linux-2.6.15.patch* | patch -p1**

Ahora modificamos la configuración del kernel con los parámetros descritos en el apartado IV.2.1.

**#make menuconfig**

Ahora ya podemos compilar el kernel de la manera Ubuntu

**#make-kpkg clean**
**#fakeroot make-kpkg --initrd --append-to-version=-mipv6 kernel_image kernel_headers**

Instalamos el kernel.

**#cd /usr/src**
**#ls –l**
**#dpkg -i "nombre completo de la imagen".deb**
**#dpkg -i "nobmre complete de las cabeceras".deb**

Una vez instalado ya podemos reiniciar el ordenador y comprobar que hemos instalado el kernel correctamente mediante **#uname -r**

## IV.2.2. Instalación del demonio mip6d

Después de modificar y recompilar el kernel, para adaptarlo al protocolo MIPv6 tenemo que instalar el demonio mip6d que gestionará y aplicará las funciones definidas en el protocolo.

Descomprimimos el paquete con los archivos de instalación y el código fuente.

***# gunzip mipv6-2.0.1.tar.gz***
***# tar –zxvf mipv6-2.0.1.tar***

Una vez creado el directorio mipv6-2.0.1. se ejecutará el *script ./configure* que contiene este directorio pasándole como parámetro la ubicación del código

fuente del kernel compilado. El script configura el kernel a partir de esta información.

***# CPPFLAGS='-isystem /usr/src/linux/include' ./configure***

Una vez configurado el *Makefile* ya podemos compilar e instalar el demonio.

***# make***
***# make install***

Una vez instalado, ya podemos iniciar el demonio llamándolo desde la *shell* con la comanda **mip6d**.

## IV.3 Configuración de MIPv6

Gracias a utilizar *linux* podemos configurar desde el *user space* utilizando el archivo */usr/local/etc/mip6d.conf*, una gran serie de parámetros que harán posible la personalización del comportamiento de MIPv6. Los parámetros los hemos subdividido en grupos según pertenezcan al HA, MN, CN o el conjunto de ambos. Subrayaremos los parámetros que en nuestro caso hemos configurado.

| **Parámetros comunes** | **Opciones** | **Descripción** |
|---|---|---|
| NodeConfig | CN \| HA \| MN | Fija el comportamiento del nodo dentro de MIPv6 |
| DebugLevel | 0 – 10 (10 maximo) | Activa la información en tiempo real del demonio por pantalla |
| DoRouteOptimizationCN | Enabled \| Disabled | Activa el mecanismo de Route Optimization con los MN |
| **Parametros communes en el HA I en el MN** | | |
| Interface | "Name" (ethX) MnIfPreference IfType | Indica la interficie que se utilizará para el protocolo |
| UseMnHaIPsec | Enabled \| Disabled | Activa la opción de utilizar IPsec para proteger la señalización entre HA i MN |
| KeyMngMobCapability | Enabled \| Disabled | Activa la utilización de las claves dinámicas para proteger la señalización |
| IPsecPolicySet | HomeAgentAddress HomeAddress IPsecPolicy | Define la política de seguridad de la comunicación para cada HA i HoA |
| **Parámetros del HA** | | |
| HaMaxBindingLife | Number (segons) | Limita el tiempo màximo durante el cual es valido un BU de un MN |

| | | |
|---|---|---|
| SendMobPfxAdvs | Enabled \| Disabled | Activa el enviamiento de mensajes Mobile Prefix Advertisement a los MN |
| SendUnsolMObPfxAdvs | Enabled \| Disabled | Activa el re-envío de MP Advertisements periódicamente |
| MinMobPfxAdvInterval | Number (segons) | Indica el intervalo mínimo entre transmisiones de MP Advertisements |
| BindingAclPolicy address | Allow \| Deny | Define el comportamiento del HA con un MN determinado por address |
| DefaultBindingAclPolicy | Allow \| Deny | Define el comportamiento por defecto del HA en cuanto a aceptación de BU |
| **Parametros del MN** | | |
| MnMaxHaBindingLife | Number (segons) | Indica el tiempo máximo durante el cual son validos los BU con el HA |
| MnMaxCnBindingLife | Number (segons) | Indica el tiempo máximo durante el cual son validos los BU con los CNs |
| MnDiscardHaParamProb | Enabled \| Disabled | Activa la opción de ignorar los paquetes ICMPv6 de problemas con los parámetros del BU del HA. Evita ataques de DoS |
| SendMobPfxSols | Enabled \| Disabled | Activa la transmisión de mensajes Mobile Prefix Solicitations al HA |
| DoRouteOptimizationMN | Enabled \| Disabled | Activa el mecanismo Route Optimization con los CNs |
| MnUseAllInterfaces | Enabled \| Disabled | Indica que todas las interfícies se utilizarán para la mobilidad |
| UseCnBuAck | Enabled \| Disabled | Indica si se utilizará el bit ACK de los BU con los CN |
| MnRouterProbes | Number | Fija el numero de veces que se enviarán pruevas de NUD antes de asumir que no hay conectividad con el router |
| MnRouterProbeTimeout | Number (segundos) | Fija el tiempo durante el cual se realizará el NUD |
| MnHomeLink “name” | HomeAddress<br>HomeAgentAddress<br>MnRoPolicy | Información del HA, la HoA i la política de utilización de Router Optimization en un home link definido estáticamente |

## IV.4 Mensajes que intervienen en el traspaso MIPv6

A continuación, en las siguientes figuras se representan los mensajes utilizados por el protocolo MIPv6 analizado en el capítulo 3.

