Facultad de Ingeniería
Instituto de Ingeniería Eléctrica

WASA

WiFi Aplicado a Sitios Alejados

Proyecto de fin de carrera de Ingeniería Eléctrica

Diego Garagorry, Enrique Lev, Fernando Viera

Tutor: Dr. Pablo Belzarena

Montevideo, Uruguay
Mayo 2011

Agradecimientos

Queremos agradecer a nuestro tutor Pablo Belzarena por habernos confiado este proyecto, aconsejarnos durante el transcurso del mismo y siempre habernos guiado en los momentos en que se presentaron dificultades.

También queremos agradecer especialmente a todos los integrantes del Plan Ceibal que han colaborado con nosotros de forma rápida y eficaz siempre que lo necesitamos. Sin duda ha sido valiosísimo el aporte que han dado en todas las etapas del proyecto. Entre ellos nos gustaría reconocer especialmente a Martín Irazoqui, Franco Miceli, Claudina Rattaro y Cecilia Abalde.

Finalmente nuestro agradecimiento especial a todos quienes han colaborado de una forma u otra a lo largo del desarrollo de este proyecto y por supuesto a nuestras familias y amigos por apoyar e impulsar nuestro trabajo y sentirlo como propio.

Resumen

En los países en vías de desarrollo existen zonas, principalmente rurales, históricamente relegadas a ciertos servicios. Dada la trascendencia que ha tomado Internet como generador de oportunidades, es importante que el acceso no esté restringido por razones de costos o rentabilidad a ciertos sectores de la sociedad.

En Uruguay, a través del Plan Ceibal, se ha entregado una computadora portátil a todos los niños de educación primaria, y si bien se ha extendido la conectividad a la mayor parte de las escuelas públicas del país, al momento de comenzar el proyecto existían escuelas rurales que presentaban la problemática anteriormente planteada.

El presente trabajo estudia una alternativa de bajo costo para extender redes de datos a sitios alejados utilizando WiFi. A través de un enlace punto a punto de 30 kilómetros se conecta un sitio sin acceso a Internet a otro con conexión. Previo a establecer el enlace se realizó un estudio teórico de los requisitos a cumplir para lograr el objetivo, lo cual puede servir para futuros trabajos utilizando esta tecnología.

Se presenta además una solución para brindar calidad de servicio en el enlace, aspecto fundamental para impulsar aplicaciones como videoconferencias, educación a distancia (aulas virtuales) o simplemente comunicaciones de voz. Mediante el uso de tarjetas de red que cumplen con el estándar 802.11e y de disciplinas de colas en capa 3 se logra tratar diferenciadamente distintos flujos de tráfico.

El proyecto abarcó un estudio teórico de los estándares 802.11, 802.11e, radio enlaces de larga distancia y disciplinas de colas en capa 3 los cuales fundaron las bases para los pasos siguientes. Mediante pruebas de laboratorio se estudió el impacto de 802.11e para finalmente establecer un enlace, entre la ciudad de Florida y la localidad de San Gabriel, con las características mencionadas.

Índice general

Índice de figuras

Parte I

Introducción y Estudios Preliminares

Capítulo 1

Introducción

1.1. Contexto

Las comunicaciones han tomado en los últimos años un papel destacado generando igualdad de oportunidades a toda la sociedad. En especial Internet ha acercado a millones de personas información en las más diversas áreas, así como contenido académico. Ha brindado la posibilidad de informarse de lo que ocurre en el mundo entero instantáneamente, permitiendo la conexión y acceso a servicios que utilizan este medio como plataforma.

Indudablemente el impacto ha sido más notorio en aquellos lugares donde no solo ha significado una mejora sustancial en el acceso, sino que ha permitido llegar a servicios que antes no eran accesibles. En ciudades o pueblos rurales, el acceso a Internet y en particular las posibilidades que permiten servicios como la educación a distancia o la telemedicina, ha impactado fuertemente el día a día.

Sin embargo, dada la lejanía de estos puntos a zonas con gran densidad de población, muchas veces no forman parte del tendido utilizado por los operadores para proveer acceso a Internet por medio de fibra óptica o par de cobre. Alternativas como enlaces por microondas, redes celulares o enlaces satelitales resultan costosos o no cumplen los requisitos necesarios en cuanto a capacidad de transmisión.

Ha surgido por lo tanto, principalmente en países subdesarrollados, la necesidad de enfrentar esta problemática buscando alternativas de bajo costo. Países como Perú o India han presentado experiencias de enlaces de varios kilómetros utilizando 802.11, más conocido como WiFi[1], como protocolo base obteniendo resultados satisfactorios.

[1]WiFi es una marca de la Wi-Fi Alliance, la organización comercial que adopta, prueba y certifica que los equipos cumplen los estándares 802.11 relacionados a redes inalámbricas de área local.

Estas zonas aisladas presentan a su vez ciertas particularidades propias de la lejanía a zonas pobladas como son, por ejemplo, limitaciones en el acceso a determinados cursos o material en el ámbito educacional. Esto ha aumentado la importancia de poder establecer conexiones hacia estos centros aislados, brindando buenas tasas de transferencia, que permitan extender ciertos servicios desde los centros poblados más cercanos. Sin embargo, para soluciones como videoconferencias, aulas virtuales compartidas o telemedicina no es suficiente con enlaces de tasa de transmisión aceptables, sino que es necesario que estos presenten algún tipo de calidad de servicio para priorizar distintos tipos de tráfico o aplicaciones.

1.2. Objetivos del proyecto

En el contexto planteado, surge la inquietud por estudiar enlaces de larga distancia con control sobre la calidad de servicio, que permitan a costos accesibles extender la cobertura de las redes existentes a estos sitios que presentan las dificultades detalladas.

Este trabajo presenta el estudio del estándar 802.11, con el objetivo de brindar conexión a sitios ubicados a una distancia de hasta 30 km, brindando una tasa de transferencia de al menos 3 Mbps y con calidad de servicio basándose en el estándar 802.11e.

1.3. El Plan Ceibal

En el marco anteriormente presentado, y con el objetivo de llevar la computación e Internet a todos los sectores de la sociedad, se lanzó en Uruguay el Plan Ceibal. Este plan ha entregado más de 420.000 computadoras portátiles a estudiantes de educación primaria y secundaria del país. A su vez, ha conectado la mayor parte de estos centros permitiendo principalmente el acceso a Internet. Sin embargo aún existen escuelas, en su mayoría rurales, que dada su lejanía hacia centros poblados no cuentan, al momento, con conexión a Internet o ésta se realiza a través de enlaces satelitales o celulares. En ambos casos los altos costos o bajas tasas de transferencia han generado la necesidad de nuevas soluciones, al igual que ha sucedido en otros países como se mencionó previamente.

Dado el interés del Plan Ceibal en el uso de la solución planteada para atacar estos problemas, se realizó el proyecto trabajando en conjunto, de forma de buscar una solución a una situación real, la cual se presentará en los próximos capítulos. Como se detallará a continuación, el Plan Ceibal aportó tanto equipos como la logística para realizar la instalación, así como el soporte necesario durante las distintas etapas del proyecto.

1.4. Estructura de la documentación

Este documento se encuentra dividido en cuatro secciones pricipales. Seguido a esta introducción a lo que fue el proyecto, se presenta los aspectos considerados para la elección del estándar 802.11, así como el proceso de selección del hardware y software utilizado. A continuación se detalla los estudios realizados previo a la implementación del enlace, describiendo tanto las herramientas utilizadas como los resultados obtenidos. Estas pruebas presentan las características de la solución que se desarrollarán en los siguientes capítulos. La tercera sección describe los estudios realizados para el establecimiento del enlace y el análisis del mismo una vez implementado. La sección cuatro presenta las conclusiones del proyecto e introduce posibles líneas de trabajo a futuro. Se incluye finalmente siete anexos con detalles técnicos donde se extienden temas de interés como el estándar 802.11, disciplinas de colas y guías para la instalación y configuración de OpenWRT.

Capítulo 2

Elección del estándar 802.11

En los últimos años se produjo un crecimiento espectacular en el desarrollo de las comunicaciones móviles. Particularmente, las redes de área local inalámbricas (WLANs) han crecido exponencialmente. Este crecimiento permitió una considerable baja de precios de los dispositivos.

La elección del estándar 802.11[5], cuyo estudio más detallado se encuentra en el apéndice A, se basó principalmente en tres pilares:

1. Uso de bandas de frecuencias no licenciadas.

2. Bajo costo de equipos en comparación con otras tecnologías.

3. Existencia de trabajos previos en enlaces de larga distancia utilizando 802.11 [6] [7].

WiFi fue diseñado para redes inalámbricas de área local [8] [9]. Sin embargo, realizando un estudio del estándar, se puede ver que modificando los valores de ciertos parámetros de la subcapa MAC, no existirían impedimentos para establecer un enlace de larga distancia utilizando esta tecnología.

2.1. Capa física

A nivel de capa física (PHY) los mayores inconvenientes se deben a las problemáticas propias de los enlaces inalámbricos, como pérdidas por propagación, pérdidas en conectores e interferencias y no a restricciones propias de 802.11. Un análisis más detallado sobre estos aspectos y el cálculo del enlace se realiza en el capítulo 7. Con respecto a lo que se especifica en el estándar, las limitaciones están dadas por los rangos de frecuencia soportada y por las velocidades de transmisión disponibles para cada rango, de acuerdo a la modulación y codificación utilizada [10].

2.2. Subcapa MAC

A partir de los modos de funcionamiento que propone el estándar en subcapa MAC, se puede ver los siguientes inconvenientes para la implementación en larga distancia:

- *ACKTimeout*: luego de enviar una trama, la fuente aguarda la confirmación de recepción ACK un tiempo conocido como ACKTimeout. Si transcurrido el ACKTimeout no se recibió la confirmación de la trama, la misma se da por perdida y se retransmite. Al haber tanta distancia entre las partes, el tiempo de respuesta aumenta. Esto lleva a que si no se modifica este parámetro de forma que el ACKTimeout sea mayor al tiempo de propagación de ida y vuelta más el SIFS (ver A.4.1), se den por perdidos paquetes que se encuentran en viaje y sean retransmitidos innecesariamente [9]. El enlace entonces no funciona o lo hace con un rendimiento muy inferior al esperado.

- *SlotTime* o "duración de slot": depende de la capa física PHY y determina tiempos definidos en la subcapa MAC (más detalles ver sección A.4.1 del apéndice). En el estándar se prevé que la estación receptora reciba la trama dentro del mismo SlotTime que fue enviado [9]. Asimismo, de acuerdo a lo que surge de [8], este parámetro influye directamente en la probabilidad de que se generen colisiones en el acceso al medio.

- Detección virtual del estado del canal: para evitar el problema del nodo oculto, se utilizan mensajes RTS/CTS (ver A.4.2) donde se indica el tiempo en que el canal será utilizado por una estación. Al variar de forma considerable los tiempos con la distancia, y dado que esto no se contempla al determinar este parámetro, este método no cumpliría su función. Como contrapartida, al tratarse de un enlace PtP (punto a punto) el problema no tendría mayores consecuencias.

2.3. 802.11e

En su versión original 802.11 no preveía garantías de calidad de servicio (QoS) tanto en ancho de banda, pérdida de paquetes o retardo. En 2005 fue aprobado el estándar 802.11e [11] que da soporte de QoS a nivel MAC.

Si bien hay experiencias previas simulando enlaces de larga distancia y utilizando 802.11e [8] y existen enlaces de varias decenas de kilómetros utilizando 802.11 [9] [12] [10] [7], el fin de este proyecto es evaluar IEEE 802.11e EDCA en un enlace PtP de aproximadamente 30 kilómetros.

2.3.1. EDCA: Enhanced Distributed Channel Access

EDCA utiliza priorización de tráfico en base a cuatro categorías de acceso, donde el comportamiento de cada una de estas categorías es similar al utilizado en el modo DCF de 802.11 (ver

A.4.2).

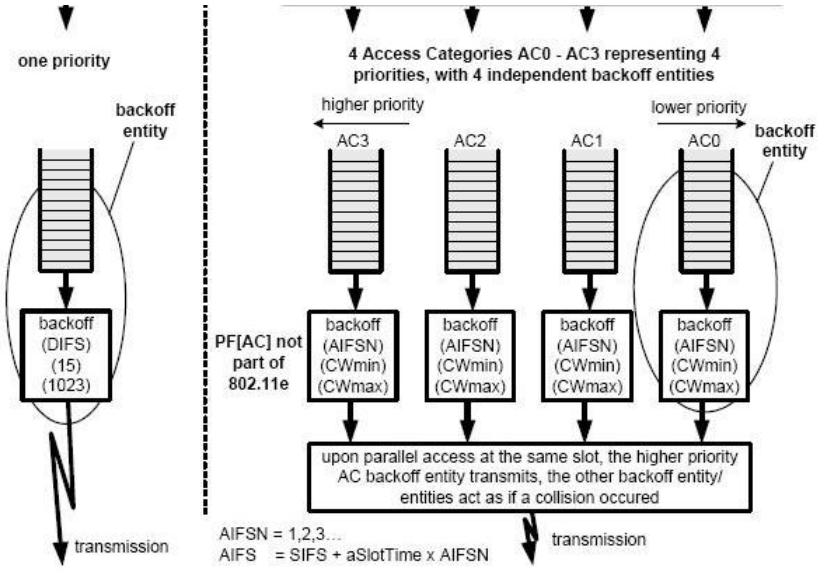

Figura 2.1: Comparación del modo de funcionamiento en 802.11 y 802.11e [1]

Cada categoría de acceso (AC) representa un nivel de prioridad diferente ya que dispone de una cola de transmisión en la subcapa MAC, con sus propios parámetros de acceso [13].

- $CWmin_i$, valor mínimo de la ventana de contienda.
- $CWmax_i$, valor máximo de la ventana de contienda.
- $TXOP_i$ (Transmission Opportunity), especifica la duración máxima que cada QSTA (estación con soporte de calidad de servicio) puede transmitir al medio.
- $AIFS_i$ (Arbitrary InterFrame Space), especifica el intervalo de tiempo entre que el medio de acceso se coloca en estado idle y el comienzo de una negociación del canal.

Se tiene entonces que cada AC se comporta como una entidad independiente de contienda. Cuando los datos llegan al punto de acceso, la subcapa MAC de 802.11e los clasifica en alguna de las AC previo a enviarlos al medio.

Las categorías de acceso y los valores por defecto de los parámetros se presentan en la siguiente tabla, desde la menos prioritaria AC_BK a la que tiene mayor prioridad AC_VO:

The default value table:

AC TYPE	Min Contention Window (2x-1; x can be 0-10)	Max Contention Window (2x-1; x can be 0-10)	Fixed Slot Time (0-15)	Transmit Opportunity (0-65535 µS)
AC_BK	4	10	7	0
AC_BE	4	6	3	0
AC_VI	3	4	1	3008 (6016 when 11b)
AC_VO (3)	2	3	1	1554 (3264 when 11b)

Figura 2.2: Parámetros por defecto en 802.11e [2]

Previo a enviar las tramas, existe un mecanismo interno de resolución de colisiones donde si 2 AC intentan acceder al medio en el mismo instante, será la que tiene mayor prioridad la que envíe el paquete y la otra lo tomará como una colisión real [8].

En capítulos posteriores se presentarán los resultados obtenidos al trabajar con el estándar 802.11e, tanto en ensayos de laboratorio como en el enlace de larga distancia.

Capítulo 3

Selección de Hardware y Software

3.1. Hardware

En el presente proyecto la elección del hardware a utilizar fue una etapa fundamental. Primero que nada porque no se disponía de equipamiento para realizar ensayos, por lo que se trató de una investigación más que de una selección. Asimismo el vínculo con Plan Ceibal marcaba la elección a equipos con los cuales ya trabajaran sus técnicos. Finalmente se precisaba que existiera compatibilidad entre todo el equipamiento seleccionado, tarea en la cual hubo que invertir muchas horas de estudio ya que, como se mencionó previamente, no se disponía de equipos para realizar pruebas. A continuación se detalla los criterios tomados para la selección del hardware.

- Equipo de bajo consumo
- Tamaño reducido
- Soporte a condiciones meteorológicas adversas
- Bajo costo económico
- Interfaz inalámbrica
- Interfaz cableada
- Capacidad de configuración de parámetros de bajo nivel
- Programación del dispositivo

3.1.1. Single Board Computers

La primer búsqueda se orientó a routers o Single Board Computers (SBC), que permitieran su posterior configuración de una cantidad razonable de funcionalidades. Un SBC es un dispositivo construido en una sola tarjeta de circuito impreso para un objetivo específico como router o bridge inalámbrico (no trae interfaces para teclado o monitor). Esta selección se basó fundamentalmente

en equipamiento con el que trabajara Ceibal por motivos de stock y reposición, dado que cumplían con los requisitos detallados. Dentro de las opciones disponibles se decidió por Mikrotik dada la experiencia que poseía Ceibal en el uso de estos equipos, así como la utilización de los mismos en proyectos similares a WASA. Los modelos evaluados fueron los siguientes:

- RouterBoard 433 [14]
- RouterBoard 433AH [14]

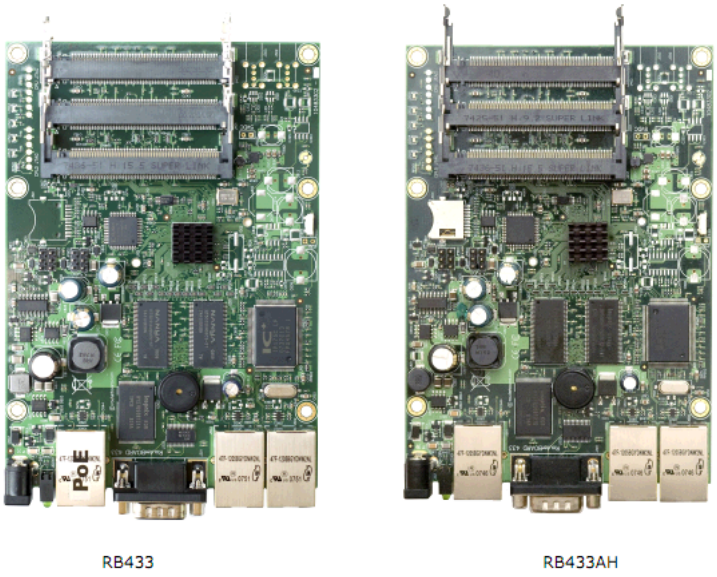

Figura 3.1: RouterBoards 433 y 433AH

Ambos modelos cuentan con 3 slots miniPCI tipo IIIA/IIIB y 3 puertos ethernet. Asimismo pueden ser alimentados mediante Power Over Ethernet (POE), aspecto fundamental si se quieren utilizar estos equipos fijados en una torre. Las diferencias significativas entre uno y otro son:

- RouterBOARD 433
  - CPU Atheros de 300Mhz
  - 64MB RAM
- RouterBOARD 433AH
  - CPU Atheros de 680Mhz
  - 128MB RAM
  - 1 microSD slot

Como último punto, es importante destacar la posibilidad de cargarles otro sistema operativo. Debido a que las diferencias relevantes entre uno y otro eran simplemente de velocidad del procesador y memoria RAM, se optó por el modelo Mikrotik 433AH.

3.1.2. Tarjetas de red

Una vez seleccionada la placa madre o router (Mikrotik 433AH), se orientó la búsqueda a tarjetas de red inalámbricas miniPCI que cumplieran 3 especificaciones básicas:

1. Sea soportada por el hardware del router.
2. Sea compatible con los controladores que proporciona el proyecto MadWifi.
3. Cumpliera con el estándar 802.11e.

Fue de particular interés el driver MadWifi ya que permite alterar parámetros a bajo nivel, incluyendo los relacionados con 802.11e, y fue desarrollado para trabajar con chipsets Atheros, incluidos en las tarjetas evaluadas.

Plan Ceibal no disponía de ninguna tarjeta que cumpliera con todas los requisitos, específicamente con el soporte del estándar 802.11e. Se realizó entonces una investigación de distintas tarjetas que cumplieran a la vez con todo lo detallado anteriormente.

Finalmente se optó por Wistron DCMA-82 [15].

Figura 3.2: Tarjeta wireless

Aunque no se mencionó específicamente, todas las tarjetas debían poder trabajar tanto en la banda de 2.4 Ghz como en la de 5.8 Ghz, dado que al momento de realizar la selección aún no estaba definida la frecuencia que se iba a utilizar.

3.1.2.1. Parámetros de las tarjetas

Si bien es fundamental asegurar la interoperabilidad entre la tarjeta de red seleccionada y el hardware y software a utilizar, también es importante tener presente los parámetros que dan los fabricantes de las mismas. Los datos más importantes que se tuvieron en cuenta son:

- Frecuencia de trabajo
  Corresponde a las bandas de frecuencias en las que puede trabajar la tarjeta de red.
- Potencia de salida
  Es la potencia emitida por la tarjeta cuando está transmitiendo. Depende de la modulación y codificación utilizada.

- Sensibilidad
  Es el nivel de potencia mínima a recibir. Cuanto menor sea la potencia recibida, menor será el ancho de banda que se puede alcanzar.

La sensibilidad de las tarjetas de red es un dato fundamental para establecer enlaces inalámbricos de larga distancia. Si bien los fabricantes brindan este dato, no es suficiente con lograr cierta potencia de la señal en recepción, sino que además es necesario lograr una buena relación señal a ruido. Este dato no es brindado por la mayoría de los fabricantes en sus especificaciones. Luego de realizadas ciertas consultas, se tomó como buen criterio, que la relación señal a ruido en recepción debía ser mayor a 15 dB. Es un valor que toman muchos fabricantes y a su vez deja un buen margen de seguridad.

3.1.3. Antenas y conectores

Tanto en las antenas como en los conectores a utilizar no se tenía demasiada opción ya que se dependía del stock de Ceibal. Como requisito evidente se debía tener antenas direccionales. Se utilizaron antenas de grilla direccionales Pacific Wireles GD58-29, de 29 dBi. Todo lo referente al cálculo del enlace se abarca en el capítulo 7.

3.2. Software

Una vez que el hardware fue identificado, se pasó a la selección del software. Las opciones evaluadas fueron las siguientes:

- RouterOS: sistema propietario, viene por defecto en los routers Mikrotik.
- OpenWRT: GNU/Linux, basado en un firmware embebido, orientado a routers y gateways residenciales.

3.2.1. RouterOS

Dado que es el sistema operativo instalado por defecto en los Mikrotik, se comenzó trabajando con el mismo.

Como ventajas, se puede destacar la posibilidad de utilizar sobre Windows una interfaz gráfica amigable (WinBox), ofreciendo todas sus funcionalidades desde la propia interfaz. La configuración del mismo es bastante intuitiva y guiada. A su vez, RouterOS cuenta con herramientas en tiempo real, tales como:

- Medición de ancho de banda
- Cálculo de relación señal a ruido

- Medición de Delay

Asimismo existe la posibilidad de acceder al router mediante SSH o telnet, sumado a la herramienta WinBox.

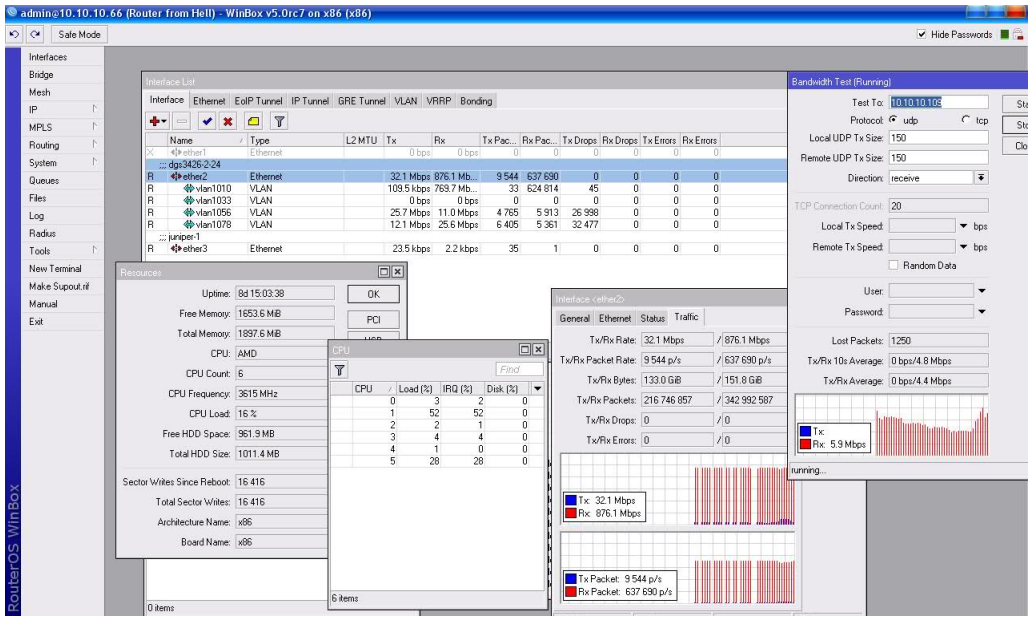

Figura 3.3: RouterOS WinBox

Por otro lado, al poseer controladores privativos para los dispositivos de red, surgió la imposibilidad de modificar parámetros de bajo nivel, tales como el tamaño de la ventana de contención o el tiempo entre tramas (AIFS).

Al momento de probar 802.11e y no poder modificar sus parámetros fue que se descartó la opción de utilizar RouterOS.

3.2.2. OpenWRT

Es un software de código abierto. En el anexo B se encuentra un manual de instalación del mismo sobre el hardware RouterBoard 433AH.

OpenWRT es un derivado directo de la distribución Debian que permite la implementación de un sistema embebido y personalizado para dispositivos inalámbricos (routers). Asimismo permite la posterior instalación de una gran cantidad de paquetes y aplicaciones para ajustar el sistema operativo a las necesidades.

El proyecto OpenWRT, cuenta con una página para soporte [16] y a su vez contiene su propio canal IRC.

