UNIVERSIDAD NACIONAL DE LA PLATA

FACULTAD DE INFORMÁTICA

# TESINA DE LICENCIATURA

**Título:** Redes inalámbricas ad-hoc autoconfigurables. Caso de estudio e implementación
**Autores:** Marcos Mazzini
**Director:** Ing. Luis Marrone
**Codirector:** Lic. Paula Venosa
**Asesor profesional:** -
**Carrera:** Licenciatura en Informática

## Resumen

*Cualquier computadora portátil medianamente moderna ya cuenta con una placa de red inalámbrica y soporte para redes ad-hoc o sea que cada placa de red puede reenviar datos a cualquier otra que esté a su alcance sin necesidad de un nodo centralizado, pero esta configuración no viene por defecto en los distintos sistemas operativos y en general se requiere instalar y configurar software adicional para que la comunicación sea simple para el usuario no experto.*
*Con la idea de facilitar el aprovechamiento del hardware con el que cuentan estas computadoras, se investigaron las tecnologías existentes que permiten configurar las placas de red en modo ad-hoc, asignar direcciones IP en forma descentralizada, resolver nombres de forma descentralizada y anunciar servicios de cada computadora en un entorno de red. Con esta información se derivó una versión Live CD de Linux con aplicaciones para redes preconfiguradas, soporte para placas de red Wi-Fi y modo ad-hoc preconfigurado de modo que si dos o más notebooks bootean este sistema puedan intercambiar archivos, compartir información, chatear y visualizar remotamente los escritorios y utilizar aplicaciones de red entre ellas sin ninguna necesidad de configurar, instalar o modificar los sistemas existentes y sin necesidad de conectarse a un Access Point.*

## Palabras Claves

*Wi-Fi Ad-Hoc LiveCD Linux Cigarra Conectar Igualdad Debian Live Zeroconf Avahi mDNS Link Local IP*

## Trabajos Realizados

*Se analizaron los estándares inalámbricos WLAN, las técnicas MAC, modo ad-hoc y su soporte en Linux ,las tecnologías de cero configuración, aplicaciones para redes LAN y distribuciones Linux Live. Se construyó un prototipo del sistema usando Debian Live. Éste incluye una conexión inalámbrica ad-hoc y descentralización de la configuración mediante Avahi (asignación de IP, resolución de nombres y anuncio de servicios). Se incluyeron las aplicaciones de red y un browser de servicios simple sobre un entorno Gnome. Se realizaron y documentaron diversas pruebas del sistema y un manual de usaurio.*

## Conclusiones

*Se verificó que las tecnologías analizadas son compatibles y que resuelven la funcionalidad esperada.*
*La problemática planteada por la configuración de la red y la falta de servidores centralizados se solucionó staisfactoriamente incluyendo y configurando las herramientas Network-Manager y Avahi en el sistema final. Debian Live resultó una herramienta sólida y flexible con amplias opciones de personalización y facilidades para crear y reversionar el sistema. Se creó el proyecto de Software Libre "Linux Cigarra" para difusión y aportes disponible en http://sourceforge.net/p/linuxcigarra*

## Trabajos Futuros

*Con base en el trabajo relizado se puede plantear perfeccionar la usabilidad de la interfaz de usuario y versionar el sistema con distintos conjuntos de aplicaciones. También Medir la escalabilidad del sistema y posibilitar la instalación permanente del sistema y/o opciones de persistencia.*
*Como alternativa se propone cambiar la red ad-hoc por una red MESH para permitir el ruteo hacia otras redes.*
*También se trabajará con el Instituto Telecom Bretagne de Francia en el marco de las actividades acordadas con el LINTI.*

**Fecha de la presentación:** NOVIEMBRE 2011

# Redes inalámbricas ad-hoc autoconfigurables. Caso de estudio e implementación

Tesina de Grado - Lic. en Informática

Marcos Mazzini
Director: Luis Marrone
Codirector: Paula Venosa

Noviembre 2011

# Agradecimientos

Hay muchas personas a quienes quisiera agredecer, entre ellas:

- A Sole por el aguante y porque la Amo.
- A mi familia y la familia Palacios.
- A Leticia Tori por su gran apoyo en mis comienzos en la vida universitaria.
- A mis compañeros de trabajo del INIFTA.
- A la comunidad de Software Libre por hacer realidad una nueva forma de concebir el software y el conocimiento.
- A todos los compañeros que en algún momento de su vida militaron en las filas de Suma Informática.
- A los compañeros que me iniciaron en la vida política y a todos los que luchan día a día convencidos de que una Patria más digna y una sociedad más justa son posibles.

# Índice general

# Capítulo 1

# Objetivos

## 1.1. Motivación

En los últimos años el costo de las computadoras portátiles ha disminuido y junto a la reaparición del crédito se ha posibilitado a mayor cantidad de personas acceder a estos dispositivos, tanto es así que en la Argentina se venden 300 notebooks por día [1]. Por otro lado el programa estatal conectar igualdad ya ha entregado más de 1 millón de netbooks a estudiantes secundarios del país [2] con lo que tenemos una gran cantidad de computadoras portátiles con placas de red inalámbricas incorporadas en diversos ámbitos.

En estos ámbitos existe muchas veces conectividad limitada a Internet o políticas restrictivas en los access points que dificultan el intercambio de información de notebook a notebook aún estando en la misma red. También existe la situación en la que se encuentran varias personas con estos dispositivos pero no cuentan con un router inalámbrico o un Acess Point (AP) para conectarlos.

Por otra parte, en el campo de la educación, cada vez más estudiantes y profesores que cuentan con computadoras portátiles. Estos terminales se han convertido en una herramienta esencial en el aprendizaje a través de su capacidad para promover la cola-

boración y facilitar el intercambio de información. Sin embargo, las plataformas actuales se basan en la centralización de la conectividad y los servicios, dejando la colaboración de decisiones y el intercambio de información más bien limitada.

También existen muchas aplicaciones para redes que no se utilizan en estos ámbitos por la necesidad de instalarlas y configurarlas, lo que restringe el potencial que podrían tener los dispositivos. Entre dichas aplicaciones se incluyen servidores FTP u otras para intercambio de archivos, mensajería instantánea, asignación y resolución de nombres de los dispositivos conectados a la red además de herramientas para poder listarlos o visualizarlos, etc.

La situación que se presenta es que cualquier computadora portátil medianamente moderna ya cuenta con una placa de red inalámbrica y soporte para redes ad-hoc o sea que cada placa de red puede reenviar datos a cualquier otra que esté a su alcance sin necesidad de un nodo centralizado, pero esta configuración no viene por defecto en los distintos sistemas operativos y en general se requiere instalar y configurar software adicional para que la comunicación sea simple para el usuario no experto.

## 1.2. Propuesta

Con la idea de facilitar el aprovechamiento del hardware con el que cuentan estas computadoras, se propone investigar las tecnologías existentes que permitan configurar las placas de red en modo ad-hoc, asignar direcciones IP en forma descentralizada, resolver nombres de forma descentralizada y anunciar servicios de cada computadora en un entorno de red. Con esta información se creará o derivará una versión Live CD de Linux con aplicaciones para redes preconfiguradas, amplio soporte para placas de red Wi-Fi y modo ad-hoc preconfigurado de modo que si dos o más notebooks bootean este sistema puedan intercambiar ar-

chivos, compartir información, chatear o utilizar aplicaciones de red entre ellas sin ninguna necesidad de configurar, instalar o modificar los sistemas existentes y sin necesidad de conectarse a un AP.

El nuevo paradigma a desarrollar en este proyecto, basado en la descentralización, la movilidad y el oportunismo tiene el objetivo de innovar y mejorar las comunicaciones, ofreciendo también aplicaciones dedicadas a este propósito. El trabajo a desarrollar se incluye en el marco de las actividades educativas acordadas entre los campus del Instituto Telecom Bretagne (Francia) y el LINTI de la UNLP (Argentina) para dar un contexto de uso real al mismo, donde se creará una red dinámica entre todos los estudiantes que asisten a un curso.

Se buscan distintos objetivos con este desarrollo, por un lado que se pueda fomentar el uso de este sistema en entornos académicos ya sea en una cursada en la facultad o en las escuelas donde los alumnos disponen de netbooks, a fin de aprovechar la estructura existente. Por otro lado se esperaría generar una comunidad que de soporte a este sistema, tanto en asistencia, soporte técnico, difusión, etc. como en futuros desarrollos para mejorar o expandir el sistema. Es de gran importancia mantener el sistema libre en todos sus sentidos. Esto significa libertad para usar el sistema con cualquier propósito, libertad de estudiar su funcionamiento y adaptarlo a las distintas necesidades y libertad de distribuir copias para poder ayudar a otros [3].

Para lograrlo se investigarán las distintas tecnologías involucradas así como también las herramientas necesarias para derivar el sistema. Esto representa un aporte original como concepto uniendo aportes de diversas áreas. La combinación de estas tecnologías como un todo no se ha publicado como distribución libre [4] ni se encontraron productos comerciales con el mismo fin, aunque la misma funcionalidad se puede obtener instalando y configurando distintas herramientas en entornos existentes.

En este trabajo se valoró poder presentar un prototipo funcional como "prueba de concepto", por lo tanto en la amplia variedad de temas abarcados se profundizó lo necesario y relevante para tener un conocimiento aplicado de cada tema. Tradicionalmente los aportes nuevos se plantean desde un punto particular de un tema específico y en general no son ponderados muchos otros aportes importantes que terminan influyendo realmente en relación con la sociedad pero que son generados en la lógica de combinar creativamente conocimientos ya existentes.

Por último, como objetivo adicional, cabe mencionar que este sistema tiene la capacidad de ser utilizado en catástrofes o situaciones excepcionales dado que permite que varios dispositivos puestos en funcionamiento con este SO puedan comunicarse sin necesidad de ningún otro servicio como Internet o incluso energía eléctrica.