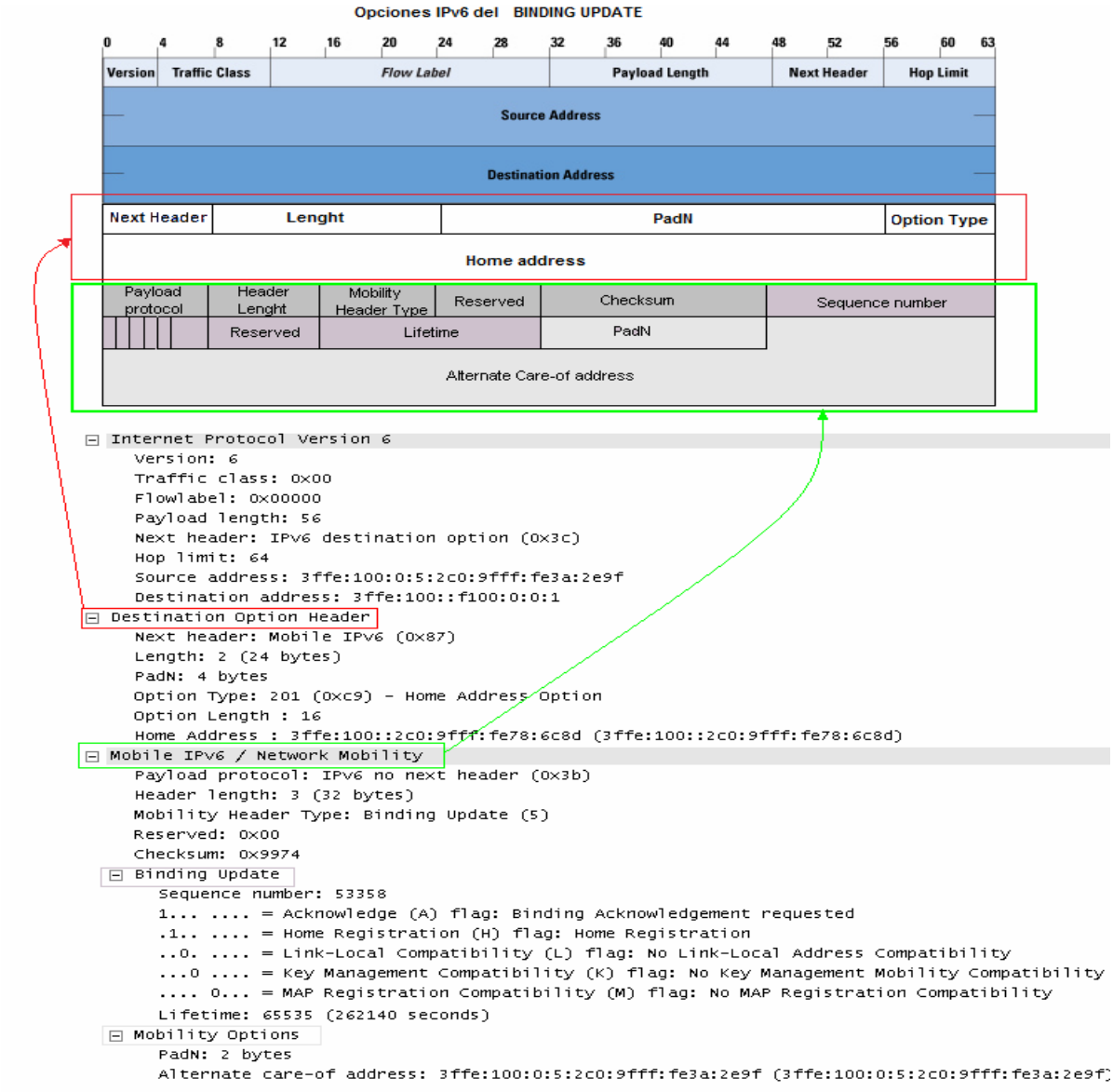


**Fig. IV.2.** Binding Update

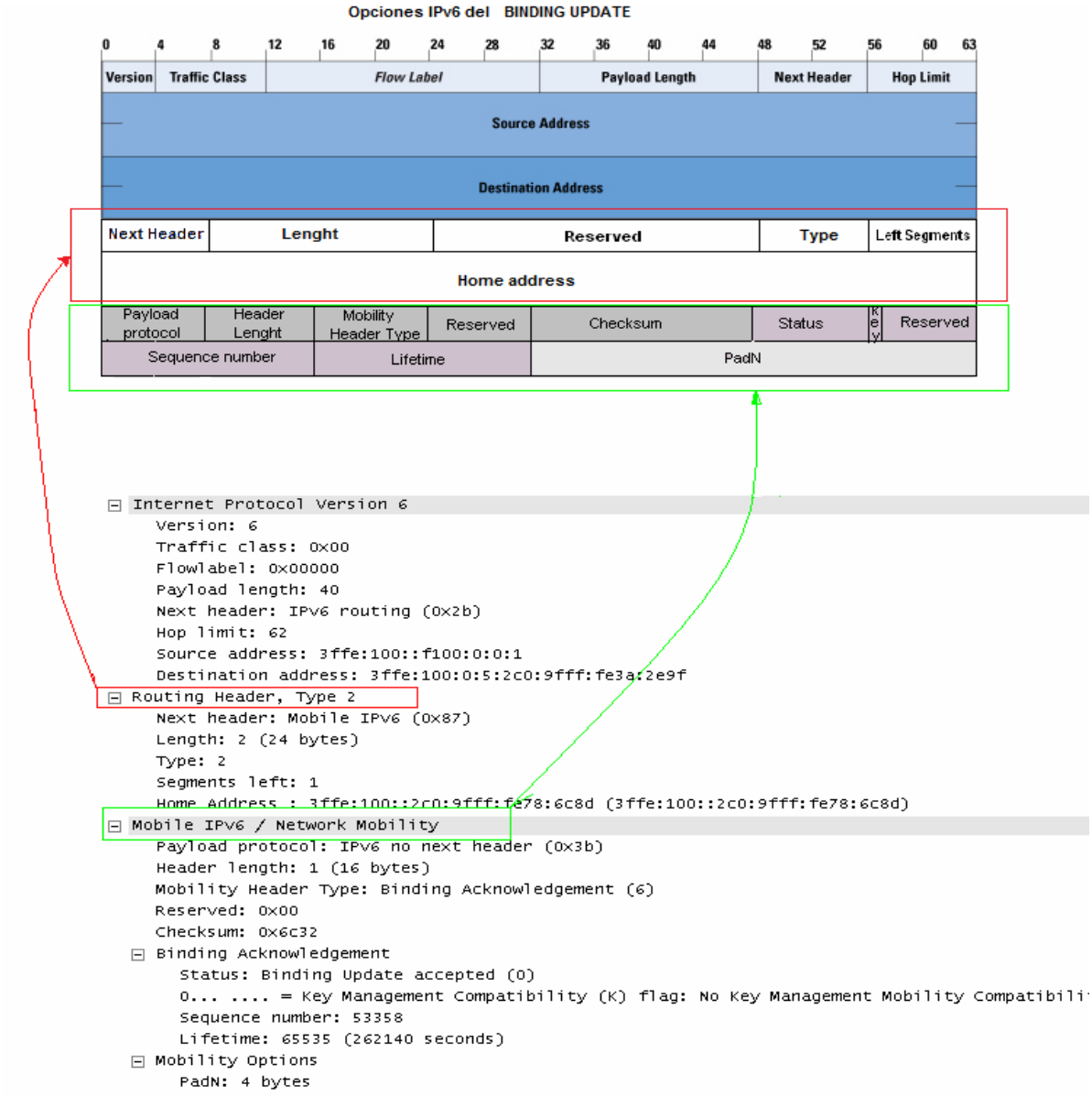


**Fig. IV.3**. Binding Ack

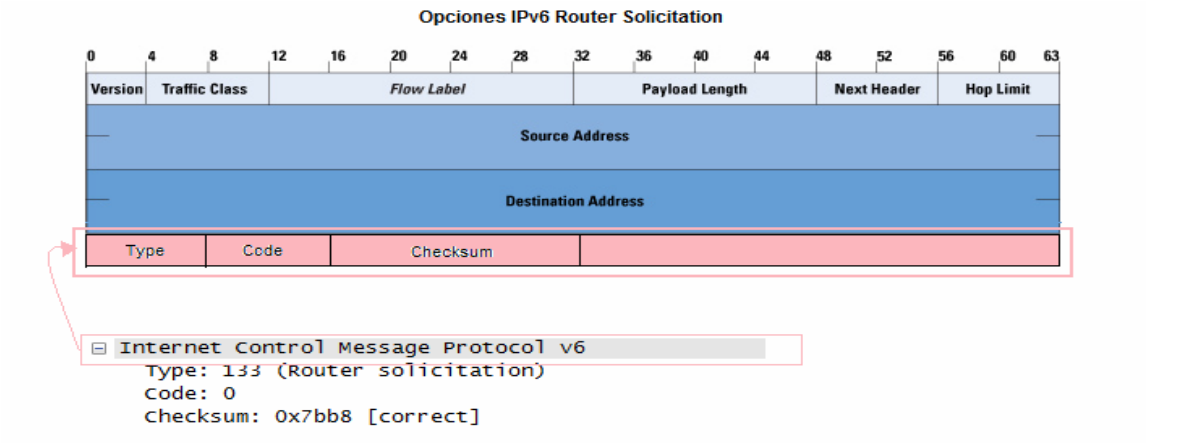


**Fig. IV.4**. Router Solicitation

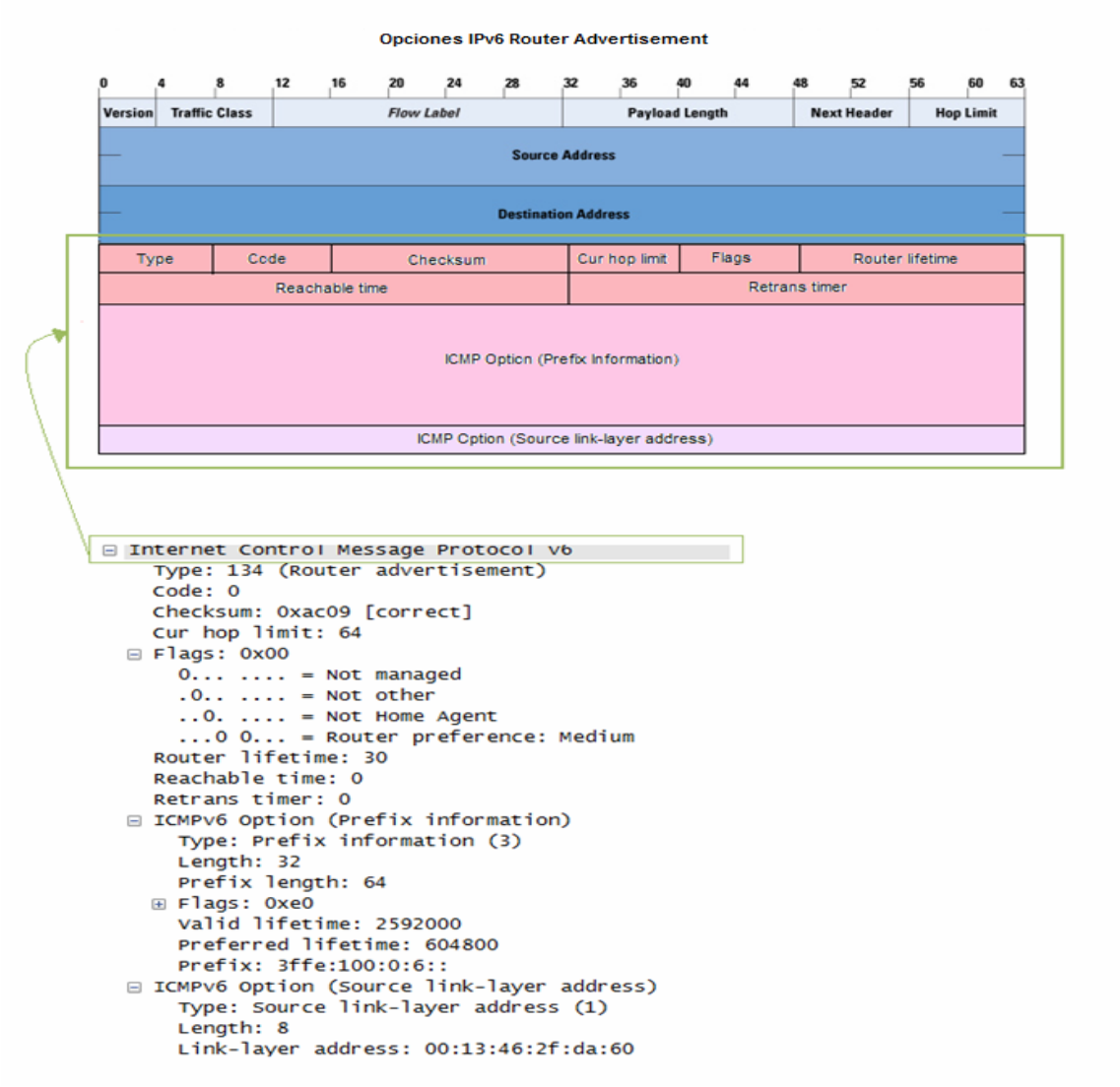


**Fig. IV.5**. Router Advertisement

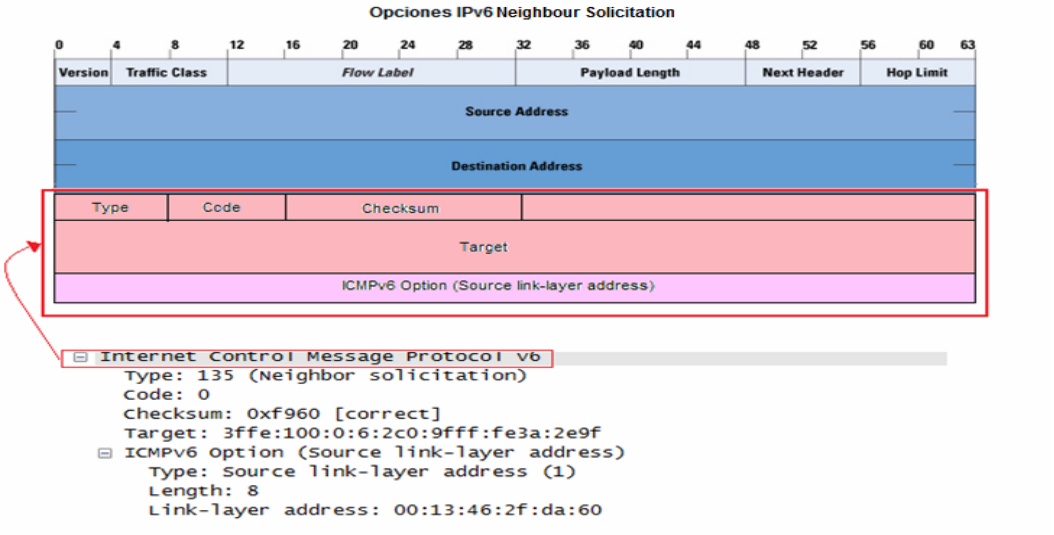


**Fig. IV.6**. Neighbor Solicitation

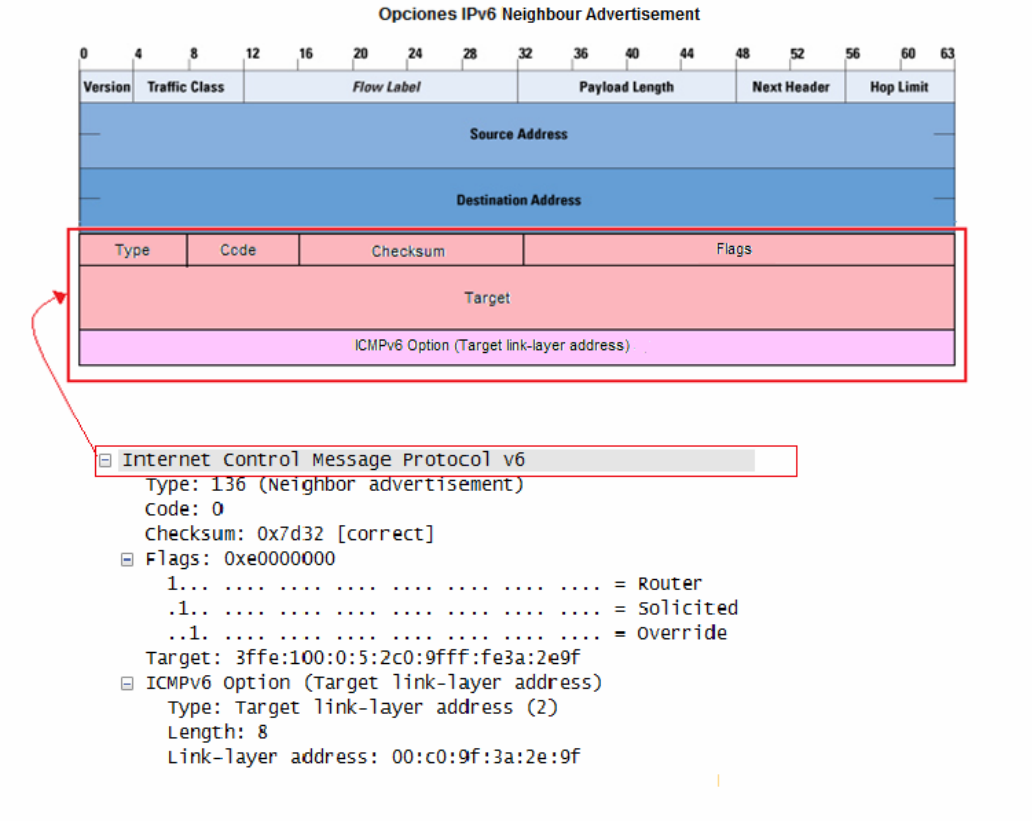


**Fig. IV.7**. Neighbor Advertisement

## IV.5 Direcciones origen y destino

En la captura de la Figura IV.8 se puede comprobar la comunicación necesaria de este protocolo. Como puede apreciarse, las direcciones origen y destino no son siempre la misma en los distintos mensajes. Esto es porque dependiendo del mensaje, se utilizará una dirección u otra, *multicast* o *unicast,* global o local.

El *Neighbour Discovery Protocol* (NPD) utiliza la dirección *multicast ff02::2* para enviar a todos los *routers* de la red local un *Router solicitation*. El *Router advertisement* lo responde el *router* conectado localmente al host y su dirección origen es de tipo *link-local.