Como interfaz con el usuario, utiliza básicamente un sistema de línea de comando, pero también existe la posibilidad de instalar un paquete que contiene una interfaz web:

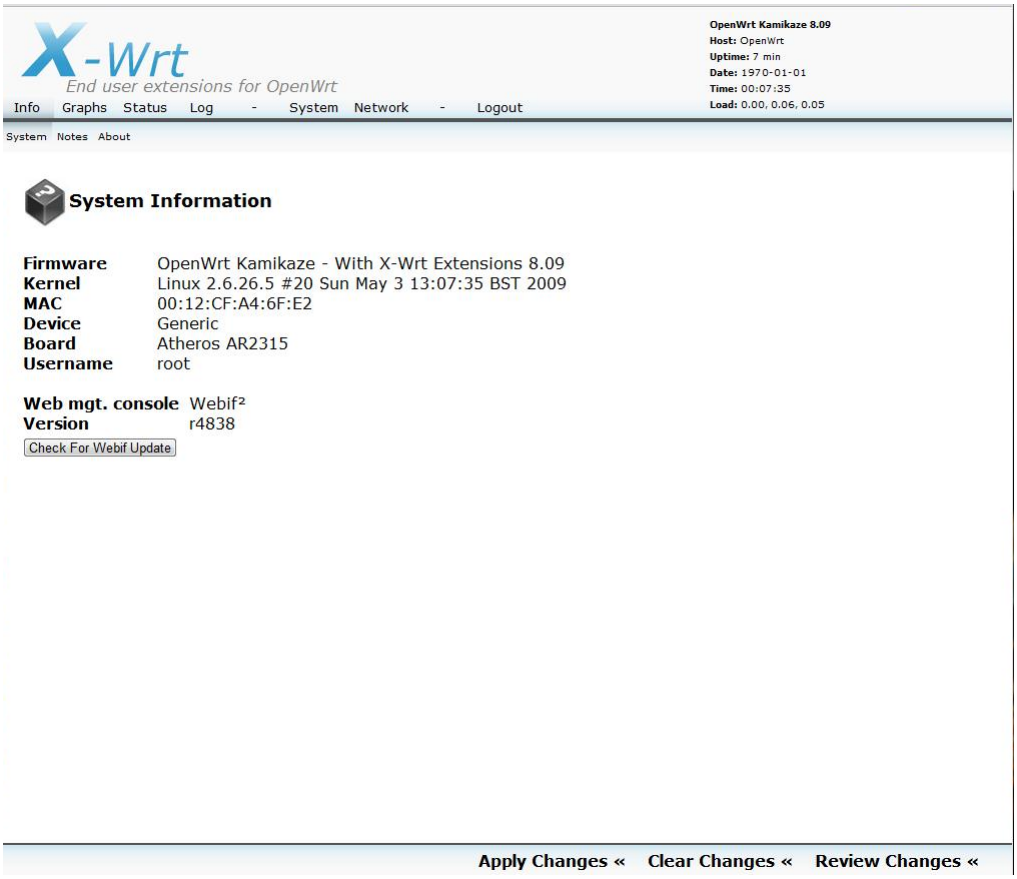

Figura 3.4: OpenWRT web interface

Dentro de las funcionalidades se puede destacar:

- Firewall
- Port Forwarding
- Traffic Shaping
- Monitoreo en tiempo real
- DHCP
- DNS dinámico
- Posibilidad de usar como repetidor, access point o puente
- Redes Mesh
- Más de 2000 paquetes para distintas aplicaciones

Parte II

Pruebas de Laboratorio

Capítulo 4

Descripción de herramientas a utilizar

El objetivo del presente proyecto fue establecer y evaluar un enlace inalámbrico de larga distancia utilizando WiFi. Para ello se necesitaron herramientas informáticas que permitieran medir las prestaciones del enlace establecido. Las mismas se utilizaron tanto en las pruebas de laboratorio como en larga distancia.

Los parámetros de red fundamentales a medir en el enlace fueron: *throughput*, *delay o RTT* y *jitter*. En este capítulo se describirán las herramientas utilizadas para evaluar el enlace, así como otras que fueron muy útiles para la realización de las pruebas. No se pretende realizar un manual de cada herramienta utilizada sino que se brinda una breve descripción de las mismas. Existen otras herramientas disponibles, por más referencias consultar apéndice G.

4.1. Throughput, delay o RTT y jitter

Throughput es la información que efectivamente se puede cursar por el canal por unidad de tiempo. Es quizás el parámetro más importante, ya que nos permite evaluar qué tipo de aplicaciones y con qué carga podremos utilizar en el enlace. Se mide en Mbit/s (Mbps).

Delay es el tiempo que tarda un paquete en ir desde el origen hasta el destino, en tanto que el *RTT* (round-trip time) es el tiempo de ida y vuelta. Estos parámetros son fundamentales para aplicaciones en tiempo real, tales como tráfico de voz y videollamadas, aunque no tan importantes para otros tipos de tráfico como son navegación web, correo electrónico, etc. Se trabaja fundamentalmente con el *RTT*, el cual, al tratarse de un tiempo, se mide en ms (milisegundos).

Jitter es la variación del *delay* e interesa básicamente el *jitter* promedio y el máximo.

4.2. Herramientas de red

A continuación se describe las aplicaciones utilizadas para configurar dispositivos y evaluar el enlace. Las mismas sirven para evaluar el throughput, RTT o jitter.

ping: es quizás de las primeras herramientas que se aprenden cuando se empieza a trabajar en redes de datos; utiliza el protocolo ICMP. Se utilizó para medir el RTT entre los equipos del enlace. El *ping* permite verificar si hay conectividad con un dispositivo y si bien su utilización es muy sencilla, existen muchas opciones y ajustes que se pueden realizar al utilizarlo. Se destaca que se puede cambiar el TOS (*Type of Service*) y el tamaño del paquete enviado. Más información se puede obtener del manual que hay en cualquier distribución Linux (*man ping*).

iperf: [17] es una herramienta para inyectar tráfico UDP o TCP y medir tanto el throughput obtenido como el jitter. Funciona como cliente-servidor, por lo tanto, se debe levantar un servidor para recibir tráfico y un cliente para generarlo. Es una aplicación muy sencilla que se utilizó en modo comando si bien existe una interfaz gráfica, jperf. Entre muchas opciones se destaca que con *iperf* se puede elegir el tiempo de prueba, cambiar el puerto destino y ajustar el flujo UDP generado. Más opciones sobre esta herramienta se pueden obtener en el manual que trae la aplicación cuando se instala (*man iperf*).

iptables: Es una herramienta muy completa para manipular paquetes de red, sobre la cual existe vasta documentación. Fue necesaria para el marcado de bits en paquetes IP. Al igual que las mencionadas anteriormente también se puede consultar el manual (*man iptables*).

wireshark: es un analizador de protocolos utilizado en redes de datos. Tiene interfaz gráfica y variedad de filtros configurables para una visualización más ajustada. Herramienta fundamental para corroborar el correcto marcado de paquetes. Funciona en la mayoría de los sistemas operativos. Por más información consultar [18].

Wifislax: es un sistema operativo Linux que se distribuye en un live CD orientado al análisis de redes inalámbricas. Se utilizó la tarjeta de red en modo monitor para analizar los paquetes en el enlace de laboratorio y verificar el correcto marcado de los mismos a nivel de capa IP y el mapeo a nivel de capa de enlace. La versión utilizada fue la 3.1 [19].

route: herramienta para configuración de rutas en sistemas Linux. Por más referencias (*man route*).

Comandos básicos: tanto para la configuración de los PC, de los routers o la gestión de la red es necesario el manejo de comandos como *ifconfig*, *iwconfig*, *traceroute*, *ssh*, etc.

4.3. Otras herramientas utilizadas

La mayor parte de los ensayos realizados sobre el enlace inalámbrico se realizaron automáticamente. Se describen algunas herramientas necesarias.

Servidor NTP: servidor de horario instalado en un PC que permite la sincronización de los dispositivos. Es fundamental la sincronización horaria porque la herramienta *iperf*, al ser cliente servidor, necesita ejecutarse simultáneamente en ambos PC.

Crontab (Cron Table): es un archivo que contiene las entradas *cron* del sistema. Cada usuario puede tener su propio *crontab*, el cual le permite ejecutar procesos o scripts en horarios y días programados.

Scripting en Shell: fue necesario para la automatización de las pruebas.

También se utilizaron herramientas para compresión de archivos y copia a través de la red como son *tar* y *scp* respectivamente.

Capítulo 5

Escenario propuesto

5.1. Arquitectura

La arquitectura del enlace para las pruebas de laboratorio se muestra en la siguiente figura:

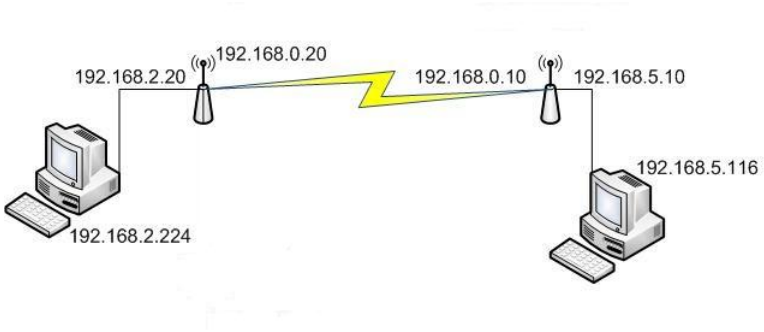

Figura 5.1: Arquitectura de pruebas de laboratorio

Las pruebas fueron realizadas en ambos sentidos, sin necesidad de diferenciarlos, dada la simetría en el esquema propuesto.

5.2. Descripción general de las pruebas

La forma que se adoptó para denominar los flujos fue prioritario, si este tenía como destino el puerto 5001, y no prioritario, si este tenía por destino el puerto 5005. En los casos donde no había calidad de servicio habilitada sería simplemente una forma de denominarlos. En las pruebas donde se configuró algún parámetro para ver diferencias entre flujo prioritario y no prioritario, debería ser siempre el flujo al puerto 5001 el que obtuviera mayores recursos y por lo tanto mayor throughput o menor delay. Se eligieron estos puertos arbitrariamente, el análisis es igualmente válido si se priorizan otros puertos o protocolos.

Se estudió qué sucede con el throughput y el delay a medida que se cambian las prioridades y se implementa calidad de servicio. Como TCP se adapta a la capacidad del canal, se trabajó con flujos UDP, ya que se pueden inyectar a tasas constantes y la contienda por el canal es entre flujos iguales.

Para medir el throughput se utilizó *iperf* y para medir el RTT se utilizaron paquetes de 64 bytes (más los encabezados IP y de capa de enlace).

5.2.1. Condiciones de trabajo

Cuando el tráfico generado es menor a la capacidad del enlace, el mismo es cursado por el canal y no se pueden realizar conclusiones de lo observado. Por lo tanto, se trabajó con flujos saturando el enlace de forma de ver el resultado de las distintas priorizaciones realizadas.

5.3. Marcado y mapeo

En el esquema propuesto, el marcado de paquetes se realizó en los PC mediante la herramienta *iptables* (ver capítulo D del apéndice). Luego en los routers, paquetes con destino el puerto 5001 se mapearon en la clase AC_VO de 802.11e y paquetes con destino el puerto 5005 en la clase AC_BK.

Cuando se utilizaron disciplinas de colas, la clasificación se realizó a partir del encabezado IP (ver capítulo D del apéndice).

Para verificar el correcto marcado de los paquetes entrantes y salientes, se utilizó *wireshark* en cada uno de los extremos del enlace. Asimismo se analizaron los paquetes en el aire configurando un tercer equipo con una tarjeta de red inalámbrica en modo monitor y utilizando *Wifislax*.

5.4. Validez de los resultados

En los enlaces inalámbricos, el medio (aire) es compartido por todo equipo que quiera transmitir. Los resultados obtenidos al evaluar el enlace pueden ser muy variables dependiendo las condiciones puntuales en cada ensayo. Al no contar con un ambiente libre de interferencia para la realización de las pruebas, se realizaron las mismas en reiteradas oportunidades, de forma de verificar los resultados y validar las conclusiones obtenidas. En las pruebas realizadas en el capítulo 6, incluso se verificaron algunos ensayos tanto en la banda de 2.4 GHz como en la de 5.8 Ghz. Igualmente, más allá de los resultados cuantitativos de las pruebas, se pretendió realizar un análisis cualitativo de las mismas y comparar lo que se obtuvo en distintas configuraciones.

Capítulo 6

Impacto de 802.11e

Como se describió en secciones anteriores, uno de los objetivos del presente proyecto fue el estudio del impacto de utilizar el estándar 802.11e o WiFi Multimedia[1] (*WMM*) en el enlace inalámbrico. En el presente capítulo se describirán las pruebas realizadas en un ambiente de laboratorio, de forma de comprender y analizar las distintas configuraciones posibles y sus efectos.

6.1. Pruebas sin WMM

Las primeras pruebas realizadas son con la configuración estándar de OpenWRT, sin calidad de servicio habilitada a nivel de subcapa MAC. Estas pruebas sirvieron como referencia para la comparación con resultados posteriores y poder observar los efectos y beneficios de aplicar QoS.

6.1.1. Se inyecta un flujo UDP

Resultados: el throughput obtenido del canal es 27,4 Mbps, mientras que el RTT es 45 ms cuando hubo tráfico saturando el canal.

6.1.2. Se inyectan 2 flujos UDP a distintos puertos

Se inyectaron 2 flujos UDP utilizando iperf (descrito en capítulo 4), uno al puerto 5001 (prioritario) y otro al 5005 (no prioritario). Como ya se mencionó, se utilizó esta nomenclatura dado que en pruebas posteriores si habrá un puerto prioritario y otro no prioritario, si bien, en la que se realizó a continuación no hay ningún flujo con mayor prioridad que otro.

Resultados: flujo prioritario: 13.6 Mbps, flujo no prioritario: 13.6 Mbps

[1] Wireless Multimedia Extensions (WME), también llamado WiFi Multimedia (WMM) es una certificación de interoperabilidad basada en el estándar 802.11e.

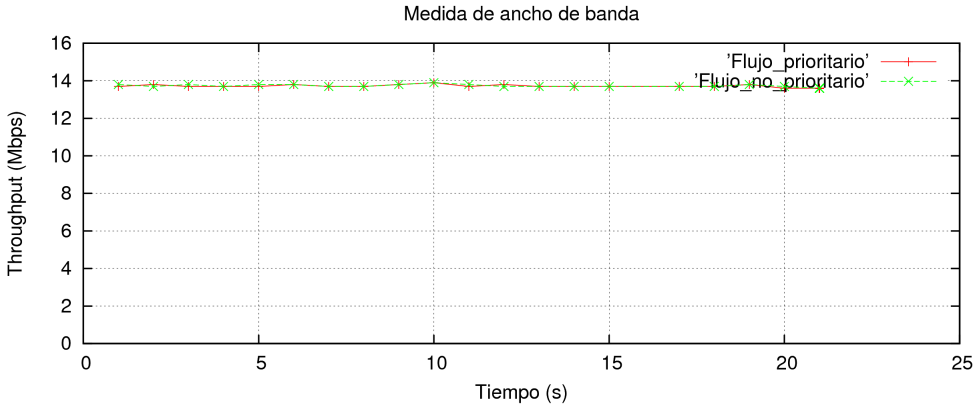

Figura 6.1: 2 flujos UDP a distintos puertos

Observaciones: el flujo se dividió por igual para cada tráfico y en la gráfica se puede ver a intervalos de 1 segundo.

6.1.3. Análisis

El ancho de banda disponible en el canal se dividió por igual cuando se inyectaron 2 flujos simultáneamente. Ésto era esperado ya que ningún flujo tenía prioridad sobre el otro.

6.2. Pruebas con WMM

Se habilitó WMM en los routers inalámbricos (ver C.2.2). Se inyectaron un flujo prioritario (que se mapeó en AC_VO) y otro no prioritario (que se mapeó en AC_BK) y se utilizaron los valores por defecto de 802.11e para estas pruebas. Más detalles sobre el marcado de paquetes se pueden ver en el anexo D.

6.2.1. Se inyecta un flujo UDP

Se inyectó primero un flujo marcado como prioritario y luego otro como no prioritario (no simultáneos).
Flujo prioritario.
Resultados: el ancho de banda obtenido fue de 28 Mbps.
Flujo no prioritario.
Resultados: el ancho de banda obtenido fue de 22.2 Mbps.

Observaciones: el flujo prioritario obtuvo un mayor throughput en el canal. Esto es exclusivamente por aplicar 802.11e, ya que cada una de las colas tenía distintos parámetros de contienda.

Para verificar lo observado en la prueba anterior, se realizó otra prueba con un solo flujo. Se inyectó un flujo no prioritario pero se modificaron los parámetros de la clase AC_BK, aumentando la probabilidad de acceso al medio ($CWmin = 2$, $CWmax = 3$, $TXOP = 1504$, $AIFS = 0$).
Flujo no prioritario.
Resultados: el ancho de banda obtenido fue de 29.1 Mbps.

Observaciones: se pudo corroborar que el cambio de parámetros de la clase AC_BK aumentó el throughput. Reduciendo el tamaño de ventana y el tiempo entre tramas y aumentando la oportunidad de transmitir una vez obtenido el acceso al medio, se obtuvo en esta prueba casi 7 Mbps más que en la prueba realizada anteriormente.

6.2.2. Se inyectan 2 flujos UDP a distintos puertos

En las siguientes pruebas 2 flujos UDP compitieron por el acceso al medio. De acuerdo a las experiencias realizadas en [20] se esperaba obtener una diferencia entre el throughput obtenido por el flujo prioritario y el no prioritario.

6.2.2.1. 802.11e, parámetros por defecto

Resultados: el flujo prioritario obtuvo un throughput de 11.8 Mbps mientras que el no prioritario 11.9 Mbps.

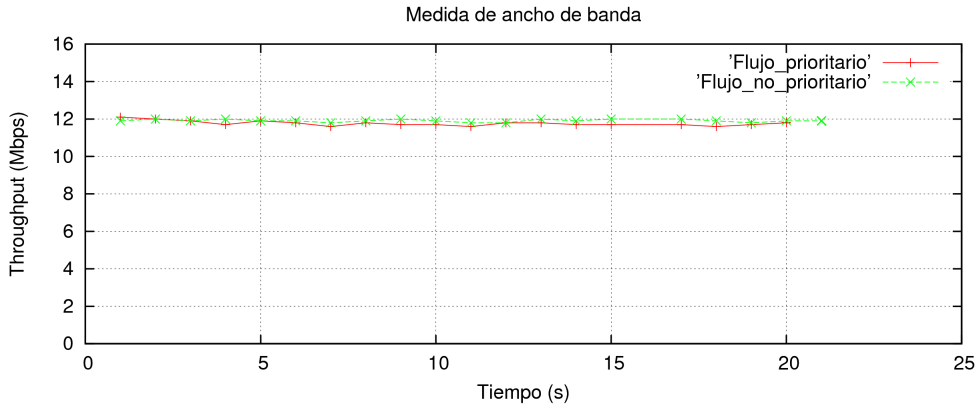

Figura 6.2: WMM, parámetros por defecto

Observaciones: el flujo se dividió por igual para cada tráfico. Habilitar WMM no solo no permitió diferenciar un tráfico de otro, sino que además el throughput total (suma de ambos) fue menor al de las pruebas vistas en 6.1.

Este resultado no era el esperado ya que se pretendía que al habilitar WMM se obtuviera alguna diferencia en el throughput obtenido para cada uno de los flujos.

Pérdida de paquetes y RTT: la pérdida de paquetes fue similar en cada uno de los flujos, alrededor de un 35 % de pérdidas. A continuación se muestran los resultados obtenidos para el RTT.

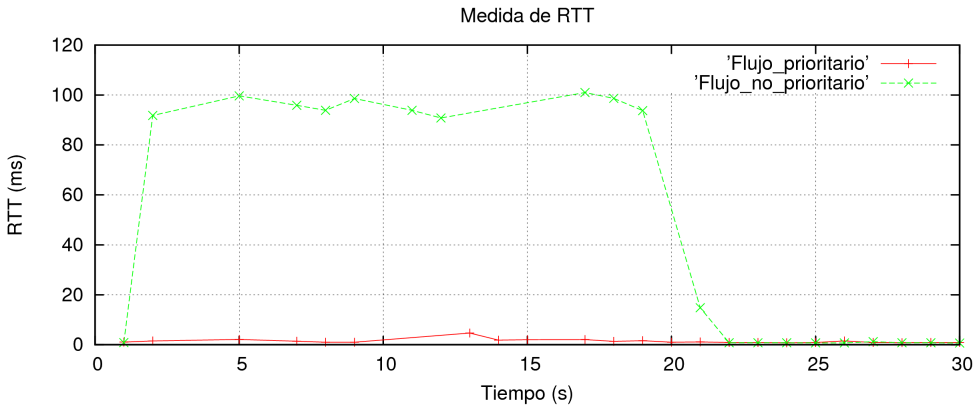

Figura 6.3: WMM, parámetros por defecto

Observaciones: la pérdida de paquetes era previsible ya que se saturó el enlace, aunque se esperaba ver una diferencia en las pérdidas del flujo por la cola AC_VO y el flujo por la cola AC_BK. Igualmente, estas pérdidas coincidieron con las observaciones realizadas a raíz del throughput obtenido. En el RTT sí hubo una clara diferencia como se puede apreciar en la gráfica 6.3.

6.2.2.2. 802.11e, se degradó AC_BK

Dado que habilitando WMM no se percibió ninguna diferencia en el reparto del canal, se cambiaron los valores de AC_BK poniendo $CWmin = 13$, $CWmax = 15$, $TXOP = 0$, $AIFS = 7$. Con esta prueba se esperaba ver degradado el throughput por AC_BK y mantenerse el de AC_VO.

Resultados: el flujo prioritario obtuvo un throughput de 2.29 Mbps mientras que el no prioritario 2.30 Mbps.

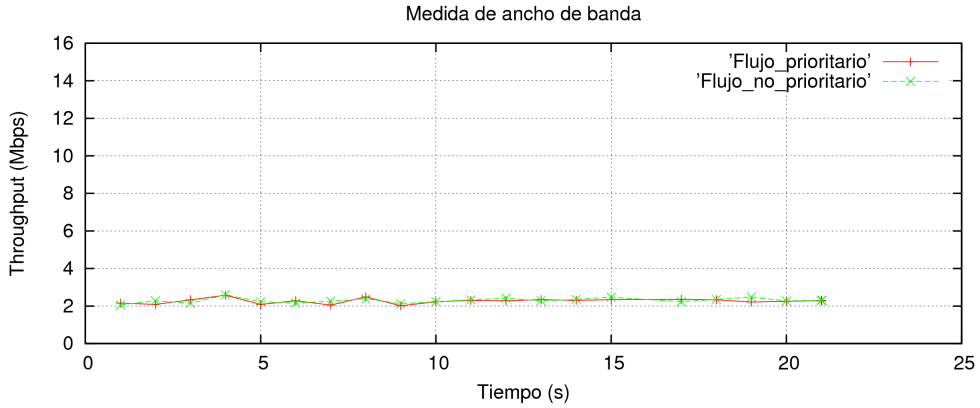

Figura 6.4: WMM, AC_BK degradado

Observaciones: se nota claramente un descenso en el throughput de cada flujo pero el canal se repartió de igual forma. El resultado no fue el esperado, ya que no solo se afectó el flujo no prioritario sino también el prioritario.

6.2.2.3. 802.11e, se degradó AC_BK pero se aumentó TXOP limit

Dado que en las pruebas anteriores no se percibió ninguna diferencia en el reparto del canal, se cambiaron los valores de AC_BK poniendo:
$CWmin = 13$, $CWmax = 15$, $TXOP = 8192$, $AIFS = 7$.

Se esperaba ver una mejora en el throughput por la cola AC_BK respecto de la prueba anterior ya que se aumentó el txop manteniendo igual el resto de los parámetros.

Resultados: el flujo prioritario obtuvo un throughput de 8.59 Mbps mientras que el no prioritario 8.88 Mbps.

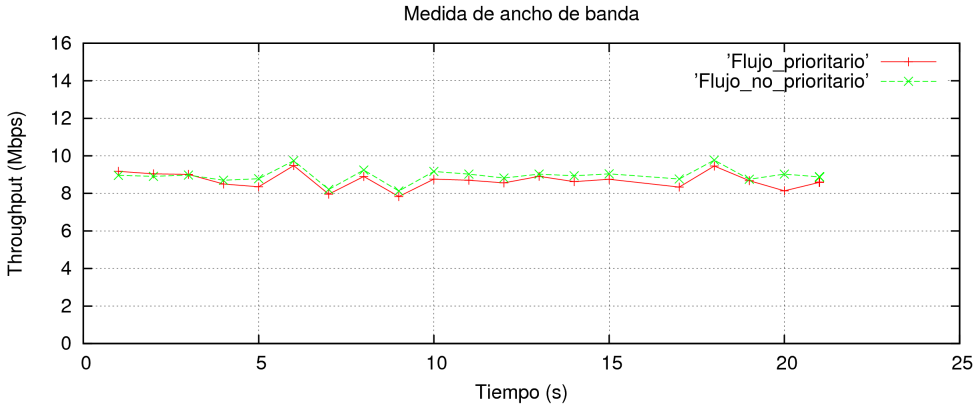

Figura 6.5: WMM, AC_BK degradado pero txop = 8192

Observaciones: se pudo apreciar un aumento en el throughput de cada flujo respecto de la prueba anterior, pero se repartieron de igual forma el canal. El resultado, nuevamente, no afectó solamente el flujo no prioritario (AC_BK) sino también el prioritario (AC_VO).