# Capítulo 2

# Estándares, Tecnologías y Herramientas Aplicables

A continuación se analizan las distintas tecnologías y configuraciones involucradas en cada nivel del sistema. Se organizaron en forma ascendente partiendo desde el funcionamiento del hardware, pasando por las distintas opciones para su uso y configuración y luego las componentes de software que incluyen las aplicaciones que se usarán dentro del sistema como también las necesarias para su desarrollo y creación. En cada punto se describen por un lado los estándares y definiciones y por otro lado la implementación o contraparte aplicada, en especial sobre Linux que es el soporte elegido para el sistema.

## 2.1. WLAN

Lo primero que se analizarán son las redes inalámbricas de área local o WLAN[7], su especificación e implementación hacen posible que los dispositivos puedan transmitir datos sin infraestructura de acceso. Se verá el estándar que rige su comportamiento y en particular las especificaciones para la capa física y de acceso al medio.

La especificación base es la IEEE 802.11 [5] y es un estándar

internacional que define las características de una red de área local inalámbrica (WLAN). En general están asociadas el término Wi-Fi que significa "fidelidad inalámbrica", a veces incorrectamente abreviado WiFi. Éste es el nombre de la certificación otorgada por la Wi-Fi Alliance, grupo que garantiza la compatibilidad entre dispositivos que utilizan el estándar 802.11. Por el uso indebido de los términos (y por razones de marketing) el nombre del estándar se confunde con el nombre de la certificación. Una red Wi-Fi es en realidad una red que cumple con el estándar 802.11.

Con Wi-Fi se pueden crear redes de área local inalámbricas de alta velocidad siempre y cuando el equipo que se vaya a conectar no esté muy alejado del punto de acceso. En la práctica, Wi-Fi admite computadoras portátiles, equipos de escritorio o cualquier otro tipo de dispositivo de alta velocidad con propiedades de conexión también de alta velocidad (11 Mbps o superior) dentro de un radio de varias docenas de metros en ambientes cerrados (de 20 a 50 metros en general) o dentro de un radio de cientos de metros al aire libre.

El estándar 802.11 establece los niveles inferiores del modelo OSI para las conexiones inalámbricas que utilizan ondas electromagnéticas, esto incluye la capa física (a veces abreviada capa "PHY") que ofrece tres tipos de codificación de información y la capa de enlace de datos compuesta por dos subcapas: control de enlace lógico (LLC) y control de acceso al medio (MAC).

La capa física define la modulación de las ondas de radio y las características de señalización para la transmisión de datos mientras que la capa de enlace de datos define la interfaz entre el bus del equipo y la capa física, en particular un método de acceso parecido al utilizado en el estándar Ethernet, y las reglas para la comunicación entre los hosts de la red[6].

Cualquier protocolo de nivel superior puede utilizarse en una red inalámbrica Wi-Fi de la misma manera que puede utilizarse

en una red Ethernet. Los estándares 802.11a, 802.11b y 802.11g, llamados "estándares físicos", son modificaciones del estándar 802.11 y operan de modos diferentes, lo que les permite alcanzar distintas velocidades en la transferencia de datos según sus rangos. En septiembre de 2009 fue aprobado el estándar 802.11n que promete una velocidad real de transferencia de 600 Mbps y mayor alcance también gracias a la tecnología Multiple Input – Multiple Output (MIMO), que permite utilizar varios canales a la vez para enviar y recibir datos al incorporar varias antenas. A continuación se profundizarán los mecanismos del estándar 802.11b que es el más difundido y el que se utiliza en el proyecto.

### 2.1.1. Capa física

La capa física[8] trabaja a nivel de bits y cómo transformarlos a señales sobre el medio por el que se quiere transmitir, en este caso las ondas electromagnéticas.

Las ondas electromagnéticas abarcan un amplio rango de frecuencias. Este rango de frecuencias y longitudes de onda es denominado espectro electromagnético. La parte del espectro más familiar a los seres humanos es probablemente la luz, la porción visible del espectro electromagnético. La luz se ubica aproximadamente entre las frecuencias de $7{,}5 * 10^{14}$Hz y $3{,}8 * 10^{14}$Hz.

El término "radio" es utilizado para la porción del espectro electromagnético en la cual las ondas pueden ser transmitidas por medio de una antena. Esto abarca el rango de 3Hz a 300GHz.

El espectro radioeléctrico es un recurso natural limitado, un bien común, compuesto por el conjunto de ondas electromagnéticas que se propagan por el espacio sin necesidad de guía artificial. Esto implica que se encuentra regulado en todos los países y en particular en la Argentina todas las actividades relacionadas con la administración, la gestión, la planificación y el control del

uso del espectro, se reparten entre los siguientes organismos del estado nacional: la Secretaria de Comunicaciones (SECOM), la Comisión Nacional de Comunicaciones (CNC), la Autoridad Federal de Servicios de Comunicación Audiovisual (ex COMFER) y la Comisión de Defensa de la Competencia. En paralelo a estos organismos nacionales, en el plano internacional funciona la Unión Internacional de Telecomunicaciones (ITU), un órgano dependiente de las Naciones Unidas, que se ocupa de la gestión internacional del espectro radioeléctrico, de inscribir asignaciones de frecuencia y posiciones orbitales de satélite.

Las distintas organizaciones alocaron frecuencias específicas para ser usadas de manera flexible, las más antiguas y comunes son las bandas de 900MHz y 2,4GHz y se llaman bandas ISM por las siglas en ingles de Industria, Científica y Médica. La principal característica es que estas bandas no precisan licencia para operar sobre ellas por lo que se establecieron reglas de máximo poder de transmisión además de técnicas de espectro esparcido o spread spectrum para permitir la convivencia de los distintos sistemas. Como estas bandas son "libres" pueden sufrir muchas interferencias de otros sistemas sin licenciar, en particular en la banda de 2,4GHz se encuentran los hornos de microondas, ésta es una de las razones por las cuales en todos los países esta banda se encuentra disponible con la excepción de Japón que establece algunas restricciones.

Principalmente por estas regulaciones el estándar 802.11b utiliza la banda de 2,4GHz y define el siguiente comportamiento: El espectro está dividido en partes iguales distribuidas sobre la banda en canales individuales. Los canales son de un ancho de 22MHz, pero están separados sólo por 5MHz. Esto significa que los canales adyacentes se superponen, y pueden interferir unos con otros.

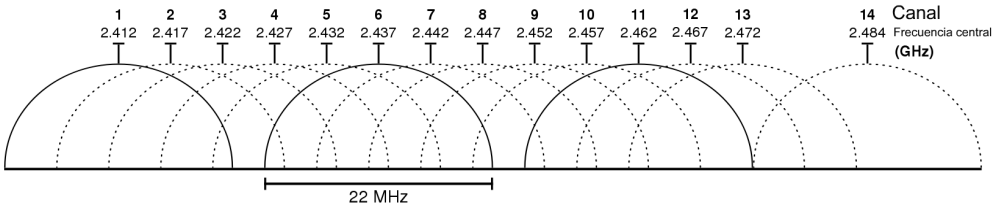


Figura 2.1: Distribución de los canales sobre la banda de 2.4GHz

Para poder transmitir de manera más robusta y debido a la normativa, se utiliza el principio de espectro esparcido por secuencia directa o Direct-Sequence Spread Spectrum (DSSS). El funcionamiento de esta técnica se basa en esparcir la señal en una banda más grande multiplexándola con un patrón de bits redundantes (el código). Cada bit a transmitir se modula con dicho código resultando 11 bits finales por cada bit transmitido y además se atenúa la señal. Esto permite que los sistemas que no están funcionando con el código correspondiente perciban la señal como ruido y los que sí lo hacen puedan demodular la señal con dicho código. Para una transmisión de 2Mb/s modulada por 11 códigos el resultado es una señal esparcida sobre 22MHz de ancho de banda. Para incrementar la velocidad de transferencia se utiliza la modulación por código complementario o Complementary Code Keying (CKK) que llega a 11Mb/s. Esta modulación en vez de utilizar 11 bits por cada uno del original que se quería transmitir usa 64 códigos únicos para cada grupo posible de 6 bits incrementando así la cantidad de información codificada.

La implementación de esta capa se da a nivel de hardware y está compuesta principalmente por componentes analógicas que modulan datos en frecuencias. Existen varias clases de hardware que cumplen dicha función y sus componentes incluyen antenas, amplificadores, sintetizadores, etc. Entre los dispositivos tenemos por un lado las placas de red inalámbricas que cumplen con

el estándar y están disponibles como placas PCI, placas PCMCIA, adaptadores USB o compact flash, ya sea por separado o embebidos en los distintos dispositivos.

Por otro lado tenemos los puntos de acceso o access points (AP) a veces denominados zonas locales de cobertura, que permiten a los dispositivos equipados con Wi-Fi cercanos acceder a una red a la que el AP se conecta directamente. Estos dispositivos pueden ser entre otros routers Wi-Fi, módems o PC servidores configurados particularmente.

### 2.1.2. Capa de enlace

En la capa anterior se definió cómo se codifican los bits, en esta capa el objetivo principal regular el uso del medio o Medium Access Control (MAC)[9]. El protocolo definido para esta capa provee un mecanismo de acceso al canal que divide el recurso principal (el canal de radio) entre los nodos y rige su uso. Establece cuando puede transmitir cada nodo y cuando se espera que reciba datos.

El mecanismo de acceso al canal utilizado por las WLAN se traduce como acceso múltiple por detección de portadora para evitar colisiones o en inglés Carrier Sense Multiple Access / Collision Avoidance (CSMA/CA). Los principios básicos de este mecanismo son el de escuchar antes de transmitir y la competencia por el acceso. Es un mecanismo de pasaje de mensajes asincrónico o sea sin conexión que da un servicio de "mejor esfuerzo" que no asegura ancho de banda ni latencia. Sus principales ventajas son que es adecuado para protocolos de red como TCP/IP, se adapta bien a la variabilidad del tráfico y es bastante robusto frente a las interferencias.