* Como destino utiliza la dirección *multicast ff02::1* perteneciente a todos los hosts de la red local. El protocolo *DAD* se aplica con el envío del *Neighbour Solicitation* como ya dijimos en el apartado 4.1.2.1. Este mensaje tiene como destino la dirección *multicast Solicited Node* (SN) *ff02::1:ff3a:2e9f* formada por la dirección *multicast* de todos los nodos mas los 24 últimos bits de la dirección IPv6 *link-local* que permitirá comprobar su unicidad. El BU que envía el MN tiene como destino la dirección IP del HA *3ffe:100::f100::1* y como origen la que se ha formado con la autoconfiguración, 64 bits de la dirección global mas los últimos 64 bits de la dirección *link-local.*

Antes de Recibir el *BAck*, para conocer el camino que deberá seguir, el *Access Router,* este envía un *Neighbour Solicitation* con origen su dirección *link-local* y destino el SN. El *router* recibe la respuesta del MN con su dirección global como origen. Ahora ya puede enviarse desde el HA el *BAck*.

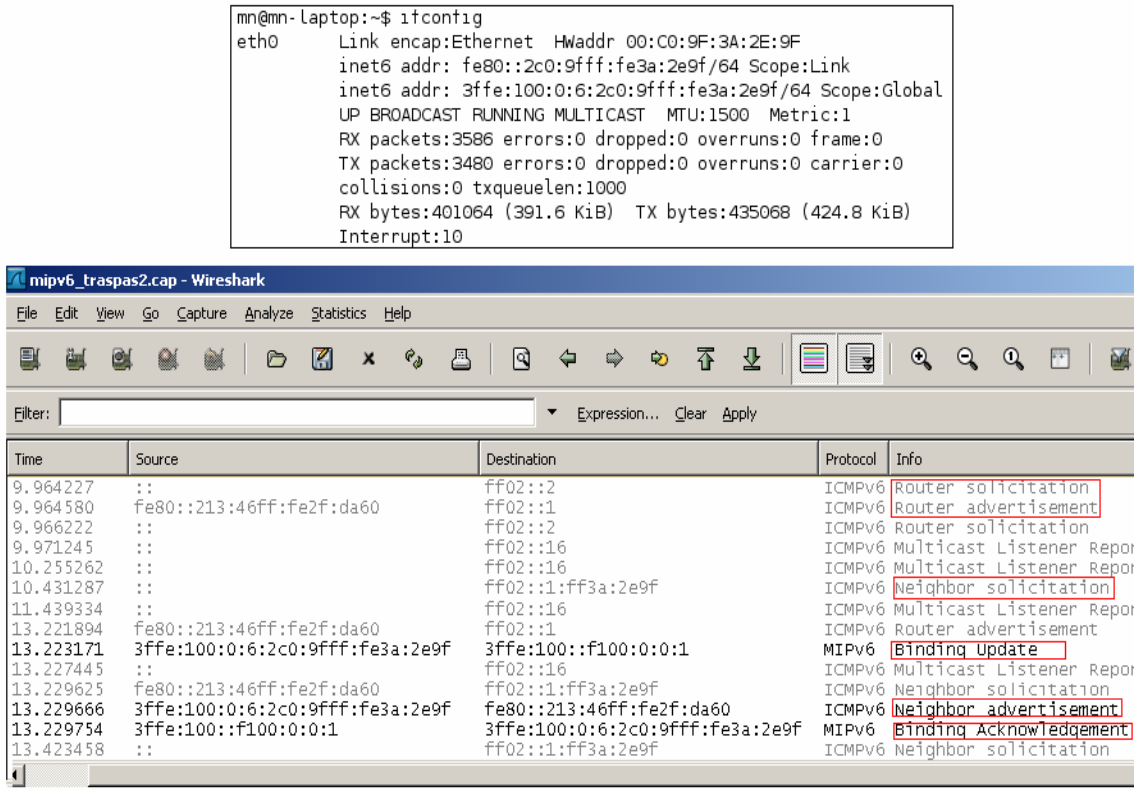


**Fig. IV.8**. Captura de un traspaso MIPv6

## IV.6 IEEE802.11

En este apartado se analizará una captura de un traspaso a nivel de enlace utilizando el protocolo IEEE802.11 para extraer la información necesaria y realizar los cálculos que nos servirán para programar la aplicación de análisis de traspasos. Cabe decir que en cada traspaso MIPv6 también se producirá un traspaso IEEE802.11 y por tanto también es necesario caracterizarlo.