6.2.3. Análisis

Únicamente habilitando WMM no se consiguió realizar una división del canal deseada. Si bien hubo efecto al modificar los valores de las colas de 802.11e, los resultados no fueron satisfactorios para la división diferenciada del canal ante un flujo prioritario y otro no prioritario. Modificar los parámetros de la cola AC_BK de 802.11e afectó el throughput obtenido para el tráfico de AC_VO y AC_BK, no lográndose una independencia entre ambos como era esperado a partir de lo que surge del estándar [5] y de los resultados en [20].

No ocurrió lo mismo con el RTT obtenido. De acuerdo a lo que muestra la figura 6.3, se notó una diferencia entre el RTT de los paquetes de la clase prioritaria y los de la clase no prioritaria.

A raíz de los resultados obtenidos y de la implementación utilizada en [21] es que se estudió el uso de disciplinas de colas (*qdisc*) y se buscó la realización de QoS utilizando qdisc con clases y WMM.

Las disciplinas de colas son algoritmos que permiten controlar la cola de un dispositivo, en este caso tarjetas de red. Con estas disciplinas se puede realizar un control de tráfico mediante el ajuste, ordenamiento y/o descarte de paquetes. Por más detalles ver apéndice E.

6.3. Pruebas utilizando qdisc sin prioridades

En una primera etapa, se realizaron las pruebas para una configuración de qdisc con 2 colas sin prioridades entre cada una de ellas, de forma de ver el impacto de 802.11e sin que haya una previa priorización por las qdisc. Se puede ver la configuración en la sección F.1 del apéndice.

Se repitieron varias de las pruebas realizadas sin qdisc para ver el efecto que tuvo implementar las mismas. Lo que difirió con las pruebas anteriores es que ahora cada tráfico tuvo una cola independiente para el manejo de paquetes, previo al pasaje a capa de enlace.

Se comenzó con pruebas sin WMM habilitado y luego se habilitó para ver el resultado de utilizar qdisc y 802.11e. Gran parte de las pruebas se centraron en qué sucedió con la división del throughput, a raíz de lo obtenido en 6.2.2.

6.3.1. Pruebas sin WMM

6.3.1.1. Se inyecta un flujo UDP

En esta prueba se inyectó primero un flujo marcado como prioritario y luego otro como no prioritario (no simultáneos).
Flujo prioritario.
Resultados: se obtuvo un throughput de 25.3 Mbps
Flujo no prioritario.
Resultados: se obtuvo un throughput de 25.3 Mbps

Observaciones: tal cual era esperado, ningún flujo tuvo mejor throughput ya que no había prioridades en las qdisc y WMM estaba deshabilitado

6.3.1.2. Se inyectaron 2 flujos UDP a distintos puertos

Resultados: el flujo prioritario obtuvo un throughput de 12.5 Mbps al igual que el no prioritario.

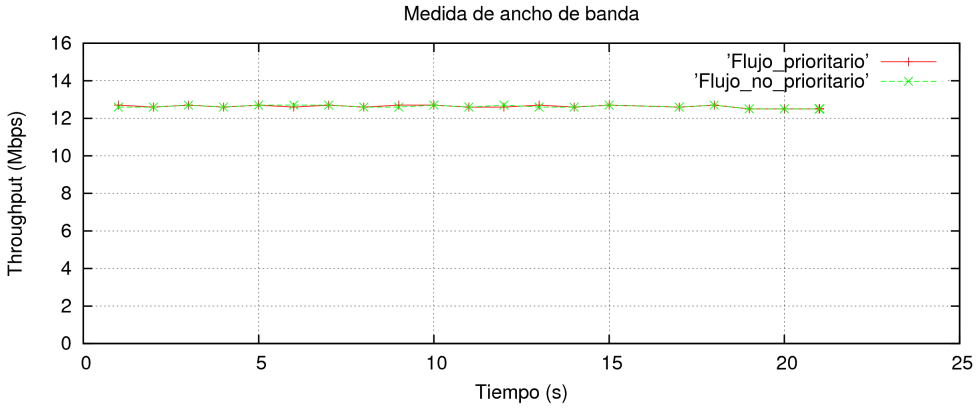

Figura 6.6: 2 flujos, qdisc sin prioridad, WMM desactivado

Observaciones: tal cual era esperado, ningún flujo tuvo mejor throughput ya que no había prioridades en las qdisc y WMM estaba desactivado.

Análisis: al no haber prioridades en la configuración de qdisc y estando WMM deshabilitado, no hay mucho más que agregar a las observaciones ya que, como era esperado, los flujos se repartieron equitativamente el canal.

6.3.2. Pruebas con WMM

6.3.2.1. Se inyectó un flujo UDP, WMM por defecto

En esta prueba se inyectó primero un flujo marcado como prioritario y luego otro como no prioritario.
Flujo prioritario.
Resultados: se obtuvo un throughput de 25.3 Mbps
Flujo no prioritario.
Resultados: se obtuvo un throughput de 22.3 Mbps
Observaciones: al igual que en las pruebas realizadas en 6.2.1, se obtuvo un mayor throughput para el flujo por la cola AC_VO.

6.3.2.2. Se inyectaron 2 flujos UDP a distintos puertos, WMM por defecto

Resultados: se obtuvo un throughput de 11.8 Mbps para cada uno de los flujos.

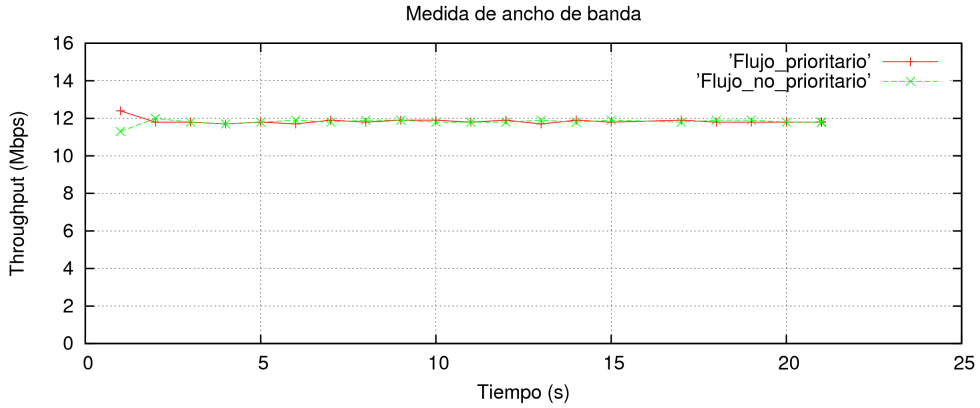

Figura 6.7: 2 flujos, qdisc sin prioridad, WMM parámetros por defecto

Observaciones: el throughput obtenido para ambos flujos fue el mismo. Al no haber ninguna priorización en qdisc, la contienda por el medio realizada a nivel de subcapa MAC es lo que diferenció cada tráfico. Los parámetros por defecto de WMM no afectaron en el reparto del canal. El utilizar colas con clases para el manejo de tráfico en el kernel, previo a enviar los paquetes al hardware, no permitió diferenciar los resultados obtenidos en esta prueba de lo obtenido en 6.2.2.1.

Pérdida de paquetes y RTT: hubo pérdida de paquetes para ambos flujos lo cual era previsible ya que se saturó el canal. A continuación se muestra el resultado obtenido para el RTT.

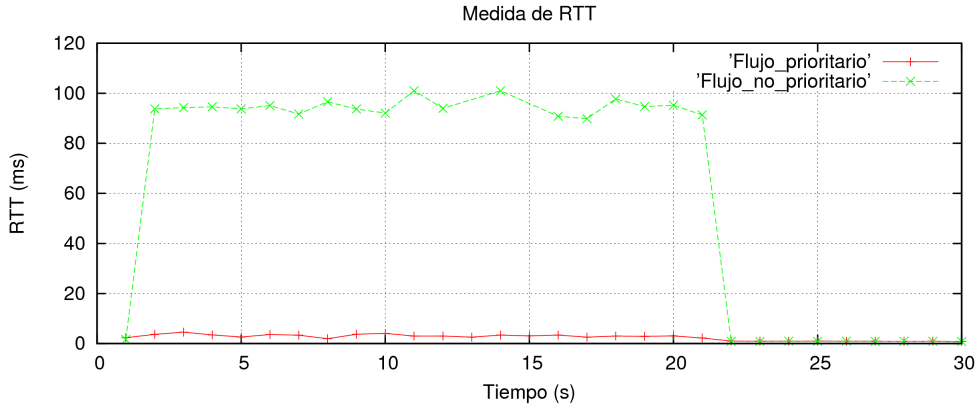

Figura 6.8: RTT de 2 flujos, qdisc sin prioridad, WMM parámetros por defecto

Observaciones: nuevamente se notó una diferencia en el RTT del flujo por la cola AC_VO y la AC_BK.

6.3.2.3. Se inyectaron 2 flujos UDP pero se degradó la cola AC_BK

De la prueba anterior surgió que utilizando colas con clases en el kernel y los parámetros por defecto de 802.11e no se logró un reparto diferenciado del canal. Al igual que en la sección 6.2.2.2, se empeoraron los valores de la cola AC_BK para ver si se lograba diferenciar los throughputs obteniendo menor ancho de banda para el flujo por AC_BK y mayor para el AC_VO.
Se utilizó: $CWmin = 13$, $CWmax = 15$, $AIFS = 7$ y $TXOP = 0$.

Resultados: se obtuvo un throughput de 2.24 Mbps para el flujo prioritario y 2.25 Mbps para el no prioritario.

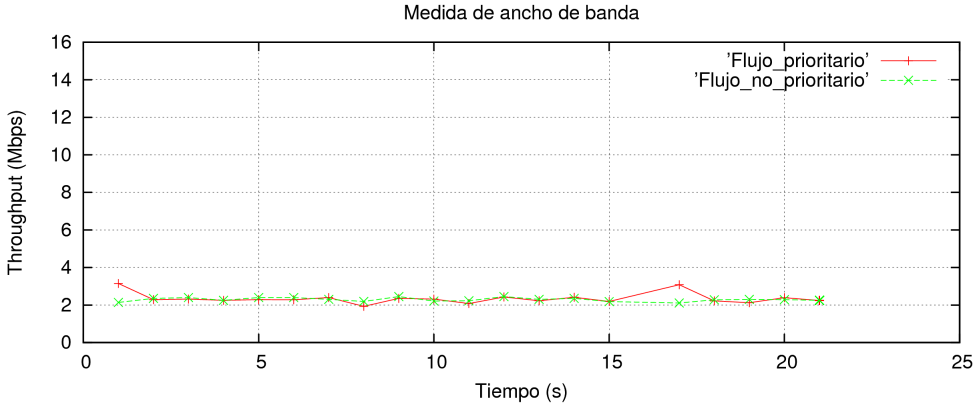

Figura 6.9: 2 flujos, qdisc sin prioridad, clase AC_BK de 802.11e degradada

Observaciones: se pudo apreciar que se degradó el throughput de ambas colas al igual que en 6.2.2.2.

Análisis: los resultados de los ensayos realizados en esta sección no fueron los esperados. Implementar 2 colas (independientes) con clases en el kernel para el manejo de los paquetes previo al pasaje de los mismos a la interfaz de red, no permitió obtener resultados distintos a los obtenidos en 6.2. No se ha logrado repartir el throughput disponible de forma diferenciada entre ambos flujos. El cambio de los parámetros de contienda de una clase de 802.11e afectó los resultados obtenidos por ambos flujos, no lográndose una independencia entre los mismos como era esperado.

6.4. Hipótesis, ensayos y verificación

Hasta aquí, las pruebas realizadas no permitieron dividir el ancho de banda disponible en el canal de forma selectiva. Habilitar WMM y cambiar los parámetros de contienda de AC_BK mejoraron el throughput obtenido por un flujo UDP, prueba 6.2.1, pero cuando se inyectaron

dos flujos no se pudo diferenciar el throughput obtenido por los mismos, pruebas 6.2.2, 6.3.2.2 y 6.3.2.3.

En los documentos [22] y [3] se estudió el uso de dispositivos con múltiples colas de hardware en sistemas Linux. Si bien no se pretendió entrar en detalles sobre el manejo de paquetes en el kernel de Linux, ni en la implementación del driver utilizado, la comprensión de lo expresado en estos documentos permitió formular una hipótesis sobre lo observado en las pruebas anteriores.

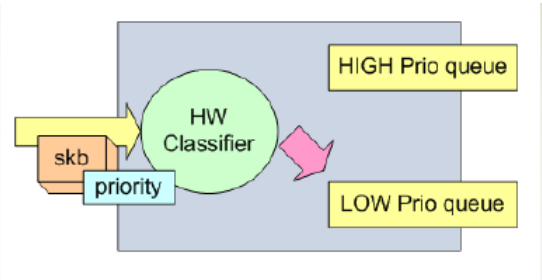

Figura 6.10: 2 colas en hardware [3]

Las pruebas realizadas en 6.2 coinciden con este diagrama. La situación es similar en 6.3.2, salvo que hay 2 colas donde se mapea cada tráfico previo al pasaje al clasificador en hardware.

6.4.1. Hipótesis

Los flujos UDP son tratados por colas distintas en el kernel. Estos paquetes se envían a la tarjeta de red como se muestra en la figura 6.10. Cuando una de las colas de hardware se llena, AC_BK en este caso, el driver llama a la función `netif_stop_queue` y tanto el flujo prioritario como el no prioritario dejan de enviarse al dispositivo de red.

Bajo esta hipótesis la cola AC_BK queda saturada mientras la cola AC_VO está vacía. Al bloquearse el clasificador, la cola AC_VO no recibe más paquetes a transmitir, lo que finalmente implica que los throughputs obtenidos por ambos flujos sean similares.

En cuanto al RTT, la hipótesis planteada se ajusta a lo observado en las gráficas 6.3 y 6.8. El flujo por la clase AC_VO tuvo un menor delay ya que luego de ingresar a la cola en hardware, el acceso al medio se realizaba más rápido que para el flujo por AC_BK. También coincide la similitud entre los paquetes perdidos por cada tráfico.

6.4.2. Ensayos y verificación de la hipótesis

Para verificar la hipótesis planteada se realizaron 2 pruebas. En ambas se degradaron los valores de la clase AC_BK dejando:

$CWmin = 13$, $CWmax = 15$, $AIFS = 7$, y se modificó, primero el TXOP de la clase AC_BK y luego el de la clase AC_VO.

Si la hipótesis era correcta, aumentando el TXOP de la clase AC_BK (txop de AC_VO por defecto) se debería notar un aumento del throughput obtenido en comparación con la prueba 6.3.2.3. En cambio, aumentar el txop de AC_VO (con TXOP de AC_BK por defecto) no debería modificar lo obtenido en la prueba 6.3.2.3 ya que esta cola no debería tener tráfico a transmitir.

TXOP de AC_BK = 8192

Resultados: el flujo prioritario obtuvo un throughput de 8.81 Mbps mientras que el no prioritario 8.88 Mbps.

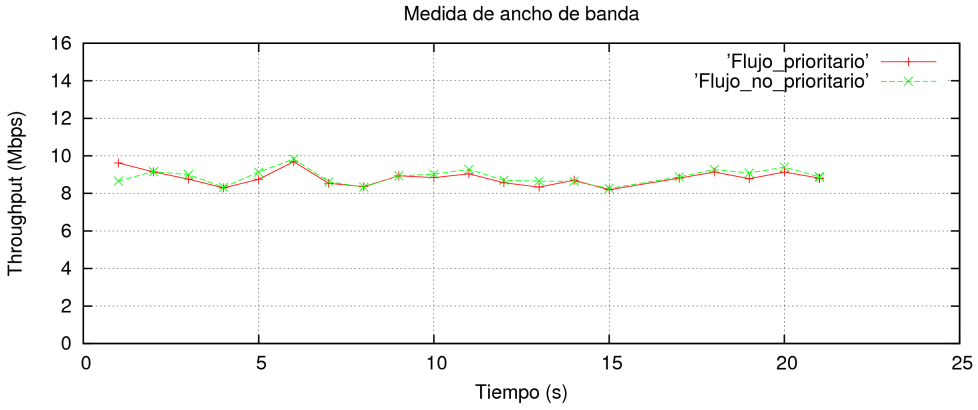

Figura 6.11: 2 flujos, qdisc sin prioridad, clase AC_BK de 802.11e degradada con txop de AC_BK = 8192

TXOP de AC_VO = 8192

Resultados: el flujo prioritario obtuvo un throughput de 2.22 Mbps mientras que el no prioritario 2.25 Mbps.

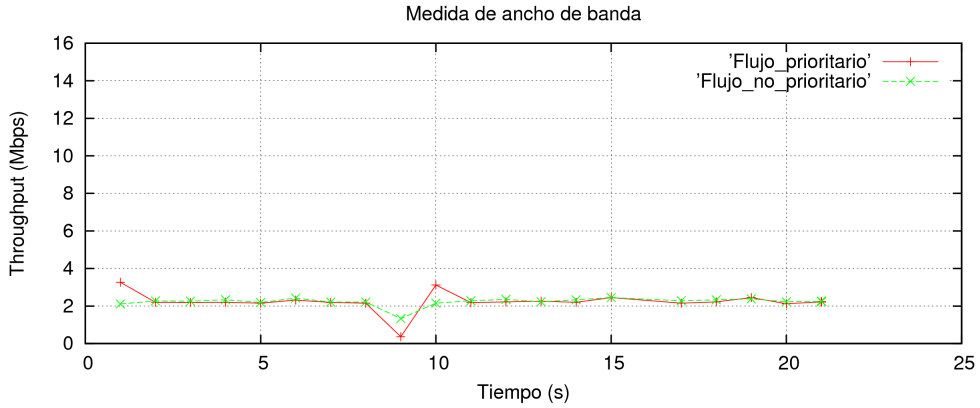

Figura 6.12: 2 flujos, qdisc sin prioridad, clase AC_BK de 802.11e degradada con txop de AC_VO = 8192

Observaciones: en las pruebas realizadas en esta sección, cuando se modificó el *TXOP limit* de AC_VO el throughput no se vio afectado (comparado con la prueba realizada en 6.3.2.3), en cambio cuando se modificó el TXOP de AC_BK, mejoró el throughput de ambos flujos. Los resultados coincidieron con lo esperado al realizar los ensayos.

6.4.3. Validez de la propuesta

De acuerdo a lo expuesto en [22] y en [3] y a las pruebas realizadas, se pudo verificar que la hipótesis plantead se ajustó a los resultados obtenidos en los ensayos. El cambio de parámetros de contienda de 802.11e afectó el acceso al medio, pero el manejo de paquetes entre el kernel y el driver provocó que cuando una cola se llenaba el clasificador de hardware se bloqueaba y ningún otro paquete era enviado a la interfaz de red.

En la situación descrita no se podía lograr una división diferenciada del ancho de banda disponible en el canal. A raíz de esto se implementó qdisc con clases de distintas prioridades. Se buscó poder priorizar un flujo sobre el otro.

6.5. Qdisc, una cola prioritaria y la otra no prioritaria

Para realizar la política de encolamiento se utilizó la herramienta *tc* de linux. Para ello se emplearon disciplinas de colas con clases, de forma de lograr un trato diferenciado a los diferentes tipos de tráfico. Las qdisc tienen una estructura jerárquica, en forma de árbol, en donde se optó por una disciplina muy conocida para la clase raíz, llamada *Hierarchical Token Bucket* o HTB, la cual permite dividir el ancho de banda disponible entre un mínimo y un máximo. El algoritmo asegura la disponibilidad del mínimo, y si se puede, alcanza el máximo. Luego de distribuir dentro de una misma clase con HTB, se optó por utilizar *Stochastic Fairness Queueing* o SFQ, el cual permite mantener un gran número de colas FIFO, para cada uno de los flujos dados, entre emisor

y receptor.

La configuración de colas utilizada se encuentra en la sección F.2 del apéndice. Los resultados obtenidos se detallan a continuación.

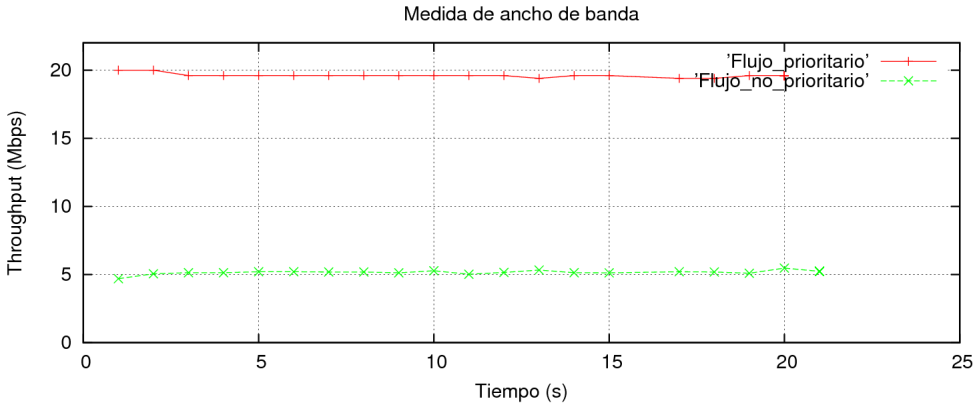

Figura 6.13: 2 flujos, qdisc con prioridad, WMM por defecto

Observaciones: hubo una clara diferencia entre el throughput obtenido por cada uno de los flujos. La utilización de colas con prioridades permitió diferenciar el throughput de cada tráfico.
¿Qué sucedió con el RTT?
Para ello se midió el retardo en cada una de las colas, cuando WMM estaba activado y cuando WMM no estaba activado.

Para WMM habilitado

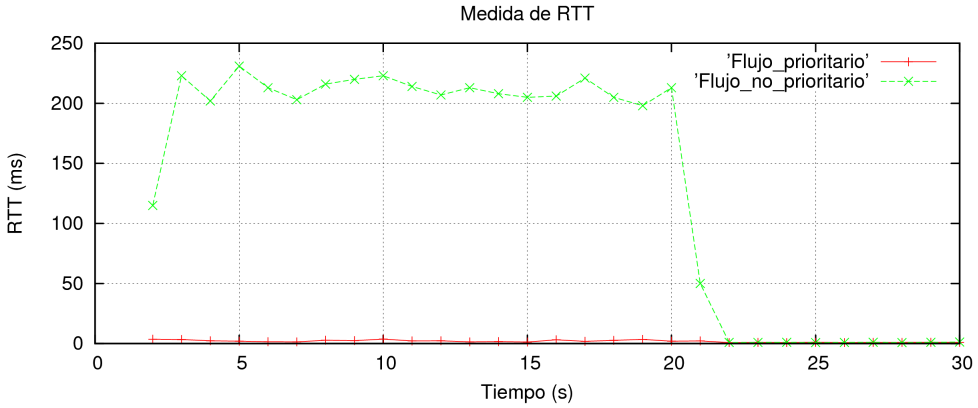

Figura 6.14: 2 flujos, qdisc con prioridad, WMM por defecto

Jitter obtenido con iperf:

Flujo	jitter promedio(ms)	jitter máximo(ms)
Prioritario	0,242	0,289
No prioritario	1.79	18.175

Para WMM deshabilitado

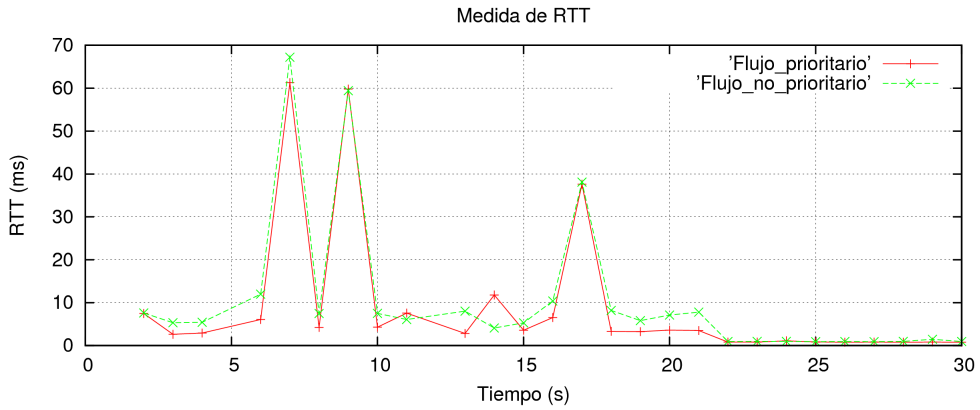

Figura 6.15: 2 flujos, qdisc con prioridad, WMM deshabilitado

Observaciones: si bien las qdisc permitieron obtener una división del canal diferenciada para cada uno de los flujos, se apreció que habilitando WMM el RTT y jitter de la clase prioritaria estaban muy por debajo de los obtenidos para la no prioritaria. Esto no sucedió cuando 802.11e estaba desactivado.

6.6. Conclusiones

Si bien no se pudo obtener los resultados esperados a partir del estándar 802.11e y de los ensayos en [20], el objetivo se cumplió implementando disciplinas de colas en capa de red y WMM en subcapa MAC.

La diferencia en los throughputs obtenidos por cada flujo se logró a partir de las prioridades en las clases a nivel de capa de red.

Asimismo el RTT y jitter son mucho menores para la clase AC_VO que para AC_BK cuando se habilita WMM. Si bien en los ensayos de laboratorio estos tiempos parecen no tener mucha trascendencia, en larga distancia podrían ser la causa de que una conversación se pueda realizar

o no.

Se logró entonces diferenciar el throughput obtenido y minimizar el retardo de la clase prioritaria, utilizando qdisc con prioridades y habilitando WMM.

Todas las pruebas se realizaron marcando el tráfico según el puerto de destino, esto es arbitrario y se puede extender para cualquier protocolo, en particular marcando los paquetes de aplicaciones en tiempo real (voz, video).

Parte III

Establecimiento del Enlace

Capítulo 7

Radioenlace

A la hora de establecer un enlace inalámbrico existen diversos factores a tener en cuenta para su realización:

- Frecuencia del enlace
- Zonas de Fresnel
- Friis
- Pérdidas
  - Pérdidas por propagación en espacio libre
  - Otras pérdidas: factores atmosféricos, desalineamiento de antenas, pérdidas en conectores
- Disponibilidad
- Efectos multitrayectoria
- Curvatura de la Tierra

En las siguientes secciones se describirán los aspectos fundamentales, así como las principales limitaciones para establecer enlaces inalámbricos.