CSMA/CA se deriva de CSMA/CD que usa detección de colisiones o Collision Detection, la base de Ethernet. La diferencia principal es la evasión de colisiones: en un cable el transmi-

sor/receptor tiene la posibilidad de escuchar mientras transmite y de esta forma detectar las colisiones. Pero aunque un nodo de radio pudiera escuchar en un canal mientras transmite, la potencia de su propia transmisión enmascararía todas las otras señales (en el cable todas las transmisiones tienen aproximadamente la misma potencia). Entonces el protocolo no puede detectar las colisiones y sólo trata de evitarlas.

El protocolo empieza por escuchar en el canal, lo que se denomina Carrier Sense y si lo encuentra libre manda el primer paquete en la cola de transmisión, si está ocupado, ya sea por otra transmisión o por interferencia, el nodo espera el final de la transmisión y empieza la etapa de competencia, esto es esperar un tiempo aleatorio. El nodo que haya elegido el menor tiempo de espera gana y transmite su paquete (siempre y cuando el canal siga libre). Los otros nodos esperan la próxima competencia que comenzará al finalizar la transmisión de este paquete. Como la competencia se define por números aleatorios y se realiza por cada paquete, cada nodo tiene en promedio igual chance de acceder al canal.

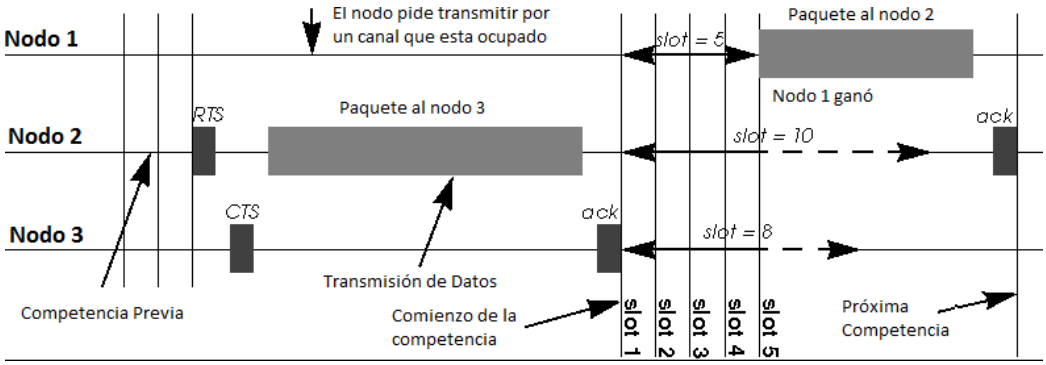


Figura 2.2: Competencia CSMA (contention)

Adicionalmente, cuando se consigue transmitir, el emisor y el receptor se envían paquetes específicos para hacer saber al emisor que el receptor esta libre u ocupado y del receptor al emisor para confirmar la recepción de los datos. Como veremos

más adelante dichos paquetes (en concreto Request To Send (RTS), Clear to Send (CTS) y Acknowledge (ACK)) se utlizan combinados en técnicas para resolver problemas de colisiones inesperadas o no detectables.

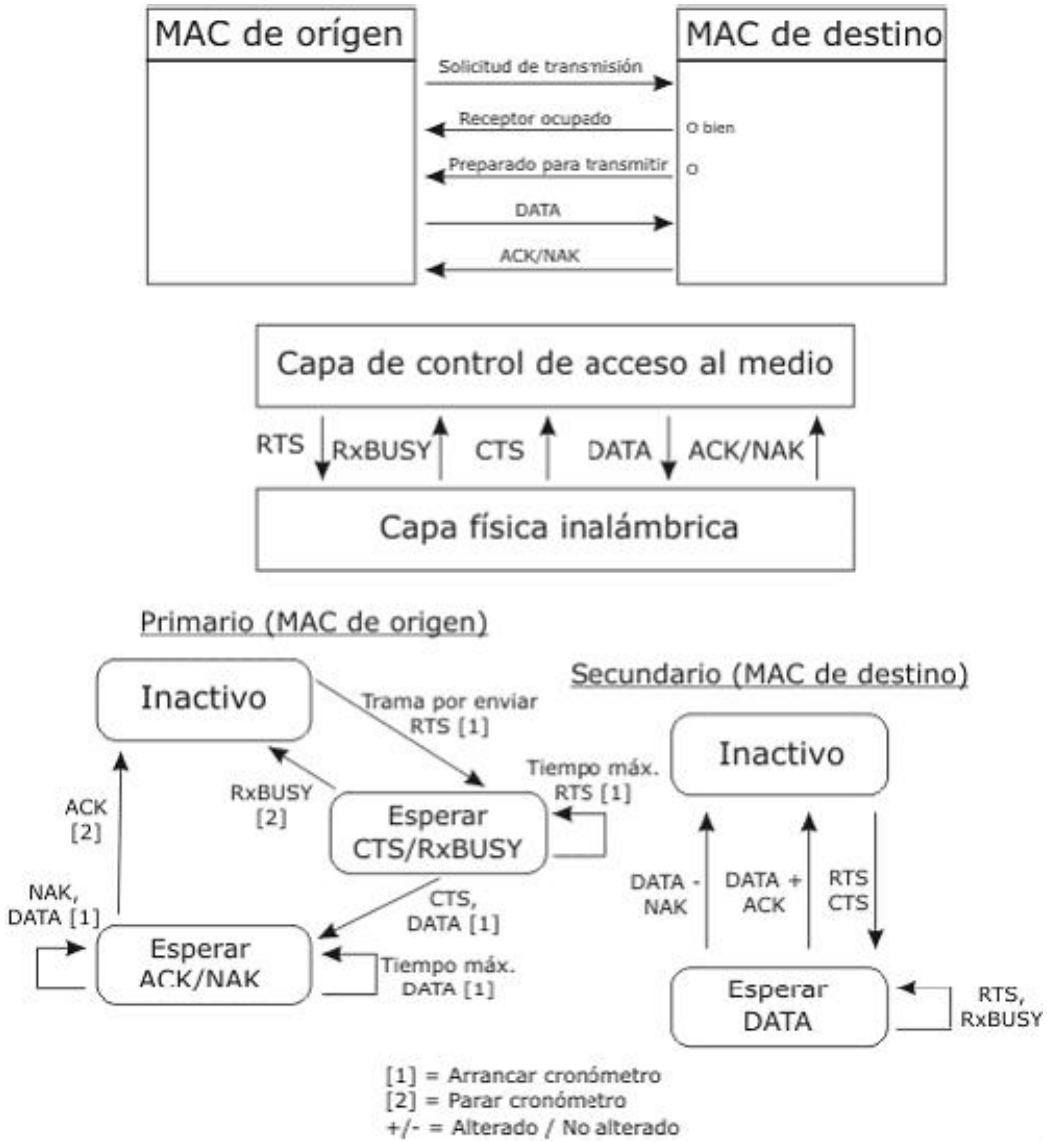


Figura 2.3: Diagrama de estados del protocolo MAC para 802.11

Existen situaciones en las que si los nodos solamente censan el medio (carrier sense) y a partir de esa información comienzan

a transmitir se generarán colisiones no esperadas. La situación que se describe se conoce como problema del 'nodo oculto' o 'hidden node problem'. Cuando un nodo A se encuentra en el límite del área de cobertura del AP pero no está al alcance de otro nodo C conectado al mismo AP se plantea que estos nodos están 'ocultos'. Cuando A y C comienzan a enviar paquetes simultáneamente CSMA/CA no funciona y ocurren colisiones que corrompen los datos. En una red ad-hoc la situación es análoga si pensamos que en vez de un AP los nodos A y C, ocultos entre sí, desean transmitir simultáneamente a un nodo B que está al alcance de ambos. Dependiendo de la ubicación física de los nodos varios pueden quedar ocultos unos de los otros.

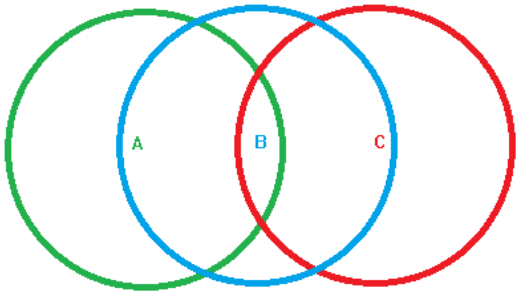


Figura 2.4: Problema del Nodo Oculto

Para resolver este problema el estándar propone un mecanismo de intercambio o handshaking [23] usando los paquetes Request To Send (RTS) y Clear To Send (CTS) que se mencionaron cuando se describió la capa MAC.

Este protocolo funciona de la siguiente manera: Un nodo que quiere enviar datos inicia el proceso enviando una trama RTS. El nodo destino responde con CTS. Cualquier otro nodo que reciba el RTS o el CTS debe abstenerse de enviar datos por un tiempo dado. Este intervalo está indicado en las trama RTS y CTS y representa la duración esperada de la transmisión entre los nodos que iniciaron el proceso. De esta forma se evitan las colisiones producidas por el problema del nodo oculto.

Un último detalle es que además se establece un umbral o threshold a partir del cual se activará el mecanismo. Típicamente las tramas RTS/CTS no se envían salvo que el tamaño del paquete a transmitir exceda el threshold (0 a 2347 octetos).

La capa descripta se imlpementa como otra componente del dispositivo inalámbrico, en genereal dentro de un microcontrolador de cada placa aunque alguna funcionalidad tambien se maneja en el driver del dispositivo. En algunos casos toda la funcionalidad de la capa MAC pasa al driver para abaratar costos.

Un driver adapta una interfaz común usada por los programas o el sistema operativo para un dispositivo específico. En Linux el driver mismo puede ser parte del sistema operativo o estar disponible como un módulo a cargarse por demanda o al detectar el dispositivo.

Desde Julio de 2007 se incluyó en el kernel de Linux (2.6.22) el módulo mac80211.ko que provee una API para desarrollar drivers donde parte del protocolo de acceso al medio se maneja por software (SoftMAC)[14]. Esto representó un gran avance en el soporte para placas wireless en Linux y casi todos los drivers han sido portados para usar este módulo.

Muchos dispositivos requieren un firmware para poder funcionar, tradicionalmente este bloque de código se grababa en una memoria ROM o Flash del dispositivo pero cada vez más frecuentemente el firmware debe ser cargado en el dispositivo por el driver durante la inicialización del dispositivo. Algunos de estos firmwares son libres y otros no, lo que implica que no siempre están disponibles en los repositorios oficiales de las distribuciones.