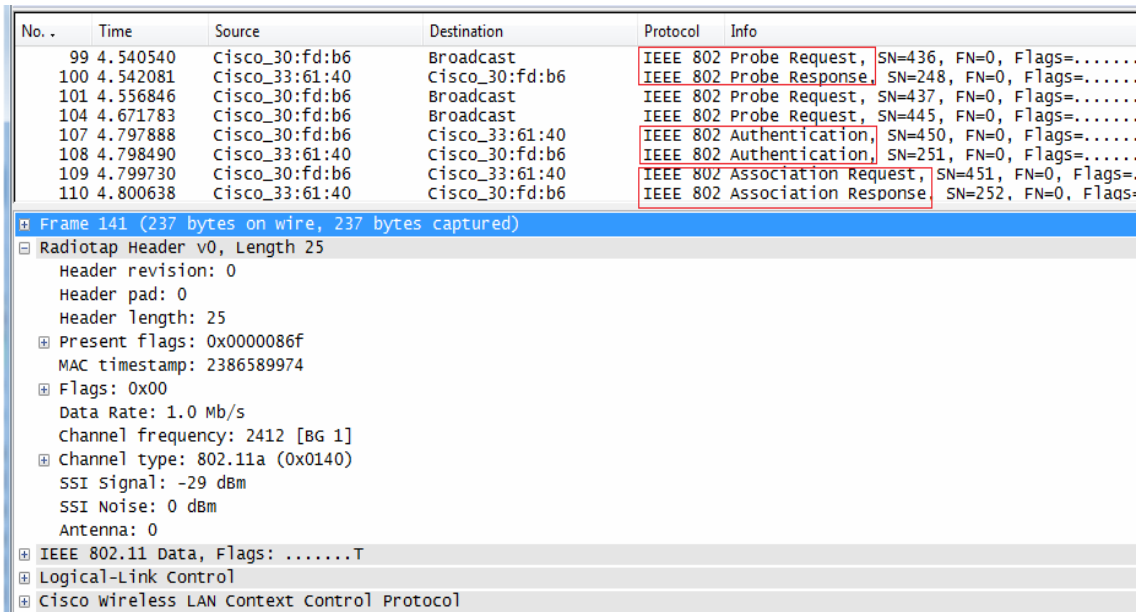


**Fig. IV.9.** Captura de un traspaso IEEE802.11

Para calcular cada uno de los mensajes hemos descartado la cabecera *Radiotap* (25 bytes) introducida por el driver de la tarjeta. Esta cabecera se añade para completar la información del medio radio añadiendo entre otros, la velocidad a la que fue capturado el paquete así como el canal en el que opera.

En la Figura IV.9 puede verse los mensajes que intervienen en el traspaso marcados en rojo. Tal y como se muestra en el gráfico 3.22, el tiempo de espera entre el *Probe Response* y el *Authentication Request* es de unos 255 ms.

# ANEXO V: Resultados de consumo MIPv6

En este anexo se presentan los resultados en formato numérico obtenidos por la aplicación creada para analizar traspasos (ver Anexo VI) utilizando los valores de consumo de la tarjeta PCMCIA *Cisco Airones 350 (ver Tabla V1)*. Con las tablas presentadas en este anexo podrá verse con mayor detalle el consumo producido por cada uno de los mensajes que intervienen en el traspaso. .

**Tabla V.1**. Valores de consumo de la tarjeta Cisco Airones 350

| | mA | Vcc (V) | mW |
|---|---|---|---|
| **Recepción** | 260 | 5 | 1300 |
| **Transmissión** | 375 | 5 | 1875 |
| **Idle** | 216 | 5 | 1080 |

Gracias a los cálculos de (4.1) hemos obtenido los tamaños de paquete y trama. Para ello, se han tenido en cuenta tanto los valores teóricos estandarizados como los obtenidos en las capturas realizadas. Después de obtener los tamaños de trama, calculamos el tiempo empleado por la STA para enviar o recibir las tramas utilizando como parámetros las características de IEEE802.11b que utiliza la tarjeta Cisco (4.2). Con los valores de la Tabla V.1 se ha obtenido el consumo de los mensajes y tiempos de espera de los traspasos. En (V.1) se pueden ver las características de las tramas 802.11b y 802.11g que transportan paquetes IP.

**(V.1)**

Trama 802.11b = Long Preamble + PLCP H. + MAC H. + LLC + FCS + paquete IPv6
Trama 802.11b = 18 + 6 + 24 + 8 + 4 + paquete IPv6 = 60 bytes + paquete IPv6
**Trama 802.11b =** 24 bytes (**1Mbps**) + 36 bytes (**11Mbps**) + paquete IPv6
**Trama 802.11g =** Short Preamble + " " = 9 bytes (**1 Mbps**) + 6 bytes (**2Mbps**) + 36 bytes (**54Mbps**) + paquete IPv6
**Paquet IP** = Header + Payload = **40 bytes + Payload**

**Tabla V.2.** Consumo de los mensajes IP de MIPv6

| | Paquete IP(B) | Tiempo (ms) | Consumo Rx (mWh) | Consumo Tx (mWh) |
|---|---|---|---|---|
| **RtSol** | 48 | 0,03491 | | 0,000018 |
| **RtAdv** | 96 | 0,06982 | 0,000025 | |
| **BU** | 96 | 0,06982 | | 0,000036 |
| **Back** | 80 | 0,05818 | 0,000021 | |
| **NbSol** | 72 | 0,05236 | 0,000019 | |
| **Nadv** | 86 | 0,06255 | | 0,000033 |
| **TOTAL** | 478 | 0,34764 | **0,000065** | **0,000087** |

**Tabla V.3.** Consumo de tramas MIPv6 y tiempos de espera

| | Tiempo (ms) | Consumo Rx (mWh) | Consumo Tx (mWh) | Consumo Idle (mWh) |
|---|---|---|---|---|
| T. espera | 70 | | | 0,021 |
| **RtSol** | 0,25309 | | 0,000132 | |
| T. espera | 0,3 | | | 0,00009 |
| **RtAdv** | 0,288 | 0,000104 | | |
| T. espera | 465 | | | 0,1395 |
| **NbSol** | 0,27055 | | 0,000141 | |
| T. espera | 100 | | | 0,03 |
| **NbSol** | 0,27055 | | 0,000141 | |
| T. espera | 2671 | | | 0,8013 |
| **BU** | 0,28800 | | 0,00015 | |
| T. espera | 6,5 | | | 0,00195 |
| **NbSol** | 0,27055 | 0,000098 | | |
| T. espera | 0,1 | | | 0,00003 |
| **NbAdv** | 0,28073 | | 0,000146 | |
| T. espera | 0,2 | | | 0,00006 |
| **Back** | 0,27636 | 0,0001 | | |
| **TOTAL** | **3315,298** | **0,000302** | **0,00071** | **0,99393** |

De la misma manera que en las anteriores tablas, los resultados de la Tabla V.4 y V.5 se han obtenido mediante (4.3) y (4.4). El programa *Kismet* nos ha permitido obtener el tamaño de los *tagged parameters* de las tramas IEEE802.11. En (V.2) vemos la formación de una trama 802.11 correspondiente a un traspaso

**(V.2)**

**Trama 802.11 =** Long Preamble + PLCP H. + MAC H. + FCS + Tagged parameters = 18 + 6 + 24 + 4 + variable = **52 bytes + Tagged parameters**
**Trama MAC** = MAC H. + FCS + Tagged parameters = 24 + 4 + variable = **28 bytes + Tagged parameters**

**Tabla V.4**. Consumo de los mensajes IEEE802.11 sin preámbulo

| | Trama MAC (B) | Tiempo (ms) | Consumo Rx (mWh) | Consumo Tx (mWh) |
|---|---|---|---|---|
| **PbReq** | 80 | 0,64 | | 0,000333 |
| **Pb Resp** | 310 | 2,48 | 0,000896 | |
| **Auth. Req** | 64 | 0,512 | | 0,000267 |
| **Auth. Resp** | 64 | 0,512 | 0,000185 | |
| **ReassReq** | 182 | 1,456 | | 0,000758 |
| **ReassResp** | 192 | 1,536 | 0,000555 | |
| **TOTAL** | 892 | 7,136 | **0,001636** | **0,001358** |

**Tabla V.5.** Consumo de tramas IEEE802.11b con preámbulo y tiempos de espera

| | Tiempo (ms) | Consumo Rx (mWh) | Consumo Tx (mWh) | Consumo Idle (mWh) |
|---|---|---|---|---|
| **PbReq** | 0,64 | | 0,000433 | |
| T. espera | 1,5 | | | 0,00045 |
| **PbResp** | 2,48 | 0,000965 | | |
| T. espera | 255,8 | | | 0,07674 |
| **Auth. Req** | 0,512 | | 0,000367 | |
| T. espera | 0,6 | | | 0,00018 |
| **Auth. Resp** | 0,512 | 0,000254 | | |
| T. espera | 1,3 | | | 0,00039 |
| **ReassReq** | 1,456 | | 0,000858 | |
| T. espera | 0,9 | | | 0,00027 |
| **ReassResp** | 1,536 | 0,000624 | | |
| **TOTAL** | 7,136 | **0,001843** | **0,001658** | **0,07803** |

La Tabla V.6 contiene los valores del consumo producido por el escaneo dependiendo de los canales ocupados y el número de AP por canal.