7.1. Zonas de Fresnel

Cuando se estudia la propagación de ondas de radio entre 2 puntos, el concepto de "línea de vista RF", no significa simplemente que desde un punto del enlace se pueda "ver" el otro sin que haya algún obstáculo que obstruya la visual. Para radioenlaces, el espacio se divide en una familia de elipsoides, conocidas como elipsoides de Fresnel, que son áreas concéntricas alrededor de la línea directa entre ambos puntos del enlace.

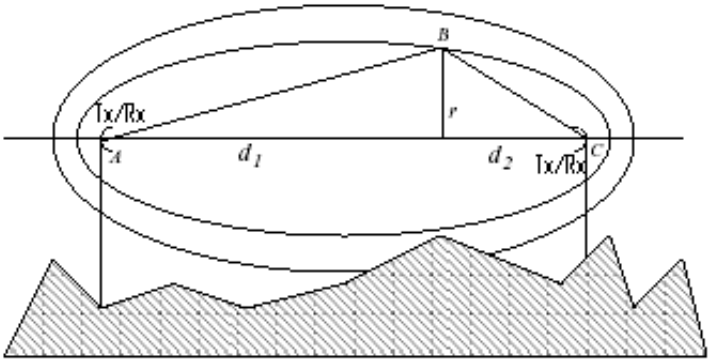

Figura 7.1: Elipsoides de Fresnel

El radio de la enésima zona de Fresnel puede calcularse de la siguiente manera:

$$r_n = \sqrt{\frac{n.\lambda.d_1.d_2}{d}} \tag{7.1}$$

λ: longitud de onda en metros (m).
d_1 y d_2 : distancia en m desde la antena 1 y 2 al punto donde queremos calcular el radio de Fresnel.
d: distancia entre el emisor y el receptor en m.

Para establecer línea de vista RF es necesario tener al menos un 60 % de la primera zona de Fresnel libre, dado que allí se transmite cerca del 95 % de la energía. No cumplir con esta restricción agrega pérdidas al enlace. A partir de los cálculos desarrollados en [23], la ecuación del radio para la primera zona de Fresnel puede escribirse:

$$r_1 = 548.\sqrt{\frac{d_1.d_2}{f.d}} \tag{7.2}$$

donde:
r_1 : radio de la primera zona de Fresnel en m.
d_1 y d_2 : distancia en km desde la antena 1 y 2 al punto donde se quiere calcular el radio de Fresnel.
d: en km
f: en Mhz

7.2. Pérdidas

7.2.1. Pérdidas en espacio libre

Se llama pérdidas en espacio libre a la pérdida de potencia (atenuación) que sufre una onda electromagnética al desplazarse en espacio abierto. Cuando la propagación es en espacio libre, existe una expresión que permite calcular las pérdidas teóricas del enlace. Esta fórmula, que depende de la frecuencia del enlace así como de la distancia, se puede aplicar cuando hay "línea de vista" despejada, tal como se explicó en 7.1.

$$L_p(dB) = 92,45 + 20\log f(GHz) + 20\log d(Km) \tag{7.3}$$

Las pérdidas en espacio libre (FSL: *Free Space Loss*) son el factor más importante de pérdidas en un enlace inalámbrico, pero no el único.

7.2.2. Otras pérdidas

Existen otros factores que introducen pérdidas en un radioenlace. En los sistemas transmisores y receptores se tienen pérdidas en las conexiones y en las guías de ondas, las cuales deben tenerse en cuenta cuando se dimensiona el enlace a establecer.

Otros factores tales como efectos multitrayectoria, condiciones climáticas y desalineamiento de antenas también introducen pérdidas. De estas últimas no se hará un análisis previo.

7.3. Ecuación de Friis

La potencia recibida por el receptor, cuando la propagación se da en espacio libre y en condiciones ideales, se puede calcular a partir de la fórmula de Friis:

$$P_{Rx} = P_{Tx} + G_{Tx} + G_{Rx} - L_{ccTx} - L_{ccRx} - L_p \tag{7.4}$$

siendo:

P_{Rx} : potencia recibida por el receptor (dBm)
P_{Tx} : potencia transmitida por el transmisor (dBm)
G_{Tx} : ganancia de la antena transmisora en la dirección del enlace (dBi)
G_{Rx} : ganancia de la antena receptora en la dirección del enlace (dBi)
L_{ccTx} : pérdida en el sistema transmisor (dB)
L_{ccRx} : pérdida en el sistema receptor (dB)
L_p : pérdidas en espacio libre (dB)

Esta ecuación permite realizar un balance del enlace y estimar la potencia recibida en condiciones ideales. En la práctica, la potencia recibida difiere de la esperada por múltiples razones,

pero básicamente se debe a que la propagación de las ondas electromagnéticas no se da en condiciones ideales.

Friis sirve para calcular la potencia en recepción y evaluar si la realización del enlace es viable.

7.4. Señal en recepción

Para saber si un enlace es realizable o no, se debe tener en cuenta 2 factores en la potencia de la señal recibida:

- Sensibilidad del receptor (Potencia recibida (P_{Rx}) *vs* sensibilidad del receptor).
- Relación señal a ruido en recepción (SNR: potencia recibida P_{Rx} *vs* ruido en recepción).

La sensibilidad del receptor es un parámetro de las tarjetas inalámbricas y es un dato aportado por el fabricante (ver sección 3.1.2.1). Para poder establecer el enlace se requiere que la potencia en recepción (P_{Rx}) sea mayor que la sensibilidad del receptor [1].

Solo con la potencia en recepción no alcanza para garantizar el establecimiento de un enlace. Se tiene que evaluar además, la relación señal a ruido en recepción ($SNR = \frac{P_{Rx}}{N}$). El SNR es un parámetro que indica la calidad de la señal en el receptor. Un buen criterio es tomar SNR $\geq$ 15 dB, lo cual, como se explicó en la sección 3.1.2.1, es el que toman la mayoría de los fabricantes para armar la cartilla de datos de las tarjetas.

Cumpliendo con estas 2 condiciones, no solo se tiene que el enlace es realizable, sino que también se sabe la velocidad de transmisión, dado que es un dato aportado por el fabricante.

7.5. Disponibilidad

La disponibilidad es un factor de calidad de un enlace. En grandes términos indica el porcentaje del tiempo en que un enlace debería estar funcionando, evaluando cuántas veces y por cuánto tiempo la potencia recibida se encuentra por debajo del umbral de recepción.

Definir umbral $\Rightarrow$ Disponibilidad del sistema $\Rightarrow$ Fade Margin (FM)

Existen varios modelos para determinar el FM. Se utilizó el de Barnett-Vigants. El Fade Margin necesario se calcula para una cierta disponibilidad teniendo en cuenta distintos parámetros tales como: factores climáticos (temperatura, humedad), frecuencia de trabajo, distancia y condiciones geográficas del terreno.

[1] Ver que distintas velocidades de transmisión implican distintas sensibilidades del receptor.

La ecuación para calcular la disponibilidad utilizando el modelo de Barnett-Vigants:

$$FM_{dB} = 30.\log(d) + 10.\log(6.A.B.f) - 10.\log(1-R) - 70 \tag{7.5}$$

donde:
FM_{dB} : Fade Margin
d: distancia entre antenas en km
A: factor geográfico o de aspereza
B: factor climático
f: frecuencia de trabajo en GHz
R: confiabilidad esperada o convenida en decimales

La confiabilidad es el parámetro que indica cuánto tiempo se espera que el enlace esté activo (p.e. confiabilidad del 99,99 % indica que el enlace está activo el 99,99 % del tiempo o lo que es lo mismo, que el enlace está interrumpido un 0.01 % del tiempo).

Para que un enlace esté activo un tiempo X se debe cumplir:

$$P_{Rx} - FM \geq S \tag{7.6}$$

Donde S es la sensibilidad del receptor y el FM se calcula para el tiempo X.

7.6. Cálculo del enlace

A continuación se presentan los resultados de los cálculos realizados teniendo en cuenta los factores mencionados anteriormente y los datos del hardware utilizado.

En cuanto a los conectores, tipo N, se tomaron valores de pérdidas de 3 dB en cada uno, de forma de dejar un buen margen para el cálculo.

Frecuencia (Ghz)	Distancia (km)	Radio Fresnel max (m)	Pérdidas espacio libre (dB)	Prx por Friis (dBm)	Enlace realizable	Fade Margin (dB)
2,4	30	30,6	129,5	-53,5	SI	36,5
5,8	30	19,7	137,2	-62,7	SI	19,3

La sensibilidad del receptor es -90 dBm y se tomó como potencia del ruido -97 dBm (dato utilizado por recomendación del Plan Ceibal). De acuerdo a lo señalado en 7.4 se tiene que la

Prx debe ser mayor a -82 dBm en el caso más restrictivo.

El FM queda en 36,5 dB para 2,4 Ghz y 19,3 dB para 5,8 Ghz. Luego utilizando A = 2 y B = 0,5 [2] en la ecuación 7.5, se obtiene una confiabilidad de 99,89 % para 5,8 Ghz y mayor al 99,98 % en 2,4 Ghz.

Todos los valores tomados fueron los más conservadores posibles por lo tanto el enlace es realizable tanto en 2,4 Ghz como en 5,8 Ghz, con una confiabilidad mayor al 99,89 % en ambas frecuencias.

[2] Factores utilizados por Plan Ceibal para el cálculo de radioenlaces.

Capítulo 8

Instalación

8.1. Introducción

Uno de los objetivos iniciales del proyecto fue poder extender la red del Plan Ceibal a algún centro educativo que, dada la lejanía con respecto al sitio con conectividad más cercano, no estuviera integrado a la red. Se identificó la escuela 101, en la localidad de San Gabriel, sin conectividad al momento de iniciar el proyecto. Para poder integrar este centro se debería realizar un enlace desde Florida, a 30km.

Figura 8.1: Ubicación geográfica de los extremos del enlace [4]

8.2. Frecuencia de trabajo

Uno de los principales temas a definir luego de conocer los puntos a conectar fue la frecuencia de trabajo. Como se presentó anteriormente dentro del estándar 802.11 existen 2 bandas de

frecuencia, una en 2.4 GHz y otra en 5.8 GHz. Entre las ventajas consideradas que presenta la primera se destaca:

- Menores pérdidas en el espacio libre. Como se indicó en el capítulo 7 de la ecuación de Friis podemos ver que las pérdidas en esta banda son menores, existiendo una diferencia de aproximadamente 7.7 dB.

Dentro de las ventajas que se consideraron en la banda de 5.8 GHz se encuentran las siguientes:

- Menor radio de Fresnel. Como se indicó en el capítulo 7 el radio de la primera zona de Fresnel es inversamente proporcional a la raíz cuadrada de la frecuencia, lo que hace que el radio en 2.4 GHz sea aproximadamente un 50 % mayor que en 5.8 GHz.

- Menor interferencia. El uso de la banda de 2.4 GHz ha tenido una explosión muy importante que ha llevado a que el uso de sus canales sea comúnmente compartido entre teléfonos inalámbricos, routers, enlaces, etc. Esto, sumado a que en 5.8 GHz existen más canales disponibles, lleva a que en esta banda haya menor probabilidad de que se produzcan colisiones con otros dispositivos y presente una mejor performance.

Para determinar la frecuencia de trabajo se estudió el impacto de estos factores. En la siguiente figura se observa las primeras zonas de Fresnel simuladas con Radio Mobile a partir del relieve del terreno y sabiendo que la altura máxima a la que era posible instalar las antenas era de 80 m en Florida y 60 m en San Gabriel. Como se observa en ambos casos se cumple que el 60 % de la primera zona de Fresnel no está obstruido tanto en 2.4 GHz como en 5.8 GHz.

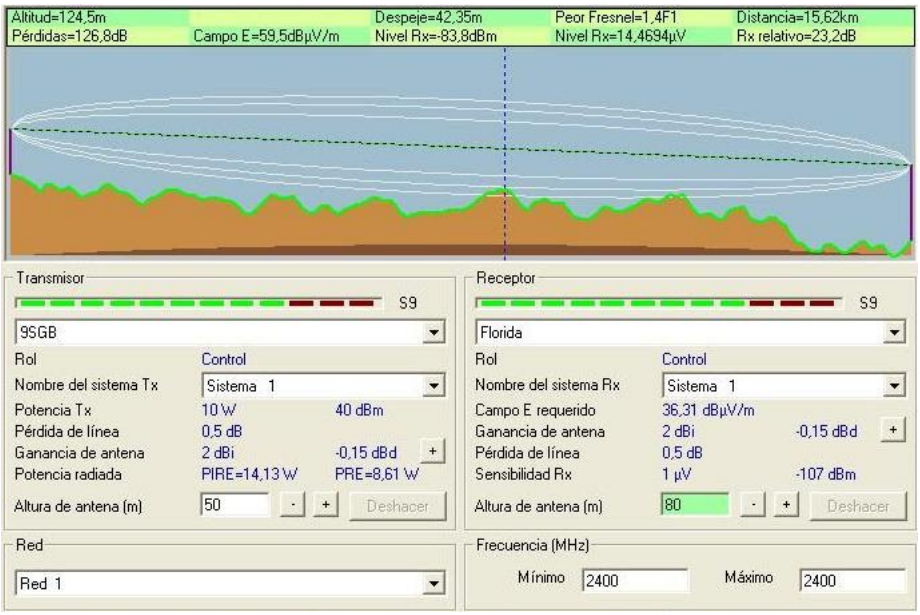

Figura 8.2: Análisis de Fresnel para 2.4 GHz

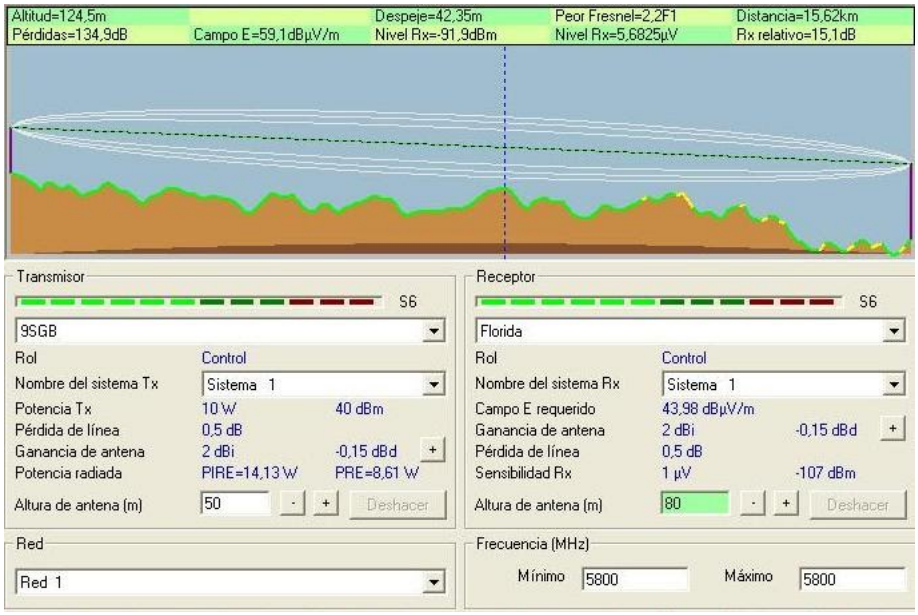

Figura 8.3: Análisis de Fresnel para 5.8 GHz

Sin embargo, ya que este análisis solo toma en cuenta el relieve se complementó el estudio con un relevamiento del perfil determinándose que, debido a la existencia de un monte en el trayecto, en el caso de 2.4 GHz no se cumpliría con la necesidad planteada. En la siguiente figura se observa el perfil realizado.

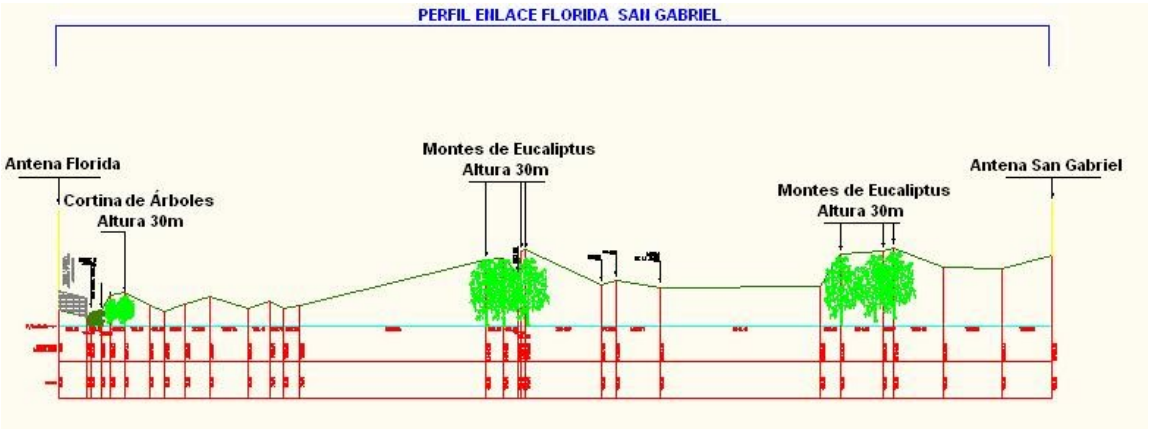

Figura 8.4: Perfil

8.3. Alineamiento de antenas

El alineado se realizó en varias etapas comenzando una aproximación gruesa a partir de las posiciones y finalizando con el alineado fino a partir de relevamiento de señal y ruido [24]. Las

etapas fueron las siguientes:

- Inicialmente se realizó la instalación de las antenas a la altura definida y de acuerdo al azimut calculado.
- Luego, a partir de un dispositivo GPS, se identificaron puntos en el trayecto que fueran visibles desde cada extremo y se ajustaron las antenas en esa dirección.
- Finalmente se realizó el alineado fino de la siguiente forma: primero se fijó un extremo y a medida que se movía lentamente el otro se observaba la variación de señal y ruido obtenido hasta lograr maximizarlo. Luego se fijó ese extremo y se realizó el mismo procedimiento en el que inicialmente estaba fijo.

En las siguientes figuras se pueden observar ambas antenas una vez finalizada la instalación.

Figura 8.5: Imágenes de antenas GD58-29 instaladas en Florida (izq) y San Gabriel (der).

8.4. Topología

En la siguiente figura se puede observar los dispositivos instalados.

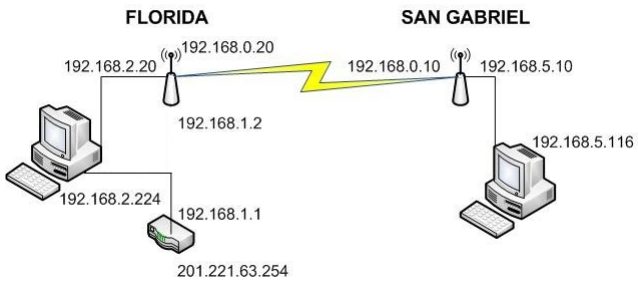

Figura 8.6: Diagrama de topología

Como se observa en cada extremo del enlace se conectaron equipos a los routers Mikrotik desde los que se realizaron pruebas automáticas y se almacenaron los resultados. Para poder trabajar de forma remota sobre el enlace se instaló en el extremo ubicado en Florida un modem con conexión 3G e IP fija hacia donde era posible conectarse directamente a través de Internet. Estos equipos fueron instalados en las bases de las antenas, en gabinetes adecuados que pueden observarse en las siguientes figuras.

Figura 8.7: Gabinetes con equipos instalados en Florida (izq) y San Gabriel (der)

Capítulo 9

Análisis

Como se presentó en capítulos anteriores el objetivo fue establecer el enlace de larga distancia para extender la red del Plan Ceibal. Asimismo, para poder ofrecer prestaciones como videoconferencias o educación a distancia es de particular importancia realizar un trato diferenciado de los flujos transmitidos. Requerimientos sobre el ancho de banda o "delay" son fundamentales para el correcto funcionamiento de estas aplicaciones.

9.1. Características del enlace

Del mismo modo que en las pruebas de laboratorio, se estudió el enlace en saturación para analizar el comportamiento del mismo en presencia de un tráfico sin prioridad frente a uno prioritario. Las características estudiadas fueron el throughput obtenido y el RTT promedio en cada caso. A su vez, se estudió tanto la configuración realizada utilizando qdisc y 802.11e como el impacto de cada uno de ellas por separado. Los tráficos utilizados fueron de 15 Mbps en cada caso marcados como voz y background.

9.1.1. Configuración seleccionada

Como se planteó anteriormente, implementar qdisc con prioridades permitió dividir el throughput de acuerdo a los objetivos planteados y con 802.11e se obtuvieron mejores tiempos de RTT para el tráfico prioritario. La configuración realizada en el enlace consistió en una solución conjunta.

Resultados: en la siguiente gráfica se observa como el flujo prioritario obtuvo la mayor parte del canal: 11,19 Mbps de throughput contra los 0,49 Mbps obtenidos por el flujo no prioritario. El throughput total obtenido fue de aproximadamente 11,68 Mbps.

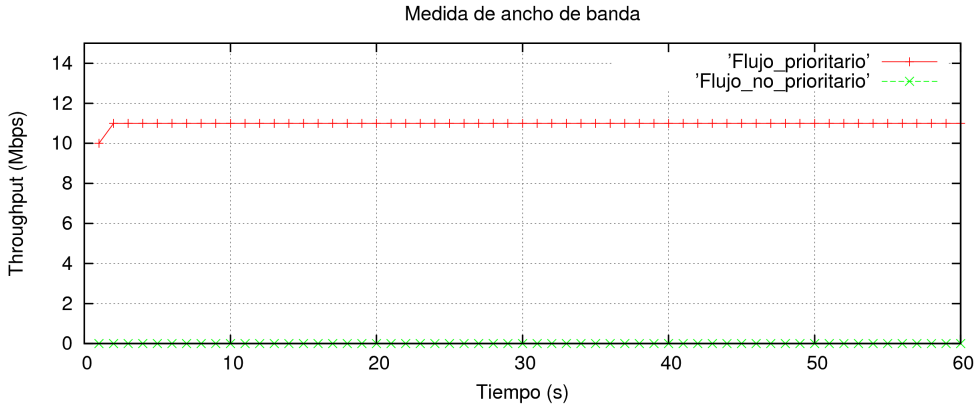

Figura 9.1: Bandwidth obtenido con la configuración seleccionada

Como se observa en la siguiente gráfica, el RTT promedio del flujo prioritario fue de 3,93 ms contra los 48,32 ms del no prioritario. Estos promedios se calcularon hasta que finalizaron de enviarse los flujos UDP (segundo 60).

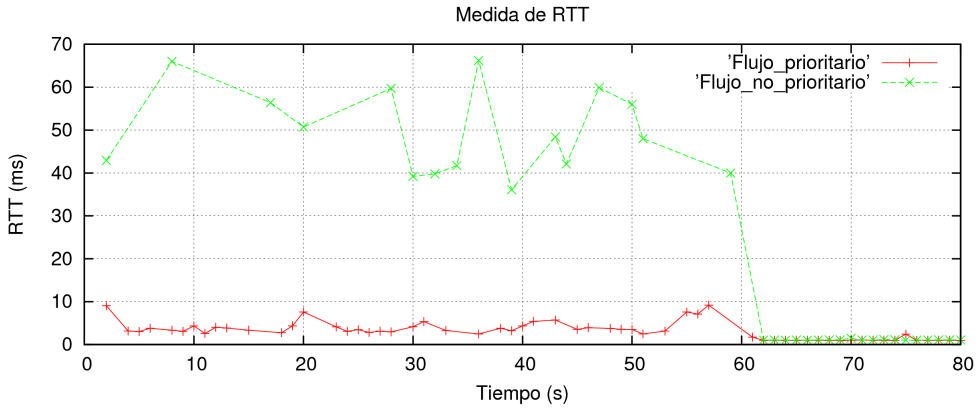

Figura 9.2: RTT obtenido con la configuración seleccionada

Observaciones: se verifica lo esperado a partir de las pruebas de laboratorio, observando tanto el impacto de la configuración de qdisc en el throughput como los efectos de 802.11e en el RTT de cada uno de los flujos.

9.1.2. Efecto de diversas configuraciones en el enlace

Con el objetivo de verificar los efectos de qdisc y 802.11e en el enlace, se procedió a realizar diversas pruebas de modo de verificar lo observado en las pruebas de laboratorio.

9.1.2.1. Configuración seleccionada

Como se presentará en la sección 9.2, si bien el enlace fue estable, presentó variaciones en su capacidad por lo que las siguientes pruebas se realizaron en un período reducido de forma de poder comparar cuantitativamente sus resultados.

Inicialmente se repitió la prueba con la configuración seleccionada con iguales resultados, como se observa en la gráfica a continuación. Esta vez con un tráfico prioritario se obtuvo un throughput de 7,35 Mbps, mientras que el no prioritario obtuvo 0,33 Mbps.

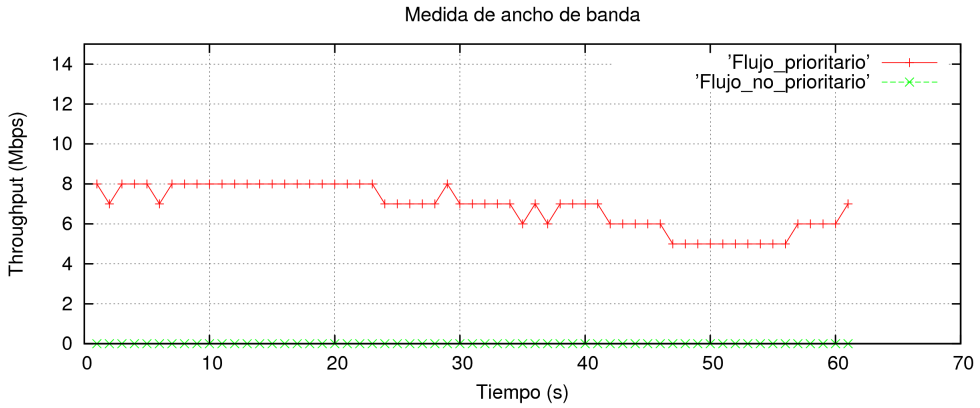

Figura 9.3: Bandwidth obtenido con la configuración seleccionada

En este caso el RTT fue de 3,5 ms para el flujo prioritario y de 43,3 ms para el no prioritario.