La API completa para redes inalámbricas en Linux lleva el nombre de Wireless Extensions[15], fue desarrollada por Jean Tourrilhes e introducida en el kernel en el año 1997. Esta com-

puesta por una entrada en /proc/net/wireless (pseudo filesystem que brinda estadísticas sobre el sistema) y tres herramientas: iwconfig (clon de ifconfig de redes cableadas), iwspy (diseñado para IP Móvil) y iwpriv (para soportar extensiones específicas de los dispositivos). El desarrollo de esta API está discontinuado y sólo se aceptan correcciones de bugs, principalmente porque utiliza la system call IOCTL() actualmente se están planteando como reemplazo cfg80211 y nl80211 que utilizan sockets netlink para intercambiar información entre el espacio de usuario y el espacio del kernel[16].

### 2.1.3. Evolución de las técnicas inalámbricas - Estándares 802.11g y 802.11n

Si bien a grandes rasgos ya se describieron las capas inferiores de las redes inalámbricas cabe mencionar las técnicas más recientes sobre la materia. Estas se engloban en el estándar 802.11g y el estándar 802.11n.

#### 802.11g

Este estándar no es tan reciente, se ratificó en 2003 y representa una evolución del 802.11b, utiliza la misma banda de 2.4Ghz pero opera a una velocidad teórica máxima de 54 Mbit/s, que en promedio es de 22 Mbit/s de velocidad real de transferencia.

A diferencia de 802.11b que usa DSSS para modular, este estándar utiliza Multiplexación por División de Frecuencias Ortogonales u Orthogonal Frequency Division Multiplexing (OFDM). Esta técnica combina modulación con codificación. Para trabajar divide un canal de RF en pequeños sub-canales, cada uno con su sub-carrier (sub-portadora). Cada sub-carrier es una portadora en sí usada para transportar datos de streams de información en paralelo.

En OFDM los canales se superponen parcialmente pero no interfieren entre sí (como ocurre en DSSS). Las frecuencias fundamentales son ortogonales (orthogonal sub-carriers), están separadas espacialmente de forma mínima, permitido por la ortogonalidad entre ellas, de esta forma se evita la interferencia entre los distintos streams.

El estándar IEEE 802.11a/g utiliza 52 sub-carriers en un canal con un ancho total de 20MHz. Se usan 48 sub-carriers para datos y 4 para pilots (canales de sincronización entre los carriers de información) y se utilizan sub-bandas exteriores reservadas para protección entre diferentes canales.

Los anchos de analógico de cada sub-canal son de 300 kHz. Los canales completos ocupan un ancho de 20MHz de forma similar a DSSS.

La parte de codificación de OFDM trabaja codificando los datos en los diferentes sub-carriers que genera. El método de codificación se conoce como Convolution Coding (CC). Esta codificación utiliza algunos sub-carriers para llevar datos y otros para generar códigos de corrección de errores, de forma similar a los bits de paridad o un CRC.[18]

802.11g tiene la ventaja de poder coexistir con los estándares 802.11a y 802.11b, esto debido a que puede operar con las Tecnologías DSSS y OFDM simultaneamente. Sin embargo, si se utiliza en conjunto con usuarios que trabajen con el estándar 802.11b, el rendimiento de la celda inalámbrica se verá afectado por ellos, permitiendo sólo una velocidad de transmisión de 22 Mbps. Esta degradación se debe a que los clientes 802.11b no comprenden OFDM.

Como veremos a continuación OFDM también es utilizado en 802.11n con algunos agregados.

**802.11n**

En enero de 2004, el IEEE anunció la formación de un grupo de trabajo 802.11 (TGn) para desarrollar una nueva revisión del estándar 802.11 y la versión n fue ratificada septiembre de 2009 anunciando una velocidad de 600 Mbps en capa física. 802.11n trabaja sobre las frecuencias de 2,4 GHz y/o de 5 GHz, siendo compatible hacia atrás. Es un estándar abierto, certificado por Wi-Fi Alliance (desde Draft 2.0). El objetivo del estándar es lograr mayores tasas de transferencias que las propuestas hasta el momento para redes WLAN, logra potencialmente hasta 600Mbps nominal (100 y 200Mbps real). Para lograr esto la subcapa PMD trabaja con OFDM donde los principales agregados a este nivel son: MIMO (Multiple Input - Multiple Output) y canales más anchos: de 20Mhz a 40Mhz.

La técnica MIMO permite aprovechar el uso de múltiples antenas para transmisión y recepción dando lugar a varios Radio Chains: múltiples caminos físicos. En los estándares previos, a/b/g el multipath, es "destructivo" ya que aumenta el ruido, pues se produce interferencia entre las mismas transmisiones al llegar a destiempo. En cambio MIMO aprovecha este fenómeno físico porque dispone de más de una antena independiente. Por otro lado esto implica un mayor consumo eléctrico para estos dispositivos. Cabe aclarar que algunos dispositivos de estándares anteriores cuentan con más de una antena pero no son independientes, sino simplemente toman la señal de la antena que reciba con mayor potencia.

En cuanto a los canales, la modificación para obtener mejor rendimiento permite agrupar 2 (dos) canales de 20MHz en uno de 40MHz, esto se conoce como Channel Bonding. Para ser compatible con el estándar IEEE 802.11n un canal de 20MHz es obligatorio, la agrupación de canales es opcional. Uno de los canales es considerado de control, "master", o primario, y el otro

secundario. Este último se puede ubicar por encima o por debajo del de control. Al producirse el agrupamiento sobre 40MHz ya no se requieren sub-bandas exteriores reservadas para protección/guardas entre los canales agrupados (en el medio). De esta forma se permite tener una configuración de 114 sub-carriers, de los cuales se usan 108 para datos y 6 para pilots, a diferencia de los 48 sub-canales para datos de 802.11a/g.

Actualmente el modo 802.11n se soporta en linux para los siguientes drivers: ath9k, iwlagn y ar9170 [19]

### 2.1.4. Modos de Operación

Además de los mecanismos para transmitir por el aire descriptos anteriormente una WLAN puede conformarse a partir de dos modos operativos distintos definidos en el estándar: infraestructura y ad-hoc. El modo de infraestructura es en el que los hosts se conectan a un punto de acceso o access point (AP). Éste es por lo general el modo predeterminado para las placas de red inalámbricas. Y el modo ad-hoc es en el que los hosts se conectan entre sí sin ningún punto de acceso.

#### Modo Infraestructura

En el modo de infraestructura, cada host se conecta a un punto de acceso a través de un enlace inalámbrico. La configuración formada por el punto de acceso y los hosts ubicados dentro del área de cobertura se llama conjunto de servicio básico o Basic Service Set (BSS). Estos forman una célula. Cada BSS se identifica a través de un identificador de BSS (BSSID) que es un identificador de 6 bytes (48 bits). En el modo infraestructura el BSSID corresponde a la dirección MAC del punto de acceso.

Es posible vincular varios puntos de acceso juntos (varios BSS) con una conexión llamada sistema de distribución (o SD) para formar un conjunto de servicio extendido o ESS. Dicho

sistema de distribución puede ser una red conectada, un cable entre dos puntos de acceso o incluso otra red inalámbrica.

Un ESS se identifica a través de un ESSID (identificador del conjunto de servicio extendido), que es un identificador de 32 caracteres en formato ASCII que actúa como su nombre en la red. El ESSID, a menudo abreviado SSID, muestra el nombre de la red y de alguna manera representa una medida de seguridad de primer nivel ya que una estación debe saber el SSID para conectarse a la red extendida.

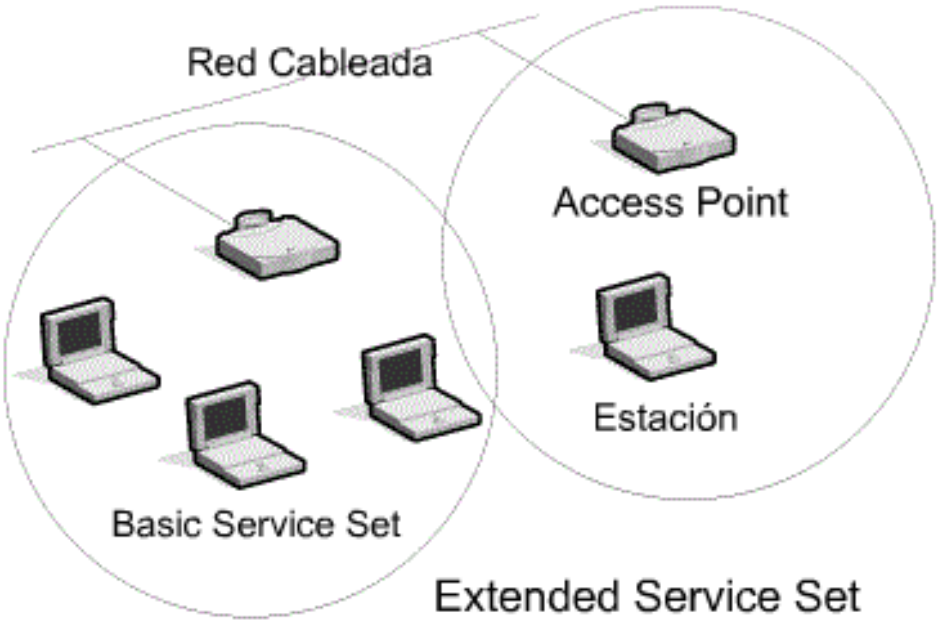


Figura 2.5: Esquema de un ESS conformado por dos BSS unidos por un SD (Red cableada)

Cuando un usuario itinerante va desde un BSS a otro mientras se mueve dentro del ESS, la placa de red inalámbrica de su equipo puede cambiarse de punto de acceso según la calidad de la señal que reciba desde distintos puntos de acceso. Los puntos de acceso se comunican entre sí a través del sistema de distribución con el fin de intercambiar información sobre los nodos.