**Tabla V.6**: Consumo producido por el escaneo en IEEE802.11b según número de canales y AP por canal.

| CONSUMO DE ESCANEO (mWh) | Canales activos | | |
|---|---|---|---|
| **AP por canal** | 1 | 2 | 3 |
| 1 | 0,0351924 | 0,0353558 | 0,0355192 |
| 2 | 0,0365558 | 0,0380826 | 0,0396094 |
| 3 | 0,0367192 | 0,0384094 | 0,0400996 |

# ANEXO VI: Aplicación para análisis consumo en los traspasos

Para analizar el consumo de los traspasos, se ha programado una aplicación Excel en lenguaje Visual Basic Aplication (VBA), que es una variante de Visual Basic para Excel. Esta aplicación recoge de un cuadro de usuario los valores de consumo de la tarjeta en recepción, transmisión e idle que se introduce. Se ha de elegir también el protocolo con el que trabaja la tarjeta, IEEE802.11b o IEEE802.11g. Con estos datos, la aplicación, según unas fórmulas programadas, calcula el consumo de los distintos mensajes que intervienen el traspaso tanto a nivel de enlace como de red. Además, gracias a las capturas presentadas en el anexo IV se ha estimado los tiempos de espera entre mensajes cuyos consumos son calculados por la aplicación. El conjunto de todos estos consumos, escaneo, traspaso a nivel de enlace y traspaso a nivel de red se representa gráficamente con otra de las opciones que tiene la aplicación. Para poder utilizar la aplicación es necesario disponer de Office 2007 y habilitar la opción de macros.

En este anexo primero se va a presentar un pequeño manual de usuario de la aplicación y a continuación se detallará el código utilizado para programarla.

## VI.1 Manual de usuario

Este es el aspecto de la aplicación:

**CONSUMO TARJETA UTILIZADA**
***IEEE 802.11b (11Mbps)***
Por defecto: CISCO AIRONET 350 (Cisco AIR-PCM350 chipset) IEEE802.11b

| | mA | Vcc (V) | mW |
|---|---|---|---|
| **Recepció** | 260 | 5 | 1300 |
| **Transmissió** | 375 | 5 | 1875 |
| **Idle** | 216 | 5 | 1080 |

Mostrar/Esconder Comentarios

Insertar valores de la tarjeta WI-FI

Mostrar Gráfico

**Paquetes MIPv6**

| | Paquete IP(B) | Trama 802.11(B) | Tiempo (ms) | Consumo Rx (mWh) | Consumo Tx (mWh) | Tiempo (ms) | Consumo Rx (mWh) | Consumo Tx (mWh) | TOTAL (mWh) |
|---|---|---|---|---|---|---|---|---|---|
| **RtSol** | 48 | 108 | 0,03491 | | 0,000018 | 0,25309 | | 0,000132 | |
| **RtAdv** | 96 | 156 | 0,06982 | 0,000025 | | 0,28800 | 0,000104 | | |
| **BU** | 96 | 156 | 0,06982 | | 0,000036 | 0,28800 | | 0,00015 | |
| **Back** | 80 | 140 | 0,05818 | 0,000021 | | 0,27636 | 0,0001 | | |
| **NbSol** | 72 | 132 | 0,05236 | 0,000019 | | 0,27055 | 0,000098 | 0,000282 | |
| **Nadv** | 86 | 146 | 0,06255 | | 0,000033 | 0,28073 | | 0,000146 | |
| **TOTAL** | 478 | 502 | 0,34764 | **0,000065** | **0,000087** | 1,65673 | **0,000302** | **0,00071** | **0,001012** |

**Paquetes portocolo IEEE 802.11 (escaneo, autenticación, asociación)**

| | Trama MAC (B) | Trama 802.11(B) | Tiempo (ms) | Consumo Rx (mWh) | Consumo Tx (mWh) | Tiempo (ms) | Consumo Rx (mWh) | Consumo Tx (mWh) | TOTAL (mWh) |
|---|---|---|---|---|---|---|---|---|---|
| **PbReq** | 80 | 104 | 0,64 | | 0,000333 | 0,832 | | 0,000433 | |
| **Pb Resp** | 310 | 334 | 2,48 | 0,000896 | | 2,672 | 0,000965 | | |
| **Auth. Req** | 64 | 88 | 0,512 | | 0,000267 | 0,704 | | 0,000367 | |
| **Auth. Resp** | 64 | 88 | 0,512 | 0,000185 | | 0,704 | 0,000254 | | |
| **ReassReq** | 182 | 206 | 1,456 | | 0,000758 | 1,648 | | 0,000858 | |
| **ReassResp** | 192 | 216 | 1,536 | 0,000555 | | 1,728 | 0,000624 | | |
| **TOTAL** | 892 | 1036 | 7,136 | **0,001636** | **0,001358** | 8,288 | **0,001843** | **0,001658** | **0,003501** |

| **Consumo de la secuencia completa de un traspaso MIPv6 con los tiempos de espera entre mensajes** | | | | | TOTAL con: | | |
|---|---|---|---|---|---|---|---|
| | **Tiempo (ms)** | **Consumo Rx (mWh)** | **Consumo Tx (mWh)** | **Consumo Idle (mWh)** | **DAD** | **ODAD** | **sin DAD** |
| T. espera | 70 | | | 0,021 | **0,995242** | **0,193942** | **0,024096** |
| **RtSol** | 0,25309 | | 0,000162 | | | | |
| T. espera | 0,3 | | | 0,00009 | | | |
| **RtAdv** | 0,288 | 0,000152 | | | | | |
| T. espera | 465 | | | 0,1395 | | | |
| **NbSol** | 0,27055 | | 0,000173 | | | | |
| T. espera | 100 | | | 0,03 | | | |
| **NbSol** | 0,27055 | | 0,000173 | | | | |
| T. espera | 2671 | | | 0,8013 | | | |
| **BU** | 0,28800 | | 0,000184 | | | | |
| T. espera | 6,5 | | | 0,00195 | | | |
| **NbSol** | 0,27055 | 0,000143 | | | | | |
| T. espera | 0,1 | | | 0,00003 | | | |
| **NbAdv** | 0,28073 | | 0,000179 | | | | |
| T. espera | 0,2 | | | 0,00006 | | | |
| **Back** | 0,27636 | 0,000146 | | | | | |
| **TOTAL** | **3315,298** | **0,000441** | **0,000871** | **0,99393** | | | |

**Consumo de la secuencia completa de un traspaso MIPv6 con los tiempos de espera entre mensajes**

| | Tiempo (ms) | Consumo Rx (mWh) | Consumo Tx (mWh) | Consumo Idle (mWh) | TOTAL | TOTAL sin escaneo | Autenticación | Reasociación |
|---|---|---|---|---|---|---|---|---|
| **PbReq** | 0,64 | | 0,000532 | | **0,082759** | **0,080367** | **0,077742** | **0,002625** |
| T. espera | 1,5 | | | 0,00045 | | | **0,001002** | **0,002235** |
| **PbResp** | 2,48 | 0,00141 | | | | **TOTAL con escaneo** | | |
| T. espera | 255,8 | | | 0,07674 | | **0,1172914** | | |
| **Auth. Req** | 0,512 | | 0,00045 | | | | | |
| T. espera | 0,6 | | | 0,00018 | | | | |
| **Auth. Resp** | 0,512 | 0,000372 | | | | | | |
| T. espera | 1,3 | | | 0,00039 | | | | |
| **ReassReq** | 1,456 | | 0,001053 | | | | | |
| T. espera | 0,9 | | | 0,00027 | | | | |
| **ReassResp** | 1,536 | 0,000912 | | | | | | |

| | | | | |
|---|---|---|---|---|
| **TOTAL** | 7,136 | **0,002694** | **0,002035** | **0,07803** |

### VI.1.1 Opción "Mostrar/Esconder comentarios"

Con esta opción podremos ver todos los comentarios explicativos que hay en las celdas de la aplicación sin tener que pasar por encima de cada una de ellas. En la Figura VI.1 se muestra el contenido de la pantalla después de presionar la opción.