9.1.2.2. 802.11e sin qdisc

Se repitió la prueba anterior sin la configuración de qdisc y con 802.11e activado, buscando validar los resultados obtenidos en las pruebas de laboratorio.

Resultados: en la siguiente gráfica se observa como el flujo total de 7,54 Mbps se divide equitativamente entre el flujo prioritario que obtuvo 3,66 Mbps y el no prioritario que obtuvo 3,88 Mbps en promedio.

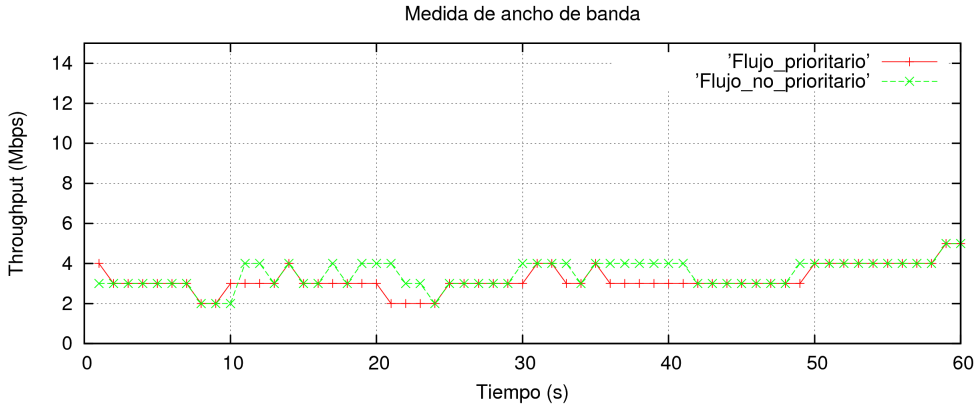

Figura 9.4: 802.11e sin qdisc

Al estar 802.11e activado sí se obtuvieron diferencias en RTT siendo de 16,49 ms el delay observado en el flujo prioritario y 302,53 ms el obtenido por el no prioritario.

Observaciones: se verifica lo esperado a partir de las pruebas de laboratorio, observando que una configuración sin qdisc no permite la división de throughput esperada pero 802.11e permite mejoras sustanciales en el RTT del flujo prioritario.

9.1.2.3. Qdisc sin 802.11e

La siguiente prueba se realizó con la configuración de qdisc que fue presentada anteriormente pero con 802.11e deshabilitado, buscando verificar lo obtenido en las pruebas de laboratorio.

Resultados: en la siguiente gráfica se observa como el flujo prioritario obtuvo la mayor parte del canal teniendo un throughput de 5.81 Mbps en promedio, frente a los 0,25 Mbps obtenidos por el flujo no prioritario.

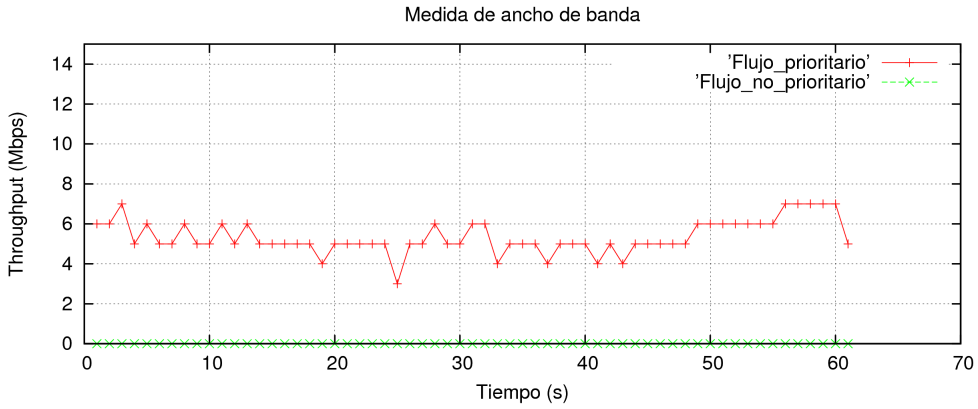

Figura 9.5: Qdisc sin 802.11e.

Al estar 802.11e desactivado esta vez los valores de RTT obtenidos en cada caso fueron similares, siendo de 217 ms en un caso y 218 ms en el otro.

Observaciones: se verifica lo esperado a partir de las pruebas de laboratorio. Se observa que una configuración con qdisc permite la división diferenciada del throughput pero no del RTT. Solamente con qdisc no se logró minimizar el RTT de la clase prioritaria y con los valores obtenidos no se lograría establecer una comunicación de voz sobre IP (ver sección 9.4.2).

9.1.2.4. Configuración sin qdisc ni 802.11e

Finalmente se evaluó el enlace sin qdisc ni 802.11e para poder estudiar efectivamente los efectos de aplicar la configuración obtenida. Así, al igual que en las pruebas anteriores, se enviaron 2 flujos UDP y se analizaron los resultados obtenidos a nivel de throughput y RTT.

Resultados: en la siguiente gráfica se observa como ambos flujos se dividieron el canal de igual forma obteniendo un throughput promedio de 5,34 Mbps en ambos casos.

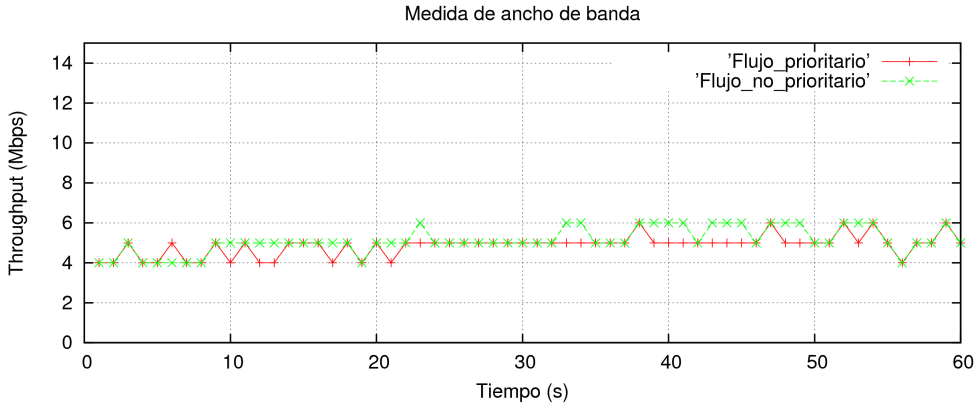

Figura 9.6: Configuración sin qdisc ni 802.11e.

Así como en la prueba anterior, los valores de RTT promedio en ambos casos fueron similares, siendo de 68 ms para uno de los flujos y 67 ms para el otro. En la siguiente gráfica se observan los resultados obtenidos.

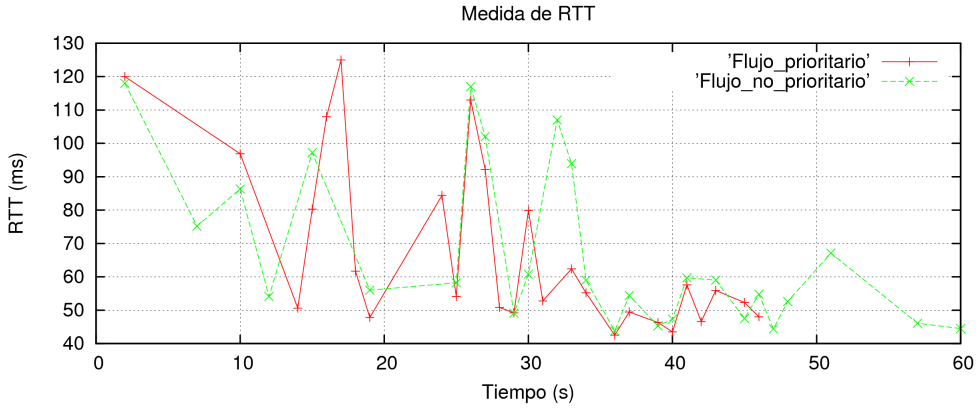

Figura 9.7: Configuración sin qdisc ni 802.11e.

Observaciones: como se esperaba, sin qdisc el throughput total se divide de igual forma entre ambos flujos y el RTT es también igual en ambos casos, considerablemente mayor al obtenido al activar 802.11e para la clase prioritaria.

Conclusiones: vemos que la configuración seleccionada tiene un impacto considerable tanto en el throughput obtenido por el tráfico prioritario como en su RTT. Si bien hay variaciones en la capacidad obtenida en las pruebas, se observa un aumento en el throughput obtenido por el tráfico prioritario pasando de 5,34 Mbps a 7,35 Mbps. En cuanto al RTT el impacto también es significativo pasando de 67 ms a 3,5 ms. Es importante destacar este resultado ya que sería posible mantener varias conversaciones de VoIP (Voice over IP), gracias a que el RTT se encuentra muy por debajo de los 150 ms necesarios para mantener una conversación aceptable.

9.2. Estabilidad del enlace

Durante el estudio del enlace se observó una variación en el throughput medido a lo largo del día, obteniendo resultados menores durante las horas en las que la temperatura que alcanzaban los equipos era mayor. Para verificar este comportamiento se realizaron medidas de throughput en la mañana, mediodía, tarde y noche durante 8 días consecutivos registrando los valores que se presentan en la siguiente gráfica.

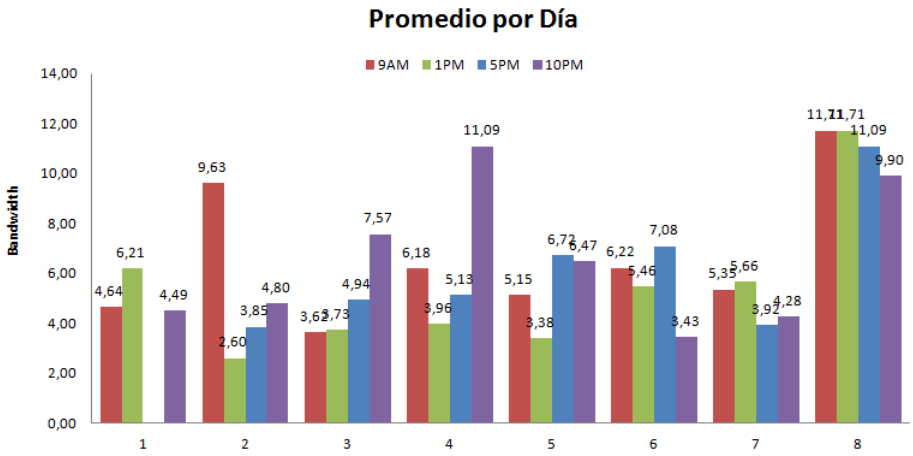

Figura 9.8: Promedios diarios según hora a la que se realizaron las pruebas

Como se observa, los throughputs promedio medidos tanto en la mañana como en la noche son similares. Hay en general un descenso considerable en las primeras horas de la tarde y luego una mejora paulatina hacia la noche.

Los resultados promedios totales fueron de 6,56 Mbps y 6,50 Mbps para la mañana y noche respectivamente. Para las pruebas realizadas a la 1PM, el valor promedio fue de 4,21Mbps y se observa una mejora a las 5 PM con un promedio de 6,11 Mbps[1].

También se puede observar que, si bien la capacidad del enlace no fue constante, presentando variaciones de hasta 7 Mbps en un mismo día, los valores mínimos observados son aceptables. Por otro lado se registraron picos con tasas de transferencia de 11 Mbps que, como se observa en la gráfica para el día número 8, se mantuvieron constantes por períodos prolongados.

Durante el tiempo en el que se ha evaluado el enlace éste siempre ha estado activo y los problemas enfrentados han sido principalmente debido a problemas eléctricos en los equipos de borde. Por otro lado, luego de 3 meses de establecido el enlace, las tasas de transferencia se han mantenido en los niveles apreciados en la gráfica por lo que no se sospechan problemas en el alineado de las antenas.

[1]El primer día no hay registros a las 5 PM debido a un problema eléctrico en San Gabriel.

9.2.1. Nivel de potencia en recepción

Del mismo modo que en la sección anterior, se estudió la estabilidad del enlace a partir del nivel de potencia en recepción para 60 días consecutivos. Como se observa en la siguiente gráfica estos valores se mantuvieron estables con un promedio de -70,38 dBm, presentando un mínimo de -76,5 dBm. Considerando valores conservadores de ruido de -97 dBm y de SNR en recepción de al menos 15 dB, como se detalló en la sección 7.4, se constata que la potencia recibida siempre se mantuvo en los rangos esperados. También se cumple que los niveles de potencia se mantienen por encima de la sensibilidad de la tarjeta de red especificados en -90 dBm [25] para las condiciones de uso del enlace.

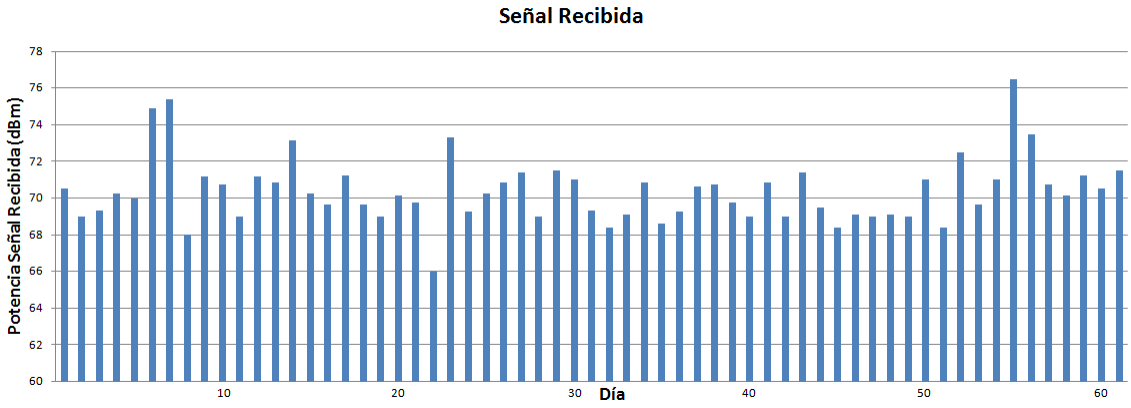

Figura 9.9: Potencia de señal recibida en valor absoluto.

9.3. Comparación con resultados de laboratorio

A partir de los resultados que se han presentado tanto en el presente capítulo como en las pruebas realizadas en un ambiente controlado, se puede observar que el comportamiento del enlace en larga distancia es similar al estudiado en laboratorio. Si bien el throughput obtenido es sensiblemente inferior se mantiene en valores aceptables para los objetivos del enlace.

También se ha verificado el comportamiento esperado a partir de las pruebas de laboratorio para los casos sin 802.11e y qdisc, así como los efectos de incorporar su uso de forma independiente.

9.4. Análisis con tráfico TCP

Previo a analizar las pruebas realizadas, se detalla algunos aspectos sobre las exigencias de throughput y delay de la voz sobre IP.

9.4.1. Ancho de banda de voz sobre IP

Cuando se establece una conversación sobre redes de datos, la voz viaja en "paquetes". Si bien la comunicación es bidireccional, se hará un breve análisis para un solo sentido ya que el resultado total se obtiene multiplicando por 2 (existen además técnicas de supresión de silencios) [26].

Los bytes de voz por trama se pueden calcular sabiendo el codec utilizado y la ventana de la trama. Si a este valor le sumamos los bytes del encabezado UDP (8 bytes), encabezado RTP (12 bytes), encabezado IP (20 bytes), encabezado MAC (28 bytes) y cabecera PLCP (24 bytes) obtenemos el tamaño total del paquete de voz [8].

Por ejemplo, si se utiliza un codec G711u con tasa de 64 Kbps y una ventana de 20 ms se obtiene que el tamaño del paquete es de 160 bytes. Sumando los encabezados mencionados previamente se obtienen 252 bytes [8]. Finalmente el ancho de banda requerido para un sentido de la comunicación es 101 Kbps.

9.4.2. Retardo o latencia de voz sobre IP

Existen restricciones al retardo de paquetes en una red para poder establecer comunicaciones de voz. Por encima de los 100 ms el retardo comienza a ser notorio, aceptándose generalmente hasta 150 ms. Retardos de 300 ms o más se consideran inaceptables para voz.

9.4.3. Ensayo con tráfico TCP y UDP

Con la configuración de qdisc y WMM utilizada se realizó un último análisis, comparando el throughput obtenido por tráfico TCP y UDP simultáneos. Como se mencionó previamente, las aplicaciones de voz y video tienen mayores exigencias en cuanto a ancho de banda y RTT. Generalmente el tráfico de navegación web, correo electrónico o descarga de archivos es TCP y las aplicaciones en tiempo real flujos UDP. TCP se adapta a las disponibilidades o propiedades del canal ya que fue diseñado para transmitir confiablemente un flujo de datos de extremo a extremo [27].

Se configuró un tráfico no prioritario TCP y uno prioritario UDP. La prueba consistió en enviar flujos UDP de mayor tamaño a medida que aumentaba el tiempo, espaciados entre sí, mientras había tráfico TCP en el canal. Se utilizó un tamaño de paquetes UDP similar al utilizado por la voz codificada con el codec G.711u, en las condiciones mencionadas en 9.4.1. En la siguiente gráfica se puede ver los resultados obtenidos para el throughput.

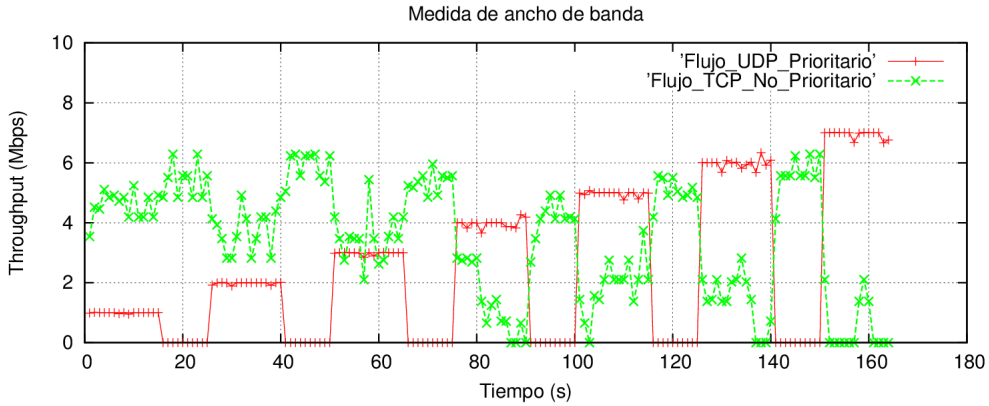

Figura 9.10: Throughput flujo TCP y UDP

Observaciones: todo el flujo UDP generado pudo ser enviado a tasas constantes. En cambio TCP varió dependiendo del tráfico en el canal.

Estos resultados permitirían afirmar que dado un flujo UDP (e.g. flujo de voz) y la configuración utilizada para el enlace inalámbrico, se podrían asegurar un número X de conversaciones simultáneas dependiendo de la calidad de voz deseada.

A continuación se presenta los resultados obtenidos para el retardo:

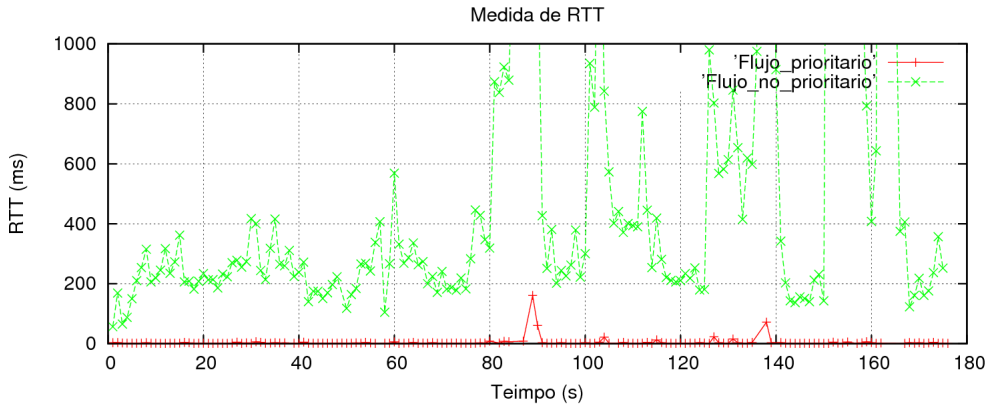

Figura 9.11: Retardo flujo TCP y UDP

Observaciones: el retardo obtenido para la clase prioritaria es muy bajo. Esta situación es óptima para aplicaciones de voz o en tiempo real.

9.4.4. Ensayo con tráfico TCP y UDP sin qdisc ni WMM

Sin realizar todas las pruebas para qdisc y 802.11e habilitado y deshabilitado, se agrega esta sección de forma de comparar los resultados obtenidos utilizando qdisc y WMM, con los que se obtendrían con la configuración estándar de OpenWrt (sin QoS).

A continuación se grafica el throughput obtenido:

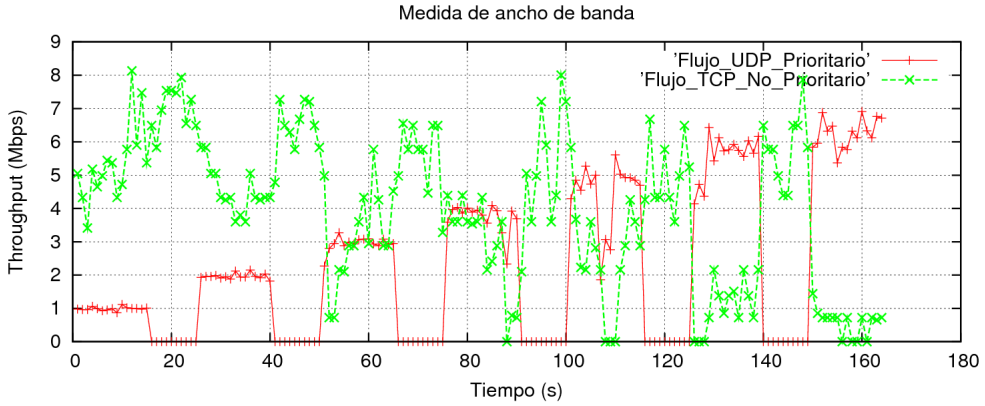

Figura 9.12: Throughput flujo TCP y UDP sin QoS

En cuanto al retardo:

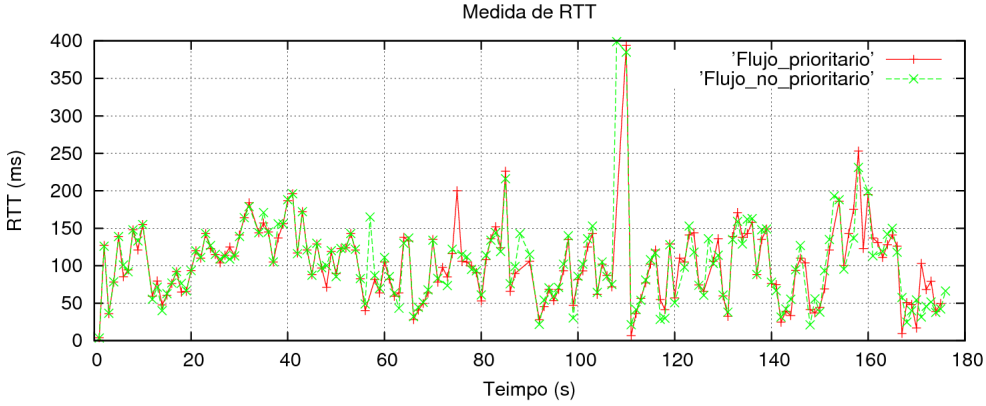

Figura 9.13: Retardo para TCP y UDP sin QoS

Observaciones: si bien existen unas pequeñas diferencias en cuanto al throughput obtenido, las ventajas de la configuración seleccionada se manifiestan claramente en el retardo.

9.5. Comentarios finales sobre el enlace

Como se mencionó en la sección 9.2 no hubo períodos apreciables de no disponibilidad del enlace durante el tiempo en el que se evaluó. Se registraron fallas en los equipos de borde lo cual, dada la lejanía existente al lugar donde se implementó el enlace y dependencias para acceder a estos equipos, generaron dificultades en algunos momentos. Sin embargo durante estos períodos, si bien no se pudieron estudiar las características del enlace, sí se pudo verificar que el mismo seguía activo.

Parte IV

Conclusiones

Capítulo 10

Conclusiones

El presente proyecto consistió en el estudio e implementación de un enlace de larga distancia utilizando WiFi, así como el uso tanto de qdisc como de 802.11e para proveer calidad de servicio sobre el mismo. Durante las distintas etapas de este proyecto se investigó el estándar 802.11, cubriéndose también el uso de qdisc, manejo de tráfico en Linux, iptables, scripting en Shell y manejo de aplicaciones para generación de tráfico. El estudio teórico, cuyos puntos más importantes se encuentran en las secciones anexas, permitió una adecuada selección de hardware y software que llevó al cumplimiento de los objetivos previamente planteados, que se repasarán a continuación.

Se pudo establecer un enlace de 30 km con buenas tasas de transferencia para el uso esperado. A su vez, se observó durante los 3 meses de monitoreo una muy buena estabilidad, teniendo en el período de evaluación presentado en 9.2 un throughput de 5,85 Mbps de promedio, encontrándose por encima de los 3 Mbps requeridos en un 96,8 % de las pruebas.