Esta característica que permite a los nodos moverse de forma transparente de un punto de acceso al otro se denomina itinerancia, un host dentro del rango de muchos puntos de acceso (que tengan el mismo SSID) puede elegir el punto que ofrezca la mejor proporción entre capacidad de carga de tráfico y carga de tráfico actual. Si los puntos de acceso no tienen el mismo SSID es el cliente el que decide a cual conectarse.

#### Modo Ad-Hoc

En el modo ad-hoc cada host inalámbrico se conecta con cada uno de los otros para formar una red punto a punto, es decir, una red en la que cada equipo actúa como cliente y como punto de acceso simultáneamente. La configuración que forman los hosts se llama conjunto de servicio básico independiente o IBSS.

Un IBSS es una red inalámbrica que tiene al menos dos estaciones y no usa ningún punto de acceso. Por eso, el IBSS crea una red temporal que le permite a los usuarios que estén en la misma sala intercambiar datos. Se identifica a través de un SSID de la misma manera en que lo hace un ESS en el modo infraestructura [22].

En una red ad hoc, el rango del BSS independiente está determinado por el rango de cada host. Esto significa que si dos hosts de la red están fuera del rango del otro, no podrán comunicarse, ni siquiera cuando puedan "ver" a otros hosts. A diferencia del modo infraestructura, el modo ad hoc no tiene un sistema de distribución que pueda reenviar tramas de datos desde un host a otro.

El cambio de modo no tiene impacto en la capa física y dentro de la capa MAC se mantiene CSMA/CA y la mayoría de los tipos de mensajes y su uso. Pero la carencia de un AP implica que la WLAN debe encargarse de más responsabilidades de la capa MAC.

La red se inicia con la primer estación ad-hoc que se activa. Ésta establece el IBSS y comienza a enviar mensajes Beacon (baliza o faro), que son necesarios para establecer la sincronización entre las estaciones. En modo infraestructura estos mensajes los envía solamente el AP. Estos mensajes de la capa MAC incluyen un timestamp a partir del cual todas las estaciones sincronizan sus clocks, un intervalo, que especifica cada cuanto se enviarán los mensajes Beacon, prestaciones del dispositivo o la red (Capability Information) donde se señala que el modo está seteado en ad-hoc, el SSID y el IBSS que fue generado al azar (dado que no hay dirección de MAC de ningún AP), entre otros. Otras estaciones ad-hoc pueden unirse directamente a la red luego de recibir un mensaje Beacon y aceptar los parámetros que especifica, la autenticación no está definida para el modo ad-hoc. A partir de ese momento todas las estaciones deben enviar un mensaje Beacon de manera periódica si no lo reciben de otra estación en el período esperado. Esta espera no es exacta se le agrega un breve delay al azar que reduce la cantidad de estaciones que transmiten efectivamente el mensaje. Si una estación no recibe el Beacon asume que no hay otras estaciones activas y por lo tanto debe enviarse un Beacon. Si en cambio lo recibe, cada estación usa la información para sincronizar su clock interno de forma de asegurar que todas las estaciones son capaces de operar.

Otra situación a analizar es cuando una estación entra en estado PS (Power Save). Al comenzar el ahorro de energía setea el bit de control de power management dentro de las tramas, de la misma forma que lo hace en modo infraestructura. Todas las otras estaciones toman concimiento de esto monitoreando dicho bit en todas las tramas que reciben. En base a esa información dejan de transmitir hacia esa estación y almacenarán los paquetes correspondientes de manera local.

Regularmente todas las estaciones dormidas se despiertan pa-

ra escuchar los Beacons. Si una estación está reteniendo paquetes que deben ser enviados a otra dormida, la primera envía una trama que incluye en el campo Traffic Indication Map (TIM) la identificación de la estación a la cual se le quieren entregar paquetes. La estación dormida se entera de que debe mantenerse despierta durante el intervalo de envío del próximo Beacon esperando que sea lo suficientemente largo como para que la estación que está almacenando los paquetes pueda enviárselos. Luego de recibirlos y avisar que fueron aceptados la estación puede volver a dormir[20].

En cuanto al ahorro de energía se pierde funcionalidad con respecto al modo infraestructura debido a que en ese caso la estación que duerme no debe continuar escuchando periódicamente sino que al despertarse le solicita al AP que le envíe los paquetes que haya almacenado. En el modo ad-hoc tampoco se tiene la capacidad de redireccionar paquetes entre dos estaciones que no están al alcance una de otra, por lo menos a nivel de MAC. Y por último otra funcionalidad que no está disponible es la intensidad de señal, dado que no es posible manejar esta información para todas las estaciones.

En concreto, para setear una placa de red en el modo ad-hoc, debemos acceder a las opciones proporcionadas por el driver de la misma. Cada sistema operativo provee mecanismos para acceder a esta configuración.

En Linux con wireless-extensions podemos configurar temporalmente una placa inalámbrica detectada como wlan0 con los siguientes comandos:

```
ifconfig wlan0 down          #desactivamos la placa
iwconfig wlan0 mode ad-hoc   #forzamos el modo ad-hoc
iwconfig wlan0 essid MIRED   #identificador de red <- MIRED
```

En este caso la dirección IP de la placa debe ser manejada de alguna manera (algún nodo puede instalar un servidor DHCP u otra configuración manual).

Si esta configuración quiere hacerse permanente debe editarse el archivo /etc/network/interfaces agregando el siguiente contenido:

```
auto wlan0
iface wlan0 inet static    #estatica -> sin dhcp
    address 192.168.1.1    #direccion IP fija
    netmask 255.255.255.0  #mascara de subred correspondiente
    wireless-channel 1     #canal por el cual transmitir
    wireless-essid MIRED   #essid puede ser cualquiera
    wireless-mode ad-hoc   #forzar el modo ad-hoc por defecto
```

Con estas configuraciones se debe tener en cuenta que cada uno que quiera conectarse a la red debe conocer de antemano el ESSID, distintos ESSID generarán distintas redes. Como comentamos anteriormente no se define el BSSID (que en infraestructura correspondía a la MAC del AP), sino que es generado automáticamente y su contenido es aleatorio.

Otra alternativa para la configuración es seguir los siguientes pasos. Primero el sistema debe haber detectado la placa y debe estar instalado el driver correcto. Si contamos con una interfaz Gnome y Network-Manager instalado podemos verificar esto y setear manualmente el modo ad-hoc desde la interfaz gráfica, mediante el aplett del Network-Manager eligiendo Editar Conexiones e ir al tab de Wireless. Luego clickear "Agregar" para editar una nueva conexión, en el dialogo introducimos los datos correspondientes, SSID lleva el nombre de la red que será visible por los otros nodos, en Modo colocamos Ad hoc y MTU queda seteado en automático. El resto de los valores quedan en blanco y sólo falta colocar una IP en el tab de IPv4.

### 2.1.5. Seguridad

Los estándares de seguridad wireless han ido evolucionando y son un tema de estudio muy activo actualmente. Aquí mencionaremos brevemente las técnicas de autenticación y encriptación y el análisis de seguridad se hará específicamente sobre el sistema

Live desarrollado. Analizar todas las consideraciones de seguridad a nivel general en entornos wireless va más allá del alcance de este trabajo. Se han tomado las consideraciones generales de [17] y allí pueden profundizares estos temas.

**WEP (Wired Equivalent Protocol)**

El protocolo WEP es un sistema de encriptación estándar propuesto por el comité 802.11, implementada en la capa MAC y soportada por la mayoría de vendedores de soluciones inalámbricas. WEP comprime y cifra los datos que se envían a través de las ondas de radio dentro de la placa de red. Encripta el cuerpo y el CRC de cada trama 802.11 antes de la transmisión utilizando el algoritmo de encriptación RC4 proporcionado por RSA Security. La estación receptora, sea un punto de acceso o una estación cliente, es la encargada de desencriptar la trama.

Como parte del proceso de encriptación, WEP prepara una estructura denominada 'seed' obtenida tras la concatenación de la llave secreta proporcionada por el usuario de la estación emisora con un vector de inicialización (IV) de 24 bits generada aleatoriamente. La estación cambia el IV para cada trama transmitida.

A continuación, WEP utiliza el "seed" en un generador de números pseudo-aleatorio que produce una llave de longitud igual a el payload (cuerpo más CRC) de la trama más un valor para chequear la integridad (ICV) de 32 bits de longitud.

El ICV es un checksum que utiliza la estación receptora para recalcularla y compararla con la enviada por la estación emisora para determinar si los datos han sido manipulados durante su envío. Si la estación receptora recalcula un ICV que no concuerda con el recibido en la trama, ésta queda descartada e incluso puede rechazar al emisor de la misma.

WEP especifica una llave secreta compartida de 40 o 64 bits

para encriptar y desencriptar, utilizando la encriptación simétrica.

Antes de que tome lugar la transmisión, WEP combina la llave con el payload/ICV a través de un proceso XOR a nivel de bit que producirá el texto cifrado. Incluyendo el IV sin encriptar sin los primeros bytes del cuerpo de la trama.

La estación receptora utiliza el IV proporcionado junto con la llave del usuario de la estación receptora para desencriptar la parte del payload del cuerpo de la trama. Cuando se transmiten mensajes con el mismo encabezado, por ejemplo el FROM de un correo, el principio de cada payload encriptado será el mismo si se utiliza la misma llave. Tras encriptar los datos, el principio de estas tramas será el mismo, proporcionando un patrón que puede ayudar a los intrusos a romper el algoritmo de encriptación. Esto se soluciona utilizando un IV diferente para cada trama.

La vulnerabilidad de WEP reside en la insuficiente longitud del Vector de Inicialización (IV) y lo estáticas que permanecen las llaves de cifrado, pudiendo no cambiar en mucho tiempo. Si utilizamos solamente 24 bits, WEP utilizará el mismo IV para paquetes diferentes, pudiéndose repetir a partir de un cierto tiempo de transmisión continua. Es a partir de entonces cuando un intruso puede, una vez recogidas suficientes tramas, determinar incluso la llave compartida.