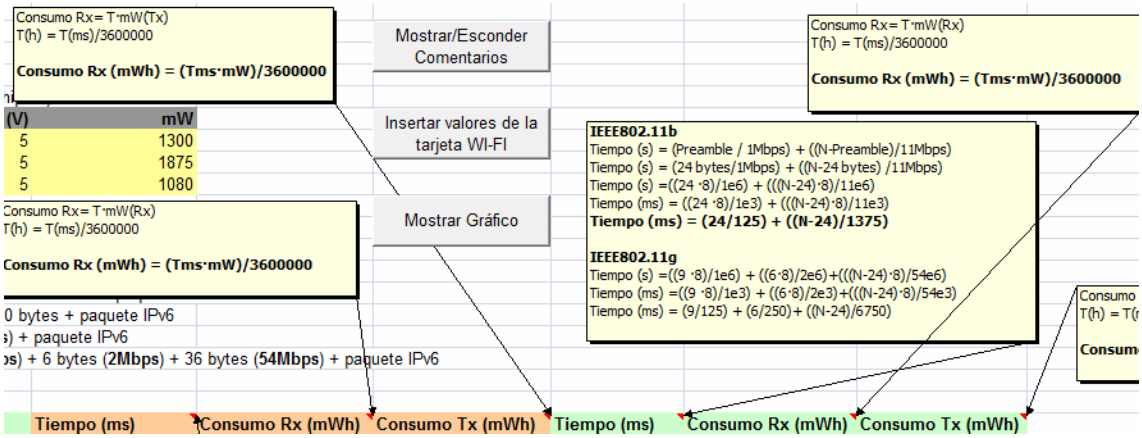


**Fig. VI.1**. Opción "Mostrar/Esconder comentarios"

### VI.1.2 Opción "Insertar valores de la tarjeta WI-FI"

Esta opción permite introducir los valores de consumo en mA de las tarjetas de las cuales se quiere medir el impacto de un traspaso MIPv6 o IEEE802.11. Se ha dado la opción de elegir también el protocolo utilizado, IEEE802.11b o IEEE802.11g. Según utilice la tarjeta un protocolo u otro, el consumo variará debido a la diferencia de velocidad y características de las cabeceras y preámbulos. En la Fig. VI.2 se muestra un ejemplo de la ventana *UserForm* que aparece al presionar la opción.


**Fig. VI.2**. Opción "Insertar valores de la tarjeta WI-FI"

### VI.1.3 Opción “Mostrar Gráfico”

Con esta opción de la aplicación podremos ver multitud de gráficos que expresan valores de consumo y tiempo en los traspasos analizados. Además se podrá ver una comparativa de los dos traspasos estudiados según las características aplicadas. En la Figura VI.3 se puede ver el aspecto de la interfaz de para generar gráficos.

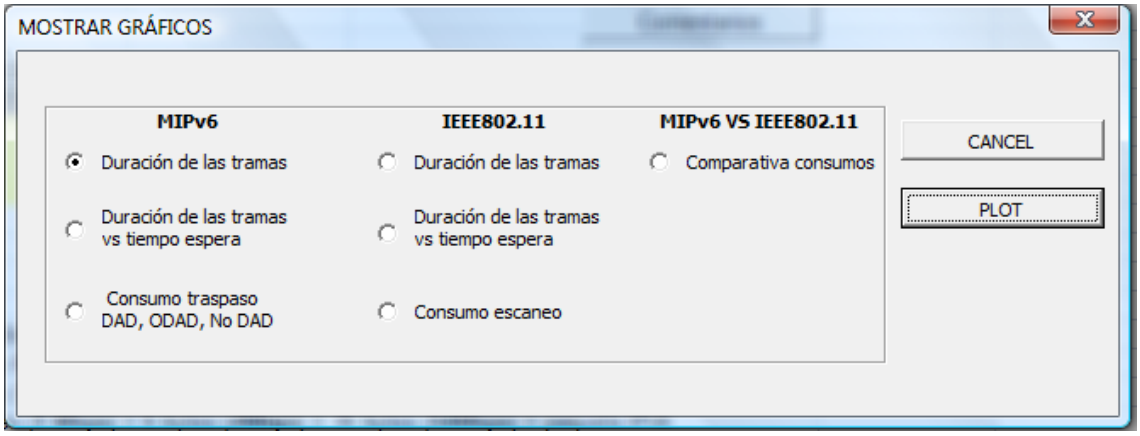


**Fig. VI.3**. Opción “Mostrar Gráfico”

A continuación, se van a mostrar algunos de los gráficos que genera la aplicación. (Ver Fig. VI.4).

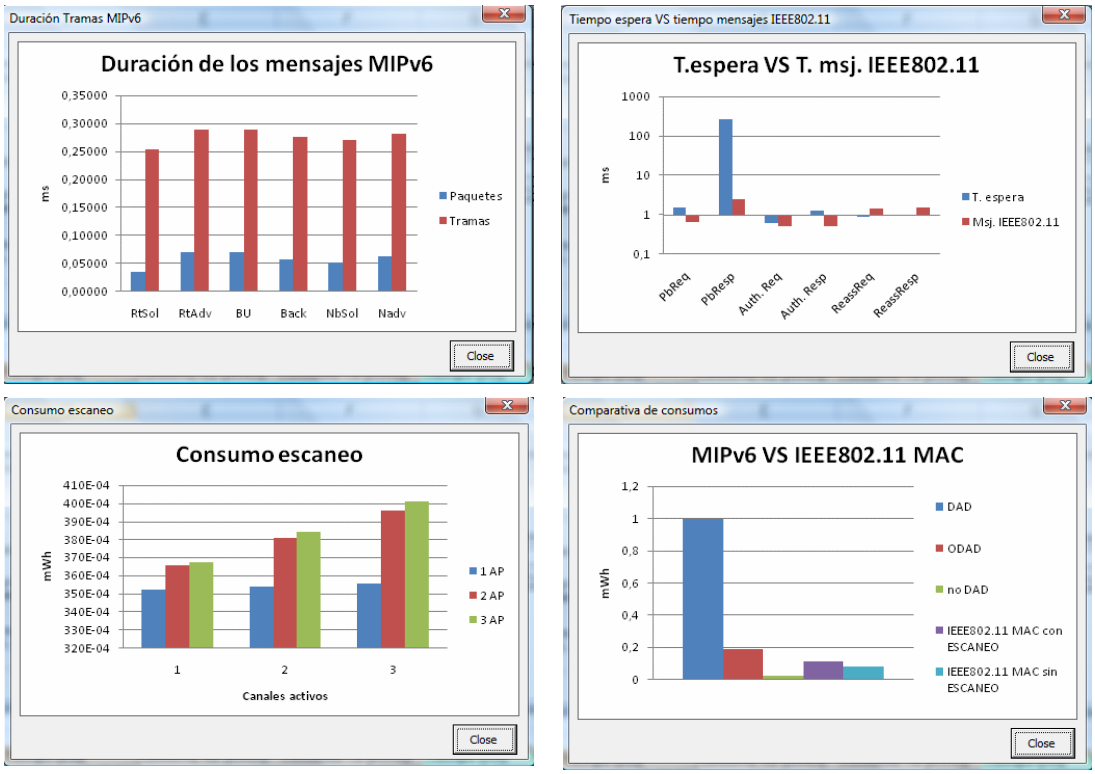


**Fig. VI.4**. Gráficos que muestra la aplicación

## VI.2 Código VBA utilizado

A continuación se muestra el código separado por UserForms y módulos de VBA.

### VI.4.1 UserForm 1

```
Private Sub CommandButton1_Click()
    Unload UserForm1
End Sub

Private Sub CommandButton2_Click()
'Sheets("Consumo MIPv6").Activate

Static Tiempo1
Dim Tiempo2 As String
Dim i
Tiempo1 = "=(C"
Tiempo2 = "7"
Cells(7, 2) = Rx.Value
Cells(8, 2) = Tx.Value
Cells(9, 2) = Idle.Value

If Optionb Then
```

```
'=REDONDEAR((B21:B27/1375);5)
For i = 21 To 26 Step 1
Worksheets("Consumo MIPv6").Cells(i, 3) = "=B" & i & " + 60"
Worksheets("Consumo MIPv6").Cells(i, 4) = "=(B" & i & "/1375)"
Worksheets("Consumo MIPv6").Cells(i, 7) = "=((24/125)+((C" & i & ")-
24)/1375)"
Next i

For j = 38 To 43 Step 1
Worksheets("Consumo MIPv6").Cells(i, 3) = "=B" & i & " + 24"
Worksheets("Consumo MIPv6").Cells(j, 4) = "=(B" & j & "/125)"
Worksheets("Consumo MIPv6").Cells(j, 7) = "=(C" & j & "/125)"
Next j
End If

If Optiong Then
'Tiempo (ms) = (18/125) + (6/250)+ ((N-24)/6750)
For i = 21 To 26 Step 1
Worksheets("Consumo MIPv6").Cells(i, 3) = "=B" & i & " + 51"
Worksheets("Consumo MIPv6").Cells(i, 4) = "=(B" & i & "/6750)"
Worksheets("Consumo MIPv6").Cells(i, 7) = "=((9/125)+(6/250)+((C" & i & ")-
24)/6750)"
Next i
For j = 38 To 43 Step 1
Worksheets("Consumo MIPv6").Cells(i, 3) = "=B" & i & " + 15"
Worksheets("Consumo MIPv6").Cells(j, 4) = "=(B" & j & "/250)"
Worksheets("Consumo MIPv6").Cells(j, 7) = "=(C" & j & "/250)"
Next j
End If

Unload UserForm1
End Sub
```

### VI.4.2 UserForm 2

```
Private Sub CommandButton1_Click()
Unload UserForm2
End Sub

Private Sub CommandButton2_Click()
Call Módulo3.ShowChart


If Not (UserForm2.Option1 Or UserForm2.Option2 Or UserForm2.Option3 Or
UserForm2.Option4 Or UserForm2.Option5 Or UserForm2.Option6 Or
UserForm2.Option7) Then
```

```
MsgBox "¡¡Debe elejir una opción!!"
End If


End Sub
```