El siguiente gran objetivo del proyecto fue proveer calidad de servicio sobre el enlace. El estudio del estándar 802.11e permitió comprender el funcionamiento del protocolo a nivel de capa 2, pero al momento de implementar la solución se identificaron los problemas presentados en el capítulo 6, llevando a incorporar un complemento a nivel de capa 3, realizado mediante qdisc. La solución final permite dividir flujos de la forma deseada teniendo impacto tanto a nivel de throughput como a nivel de RTT, aspecto fundamental en aplicaciones de tiempo real. Los resultados han sido muy satisfactorios permitiendo aumentar el throughput del flujo prioritario y reducir considerablemente el RTT, como se presentó en la sección 9.1.2.

Los resultados obtenidos en el capítulo 9 permiten observar los efectos de la solución realizada y la trascendencia de los mismos especialmente en comunicaciones en tiempo real. Se destaca por ejemplo el impacto en comunicaciones mediante voz sobre IP, donde un RTT mayor a 100 ms provocaría problemas notorios en las llamadas. La solución planteada permite bajar estos

tiempos de valores en torno a los 110 ms a 2 ms en promedio, para el flujo prioritario.

A nivel académico se destaca una buena planificación que pudo adaptarse a los problemas identificados durante el proyecto, permitiendo el cumplimiento de los objetivos planteados en los plazos previstos. También hubo una correcta interacción y colaboración con los distintos interesados en el proyecto, destacando la coordinación con el Plan Ceibal, tanto en el desarrollo de la solución como en la implementación de la misma.

Como conclusión general del proyecto se verifica WiFi como una solución real y conveniente para realizar enlaces de larga distancia en el contexto planteado y se presenta una solución conjunta de qdisc y 802.11e para la implementación de calidad de servicio sobre el mismo.

Capítulo 11

Trabajos a futuro y líneas de investigación

El presente trabajo se basó en el estudio de enlaces de larga distancia y el uso de calidad de servicio sobre el mismo. A partir de estas áreas se destacan 3 principales focos de análisis futuro.

Una de estas áreas es el comportamiento de la solución identificada para calidad de servicio con flujos generados por aplicaciones reales. Si bien las pruebas fueron realizadas estudiando el comportamiento frente a flujos UDP, sobre el que se basan las aplicaciones de tiempo real, es relevante analizar la calidad con la que se desarrollarían actividades como videoconferencias, aulas virtuales o telemedicina en contextos de saturación frente a distintos tipos de tráfico.

Por otro lado las dificultades identificadas al trabajar con 802.11e generan una nueva área de investigación, con el fin de resolver los inconvenientes generados por el despachador de paquetes que sirve a las colas en capa 2. Otros trabajos presentan experiencias dividiendo flujos a nivel de throughput desde un mismo equipo con 802.11e, por lo que sería de interés poder resolver este problema para los equipos utilizados.

Finalmente se considera interesante la investigación de incorporar múltiples saltos que permitan realizar comunicaciones de mayor distancia utilizando repetidores y eventualmente evaluar QoS en este escenario.

Parte V

Anexos

Apéndice A

El estándar 802.11

En este apartado se realiza una descripción de los aspectos generales más importantes del estándar 802.11. Sirve a modo de guía o resumen, no pretendiéndose sustituir la lectura del mismo.

Luego de su primera publicación, la norma sufrió varias extensiones con el fin de realizar modificaciones y mejoras. Particularmente, en la versión original, se especificaban 2 velocidades teóricas de transmisión de 1 y 2 Mbps y hoy en día se habla de 600 Mbps teóricos en 802.11n.

A.1. Descripción

A continuación se detalla los objetivos del estándar 802.11 [28]:

- Describe funciones y servicios requeridos por un dispositivo que cumple la norma 802.11, ya sea para trabajar en redes ad-hoc como en redes en modo Infraestructura, así como aspectos para la movilidad dentro de estas redes.
- Define los procedimientos de la subcapa MAC para soportar la entrega asíncrona de datos, MSDU (*MAC service Data Unit*).
- Define diversas formas de señalización de la capa física (PHY) y funciones controladas por la subcapa MAC.
- Permite la coexistencia de redes inalámbricas de área local (WLAN) 802.11 con otras WLANs que pueden llegar a estar superpuestas.
- Describe los requerimientos y procedimientos para brindar confiabilidad y seguridad en el transporte de los datos del usuario a través del medio inalámbrico (WM) y autenticación de dispositivos cumpliendo la norma 802.11.
- Define los mecanismos para selección dinámica de frecuencia (DFS) y control de potencia de transmisión (TPC) que pueden ser utilizados para cumplir las regulaciones locales.

- Define los procedimientos de la subcapa MAC para soportar aplicaciones de redes de área local (LAN) con requerimientos de calidad de servicio (QoS), incluyendo transporte de voz, audio y video.

A.2. Arquitectura de capas

El estándar 802.11 define el uso de las 2 capas inferiores del modelo OSI, poniendo especial énfasis en la capa física PHY y la subcapa MAC de la capa de enlace de datos.

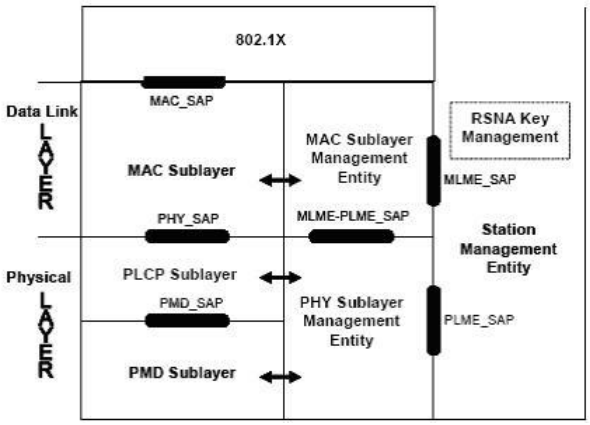

Figura A.1: Modelo de referencia.

La subcapa MAC se encarga del control del acceso al medio inalámbrico, utiliza los servicios de la capa inferior (PHY) y se encarga de evitar colisiones al acceder al WM (medio compartido). La capa PHY, como se observa en la figura A.1, se divide en la subcapa de convergencia PLCP , encargada de adaptar las tramas MAC agregándole campos de sincronismo, velocidad de transmisión y delimitadores de tramas y la subcapa PMD, encargada de la comunicación entre distintas estaciones. Esta última depende del medio físico, encargándose de enviar y recibir datos mientras que la subcapa PLCP permite operar a la subcapa MAC, teniendo una mínima dependencia con la subcapa PMD.

A.3. La capa PHY

El estándar 802.11 define distintos tipos de capa PHY. Cada PHY cumple básicamente 2 funciones:

1. Una función de convergencia que formatea los datos recibidos de la subcapa MAC de forma que la subcapa PMD pueda transmitirlos. Esta función es realizada por la subcapa PLCP.

2. La función PMD que define la forma y las características de transmisión a través del medio inalámbrico.

A continuación se detallan los 3 posibles tipos de capa PHY:

1. DFIR: Tecnología de Infrarrojos (*Diffuse Infrared*) Los sistemas de infrarrojos se sitúan en altas frecuencias y tienen las mismas propiedades que la luz visible (son reflejados en ciertas superficies, no atraviesan objetos opacos). No es demasiado usada a nivel comercial.

2. DSSS: Espectro ensanchado por Secuencia Directa (*Direct Sequence Spread Spectrum*): Consiste en la generación de un código redundante para cada uno de los bits y la posterior modulación de la señal resultante mediante una portadora de RF. [29]

3. FHSS: Espectro ensanchado por salto en frecuencia (*Frequency Hopping Spread Spectrum*): Consiste en transmitir parte de la información en una determinada frecuencia durante un intervalo de tiempo (*dwell time*) y luego se cambia a una nueva frecuencia de emisión.[29]

Las tecnologías de espectro ensanchado difunden los datos a lo largo del ancho de banda disponible en vez de concentrar la energía alrededor de una portadora. El ancho de banda es compartido con el resto de los usuarios que trabajen en la banda.

A.4. La subcapa MAC

El estándar 802.11 define principalmente 2 modos de funcionamiento en la capa MAC, uno distribuido (DCF) y otro centralizado (PCF). Ambos modos de funcionamiento pueden usarse alternativamente de manera que a cada período de contienda con DCF siga un período de polling con PCF. PCF es opcional y ha sido raramente implementado. En la siguiente figura se expone la estructura descrita. A partir de la publicación de 802.11e se agrega una tercera función de coordinación llamada HCF que se describirá más adelante.

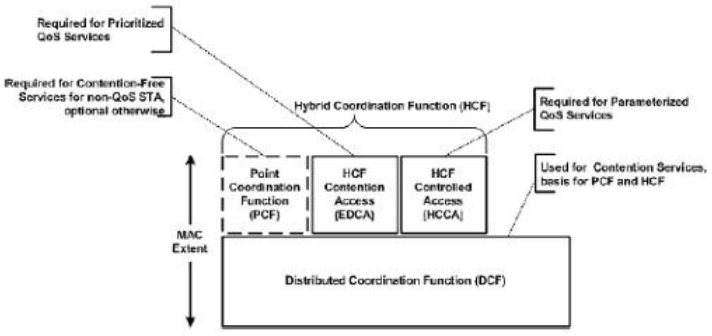

Figura A.2: Arquitectura de la subcapa MAC

A.4.1. IFS

A continuación se define los tipos de intervalos de tiempo entre tramas que utiliza el estándar 802.11. Los distintos tiempos son definidos para dar diferentes niveles de prioridad de acceso al medio inalámbrico.

- SIFS: es el mínimo tiempo de espera antes de transmitir cualquier trama. Se utiliza para los ACK, CTS, para alguna estación respondiendo un polling de PCF o para una estación que ganó el canal y necesita mantenerlo.

- PIFS (SIFS + aSlotTime): se utiliza en modo PCF para dar prioridad al inicio de un CFP (*Contention free period*).

- DIFS (PIFS + aSlotTime): se utiliza en DCF para la transmisión de tramas de datos (MPDUs) y tramas de control (MMPDUs).

- AIFS: se utiliza cuando se emplea calidad de servicio (QoS). Se detalla en el capítulo A.6.

- EIFS (aSIFSTime + DIFS + ACKTxTime): es el tiempo más largo. Se utiliza cuando una estación recibió una trama que no puede entender, de forma de evitar colisiones con una trama futura del diálogo actual.[10]

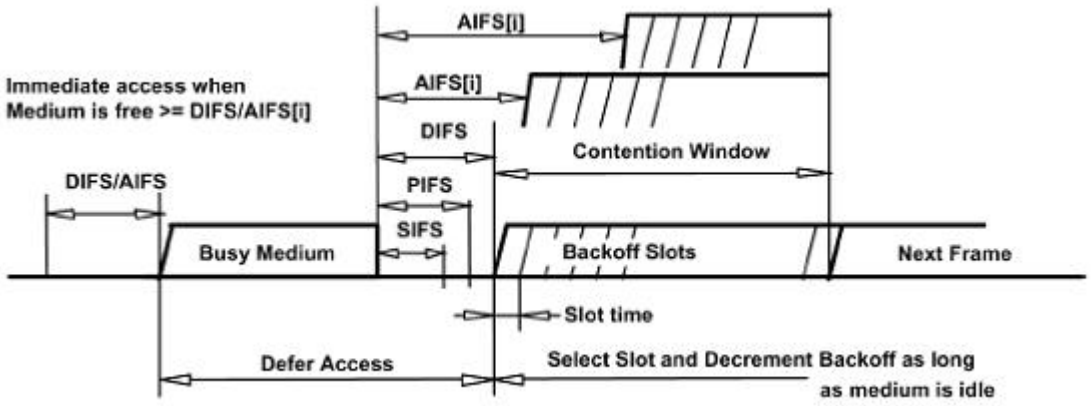

Figura A.3: Relación de distintos IFS.

A.4.2. DCF

DCF, Distributed Coordination Function, accede al medio utilizando el protocolo CSMA/CA (*carrier sense multiple access with collision avoidance*) a diferencia de 802.3 (Ethernet) que utiliza CSMA/CD (*collision detection*). La diferencia es que uno evita las colisiones censando el medio (CSMA/CA) y el otro las detecta luego de ocurridas (CSMA/CD).

Existen 2 formas de censar el canal, una real (a través de PHY), donde efectivamente se compara el nivel de energía detectada, y otra virtual (a través de MAC) donde se emplea un contador llamado NAV (*Network Allocator Vector*). El uso de mensajes de solicitud y confirmación de uso del canal (donde se indica el tiempo que se va a ocupar el mismo) permite el manejo de este contador.

Asimismo, para aumentar la fiabilidad y detectar posible colisiones, se necesita que todo envío de información sea seguido de un ACK positivo (acknowledgment) de confirmación por parte del receptor. Si la fuente no recibe esta trama en un tiempo identificado como *ACKTimeout*, se realiza una retransmisión del mensaje.

CSMA/CA:

Una estación que quiere transmitir censa el medio. Si el mismo está disponible, espera un tiempo DIFS (*DCF Interframe Space*) y si luego de ese tiempo el canal sigue libre, transmite. Cuando una estación encuentra el medio ocupado vuelve a censar el canal luego de un tiempo de backoff aleatorio.

Backoff:

El algoritmo de *backoff* es el empleado para que distintas estaciones que quieran acceder al medio al mismo tiempo no colisionen una y otra vez. Si una estación encuentra el medio ocupado, selecciona un tiempo aleatorio de espera que será de entre 0 y CW (Contention Window) slot times. Durante ese período censa el medio en cada SlotTime y cuando el mismo se encuentra libre disminuye el temporizador. Una vez que el temporizador llega a 0 la estación transmite. Al ser distintos los tiempos para cada una de las estaciones, la que tenga un menor tiempo de backoff es la primera que va a transmitir.
Si cuando una estación está censando el medio éste se ocupa, la misma detiene el contador y espera a que se libere un tiempo DIFS o EIFS (según el caso) para continuar disminuyendo el temporizador.

A continuación se muestra una figura donde varias estaciones están compitiendo por el medio.

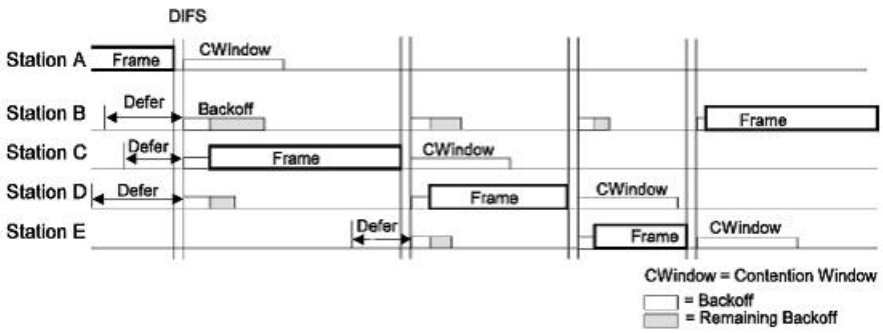

Figura A.4: Procedimiento *Backoff*.

$$\textbf{BackoffTime} = \textbf{Random()} \times \textbf{aSlotTime}$$

Random(): Número entero pseudo aleatorio en el intervalo [O,CW]
aSlotTime: valor dado en la capa PHY

RTS/CTS:

En el modo DCF se prevé la posibilidad que suceda el problema de nodo oculto. 2 estaciones pueden no verse entre sí pero quieren transmitir a una intermedia. Cuando una está transmitiendo, la otra no detecta el canal ocupado. Para evitar colisiones se define el mecanismo RTS/CTS (*Request to Send / Clear to Send*). El funcionamiento es el siguiente: una estación que desea transmitir envía un RTS y la estación receptora responde con un CTS si está disponible para aceptar los paquetes. RTS y CTS incluyen información de cuánto tiempo va a durar la transmisión por lo que todo equipo que escuche estos mensajes actualizará el NAV y determinará el canal ocupado durante ese período. De esta forma se evita el problema de nodo oculto, ya que el CTS incluye el tiempo que el canal va a estar ocupado. Durante el período NAV el dispositivo no censa el canal porque ya sabe que está ocupado, por eso mismo se dice que el CS (*Carrier Sense*) es virtual.

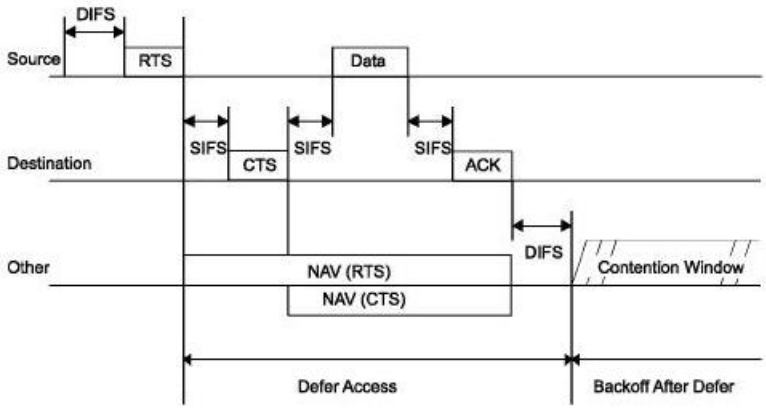

Figura A.5: Procedimiento RTS/CTS

Procedimiento ACK:

La trama ACK debe ser enviada por la estación receptora una vez recibida la trama de datos, si la misma requiere reconocimiento. Luego de recibir correctamente la trama, la estación receptora espera SIFS y envía la trama ACK.

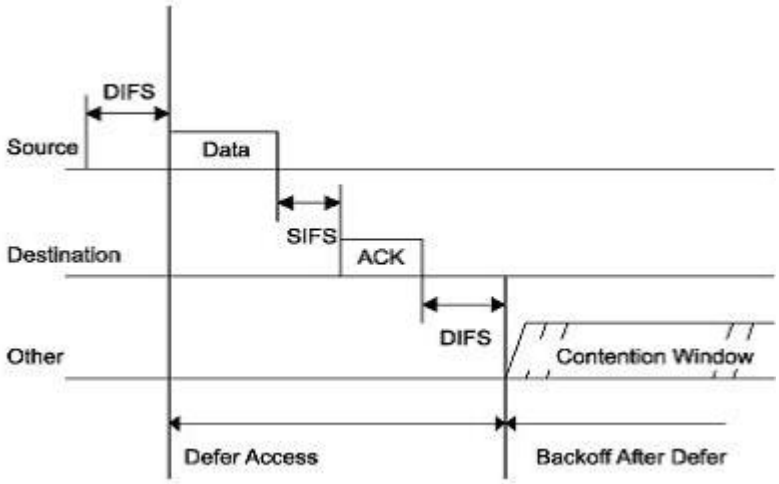

Figura A.6: Procedimiento ACK

A.4.3. PCF

PCF se utiliza en el modo infraestructura donde existe un punto coordinador (PC), que reside en el AP (*Access Point*) y determina qué estación tiene derecho a transmitir, luego de realizar un sondeo. Todas las estaciones asociadas deben respetar las reglas de PCF para transmitir y determinan su NAV en el comienzo del CFP (*Contention-Free Period*). Durante el CFP, el PC

es quién gana el acceso al medio ya que espera un tiempo PIFS (menor a DIFS, que es lo que esperan las estaciones) y es quien controla el orden de las transmisiones.

El protocolo de transferencia CF (*Contention Free*) se basa en un mecanismo de "polling", realizado por el PC, donde el mismo envía tramas "beacon" periódicamente, difundiendo su presencia. Todas las estaciones son informadas del CFP y deben actualizar su NAV en consecuencia. No se entra en detalle del funcionamiento de PCF. Por más información ver el estándar A.

A.5. Tramas

Cada trama contiene los siguientes componentes básicos:

- Una cabecera MAC, que comprende campos de control, duración, direccionamiento y control de secuencia. Para tramas de datos con QoS también se incluye tramas de control de QoS.
- Un cuerpo de trama de longitud variable, que contiene información específica del tipo y subtipo de trama.
- Una secuencia "checksum" (FCS) que contiene un IEEE 32-bit CRC.

Las tramas MAC se clasifican según 3 tipos:

- Tramas de control: por ejemplo, ACK, RTS y CTS, PS-Poll, etc.
- Tramas de datos.
- Tramas de gestión: por ejemplo, tramas Beacon, asociación, disociación, reasociación, autenticación, etc.

A.6. El estándar 802.11e

El estándar de 802.11e ha definido mejoras del estándar original 802.11 MAC(Medium Access Control) para proveer Calidad de Servicio (QoS). Introduce una nueva función de coordinación llamada HCF *(Hybrid Coordination Function)*, la cual combina las formas de funcionamiento de DCF y PCF.

A.6.1. HCF

Hybrid Coordination Function tiene 2 modos de acceso al canal:

- EDCA (Enhanced Distributed Channel Access) (ver sección 2.3.1)
- HCCA (HCF Controlled Channel Access) (ver estándar)

A.6.2. Traffic Specifications (TSPECs)

El estándar 802.11e específica el uso TPSECs como dispositivo de gestión de flujos de tráfico que proporciona al enlace una gestión ente protocolos de capas superiores, tales como *Intserv* y *Diffserv*, con las funciones de acceso al canal de 802.11e. Esta especificación describe las características de los flujos de tráfico, tales como el tamaño de los paquetes, el caudal o el retardo. La negociación TSPEC proporciona un mecanismo para el control de la admisión, establecimiento, ajuste y eliminación de flujos de tráfico [1].

A.6.3. 802.11e MAC Enhancements

Algunas de las mejoras que aparecen a nivel de subcapa MAC son [30]:

1. *Contention Free Bursts:* Permite que las estaciones QSTA/QAP puedan enviar las tramas sin tener que esperar los tiempos de contienda una y otra vez. La estación QSTA continúa transmitiendo después de un SIFS, siempre y cuando tenga tiempo garantizado por el TXOP. El método de *bursting* puede mejorar la performance significativamente ya que los overheads asociados a DIFS y backoff son reducidos.

2. *New Acknowledgment Rules:* En el estándar de 802.11 todas las tramas de datos requieren un ACK inmediato. HCF añade dos nuevas opciones para especificar las tramas de control con QoS.

   - No ACK, aumenta la eficiencia enviando un no ACK para determinadas aplicaciones. En especial puede ser útil para aplicaciones con una tolerancia muy baja de latencia, pero puedan tolerar pérdidas de paquetes (p.e. voz sobre IP)

   - Block ACK, aumenta la eficiencia mediante la agregación de ACKs múltiples en una sola respuesta. Los Block ACK se pueden distinguir en 2 tipos: *inmediatos* y *retrasados*.

3. *Direct Link Protocol:* La especificación estándar de 802.11 permitía el tráfico en una red, entre las estaciones y el AP *(Access Point)*, pero con el uso de DLP *(Direct Link Protocol)* el tráfico puede ser mandado directamente a otra estación sin tener que pasar por el AP. Esta capacidad incrementa el ancho de banda de comunicación entre 2 estaciones que estén en un mismo rango. DLP también podría incrementar la performance en el caso de que dos estaciones estén más cerca entre sí, que el AP.

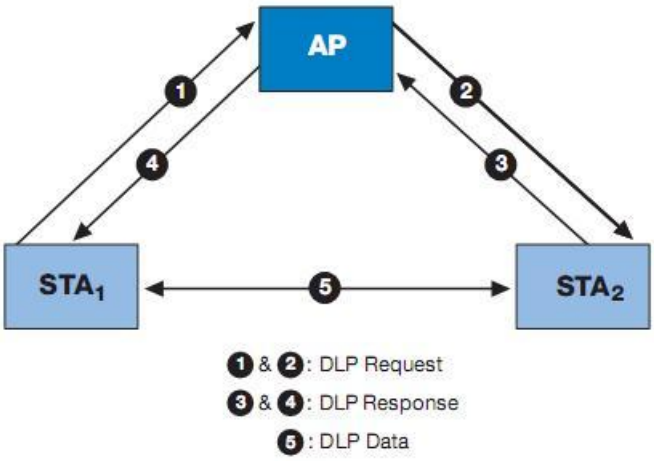

Figura A.7: Direct Link Protocol

4. *Piggybacking:* Reduce el overhead de polling y ACK mandando el dato *"piggybacked"*.

5. *Automatic Power-Save Delivery:* Permite ampliar la duración de la batería de los dispositivos permitiendo apagar sus radios durante la mayor parte del tiempo. Es una mejora al ya existente mecanismo de 802.11 ahorro de energía (*power-saving*). APSD permite a una estación la creación de un Calendario para la entrega de tramas, basada en una repetición de un patrón de un número determinado de intervalos de beacons. Esta característica sería apropiada para aplicaciones de VoIP, donde los paquetes son transmitidos en intervalos regulares.

Apéndice B

Instalación de OpenWRT

Los Mikrotik 433AH vienen con el sistema operativo propietario RouterOS. Se compiló e instaló la última versión de OpenWRT disponible al momento (Backfire). Para ello se utilizó la versión 10.10 de Ubuntu disponible en la página oficial. A continuación se detalla todos los pasos necesarios para la instalación a partir de la versión de Ubuntu utilizada.