A pesar de todo, WEP proporciona un mínimo de seguridad para pequeños negocios o instituciones educativas, si no está deshabilitada, como se encuentra por defecto en los distintos componentes inalámbricos.

#### OSA (Open System Authentication)

A diferencia del anterior no se trata de un mecanismo de encriptación sino de un mecanismo de autenticación que en realidad no realiza chequeos, todas las peticiones recibidas son au-

tenticadas. Es el algoritmo de autenticación usado cuando no se selecciona ninguno alternativo. El principal problema que tiene es que no realiza ninguna comprobación de la estación cliente, además las tramas de gestión son enviadas sin encriptar, aún activando WEP, por lo tanto es un mecanismo poco fiable.

#### WPA/WPA2 (Wi-Fi Protected Access)

WPA también es un mecanismo de autenticación de usuarios, que adopta el uso de un servidor, donde se almacenan las credenciales y contraseñas de los usuarios de la red. Para no obligar al uso de tal servidor para el despliegue de redes, WPA permite la autenticación mediante clave compartida o Pre-Shared Key (PSK), que de un modo similar a WEP, requiere introducir la misma clave en todos los equipos de la red. WPA2 es la estandarización completa de dicho mecanismo (802.11i) dado que WPA se basó en un draft por la urgencia de brindar seguridad en redes inalámbricas. WPA2 introduce CCMP, un nuevo modo de encriptación basado en AES de alta seguridad.

#### ACL (Access Control List)

Este mecanismo de seguridad es soportado por la mayoría de los productos comerciales. Utiliza, como mecanismo de autenticación, la dirección MAC de cada estación cliente, permitiendo el acceso a aquellas MAC que estén incluidas en la Lista de Control de Acceso. Evidentemente este mecanismo no es práctico si los nodos no son siempre los mismos.

#### CNAC (Closed Network Access Control)

Este mecanismo pretende controlar el acceso a la red inalámbrica y permitirlo solamente a aquellas estaciones cliente que conozcan el nombre de la red (SSID) actuando este como contraseña.

Así se evitan asociaciones no autorizadas aunque este mecanismo es vulnerable dado que el SSID se envía en las association request y estos paquetes podrían ser tomados por un dispositivo que capture los paquetes.

## 2.2. Zeroconf

En la sección anterior se describieron las tecnologías de las capas más cercanas al hardware, a partir de aquí comienza el análisis enfocado en capas superiores de abstracción. Esto implica protocolos y estándares de la capa de red y de transporte, y también aplicaciones de estas capas o que hacen uso de las mismas. Se mencionarán conceptos relacionados con direcciones IP, nombres de dominios de red y aplicaciones que hacen uso de paquetes UDP/TCP con distintas finalidades. Estos conceptos no son estrictamente de redes ad-hoc sino de redes en general. El diseño por capas permite aplicar estos conceptos tanto sobre redes WLAN que acabamos de describir, como sobre otras redes de área local.

Muchos protocolos TCP/IP como DHCP, DNS o LDAP requieren administración y mantenimiento pero hay redes en las que esto no es aceptable, por ejemplo en las redes hogareñas o en las redes ad-hoc, dado que en definitiva podrían ser solamente dos computadoras conectadas por un enlace inalámbrico. En estas redes no existen administradores y los usuarios de no tienen el tiempo o el interés en adquirir los conocimientos de administración necesarios. En cambio estas redes precisan protocolos que requieran cero configuración o cero administración por parte del usuario. Zeroconf[10] es una de las formalizaciones para definir dichos protocolos con “cero configuración” y es la que analizaremos. La otra formalización, que no es directamente comparable con Zeroconf, está planteada por el Universal Plug and Play Forum (UPnP Forum). Su principal objetivo es definir

nuevos protocolos entre los que se incluyen Viiv, DLNA, DHWG y UPnP y apuntan a la capa de aplicación. El punto de contacto o solapamiento se da en torno al anuncio servicios [24].

Zeroconf se focaliza en tres tecnologías: configuración automática de direcciones IP, búsqueda de nombres de los hosts y anuncio de servicios disponibles (también se buscaba controlar la alocación de direcciones IP multicast pero todavía no se estandarizó). Está específicamente enfocado en realizar estas tareas sin una estructura centralizada, por lo que sus propósitos en realidad son las siguientes:

- Alocar direcciones IP dinámicas sin un servidor DHCP
- Traducir nombres y direcciones IP sin un servidor DNS
- Encontrar servicios, por ejemplo impresoras, sin un servicio de directorio corriendo en un servidor.

Se propone lograr estos objetivos mediante los mecanismos que se detallan a continuación. En general se evidencia como línea general la intención de reutilizar protocolos o tecnologías preexistentes en vez de intentar definir protocolos nuevos[21].

Para el caso de la configuración automática de la dirección IP, en la estandarización[11] se especifica básicamente el siguiente comportamiento para IPv4: Los hosts eligen una dirección al azar en el rango 169.254.0.0/16 y luego mandan un requerimiento ARP para verificar si otro host está usando dicha dirección. Si es así eligen otra dirección IP. Este comportamiento se toma del mecanismo ya existente definido en el RFC 5227[12]. Esto se conoce como direcciones de enlace local auto-asignadas o (Link-Local) Self-Assigned IP addresses a veces abreviado (IPv4LL), para Microsoft lleva el nombre de Automatic Private IP Addressing (APIPA) o a veces Internet Protocol Automatic Configuration (IPAC).

Para la resolución de nombres existen dos intentos de estandarización: Multicast DNS (mDMS) y Link-Local Multicast Name Resoulution (LLMNR). Ambos protocolos son similares, LLMNR casi no se utiliza pero ha avanzado mucho más en la formalización del estándar. Estos protocolos trabajan de la misma forma: envían consultas similares a las DNS a una dirección multicast en la red local, pero los protocolos son incompatibles entre sí. Alguna de las diferencias vienen dadas por la forma en que manejan los nombres propiamente dichos. mDNS usa el espacio de nombres .local mientras que LLMNR permite a los dispositivos elegir cualquier nombre, lo que es considerado un riesgo de seguridad por la Internet Engineering Task Force (IETF) que es la encargada de regular los RFC. Por otro lado mDNS puede trabajar sin problemas con el servicio de directorio de DNS (DNS-SD) mientras que LLMNR es incompatible con éste.

Normalmente si un host quiere saber el nombre de otro host envía una consulta al servidor DNS central, como en este caso no hay servidores centrales, el host envía su requerimiento a una dirección IP multicast en la que están escuchando todos o la mayoría de los hosts de la red local. Aquel que conozca la respuesta a una consulta específica la responderá. Los demonios mDNS escuchan en la dirección 225.0.0.251 en el puerto 5353 mientras que los LLMNR escuchan en la dirección 225.0.0.252 puerto 5355.

Para anunciar y encontrar los servicios se definieron varios estándares entre los que se incluyen: DNS Service Discovery(DNS-SD), Simple Servicie Discovery Protocol (SSDP) que forma parte de Universal Plug and Play (UPnP) y Service Location Protocol (SLP). Existen otros que dependen de servidores de directorio centrales por lo que no son aptos para el sistema propuesto. DNS-SD funciona con multicast DNS así como también con un servicio de directorio dedicado DNS ordinario. Esto último se

conoce como Wide Area Bonjur en el entorno Apple. Como utiliza registros DNS funciona particularmente bien con mDNS. Esta basado principalmente en el uso de registros SRV pero usa otro nivel de indirección generando registros PTR que apuntan a registros SRV y TXT.

Por otro lado SSDP se considera más complejo dado que utiliza anuncios por notificaciones HTTP para encontrar servicios combinando un tipo de URI de servicio única y un nombre de servicio único o Unique Service Name (USN). Al ser parte de UPnP se encuentra más formalizado lo que puede ser una ventaja o desventaja dependiendo del punto de vista.

## 2.3. Avahi

Existen varias implementaciones de los protocolos Zeroconf que se comentaron en el inciso anterior, las principales son: Apple Bonjour (conocido anteriormente como Rendezvous), la implementación de Microsoft de LLMNR para Windows CE 5.0 y Avahi, la implementación para Linux y BSD. Avahi implementa IPv4LL, mDNS y DNS-SD, es parte de la mayoría de las distribuciones Linux y aunque no viene instalado por defecto en muchas de ellas esta tendencia está cambiando. Avahi implementa también librerías de compatibilidad que lo hacen interoperable con el protocolo de Apple y la implementación histórica de mDNS denominada Howl. Actualmente no ha sido portado a Windows.

En Linux la implementación está disponible a partir de los paquetes básicos que la componen y son los siguientes:

- avahi-daemon: Es el demonio que se ejecuta en el background, escucha los anuncios de servicios en la red y anuncia los servicios propios. Para otras aplicaciones u otros procesos es posible comunicarse con este demonio por D-bus, la

interfase de comunicación entre procesos (IPC) de freedesktop.org. Los administradores pueden colocar también fragmentos cortos de XML en la carpeta /etc/avahi/services para publicar servicios semi-estáticos.

- avahi-autoip: Este paquete entra en funcionamiento cuando el host no puede obtener una dirección IP. Si espera por un servidor DHCP esto se produce 30 segundos después de no obtener respuesta. Este paquete no depende de ninguna librería de avahi de modo que puede usarse aún sin instalar avahi. Para detectar los cambios en las configuraciones de red usa la interfaz IPC Netlink y en general se presenta como un plug-in para el programa cliente DHCP.

- avahi-dnsconfd: Este demonio escucha en la red por anuncios de servidores DNS y los pasa a resolvconf para usarlos en la resolución de nombres. Cada host que no tenga un nombre de dominio asignado se le asignará ".local" quedando el nombre completo del host como "hostname.local". Al querer resolverse un nombre dentro de este dominio, el requerimiento se envía a un destino multicast o sea que se le pregunta a todos los host quien posee ese nombre, el destinatario responde también a un destino multicast y de esta forma el que consultó obtiene su respuesta. Con este demonio se mantienen los nombres ya resueltos para no aumentar el tráfico en la red si ya se había consultado por estos.