### VI.4.3 UserForm 3

```
Option Explicit

Private Sub Image1_Click()

End Sub

Private Sub UserForm_Initialize()
   Dim CurrentChart As Chart
   Dim Fname As String

   Set CurrentChart = ActiveSheet.ChartObjects(1).Chart

'  Chart guardado como GIF
   Fname = ThisWorkbook.Path & Application.PathSeparator & "temp.gif"
   CurrentChart.Export Filename:=Fname, FilterName:="GIF"
   ActiveSheet.ChartObjects(1).Delete

'  Mostrar como Chart
   Image1.Picture = LoadPicture(Fname)
   Application.ScreenUpdating = True
   Kill ThisWorkbook.Path & Application.PathSeparator & "temp.gif"
End Sub

Private Sub CloseButton_Click()
   Unload Me
```

### VI.4.4 Módulo 1

```
Sub ShowDialog()
   UserForm1.Show
End Sub

Sub ShowDialog2()
   UserForm2.Show
End Sub
```

### VI.4.5 Módulo 2

```
Option Explicit

Sub ToggleComments()
   If Application.DisplayCommentIndicator = xlCommentAndIndicator
Then
      Application.DisplayCommentIndicator = xlCommentIndicatorOnly
   Else
      Application.DisplayCommentIndicator = xlCommentAndIndicator
   End If
End Sub
```

### VI.4.6 Módulo 3

```
Sub ShowChart()
   Dim UserRow As Long
   UserRow = ActiveCell.Row

   If UserForm2.Option1 Then
   Call Gráfico1
   UserForm3.Caption = UserForm2.Option1.Caption
   UserForm3.Show
   End If

   If UserForm2.Option2 Then
   Call Gráfico2
   UserForm3.Caption = UserForm2.Option2.Caption
   UserForm3.Show
   End If

   If UserForm2.Option3 Then
   Call Gráfico3
   UserForm3.Caption = UserForm2.Option3.Caption
   UserForm3.Show
   End If

   If UserForm2.Option4 Then
   Call Gráfico4
   UserForm3.Caption = UserForm2.Option4.Caption
    UserForm3.Show
   End If

   If UserForm2.Option5 Then
   Call Gráfico5
   UserForm3.Caption = UserForm2.Option5.Caption
```

```
    UserForm3.Show
    End If

    If UserForm2.Option6 Then
    Call Gráfico6
    UserForm3.Caption = UserForm2.Option6.Caption
    UserForm3.Show
    End If

    If UserForm2.Option7 Then
    Call Gráfico7
    UserForm3.Caption = UserForm2.Option7.Caption
    UserForm3.Show
    End If
End Sub
```

```
Sub Gráfico1()

    ActiveSheet.Shapes.AddChart.Select
    ActiveChart.ChartType = xlColumnClustered
    ActiveChart.ApplyLayout (9)
    'Set Chart1 = Charts.Add
    ActiveChart.SeriesCollection.NewSeries
    ActiveChart.SeriesCollection(1).Name = "='Paquetes'"
    ActiveChart.SeriesCollection(1).Values = "='Consumo
MIPv6'!$D$21:$D$26"
    ActiveChart.SeriesCollection.NewSeries
    ActiveChart.SeriesCollection(2).Name = "='Tramas'"
    ActiveChart.SeriesCollection(2).Values = "='Consumo
MIPv6'!$G$21:$G$26"
    ActiveChart.SeriesCollection(2).XValues = "='Consumo
MIPv6'!$A$21:$A$26"
    ActiveChart.Axes(xlValue).AxisTitle.Select
    ActiveChart.Axes(xlValue, xlPrimary).AxisTitle.Text = "ms"
    ActiveChart.Axes(xlCategory).AxisTitle.Select
    Selection.Delete
    ActiveChart.ApplyLayout (9)
    ActiveChart.Axes(xlCategory).AxisTitle.Select
    Selection.Delete
    ActiveChart.ChartTitle.Select
```

```
    ActiveChart.ChartTitle.Text = "Duración de los mensajes MIPv6"

End Sub

Sub Gráfico2()

    ActiveSheet.Shapes.AddChart.Select
    ActiveChart.ChartType = xlColumnClustered
    ActiveChart.SeriesCollection.NewSeries
    ActiveChart.SeriesCollection(1).Name = "='"Consumo'""
    ActiveChart.SeriesCollection(1).Values          =          "='Consumo
MIPv6'!$G$59:$I$59"
    ActiveChart.SeriesCollection(1).XValues          =          "='Consumo
MIPv6'!$G$58:$I$58"
   ' ActiveChart.PlotArea.Select
    ActiveChart.ChartArea.Select
    ActiveChart.ApplyLayout (9)
    ActiveChart.Axes(xlValue).AxisTitle.Select
    ActiveChart.Axes(xlValue, xlPrimary).AxisTitle.Text = "mWh"
    ActiveChart.Axes(xlValue).Select
    Selection.TickLabels.NumberFormat = "0,00E+00"
     Selection.TickLabels.NumberFormat = "0,0E+00"
    ActiveChart.Axes(xlCategory).AxisTitle.Select
    Selection.Delete
    ActiveChart.ChartTitle.Select
    ActiveChart.ChartTitle.Text = "Consumo traspaso MIPv6"

End Sub
    Sub Gráfico3()

    ActiveSheet.Shapes.AddChart.Select
    ActiveChart.ChartType = xlColumnClustered
    ActiveChart.SeriesCollection.NewSeries
    ActiveChart.SeriesCollection(1).Name = "='"Sin Preám.""
    ActiveChart.SeriesCollection(1).Values          =          "='Consumo
MIPv6'!$D$38:$D$43"
    ActiveChart.SeriesCollection.NewSeries
    ActiveChart.SeriesCollection(2).Name = "='"Con Preám.""
    ActiveChart.SeriesCollection(2).Values          =          "='Consumo
MIPv6'!$G$38:$G$43"
    ActiveChart.SeriesCollection(1).XValues          =          "='Consumo
MIPv6'!$A$38:$A$43"
    ActiveChart.ApplyLayout (9)
    ActiveChart.ChartTitle.Select
    ActiveChart.ChartTitle.Text = "Duración tramas IEEE802.11"
    ActiveChart.Axes(xlCategory).AxisTitle.Select
    Selection.Delete
```

```
    ActiveChart.Axes(xlValue).AxisTitle.Select
    ActiveChart.Axes(xlValue, xlPrimary).AxisTitle.Text = "ms"
End Sub



Sub Gráfico4()
    ActiveSheet.Shapes.AddChart.Select
    ActiveChart.SetSourceData Source:=Range("'Consumo
MIPv6'!$A$50:$P$51")
    ActiveChart.ChartType = xlColumnClustered
    ActiveChart.Axes(xlValue).MajorGridlines.Select
    ActiveChart.ChartArea.Select
    ActiveChart.ApplyLayout (9)
    ActiveChart.SeriesCollection(1).Delete
    ActiveChart.SeriesCollection(1).Delete
    ActiveChart.SeriesCollection.NewSeries
    ActiveChart.SeriesCollection(1).Name = "='"T. espera'""
    ActiveChart.SeriesCollection(1).Values = "='Consumo
MIPv6'!$B$59,'Consumo MIPv6'!$B$61,'Consumo MIPv6'!$B$63,'Consumo
MIPv6'!$B$65,'Consumo MIPv6'!$B$67,'Consumo MIPv6'!$B$69,'Consumo
MIPv6'!$B$71,'Consumo MIPv6'!$B$73"
    ActiveChart.SeriesCollection.NewSeries
    ActiveChart.SeriesCollection(2).Name = "='"Msj. IEEE802.11""
    ActiveChart.SeriesCollection(2).Values = "='Consumo
MIPv6'!$B$60,'Consumo MIPv6'!$B$62,'Consumo MIPv6'!$B$64,'Consumo
MIPv6'!$B$66,'Consumo MIPv6'!$B$68,'Consumo MIPv6'!$B$70,'Consumo
MIPv6'!$B$72,'Consumo MIPv6'!$B$74"
```

```
ActiveChart.SeriesCollection(2).XValues = "='Consumo
MIPv6'!$A$60,'Consumo MIPv6'!$A$62,'Consumo MIPv6'!$A$64,'Consumo
MIPv6'!$A$66,'Consumo MIPv6'!$A$68,'Consumo MIPv6'!$A$70,'Consumo
MIPv6'!$A$72,'Consumo MIPv6'!$A$74"
    ActiveChart.Axes(xlValue).AxisTitle.Select
    ActiveChart.Axes(xlValue, xlPrimary).AxisTitle.Text = "ms"
    ActiveChart.Axes(xlValue).Select
    ActiveChart.Axes(xlValue).ScaleType = xlLogarithmic
    Selection.TickLabelPosition = xlLow
    ActiveChart.Axes(xlCategory).Select
    Selection.TickLabelPosition = xlLow
    ActiveChart.Axes(xlCategory).AxisTitle.Select
    Selection.Delete
    ActiveChart.ChartTitle.Select
    ActiveChart.ChartTitle.Text = "T.espera VS T. msj. MIPv6"
```

```
End Sub

Sub Gráfico5()

    ActiveSheet.Shapes.AddChart.Select
    ActiveChart.SetSourceData Source:=Range("'Consumo
Escaneo'!$A$14:$D$24")
    ActiveChart.ChartType = xlColumnClustered
    ActiveChart.Parent.Delete
    Range("D14").Select
    ActiveSheet.Shapes.AddChart.Select
    ActiveChart.SetSourceData Source:=Range("'Consumo
Escaneo'!$A$14:$C$18")
    ActiveChart.ChartType = xlColumnClustered
    ActiveChart.ApplyLayout (9)
    ActiveChart.Axes(xlCategory).AxisTitle.Select
    Selection.Delete
    ActiveChart.Axes(xlCategory).Select
    ActiveChart.ApplyLayout (9)
    ActiveChart.Axes(xlCategory, xlPrimary).AxisTitle.Text = "Canales
activos"
    ActiveChart.SeriesCollection(1).Delete
    ActiveChart.SeriesCollection.NewSeries
    ActiveChart.SeriesCollection(1).Name = "='1 AP'"
    ActiveChart.SeriesCollection(1).Values = "='Consumo
Escaneo'!$B$22:$D$22"
    ActiveChart.SeriesCollection.NewSeries
    ActiveChart.SeriesCollection(2).Name = "='2 AP'"
    ActiveChart.SeriesCollection(2).Values = "='Consumo
Escaneo'!$B$23:$D$23"
    ActiveChart.SeriesCollection.NewSeries
    ActiveChart.SeriesCollection(3).Name = "='3 AP'"
    ActiveChart.SeriesCollection(3).Values = "='Consumo
Escaneo'!$B$24:$D$24"
    ActiveChart.Axes(xlValue).AxisTitle.Select
    ActiveChart.Axes(xlValue, xlPrimary).AxisTitle.Text = "mWh"
    ActiveChart.Axes(xlValue).Select
    Selection.TickLabels.NumberFormat = "0,00E+00"
    ActiveChart.ApplyLayout (9)
    ActiveChart.ChartTitle.Select
    ActiveChart.ChartTitle.Text = "Consumo escaneo"
End Sub

Sub Gráfico6()

    ActiveSheet.Shapes.AddChart.Select
    ActiveChart.ChartType = xlColumnClustered
```

```
    ActiveChart.SeriesCollection.NewSeries
    ActiveChart.SeriesCollection(1).Name = "='""DAD"""
    ActiveChart.SeriesCollection(1).Values = "='Consumo MIPv6'!$G$59"
    ActiveChart.SeriesCollection.NewSeries
    ActiveChart.SeriesCollection(2).Name = "='""ODAD"""
    ActiveChart.SeriesCollection(2).Values = "='Consumo MIPv6'!$H$59"
    ActiveChart.SeriesCollection.NewSeries
    ActiveChart.SeriesCollection(3).Name = "='""no DAD"""
    ActiveChart.SeriesCollection(3).Values = "='Consumo MIPv6'!$I$59"
    ActiveChart.SeriesCollection(1).XValues = "='Consumo MIPv6'!$G$7"
    ActiveChart.SeriesCollection.NewSeries
    ActiveChart.SeriesCollection(4).Name = "='""IEEE802.11 MAC con
ESCANEO"""
    ActiveChart.SeriesCollection(4).Values = "='Consumo MIPv6'!$H$86"
    ActiveChart.SeriesCollection.NewSeries
    ActiveChart.SeriesCollection(5).Name = "='""IEEE802.11 MAC sin
ESCANEO"""
    ActiveChart.SeriesCollection(5).Values = "='Consumo MIPv6'!$H$83"
    ActiveChart.ChartArea.Select
    ActiveChart.PlotArea.Select
    ActiveChart.ApplyLayout (9)
    ActiveChart.Axes(xlCategory).AxisTitle.Select
    Selection.Delete
    ActiveChart.Axes(xlValue).AxisTitle.Select
    ActiveChart.Axes(xlValue, xlPrimary).AxisTitle.Text = "mWh"
    ActiveChart.ChartTitle.Select
    ActiveChart.ChartTitle.Text = "MIPv6 VS IEEE802.11 MAC"

End Sub

Sub Gráfico7()

    ActiveSheet.Shapes.AddChart.Select
    ActiveChart.SetSourceData Source:=Range("'Consumo
MIPv6'!$A$50:$P$51")
    ActiveChart.ChartType = xlColumnClustered
    ActiveChart.Axes(xlValue).MajorGridlines.Select
    ActiveChart.ChartArea.Select
    ActiveChart.ApplyLayout (9)
    ActiveChart.SeriesCollection(1).Delete
    ActiveChart.SeriesCollection(1).Delete
    ActiveChart.SeriesCollection.NewSeries
    ActiveChart.SeriesCollection(1).Name = "='""T. espera"""
    ActiveChart.SeriesCollection(1).Values = "='Consumo
MIPv6'!$B$84,'Consumo MIPv6'!$B$86,'Consumo MIPv6'!$B$88,'Consumo
MIPv6'!$B$90,'Consumo MIPv6'!$B$92"
    ActiveChart.SeriesCollection.NewSeries
```

```
    ActiveChart.SeriesCollection(2).Name = "="""Msj. IEEE802.11"""
    ActiveChart.SeriesCollection(2).Values = "='Consumo
MIPv6'!$B$83,'Consumo MIPv6'!$B$85,'Consumo MIPv6'!$B$87,'Consumo
MIPv6'!$B$89,'Consumo MIPv6'!$B$91,'Consumo MIPv6'!$B$93"
    ActiveChart.SeriesCollection(2).XValues = "='Consumo
MIPv6'!$A$83,'Consumo MIPv6'!$A$85,'Consumo MIPv6'!$A$87,'Consumo
MIPv6'!$A$89,'Consumo MIPv6'!$A$91,'Consumo MIPv6'!$A$93"
    ActiveChart.Axes(xlValue).AxisTitle.Select
    ActiveChart.Axes(xlValue, xlPrimary).AxisTitle.Text = "ms"
    ActiveChart.Axes(xlValue).Select
    ActiveChart.Axes(xlValue).ScaleType = xlLogarithmic
    Selection.TickLabelPosition = xlLow
    ActiveChart.Axes(xlCategory).Select
    Selection.TickLabelPosition = xlLow
    ActiveChart.Axes(xlCategory).AxisTitle.Select
    Selection.Delete
    ActiveChart.ChartTitle.Select
    ActiveChart.ChartTitle.Text = "T.espera VS T. msj. IEEE802.11"
End Sub
```