B.1. Paquetes a instalar en Ubuntu

La versión básica de Ubuntu 10.10 no contiene todos los paquetes necesarios para compilar e instalar OpenWRT en los Mikrotik. A continuación se detalla los paquetes que se precisa instalar:

- subversion
- putty
- tftp
- tftpd
- dnsmasq
- apache2
- g++
- libncursesw5-dbg, libncurses-ruby1.9.1, libncurses5-dev
- zlib1g-dbg, zlib1g-dev
- flex
- gawk

B.2. Descargar Backfire y paquetes adicionales

Primero se debe bajar la última versión de Backfire. Para ello se utiliza el siguiente comando en una consola de texto:

→ svn co svn://svn.openwrt.org/openwrt/trunk

Luego se agrega los paquetes deseados que no están incluidos en el trunk. Dentro de la carpeta trunk, ejecutar:

→ ./scripts/feeds update

Al finalizar se ve la pantalla siguiente:

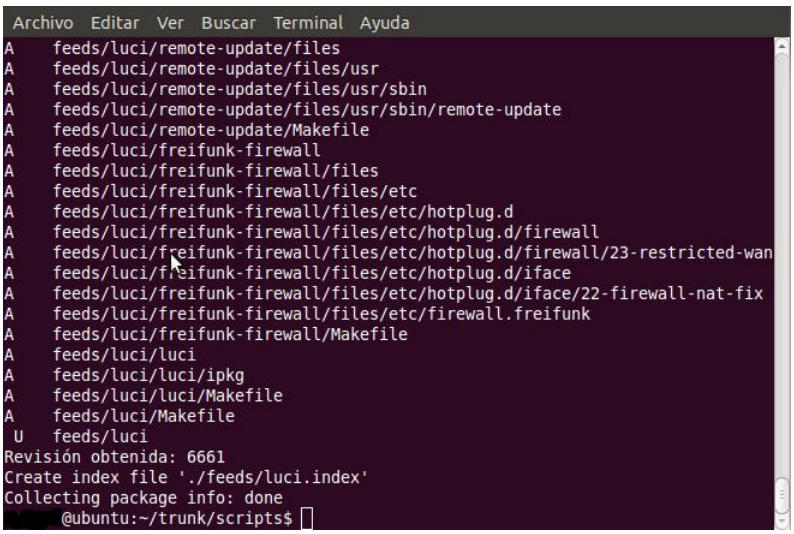

Figura B.1: Update paquetes OpenWRT.

Si se desea buscar un paquete para ver si está disponible:

→ ./scripts/feeds search *nombre paquete* (p.e. snmp)

Luego para instalarlo:

→ ./scripts/feeds install *nombre paquete*

Se instalaron los siguientes paquetes que no estaban en el trunk:

- snmpd
- snmp-utils
- snmp-static

- webif

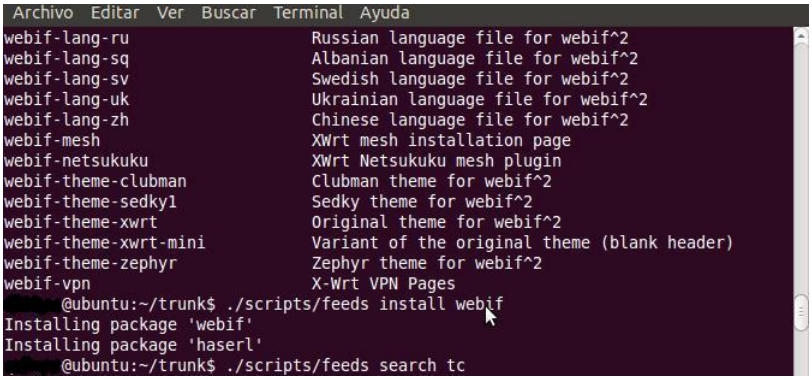

Figura B.2: Webif instalado.

- tc

- wshaper

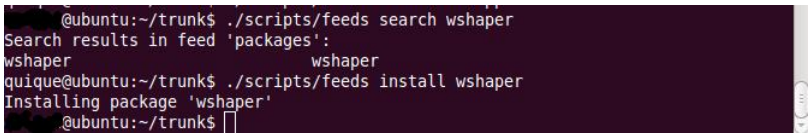

Figura B.3: Wshaper instalado

B.3. Compilación

Previo a compilar el OpenWRT, se debe modificar un archivo para aumentar la partición del kernel. Es probable que en la últimas versiones ya esté solucionado y no sea necesario.
En el directorio donde se instaló el trunk, modificar el archivo:

trunk/target/linux/ar71xx/files/drivers/mtd/nand/rb4xx.nand.c

Modificar la línea:

.size = (4 * 1024 * 1024) - (256 * 1024)

cambiando "4" por "8", de forma que quede:

.size = (8 * 1024 * 1024) - (256 * 1024)

B.3.1. Selección de paquetes

Ejecutar en una consola (sin ser usuario root):

→ make menuconfig

y aparecerá un menú como el de la siguiente figura:

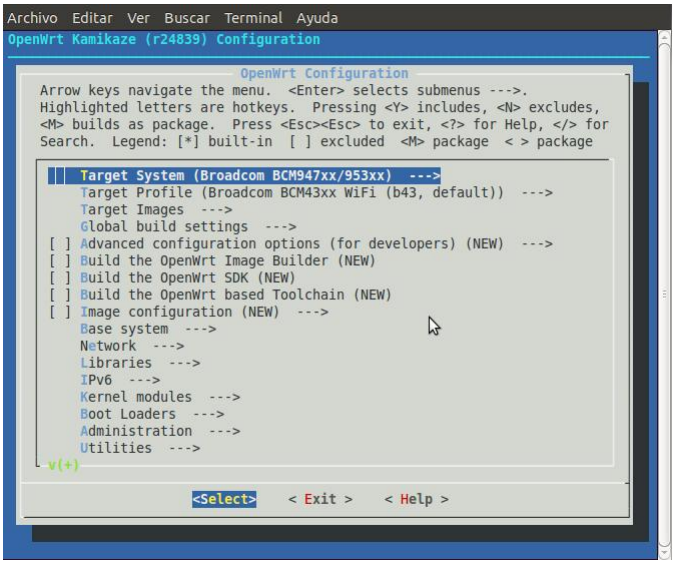

Figura B.4: Menú instalación OpenWRT.

El primer paso a realizar es cambiar el "Target System" seleccionando *ATHEROS AR71XX/AR7240/AR913x*:

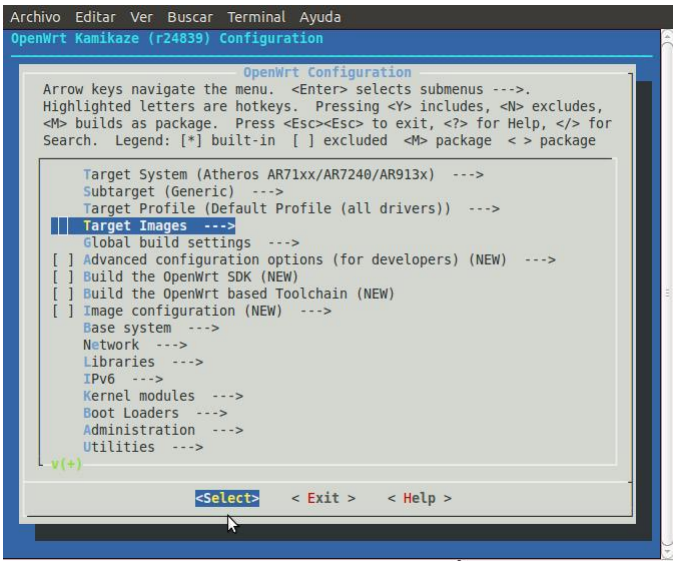

Figura B.5: Menú instalación OpenWRT.

Luego se debe seleccionar los paquetes adicionales a compilar.

Dentro del menú "Base System" marcar:

- qos-scripts

Dentro del menú "Network":

- ebtables
- snmp-utils
- snmpd
- snmpd-static

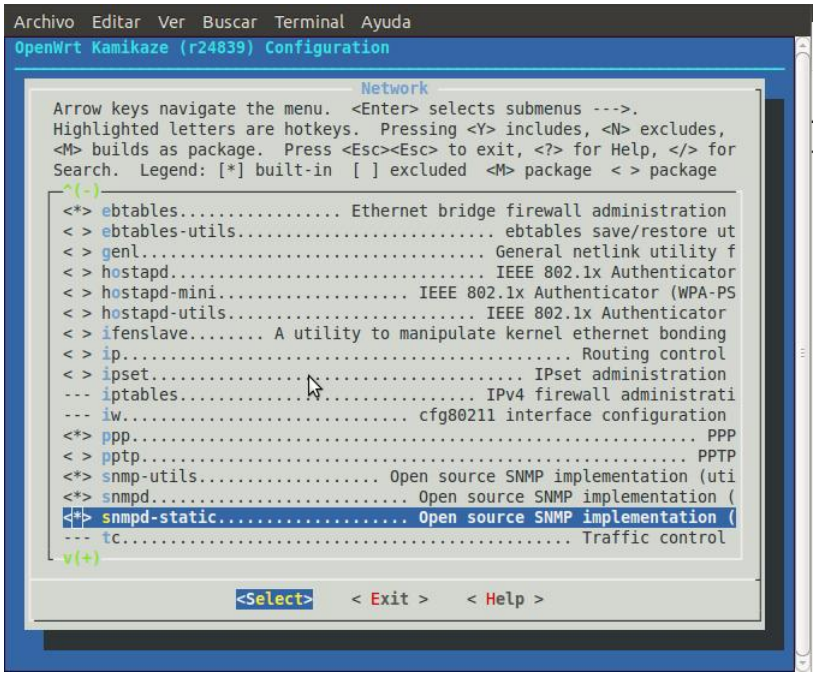

Figura B.6: Submenú Network.

- wshaper

Se guarda la configuración y se sale.

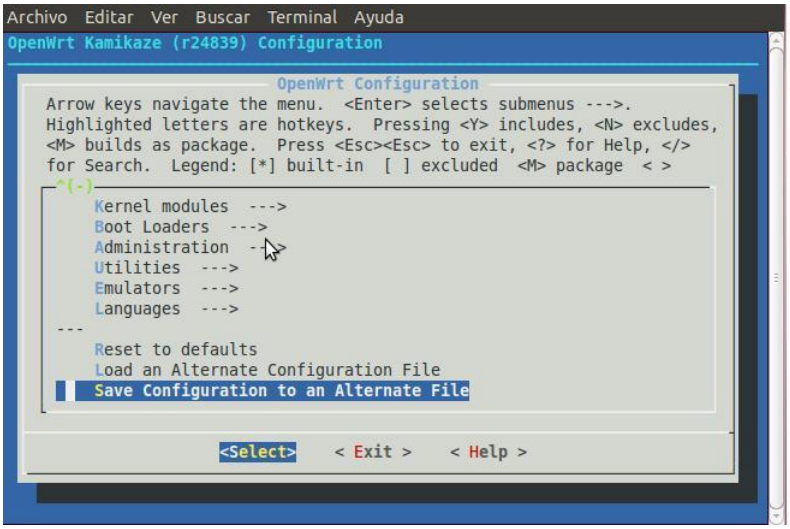

Figura B.7: Guardar y salir menú instalación OpenWRT.

Hasta aquí se ha seleccionado y guardado que paquetes se desea instalar en la versión de OpenWRT. A continuación se detalla los archivos a compilar.

B.3.2. Imágenes a compilar

Para poder instalar OpenWRT en los Mikrotik, se deberá generar 3 archivos:

- openwrt-ar71xx-vmlinux-initramfs.elf
- openwrt-ar71xx-rootfs.tgz
- openwrt-ar71xx-vmlinux.elf

El primero de ellos se utiliza para cargar una imagen de OpenWRT en memoria RAM, mientras que el segundo y el tercero son el sistema operativo y el kernel respectivamente. No se puede compilar las 3 simultáneamente. Por lo tanto, primero se genera el archivo para cargar en RAM y luego los otros dos.

Para compilar el primer archivo ejecutar:

$$\rightarrow \text{make menuconfig}$$

Luego ingresar en el menú “Target Images” y seleccionar la opción *ramdisk*.

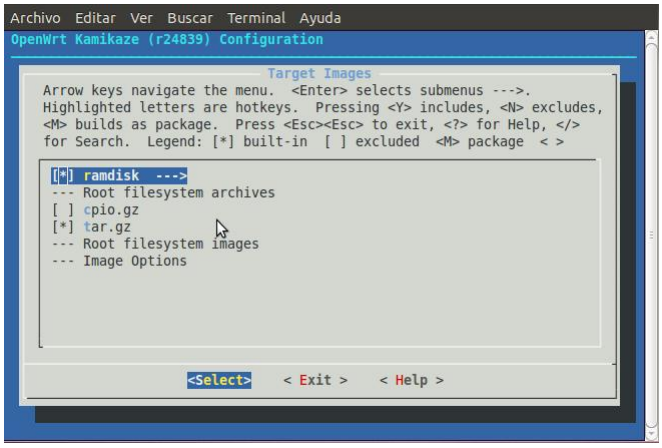

Figura B.8: Imagen para cargar en RAM.

Salir del menú y ejecutar en una consola de texto:

→ make

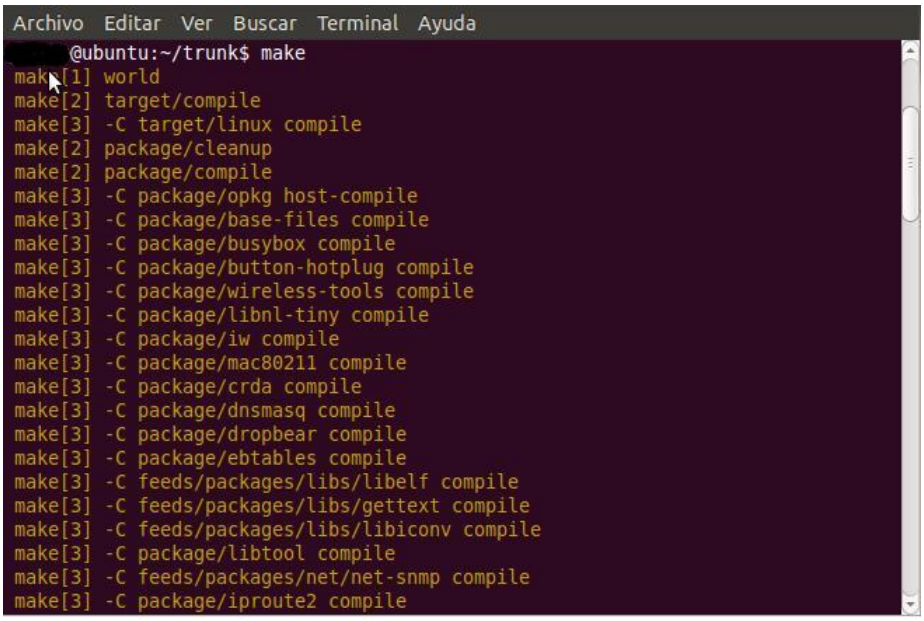

Figura B.9: Ejecutar make.

Luego de generada la imagen para cargar en RAM, lo que puede llevar mucho tiempo, se debe generar las otras 2 imágenes. Para ello se repite el proceso:

→ make menuconfig

Luego en el menú "Target Images" seleccionar las opciones *tar.gz* y *squashfs (NEW)*

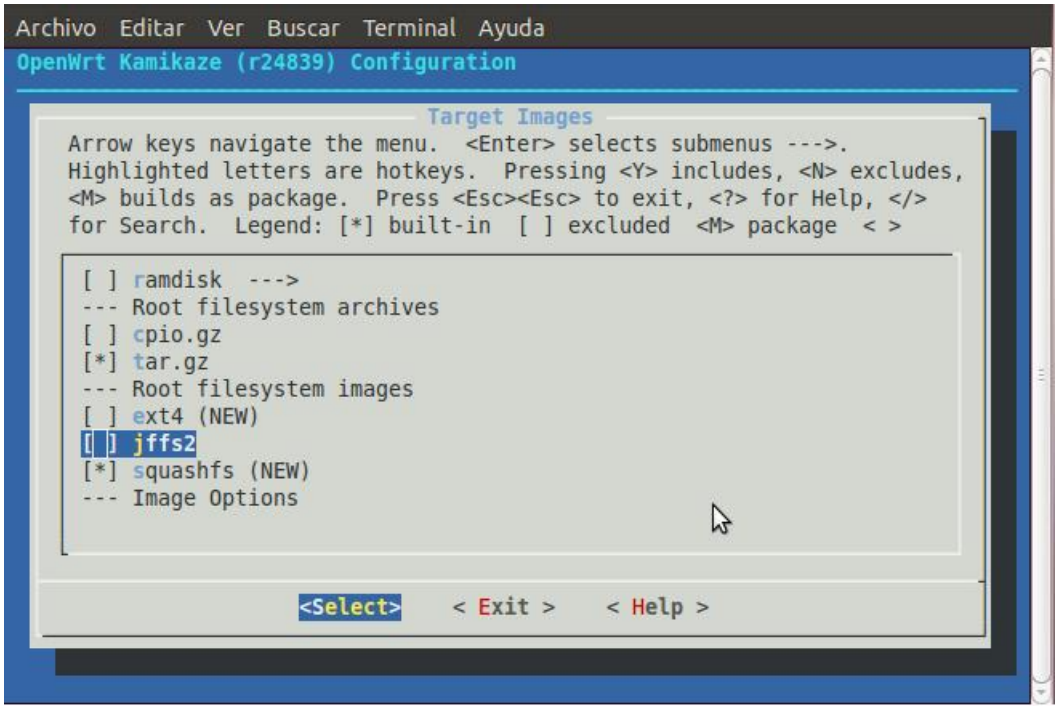

Figura B.10: Imagen de kernel y S.O.

Nuevamente salir del menú y ejecutar en una consola de texto:

→ make

B.4. Configuración del PC de trabajo

Hasta aquí se ha generado los archivos necesarios para la instalación de OpenWRT. A continuación se detalla los pasos a seguir de forma de dejar funcionando el PC de trabajo, previo a cargar las imágenes en el router. Como se mencionó en la sección anterior se tiene los siguientes archivos, los cuales se compilaron en la carpeta */home/wasa/trunk/bin/ar71xx/*:

- openwrt-ar71xx-vmlinux-initramfs.elf
- openwrt-ar71xx-rootfs.tgz
- openwrt-ar71xx-vmlinux.elf

Se crea una carpeta *tftp* dentro del *home* del usuario (no debe ser usuario root). Por claridad, se utiliza usuario wasa, el cual debe ser reemplazado por el usuario que se utilice:

→ mkdir /home/wasa/tftp

A continuación se debe copiar los archivos generados a las carpetas correspondientes. Moverse a la carpeta donde están los archivos:

→ cd /home/wasa/trunk/bin/ar71xx/

El archivo para cargar en RAM se copia a la carpeta *tftp* recientemente creada:

→ cp openwrt-ar71xx-vmlinux-initramfs.elf /home/wasa/tftp/

Los otros 2 archivos se copian a */var/www/*, creada cuando se instaló el servidor apache.

→ cp openwrt-ar71xx-rootfs.tgz /var/www/
→ cp openwrt-ar71xx-vmlinux.elf /var/www/

Luego se debe configurar el PC de forma que actúe como servidor *dhcp* para que el Mikrotik tome una dirección IP.
Primero se configura la interfaz de red del Ubuntu:

→ sudo ifconfig eth0 192.168.0.1/24

A continuación se edita el archivo *dnsmasq.conf*

→ sudo gedit /etc/dnsmasq.conf

y se agrega las siguientes líneas al final del mismo:

```
interface=eth0
dhcp-range=192.168.0.2,192.168.0.254,255.255.255.0,24h
dhcp-host=00:0c:42*:*:*,192.168.0.2
dhcp-boot=/home/wasa/tftp/openwrt-ar71xx-vmlinux-initramfs.elf
enable-tftp
tftp-root=/home/wasa/tftp
```

La primera línea corresponde a la interfaz de red donde se conecta el Mikrotik. Si la conexión no se realiza en la eth0, se debe cambiar en este archivo por la interfaz correcta. La segunda línea corresponde al rango *dhcp*, no debería ser necesario cambiarla. La tercer línea *dhcp-host=* es seguida con el prefijo de las MAC de las tarjetas de red que se está utilizando. Se puede omitir o cambiar por la MAC que se vaya a utilizar. La tercera y quinta línea depende del PATH donde se hayan guardado los archivos. En este caso se configura de forma acorde a lo que se explicó en los pasos anteriores.

Finalmente, lo único que resta es reiniciar los servicios necesarios para cargar las imágenes del sistema operativo en los Mikrotik. Para ello, se ejecuta:

→ sudo service dnsmasq restart

→ sudo service apache2 restart

NOTA: En todos los pasos donde se copian, mueven o editan archivos, se deberá revisar que los permisos asociados al usuario con el que se está trabajando sean los correctos (para cambiar permisos se utiliza el comando *chmod*). Asimismo, se debe conocer la contraseña del usuario **root** para poder seguir los pasos de instalación.

B.5. Instalación de archivos en el Mikrotik

Finalmente, en esta última sección se describe como cargar las imágenes compiladas en el Mikrotik. Para ello se trabaja con el software **Putty** el cual permite una conexión serial con el router.

- Conectar el cable serial entre el PC y el router Mikrotik.
- Conectar cable de red entre PC y router Mikrotik.
- Ejecutar el **Putty** y seleccionar conexión *serial* a una velocidad de 115200 bps.
- Encender Mikrotik y presionar alguna tecla antes de 2 segundos para que aparezcan las opciones de booteo.
- Seleccionar opción **o "boot device"**.
- Seleccionar opción **e "boot from ethernet"**
- Seleccionar opción **p "boot protocol"**.
- Seleccionar opción **2 "dhcp protocol"**.
- Seleccionar opción **x "exit"**

A continuación, deberá cargar la imagen del OpenWRT en RAM. Resta simplemente cargar las imágenes del sistema operativo en NAND, de forma de tener el sistema instalado definitivamente. Una vez en el prompt, ejecutar:

→ *ifconfig br-lan 192.168.0.3/24*, ya que cuando inicia el OpenWRT tiene otra IP por defecto, por lo tanto debemos configurar el Mikrotik en la misma red que el PC.

Finalmente,

→ *wget2nand http://192.168.0.1*, ya que fue la IP que se le asignó al PC. Luego de ejecutar este comando el Mikrotik ya fue "flasheado" con el nuevo sistema operativo.

- Reiniciar Mikrotik con el comando *reboot*
- Seleccionar opción **o "boot device"**
- Seleccionar opción **o "boot from nand only"**
- Seleccionar opción **x "exit"**

El equipo quedó configurado. Ya se puede desconectar el cable serial y cada vez que se encienda el Mikrotik cargará el sistema operativo OpenWRT.

Apéndice C

Configuración del OpenWRT

En el siguiente anexo se detalla brevemente algunos aspectos básicos de la configuración de OpenWrt. Al tratarse de un sistema Linux, la misma se realiza mediante la edición de archivos de texto. A continuación se detalla los archivos utilizados así como algunos comandos fundamentales.

OpenWrt trae habilitados algunos servicios por defecto, como son firewall y servidor dhcp, pero se deshabilitaron ya que no iban a ser utilizados.

C.1. Archivos de configuración

C.1.1. Archivos básicos

Los archivos básicos de configuración se encuentran en */etc/conf/*. Fundamentalmente se utilizaron los siguientes:

- *network*
- *wireless*

En el archivo *network* se configura las opciones básicas de las interfaces de red, como es la dirección IP y máscara (si se define una IP estática) o si la interfaz solicita dirección de red mediante DHCP.

```
config 'interface' 'loopback'
        option 'ifname' 'lo'
        option 'proto' 'static'
        option 'ipaddr' '127.0.0.1'
        option 'netmask' '255.0.0.0'

config 'interface' 'lan'
        option 'ifname' 'eth0 eth1'
        option 'type'  'bridge'
        option 'proto' 'static'
```

```
        option 'netmask' '255.255.255.0'
        option 'ipaddr' '192.168.2.20'

config 'interface' 'wan'
        option 'ifname' 'ath0'
        option 'proto' 'static'
        option 'netmask' '255.255.255.0'
        option 'ipaddr' '192.168.0.20'
```

Se creó un bridge entre las interfaces *eth0* y *eth1*.

Como su nombre lo indica, el archivo *wireless* permite configurar lo relacionado a la interfaz inalámbrica.

```
config 'wifi-device' 'wifi0'
        option 'type' 'atheros'
        option 'country' '0'
        option 'channel' '149'
        option 'diversity' '0'
        option 'txantenna' '0'
        option 'rxantenna' '0'
        option 'disabled' '0'
        option 'hwmode' '11a'

config 'wifi-iface'
        option 'device' 'wifi0'
        option 'network' 'wan'
        option 'encryption' 'none'
        option 'ssid' 'wasatest'
        option 'hidden' '0'
        option 'isolate' '0'
        option 'bgscan' '0'
        option 'macfilter' 'none'
        option 'wds' '0'
        option 'mode' 'ap'
        option 'txpower' '49'
        option 'wmm' '1'
```

No se detalla cada una de las opciones pero se señala que **"option wmm 1"** habilita Wireless MultiMedia (802.11e).

Modificar cualquiera de estos archivos no genera ningún cambio hasta que se reinicien los servicios de red o el sistema.

C.1.2. Otros archivos

El archivo *custom-user-startup* ubicado en */etc/init.d/* fue de gran utilidad ya que allí se colocan los comandos que quiera ejecutar el usuario cuando se inicia el sistema. Si se quiere

realizar una configuración que persista en cada reinicio del sistema se debe colocar aquí. En particular se agregaron:

- rutas estáticas
- script de configuración de qdisc
- comando para cambio de valores por defecto, p.e. acktimeout
- comando para sincronización horaria
- parámetros de WMM que se deseen setear al inicio del sistema

C.2. Comandos utilizados

C.2.1. Comandos básicos

A continuación se detalla algunos comandos para el manejo básico de OpenWrt.
Reiniciar el sistema: → *reboot*
Editor de texto para modificar archivos: → *vi nombre_archivo*
Bajar el servicio firewall: → */etc/init.d/firewall stop*
Deshabilitar servicio firewall: → */etc/init.d/firewall disable*
Reiniciar interfaces de red: → */etc/init.d/network restart*
Sincronización horaria: → *rdate IP_servidor_de_hora*

C.2.2. Comandos para 802.11e

Se enumera los comandos utilizados para la configuración de parámetros relacionados a 802.11e mediante el driver MadWifi. Por información más detallada sobre las opciones disponibles consultar [31].
Consulta si WMM está habilitado o no: → *iwpriv ath0 get_wmm*
Habilitar WMM si está deshabilitado: → *iwpriv ath0 wmm 1*
Consulta valores configurados para las colas de 802.11e: → *wlanconfig ath0 list wme*

Para la configuración de cada una de las colas de 802.11e, la sintaxis de los comandos es similar.
→ *iwpriv ath0* (AIFS CWmin CWmax TXOPlimit) *A B C*

A: corresponde a la cola que se quiere configurar,

- 0: Best Effort, BE

- 1: Backgroung, BK
- 2: Video, VI
- 3: Voice, VO

B: si se modifica el parámetro para el AP o la estación.
C: el valor a setear.