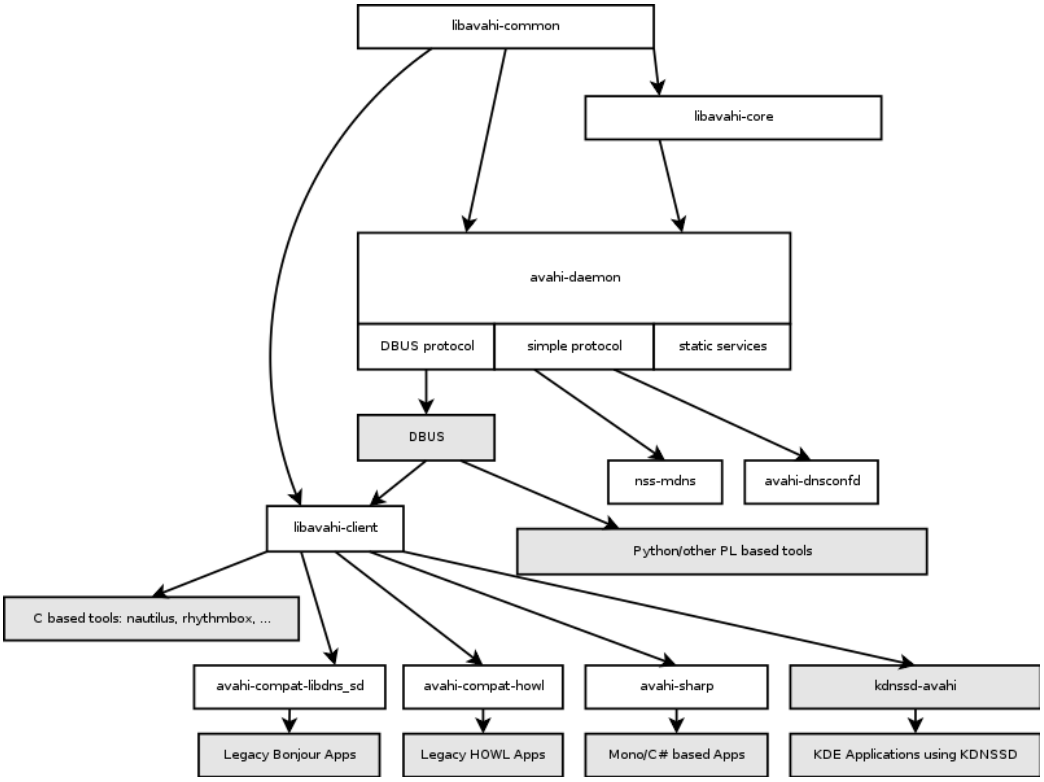


Figura 2.6: Componentes de software de Avahi

## 2.4. Aplicaciones para redes LAN – soporte avahi y aplicabilidad al sistema en base a la funcionalidad esperada

El análisis realizado hasta aquí nos permite sentar las bases para el sistema propuesto. Repasando un poco vemos que contamos con soporte en Linux para placas wireless con sus respectivos drivers, administración de la configuración de dichas placas (en particular establecer el modo ad-hoc) y herramientas para asignarles una dirección IP, resolver nombres y descubrir servicios en forma descentralizada. El paso siguiente son las aplicaciones.

El sistema de capas permite a las aplicaciones de usuario ac-

ceder a servicios de red independientemente de los detalles subyacentes. Esto abre un gran abanico de aplicaciones y protocolos específicos en este nivel.

A continuación se detallan los componentes para el sistema propuesto. Este listado es solamente orientativo y para elaborarlo se tuvieron en cuenta las necesidades técnicas para el desarrollo, la funcionalidad mínima para el sistema propuesto y la usabilidad del sistema final enfocándose esto último en la necesidad de que el sistema funcione sin la intervención del usuario para configurarlo. En los próximos capítulos se profundiza el estudio de algunos de estos componentes, que en líneas generales son los siguientes:

- Herramienta para confeccionar LiveCD que permita personalizar la imagen a crear y de ser posible que la imagen generada pueda clonarse a un pendrive.
- Sistema Operativo base con soporte para las herramientas detalladas.
- Soporte para drivers de placas de red Wi-Fi con la posibilidad de fijar el modo ad-hoc.
- Asignación de direcciones IP en forma descentralizada.
- Asignación y resolución de nombres de dominio (DNS) de forma descentralizada.
- Anuncio de los servicios disponibles en cada host.
- Herramienta administrativa de la red (visualizar hosts conectados y sus servicios).
- Aplicaciones LAN (no esenciales y preferentemente con soporte para anunciarse como servicio)

  Servidor ftp

Aplicación de Chat / videochat

Servidor web

Streaming de audio / vídeo

Blog / Microblogging

Wiki o alguna forma de trabajo colaborativo

Juegos para jugar en red

- Ruteo: como ampliación del sistema podría pensarse el caso de que algunos hosts tuvieran además acceso a otras redes (o Internet) y que sea posible que funcionen como routers para el resto de los hosts y así compartir el acceso a estas redes.

Éstas especificaciones se resolvieron con herramientas ya descriptas o que se analizan más adelante. La elección de las mismas se basa en la aplicabilidad al problema, la documentación disponible, su usabilidad (tanto para el desarrollador como para el usuario final) y los análisis realizados. En algunos casos también influyó el gusto personal.

Como distribución Linux se eligió Debian, por ser una distribución robusta que sirve como base para muchas otras aunque esta decisión podría revisarse en futuros desarrollos debido a la falta de soporte para el hardware más reciente del mercado en esta distribución (en general drivers propietarios). Para los servicios descentralizados se eligió Avahi por ser una implementación robusta y reconocida por la gran cantidad de distintas distribuciones que incorporan dicho software. Como interfaz de usuario se eligió Gnome porque tiene más aceptación para los usuarios que recién se inician en el mundo Linux aunque esta decisión también podría reconsiderarse.

Para manejar las conexiones de red se eligió Network-Manager. Si bien esta herramienta tiene detractores dentro de los administradores de redes brinda gran flexibilidad a los usuarios finales.

En general presenta algunos conflictos con configuraciones de red poco convencionales pero no hubo mayoers inconvenientes en este entorno. Elegir en cambio una configuración estática hubiera forzado a tener que editar el archivo de configuración si se quiere dejar la red ad-hoc y conectarse a otra red, algo que la mayoría de los usuarios no está acostumbrado a hacer. Esta es una de las principales razones por las cuales se elige esta herramienta, como hemos visto la misma configuración puede hacerse manualmente.

Como herramienta para la visualización de los servicios anunciados por la red se eligió Avahi Discovery, es una interfaz sencilla y permite visualizar el detalle de la información anunciada por la red, no está destinada a usuarios finales pero es útil para propósitos informativos. Adicionalmente se incluyó el cliente de chat empathy porque se anuncia usando avahi y como demostración de una aplicación para redes. Adicionalmente se trabajó con cliente y servidor de escritorios remotos o Virtual Computing Network (VNC). Aprovechando que Gnome trae por defecto un cliente vnc denominado Vinagre se agregó el servidor denominado Vino. Estas aplicaciones se anuncian mediante avahi y permiten que una computadora ejecutando el cliente pueda ver en una ventana el escritorio de otra computadora ejecutando el servidor y compartiendo su escritorio. Esta aplicación resulta muy útil en entornos educativos o para realizar streaming básico (el protocolo trabaja a nivel de transmitir cada pixel de la pantalla por la red).

A continuación se listan algunas de las muchas aplicaciones con soporte para avahi, estas en particular son para Debian Gnome:

- Servidor FTP: wzdftpd + wzdftpd-mod-avahi (no disponible en Debian Lenny pero si en Squeeze)
- Browsear los servicios: Avahi Discovery - Avahi Bookmarks

- Chat: Pidgin - Empathy
- VNC: vinagre cliente (ya viene con Gnome) - vino servidor (escritorios remotos)
- Escuchar y compartir musica: Rhythmbox
- Editor Colaborativo: Gobby - hay que trabajar bastante para lograr la personalizacion
- Juego: glChess, ajedres con soporte para red incluido en el paquete gnome-games

Podrían considerarse otras aplicaciones que sean sólo para redes y anunciarlas por la red de forma semi-estatica o agregando como desarrollo el soporte para avahi.

Por último mencionaremos otras aplicaciones accesorias que se utilizaron en el desarrollo, instalación o manipulación final del sistema. Por un lado debemos manipular los archivos de imagen del sistema final. Estos archivos son .iso para el caso de los CD y .img para el caso de los discos USB. Para escribirlo a un CD se utilizó la herramienta para grabar discos Brasero aunque puede elegirse cualquier otra que tome este formato (que actualmente está bastante difundido). Alternativamente para probar alguna funcionalidad de estas imágenes se usó Virtual Box. Este software permite virtualizar una PC y ejecuta dentro de ésta una gran cantidad de sistemas operativos usando como hardware la emulación brindada por la aplicación y un archivo de imagen del SO. En particular se configuró una virtualización para que se arranque desde la imagen .iso provista como si fuera un CD real. Sólo alguna funcionalidad del sistema pudo probarse con esta heramienta porque no brinda emulación para placas wireless. Para el caso de la imagen .img (o imagen .iso híbrida) su manipulación no está tan difundida. Si no contamos con dicha imagen para lograr que un disco USB se booteable debe formatearse el Master Boot Record (MBR) del dispositivo para que

el hardware sepa donde buscar el programa de inicio. Por otro lado se deben copiar los archivos del sistema a una partición del dispositivo a la que se la debe etiquetar como booteable. Las herramientas que soportan creación de imágenes .img ya incluyen en dicha imagen el MBR, la partición etiquetada y los archivos del sistema. Sólo resta copiar bit a bit dicha imagen sobre el dispositivo (si la imagen es más pequeña que la capacidad total del dispositivo habrá que formatear el espacio restante para poder aprovecharlo). En linux esta funcionalidad se puede obtener con la herramienta de línea de comandos "dd" que trabaja con datos en bruto o raw data. Simplemente se especifica desde qué archivo se copiará y hacia qué dispositivo y el comando realiza la escritura de bajo nivel. Existe un excelente port de esta herramienta para Windows denominada "dd for windows" (`http://www.chrysocome.net/dd`) bajo la licencia GPL. Otra herramienta más simple para el entorno gráfico de Windows es Image Writer for Windows [25].