# ANEXO VII. Listado de Acrónimos en el texto

| ACRÓNIMO ORIGINAL | SIGNIFICADO EN EL IDIOMA ORIGINAL |
|---|---|
| AC | Access Category |
| AC_BE | Best Effort Access Category |
| AC_BK | Background Access Category |
| AC_VI | Video Access Category |
| AC_VO | Voice Access Category |
| ACK | Acknowledgement |
| AID | Association Identification |
| AIFSN | Arbitration Inter Frame Space Number |
| AP | Access Point |
| APSD | Automatic Power Save Delivery |
| ARP | Address Resolution Protocol |
| Back | Binding Acknowledgement |
| BU | Binding Update |
| CFP | Contention Free Period |
| CN | Correspondent Node |
| CoA | Care-of-Address |
| CoS | Class of Service |
| CP | Contention Period |
| CPRp | Consumo Probe Response |
| CPRq | Cosnumo Probe Request |
| DAD | Duplicate Address Detection |
| DBPSK | Differential Binary Phase Shift keying |
| DCF | Distributed Coordination Function |
| DSCP | Diffserv code point |
| DTIM | Delivery Traffic Indication Message |
| ECWmin | Minimum Contention Window Value |
| EDCA | Enhanced Distributed Channel Access |
| EOSP | End Of Service Period |
| ESSID | Extended Service Set Identifier |
| FCS | Frame Check Sequence |
| Flag | sub campo |
| HA | Home Agent |
| HCCA | HCF Controlled Channel Access |
| HCF | Hybrid Coordination Function |
| Idle | tiempo de inactividad |
| LI | Listen Interval |
| LLC | Logical Link Control |
| MAC | Media Access Control |
| MaxCT | Maximum Channel Time |

| | |
|---|---|
| MD | More Data |
| MinCT | Minimum Channel Time |
| MIPv6 | Mobile IPv6 |
| MLD | Multicast Listener Discovery |
| MN | Mobile Node |
| MSDU | Mac Service Data Unit |
| NbAdv | Neighbour Advertisement |
| NbSol | Neighbor Solicitation |
| ODAD | Optimistic Duplicate Address Detection |
| payload | campo de datos |
| PbReq | Probe Request |
| PbRsp | Probe Response |
| PBS | Priority Based Scheduling |
| PE | Parameter Element |
| PHB | Per-Hop behavior |
| PLCP | Physical Layer Coverage Protocol |
| PLCP H. | Physical Layer Coverage Protocol Header |
| PPDU | Presentation Protocol Data Unit |
| PSDU | Physical layer Data Unit |
| PSM | Power Save Mode |
| PS-Poll | Power Save Poll |
| QAP | Access Point con funcionalidades WMM |
| QoS | Quality of service |
| RtAdv | Router Advertisement |
| RtSol | Router Solicitation |
| RTT | Round Trip Time |
| Rx | Recepción |
| sAPSD | scheduled Automatic Power Save Delivery |
| script | Conjunto de instrucciones para la automatización de tareas |
| STA | Station |
| TCP | Transmission Control Protocol |
| TIM | Traffic Indication Map |
| TU | Time Unit |
| Tx | Transmisión |
| TXOP | Transmision Oportunity |
| uAPSD | unscheduled Automatic Power Save Delivery |
| UDP | User Datagram Protocol |
| UMTS | Universal Mobile Telecommunications System |
| VBA | Visual Basic Application |
| VoIP | Voz sobre IP |
| WEP | Wired equivalent Privacy |
| WMM | WI-FI Multimedia |
| WMM-PS | WI-FI Multimedia Power Save |
| WPA | WI-FI Protected Access |