C.3. Otros valores

En */proc/sys/dev/wifi0/* se puede setear específicamente ciertos valores utilizados tales como *acktimeout* o *slottime*. Particularmente para el acktimeout se siguieron las recomendaciones de [32] [33]:
→ *echo 235 > /proc/sys/dev/wifi0/acktimeout*

Apéndice D

Marcado de Paquetes

Como primer elemento para la política de calidad de servicio (QoS), es necesario clasificar o identificar el tráfico que va a ser tratado de diferente forma. La clasificación y el marcado de paquetes puede hacerse en diferente capas, por ejemplo:

- Capa 2: 802.1Q Class of Service, 802.11e
- Capa 3: Differentiated Services Code Points (DSCP), Direcciones Origen/Destino IP
- Capa 4: Protocolos TCP/UDP, Puertos Origen/Destino
- Capa 7: Aplicaciones específicas

Para el marcado de tráfico se vio la necesidad de representar cómo es el mapeo de los bits DSCP (*Differentiated Services Code Point* ver RFC [34]) con 802.11e.

Para lograr una correcta implementación de los ensayos, se marcaron paquetes salientes desde los Linux con la herramienta *iptables*, creando reglas de marcado según el puerto destino del paquete. Se puede verificar el manual en cualquier distribución de Linux que contenga el paquete *iptables*, con el siguiente comando *man iptables*.

D.1. Type of Service

TOS se describe en la siguiente RFC [35] y los bits del 0 - 7 son usados para la siguiente implementación:

- Bits 0 - 2: IP Precedence
- Bit 3: Delay
- Bit 4: Throughput

- Bit 5: Reliability
- Bits 6 - 7: Reservados

D.2. DSCP

Differentiated Services Code Point está descrito en la RFC [34], donde se define el siguiente uso para el byte de TOS del encabezado IPv4:

- Bits 0 - 5: DSCP Value
- Bits 6-7: Reservados

D.3. Per-Hop Behaviour

Dentro de las distintas RFC's [34] [36] [37], se definen distintas clases de DSCP PHB (*Per Hop Behaviours*):

- RFC 2474 Class Selector (CS1 al CS7)
- RFC 3246 2597 Assured Forwarding PHBs (AF)
- RFC 3246 Expedited Forwarding (EF)

Y los bits del encabezado IP, son utilizados de la siguiente manera:

- Bits 0 - 2: PHB Class Value
- Bits 3 - 4: PHB Class Selector Value
- Bits 5 - 7: Reservados

D.4. Relación entre los diferentes estándares

En la siguiente figura se resume los mapeos de los diferentes estándares de TOS, DSCP y PHB.

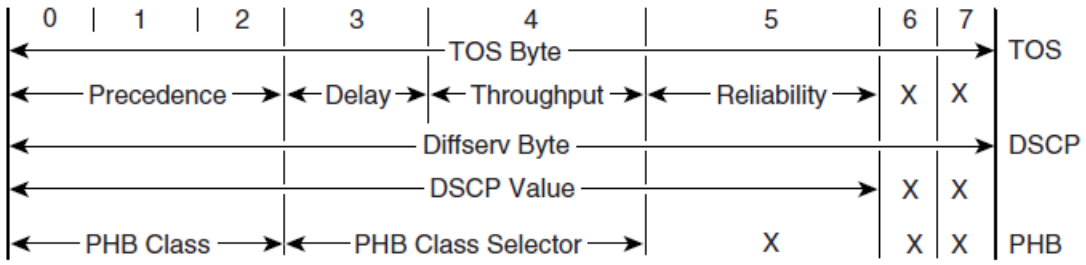

Figura D.1: Mapeo entre TOS, DSCP y PHB

D.5. Driver MadWifi

El mapeo que implementa el driver MadWifi para las AC (*Access Categories*) del estándar 802.11e es el siguiente [21]:

Valores DSCP	Clase DSCP	AC 802.11e
0x00	Default	AC BE
0x08	CS1	AC BK
0x28	CS5	AC VI
0x38	CS7	AC VO

D.6. Configuración utilizada

Es por eso que en todos los ensayos se utilizó la siguiente configuración de *iptables*, basado en lo expuesto anteriormente y utilizando la distribución Ubuntu de Linux:

- Flujo prioritario *sudo iptables -t mangle -A OUTPUT -p udp –dport 5001 -j DSCP –set-dscp 56*
- Flujo no prioritario *sudo iptables -t mangle -A OUTPUT -p udp –dport 5005 -j DSCP –set-dscp 10*

Apéndice E

Clases en qdisc

En este apéndice se dará una breve introducción a las qdisc y al uso de las mismas. Información más detallada referente a qdisc se puede encontrar en [38].

Para el control de tráfico en Linux, existe la herramienta *tc*, la cual provee diversas disciplinas de colas que pueden ser utilizadas, dependiendo de cómo se desee gestionar el tráfico. En este trabajo en particular, se utilizó disciplinas de colas con clases, las cuales son muy útiles en caso de tener diferentes tipos de tráfico a los cuales se les quiere dar un tratamiento diferenciado.

E.1. Disciplinas de colas simples o sin clases

Las disciplinas de colas sin clases son aquellas que aceptan datos y se limitan a reordenarlos, retrasarlos, o descartarlos.

La forma de encolar los datos determina cómo se envían a través de un enlace. Para una comunicación bidireccional es necesario la implementación en ambos routers de borde.

E.1.1. Pfifo fast

Como su nombre lo indica, *First In, First Out* significa: el primero que entra es el primero que sale. Ningún paquete recibe tratamiento especial o diferenciado dentro de su banda. Esta cola tiene 3 bandas y dentro de cada banda se aplican las reglas FIFO. Sin embargo, no se procesará la banda 1 mientras haya paquetes esperando en la banda 0. Lo mismo se aplica para las bandas 1 y 2.

El núcleo o *kernel* obedece la marca llamada TOS que hay en los paquetes, y se toma el cuidado de insertar los paquetes de mínimo retraso en la banda 0. La información necesaria se

encuentra en la siguiente RFC [39].

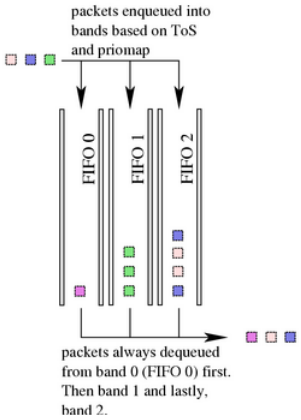

Figura E.1: Pfifo fast

E.1.2. Token Bucket Filter

El Token Bucket Filter (TBF) es una qdisc sencilla que se limita a dejar pasar paquetes que lleguen a una tasa que no exceda la tasa impuesta administrativamente, pero con la posibilidad de permitir ráfagas cortas que excedan esa tasa.

La implementación de TBF consiste en un buffer, el bucket o balde, que se llena constantemente con piezas virtuales de información denominadas tokens, a una tasa específica llamada *token rate*. El parámetro más importante del bucket es su tamaño, que es el número de tokens que puede almacenar. Cada token que llega toma un paquete de datos entrante de la cola de datos y se elimina del bucket. Asociar este algoritmo con los dos flujos (tokens y datos), deja tres posibles situaciones:

- Los datos llegan a TBF a una tasa que es igual a la de tokens entrantes. En este caso, cada paquete entrante tiene su token correspondiente y pasa a la cola sin retrasos.

- Los datos llegan al TBF a una tasa menor a la de los token. Sólo una parte de los tokens se borra con la salida da cada paquete que se envía fuera de la cola, de manera que se acumulan los tokens, hasta llenar el bucket. Los tokens sin usar se pueden utilizar para enviar datos a velocidades mayores de la tasa de tokens, en cuyo caso se produce una corta ráfaga de datos.

- Los datos llegan al TBF a una tasa mayor a la de los token. Esto significa que el bucket se quedará pronto sin tokens, lo que causará que TBF se acelere a sí mismo por un rato. Esto

se llama una *situación sobre límite*. Si siguen llegando paquetes, empezarán a ser descartados.

Esta última situación es muy importante, porque permite ajustar administrativamente al ancho de banda disponible a los datos que están pasando por el filtro.

La acumulación de tokens permite ráfagas cortas de datos extralimitados para que pasen sin pérdidas, pero cualquier sobrecarga restante causará que los paquetes se vayan retrasando constantemente y al final sean descartados.

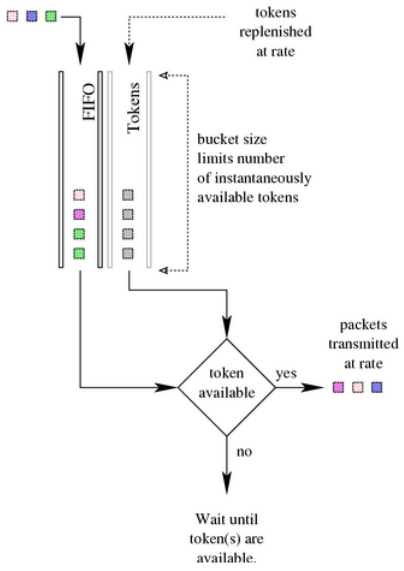

Figura E.2: Token Bucket Filter

E.1.3. Stochastic Fairness Queueing

Stochastic Fairness Queueing (SFQ) es una implementación sencilla de la familia de algoritmos de colas justas (fair queueing). Es menos preciso que los otros, pero también necesita menos cálculos.

El tráfico se divide en un número grande de colas FIFO, una por cada flujo (sesión TCP o flujo UDP). Entonces se envía el tráfico de una manera parecida a round robin (*algoritmo de planificación de procesos de forma equitativa*), dando a cada sesión por turnos la oportunidad de enviar datos. Esto lleva a un comportamiento bastante equitativo y evita que una única conversación ahogue a las demás. SFQ se llama *estocástica* porque realmente no crea una cola para cada sesión, sino que tiene un algoritmo que divide el tráfico en un número limitado de colas

usando un algoritmo de *hash*. Debido al hash, varias sesiones pueden acabar en el mismo bucket, lo que dividirá por dos las posibilidades de cada sesión de enviar un paquete, reduciendo a la mitad de esta forma la velocidad efectiva disponible. Para evitar que esta situación acabe siendo detectable, SFQ cambia a menudo su algoritmo hash de manera que dos sesiones sólo colisionen durante unos pocos segundos.

Es importante tener en cuenta que SFQ sólo es útil en caso de que la interfaz real de salida esté realmente llena. Si no lo está, entonces el kernel de Linux no encolará paquetes y no se producirá efecto alguno.

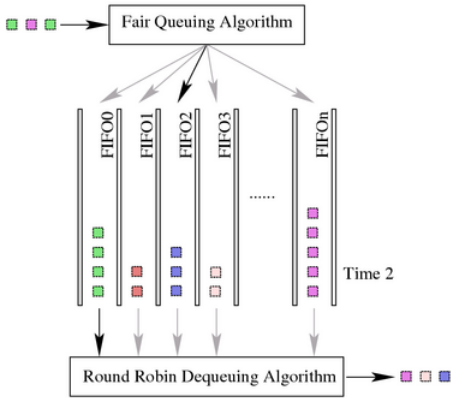

Figura E.3: Stochastic Fairness Queueing

E.2. Disciplinas de cola con clases

Las qdisc con clases son muy útiles si se tienen diferentes tipos de tráfico a los que se quiere dar un tratamiento separado.

Cuando llega tráfico (paquetes) a una qdisc con clases, se clasifica según los filtros. Estos últimos son los que toman la decisión para redirigir ese tráfico. La qdisc, se encarga de encolar el paquete en una de las clases. Cada una de las subclases puede probar otros filtros para ver si se imparten más instrucciones sobre ese paquete.

Aparte de contener otras qdisc, la mayoría de las qdisc con clases también realizan lo que denominamos *shaping*, lo que resulta útil para reordenar paquetes y para controlar tasas de tráfico.

Cada interfaz tiene una *qdisc raíz* de salida, que por defecto es la disciplina de colas *pfifo fast* sin clases que se mencionó anteriormente. A cada qdisc se le puede asignar un controlador o *handle*, que puede ser usado en sentencias posteriores para referirse a la configuración de esa

qdisc. Aparte de la qdisc de salida, la interfaz también puede tener una de qdisc entrada, que dicta las normas sobre el tráfico entrante.

E.2.1. PRIO

La qdisc PRIO no hace ajustes, sino que sólo subdivide el tráfico basándose en cómo se hayan configurado los filtros. Formalmente se puede llamar un reorganizador conservativo. Se puede considerar la qdisc PRIO como una pfifo fast mejorada, en la que cada banda es una clase separada en lugar de una simple FIFO. Cuando se encola un paquete a la qdisc PRIO, se escoge una clase basándose en las órdenes de filtrado que haya. Por defecto, se crean tres clases, al igual que pfifo. Estas clases contienen qdisc que son puras FIFO sin estructura interna, pero puede sustituirlas por cualquier qdisc que haya disponible. Siempre que se necesite desencolar un paquete, se intenta primero con la clase :1. Las clases más altas sólo se usan si no se ha conseguido encolar el paquete en las clases más bajas. Esta qdisc es muy útil en caso que se quiera dar prioridad a cierto tráfico sin usar sólo las marcas TOS sino usando los filtros de la herramienta *tc* de Linux.

E.2.2. Hierarchical Token Bucket

HTB funciona igual que CBQ (*Class Based Queueing*), pero no recurre a cálculos de tiempo ocioso para los ajustes. En su lugar, es un Token Bucket Filter con clases. Sólo tiene unos pocos parámetros en comparación con CBQ y es más rápido gracias a ello. Al realizar ingeniería de tráfico de la red se determinan cuantas fichas son necesarias para transmitir un número de bytes. Mientras haya fichas el flujo de paquetes es permitido. Por lo tanto, un flujo puede transmitir el tráfico hasta su velocidad de ráfaga máxima si hay fichas suficientes en el cubo o bucket. Más información acerca del funcionamiento e implementación de HTB puede consultarse en [40].

Apéndice F

Configuración de qdisc en OpenWRT

Las configuraciones realizadas se basaron en el trabajo realizado en [21] y en las recomendaciones en [38].

F.1. Configuración de colas con clases y sin prioridades

A continuación, se muestra la configuración utilizada para crear 2 qdisc HTB en capa 3 sin prioridades entre ellas. Un tráfico ingresó en la cola AC_VO y el otro en la AC_BK.

```
# Borro colas
tc qdisc del dev ath0 root
tc qdisc del dev ath0 ingress
tc qdisc del dev eth0 root
tc qdisc del dev eth0 ingress

#Creo clase root
tc qdisc add dev ath0 root handle 1: htb default 10

## Clase raiz ("bit" = bps, "bps" = Bps)
tc class add dev ath0 parent 1: classid 1:1 htb rate 25000kbit ceil 25000kbit

## Trafico QoS
tc class add dev ath0 parent 1:1 classid 1:12 htb rate 20000kbit ceil 25000kbit burst 17kb

## AC_VO
tc class add dev ath0 parent 1:12 classid 1:121 htb rate 2000kbit ceil 25000kbit burst 17kb
prio 0
tc qdisc add dev ath0 parent 1:121 handle 121: sfq perturb 10
tc filter add dev ath0 parent 1:0 protocol ip prio 1 u32 match ip tos 0xe0 0xff flowid 1:121
```

```
## AC_BK
tc class add dev ath0 parent 1:12 classid 1:120 htb rate 2000kbit ceil 25000kbit burst 34kb
prio 0
tc qdisc add dev ath0 parent 1:120 handle 120: sfq perturb 10
tc filter add dev ath0 parent 1:0 protocol ip prio 1 u32 match ip tos 0x28 0xff flowid 1:120
```

F.2. Configuración de colas con clases y prioridades

La siguiente configuración es la que se utilizó en el enlace de laboratorio, para diferenciar el throughput obtenido por cada uno de los flujos.

```
# Borro colas
tc qdisc del dev ath0 root
tc qdisc del dev ath0 ingress
tc qdisc del dev eth0 root
tc qdisc del dev eth0 ingress

#Creo clase root
tc qdisc add dev ath0 root handle 1: htb default 10

## Clase raiz ("bit" = bps, "bps" = Bps)
tc class add dev ath0 parent 1: classid 1:1 htb rate 25000kbit ceil 25000kbit

## Trafico QoS
tc class add dev ath0 parent 1:1 classid 1:12 htb rate 20000kbit ceil 25000kbit burst 17kb

## Todas las clases tienen un mínimo que se intenta asegurar y pueden llegar a tomar todo
el ancho de banda disponible en el canal si no hay tráfico
## AC_VO
tc class add dev ath0 parent 1:12 classid 1:121 htb rate 2000kbit ceil 25000kbit burst 17kb
prio 0
tc qdisc add dev ath0 parent 1:121 handle 121: sfq perturb 10
tc filter add dev ath0 parent 1:0 protocol ip prio 1 u32 match ip tos 0xe0 0xff flowid 1:121

## AC_VI
tc class add dev ath0 parent 1:12 classid 1:122 htb rate 1000kbit ceil 25000kbit burst 17kb
prio 1
tc qdisc add dev ath0 parent 1:122 handle 122: sfq perturb 10
tc filter add dev ath0 parent 1:0 protocol ip prio 2 u32 match ip tos 0xa0 0xff flowid 1:122

## AC_BK
tc class add dev ath0 parent 1:12 classid 1:120 htb rate 500kbit ceil 25000kbit burst 34kb
prio 2
tc qdisc add dev ath0 parent 1:120 handle 120: sfq perturb 10
tc filter add dev ath0 parent 1:0 protocol ip prio 3 u32 match ip tos 0x28 0xff flowid 1:120


##AC_BE
tc class add dev ath0 parent 1:1 classid 1:11 htb rate 1000kbit ceil 25000kbit burst 17kb
prio 3
tc qdisc add dev ath0 parent 1:11 handle 11: sfq perturb 10
```

```
tc filter add dev ath0 parent 1:0 protocol ip prio 3 u32 match ip src any flowid 1:11
```

F.3. Configuración utilizada en el enlace de larga distancia

Finalmente, para el enlace de larga distancia se modificaron algunos valores para ajustarse a las características del enlace.

```
# Borro colas
tc qdisc del dev ath0 root
tc qdisc del dev ath0 ingress
tc qdisc del dev eth0 root
tc qdisc del dev eth0 ingress

#Creo clase root
tc qdisc add dev ath0 root handle 1: htb default 10

## Clase raiz ("bit" = bps, "bps" = Bps)
tc class add dev ath0 parent 1: classid 1:1 htb rate 16000kbit ceil 16000kbit

## Trafico QoS
tc class add dev ath0 parent 1:1 classid 1:12 htb rate 10000kbit ceil 16000kbit burst 17kb

## AC_VO
tc class add dev ath0 parent 1:12 classid 1:121 htb rate 2000kbit ceil 16000kbit burst 17kb
prio 0
tc qdisc add dev ath0 parent 1:121 handle 121: sfq perturb 10
tc filter add dev ath0 parent 1:0 protocol ip prio 1 u32 match ip tos 0xe0 0xff flowid 1:121


## AC_VI
tc class add dev ath0 parent 1:12 classid 1:122 htb rate 1000kbit ceil 16000kbit burst 17kb
prio 1
tc qdisc add dev ath0 parent 1:122 handle 122: sfq perturb 10
tc filter add dev ath0 parent 1:0 protocol ip prio 2 u32 match ip tos 0xa0 0xff flowid 1:122


## AC_BK
tc class add dev ath0 parent 1:12 classid 1:120 htb rate 500kbit ceil 16000kbitburst 34kb
prio 2
tc qdisc add dev ath0 parent 1:120 handle 120: sfq perturb 10
tc filter add dev ath0 parent 1:0 protocol ip prio 3 u32 match ip tos 0x28 0xff flowid 1:120


##AC_BE
tc class add dev ath0 parent 1:1 classid 1:11 htb rate 1000kbit ceil 16000kbit burst 17kb
prio 3
tc qdisc add dev ath0 parent 1:11 handle 11: sfq perturb 10
tc filter add dev ath0 parent 1:0 protocol ip prio 3 u32 match ip src any flowid 1:11
```

Apéndice G

Herramientas adicionales

En este apartado se menciona, brevemente, algunas herramientas disponibles que se evaluaron en el presente proyecto, aunque finalmente no se utilizaron.

D-ITG: Distributed Internet Traffic Generator, es un generador de tráfico muy robusto desarrollado por el "Dipartimento di Informatica e Sistemistica de la Università degli Studi di Napoli, Federico II" [41]. Permite generar tráfico TCP, UDP, de voz con distintos codecs y setear los campos TOS y TTL, entre otros, de los distintos paquetes enviados. Incluye un logger lo que permite graficar fácilmente los resultados obtenidos.

TCPSpray: herramienta para enviar tráfico TCP y medir el throughput obtenido. Permite ajustar el tamaño y número de paquetes entre otras opciones. El manual en Linux está disponible en [42].

SmokePing: herramienta para medir retardo (RTT) y latencia en una red de datos [43]. Los resultados son fácilmente logueables y se pueden generar alarmas.

Se puede encontrar una lista de herramientas para el análisis del tráfico de red en sistemas Linux en [44].

Parte VI

Bibliografía

Bibliografía

[1] D. Carlos García García. *Propuesta de arquitectura de QoS en entorno inalámbrico 802.11e basado en Diffserv con ajuste dinámico de parámetros.* 2006.

[2] 3Com Wireless 7760 11 a/b/g PoE Access Point. *www.3Com.com.*

[3] Tudor Blaga Gabriel Lazar, Virgil Dobrota. *Performance of Wireless IEEE 802.11e-Based Devices with Multiple Hardware Queues.*

[4] *http://maps.google.com/.*

[5] IEEE 802.11. *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications.* 2007.

[6] EHAS: Enlace hispano Americano de Salud. *www.ehas.org.*

[7] Aravind eye care system. *www.aravind.org.*

[8] Sandra Salmeron Ntutumu. *Parametrizacion de IEEE 802.11e EDCA para la priorizacion del trafico VOIP en redes extensas para zonas rurales de paises en vias de desarrollo.*

[9] César Córdova Leopoldo Liñán David Chávez Luis Camacho, River Quispe. *WILD: WiFi Based Long Distance.* 2009.

[10] Ma. Fernanda Dulcey Álvaro Rendón Francisco Simó, Andrés Martínez. *Implementación de IEEE 802.11 en enlaces largos para zonas rurales aisladas.*

[11] IEEE. *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 8: Medium Access Control (MAC) Quality of Service Enhancements.* 2005.

[12] River Quispe David Segundo Javier Simo, Pablo Osuna. *Application of IEEE 802.11 technology for health isolated rural environments.*

[13] y J. Cotrina E. Lopez-Aguilera, J. Casademont. *Sobre la justicia en las redes IEEE 802.11e: Desincronizacion de su mecanismo de acceso al medio.* Octubre 2007.

[14] *http://www.routerboard.com/pdf/rb433ugA.pdf.*

[15] *http://www.reloadbg.com/pdf/cm11.pdf.*

[16] OpenWRT. *http://openwrt.org/.*

[17] Iperf. *http://sourceforge.net/projects/iperf/.*

[18] Wireshark. *www.wireshark.org/.*

[19] WifiSlax. *www.wifislax.com.*

[20] Hayoug Yoon. *Test of MADWIFI-ng WMM/WME in WLANs.* 2006.

[21] Nydia Mendiola Almaraz. *Estrategia de Integración para equipos 802.11e-EDCA (WiFi) y 802.16 (WiMAX) con soporte para QoS.* 2010.

[22] Peter P. Waskiewicz Jr Zhu Yi. *Enabling Linux Network Support of Hardware Multiqueue Devices.* 2007.

[23] Sandro Costantini. *Calculo del radio de la zona de Fresnel.*

[24] Ermanno Pietrosemoli. *TRICALCAR: Enlaces de Larga Distancia.*

[25] *http://wistrondcma82.tk/.*

[26] Ing. José Joskowicz. *Voz, Video y Telefonía sobre IP.*

[27] Andrew S. Tanenbaum. *Redes de computadoras, 4ta edicion.*

[28] Juan Rodriguez Martin Irazoqui, Hernan Susunday. *Proyecto Yacare, Analizador RF para Plan Ceibal.*

[29] Francisco López Ortiz. *El estándar IEEE 802.11.*

[30] Intel. *White Paper Providing QoS in WLANS.* 2009.

[31] *Madwifi/Atheros Wireless Linux Driver Users Guide.* 2006.

[32] *http://wiki.ehas.org/index.php.*

[33] *http://madwifi-project.org/wiki/UserDocs/LongDistance.*

[34] Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers. *www.ietf.org/rfc/rfc2474.txt.*

[35] Internet Protocol Specification. *www.ietf.org/rfc/rfc791.txt.*

[36] Assured Forwarding PHB Group. *www.ietf.org/rfc/rfc2597.txt.*

[37] An Expedited Forwarding PHB (Per-Hop Behavior). *www.ietf.org/rfc/rfc3246.txt.*

[38] *Linux Advanced Routing and Traffic Control http://larct.org.*

[39] Internet Protocol Specification. *www.ietf.org/rfc/rfc1349.txt.*

[40] Martin Devera. *HTB Home luxik.cdi.cz/ devik/qos/htb/.*

[41] D-ITG. *http://www.grid.unina.it/software/ITG/.*

[42] TcpSpray. *http://linux.die.net/man/1/tcpspray.*

[43] SmokePing. *http://oss.oetiker.ch/smokeping/.*

[44] *http://www.ubuntugeek.com/bandwidth-monitoring-tools-for-linux.html.*