## 2.5. Distribuciones Linux Live en general

La forma de proporcionar toda la funcionalidad planteada a un usuario final sin que este deba modificar su sistema operativo es proveer dicho sistema operativo ya configurado y listo para usarse. Un LiveCD es un CD o DVD que contiene un sistema operativo booteable. De esta forma se puede correr un sistema operativo moderno y completo en una computadora sin necesidad de almacenamiento secundario sobre el cual escribir. Los LiveUSB comúnmente tiene además la funcionalidad agregada de escribir los cambios en el medio de booteo de forma transparente.

Originalmente se utilizaban los diskettes como discos de booteo, esto limitaba el tamaño y las herramientas con las que se podía contar. Con la masividad de las lectoras de CD y el aba-

ratamiento de la memoria RAM se hizo posible cargar el kernel de Linux, un manejador de ventanas y aplicaciones con interfaz gráfica sin tocar ningún sistema operativo instalado en un disco rígido. Debido a que se percibía y se sigue percibiendo como difícil o riesgos agregar una partición al disco rígido además de seguir los pasos de una nueva instalación, el concepto de LiveCD comenzó a cobrar fuerza. En 1998 se lanzó DemoLinux[13], la primer distribución de Linux específicamente diseñada como LiveCD que aunque no fue actualizada desde 2002 todavía se puede descargar de Internet. Knoppix, derivado de Debian, se lanzó en 2003 y ganó mucha popularidad como disco de rescate y como distribución primaria. Desde entonces creció ampliamente la disponibilidad de distribuciones Live y a la fecha (fines de 2011) distrowatch.com lista 191 distribuciones Live activas. Por otro lado varias de las distribuciones Linux en general ofrecen un LiveCD como soporte preferido para el programa de instalación.

Una de las principales técnicas usadas por los sistemas Live es combinar el sistema de archivos de sólo lectura de los CD con un sistema de archivos temporales en la forma de un disco RAM o RAM disk, en casos de distribuciones pequeñas, todo el sistema se carga en RAM. Comúnmente las carpetas /home y /var (que contienen respectivamente los archivos personales y de configuración y los datos variables) son actualizadas frecuentemente por el sistema por lo tanto se elige colocarlas en un RAM disk. En particular Puppy Linux tiene una capa que permite salvar dichos datos en posteriores sesiones dentro del cdrom o en un archivo contenido en cualquier dispositivo en el que pueda escribirse. Debian live permite también este tipo de persistencia si se etiqueta alguna partición del LiveUSB con el texto “live-rw”.

Además de los LiveCD basados en Linux se destacan entre muchos otros BeleniX de Open Solaris, AmigaOS 4 para powerPC, FreeBSD, FreeDOS y BartPE, sistema de rescate basado

Windows PE, el entorno de pre-instalación de Windows.

## 2.6. Herramientas para la creación de LiveCD

A continuación se especifican algunas herramientas para la creación de LiveCD. En la parte de pruebas se pueden ver los resultados que se obtuvieron con algunas de estas. El análisis se profundiza sobre Debian Live debido a que es la utilidad elegida para construir el prototipo del sistema propuesto.

#### Linux Live scripts

Linux Live scripts[26] es una colección de scripts que permiten crear un Linux Live a partir de una distribución ya instalada. El sistema creado se puede bootear de un CD un disco USB, FlashDrive, etc.

Para construir el Live CD se debe instalar una distribución en una partición o carpeta, se instalan o remueven los paquetes deseados, luego se descargan los scripts y se edita el archivo .config según la configuración deseada. Cuando está todo listo se ejecuta el script "build", esto crea el árbol de directorios de la distribucion Live. Luego se ejecuta el script "make-iso.sh" o "bootints.hs" para generar una imagen para CD o para USB según corresponda.

Esta serie de scripts fueron creados por los desarrolladores de Slax, la versión Live de Slackware y proveen además un web-frontend limitado para personalizar un sistema Live basado en Slax en la url `www.slax.org/build.php`.

#### Remastersys

Remastersys [27] es un programa para sistemas basados en Debian o Ubuntu y permite remasterizar LiveCD basados en Debian o hacer un backup de un sistema completo incluyendo

los datos del usuario en un LiveCD instalable. Esto último fue su propósito original pero se utilizó ampliamente en la creación de LiveCD, actualmente el proyecto está descontinuado.

#### Woof

Woof [28] es una herramienta para Puppy Linux, destinada a versionar dicho sistema. Provee la siguiente funcionalidad: Descargar paquetes de otras distribuciones o paquetes PET de la distribución propia, incluir soporte para compilar en distintos lenguajes y automáticamente construir la imagen .iso con amplias opciones de personalización en cuanto a selección de paquetes y selección de archivos específicos dentro del proceso de versionado. El resultado final es otro Puppy Linux, lo que significa obtener un LiveCD que corre completamente en RAM, es compacto y soporta persistencia. Actualmente soporta paquetes de Debian, Ubuntu, Slackware, Arch, T2 y Puppy. También se ofrece la herramienta CD-Remaster que es más sencilla pero también más limitada.

#### Debian Live

El proyecto Debian Live[29] comenzó en el 2006, en ese momento ya existían varias herramientas para la creación de sistemas live basadas en Debian pero que presentaban algunas desventajas para la perspectiva de Debian. Partiendo de que es un sistema operativo universal (base para muchos otros) se buscó incluir Debian Live como un sub-proyecto de Debian, que refleje un solo estado de la distribución, dado que herramientas previas mezclaban distintas versiones de Debian. Por otro lado se busca portabilidad a la mayor cantidad de arquitecturas posibles y que se trabaje con paquetes Debian sin alterar, que no se incluyan paquetes que no están en el archivo oficial de Debian y que use un kernel Debian inalterado sin parches adicionales.

Esto resulta en un sistema robusto con gran soporte que genera imágenes compatibles con las distribuciones Debian y sus repositorios, entre otras ventajas.

Un sistema Live construido con Debian Live tiene varios componentes, entre los principales encontramos la imagen del kernel de linux que es un solo archivo con toda la funcionalidad de bajo nivel del sistema operativo, usualmente se nombra como vmlinuz seguido del número de versión. Otra componente es la imagen del disco RAM inicial (initrd) que es otro archivo en forma de un disco RAM preparado para el booteo de Linux. Contiene un programa de init, los módulos necesarios para montar la imagen del sistema (raíz) y scripts que usan dichos módulos. El tercer componente es la imagen del sistema, es una imagen del filesystem del sistema operativo. En sistemas de escritorio no Live, estos archivos se encuentran en una partición de un disco rígido formateada con algún filesystem con ext2/3/4 ntfs,etc. En el caso de los sistemas Live usualmente el filesystem es comprimido por SquashFS para minimizar el tamaño de la imagen de Debian Live. Esta imagen es de solo lectura de modo que durante el booteo el sistema Live usará un mecanismo "union" de dicho filesystem junto a un filesystem generado en memoria, para permitir la escritura de archivos al sistema mientras se ejecuta. Sin embargo estos cambios se perderán al reiniciar salvo que se utilice persistencia(en cuyo caso la union se hace con el filesystem especificado para la persistencia). El complemento para los componentes detallados anteriormente es el Bootloader, un pequeño código confeccionado para bootear desde el dispositivo elegido que también permite la mayoría de las veces seleccionar opciones o configuraciones desde una linea de comandos o un menú. Se encarga de cargar el kernel de Linux y su initrd para correr con su filesystem asociado. Se pueden usar distintas soluciones dependiendo del formato y el dispositivo que contenga los componentes: isolinux se utiliza para bootear desde un

CD o DVD, syslinux para un disco o dispositivo USB con sistema VFAT, pxelinux para bootear por la red (PXE netboot), GRUB para particiones ext2/3/4, etc. Sin el bootloader las distintas componentes deberían ser accesibles por los firmwares de los dispositivos en los cualees se quieren ejecutar.

Todas estas componentes se pueden generar con el comando live-build, que construye la imagen del filesystem a partir de nuestras especificaciones, prepara un kernel de Linux, su initrd y un bootloader para ejecutar todos ellos y elige automáticamente un formato dependiente del medio (CD, imagen de disco, etc.).

Específicamente Debian Live esta compuesto por tres herramientas principales: live-build, live-boot y live-config.

La herramienta live-build es una colección de scripts que conforman un framework basado en la idea de usar una estructura de directorios de configuración para automatizar y personalizar todos los aspectos que involucran construir una imagen Live. Los scripts tienen un subdirectorio central "config/" en el cual se configuran sus comportamientos. Esto también permite que los scripts sean independientes y que sea seguro ejecutar cada uno. Los scripts se denominan comandos, los tres más importantes son: "lb config" que inicializa el directorio de configuración con una estructura esqueleto, "lb build" que inicia la construcción del sistema Live y "lb clean" que remueve partes de dicha construcción.

La segunda herramienta, live-boot, es un paquete de scripts para generar un initramfs capaz de bootear el sistema Live como el que se creó con live-build. Esto es necesario para que el proceso de arranque del sistema pueda tener disponible el sistema raíz luego de haber cargado el kernel en memoria. Los scripts contienen enlaces configurables por quien está armando la imagen llamados hooks, que se pasan a la herramienta initramfs-tools para que construya la imagen de booteo con archivos personalizados. La funcionalidad permite incluir arranque desde imáge-

nes .iso, imágenes para booteo por la red o netboot y pendrives USB. Para soportar esta variedad de medios, el sistema generado se comporta de la siguiente manera: al inicio se busca un medio de solo lectura que contenga el directorio "/live" donde se aloja el filesystem raíz (generalmente una imagen comprimida como squashfs) y si la encuentra, crea un entorno para escritura usando aufs para que arranque el sistema.

La tercer herramienta, live-config, consiste en scripts que corren al iniciar el sistema luego de live-boot para configurar el sistema automáticamente. Se encarga de tareas como setear el nombre del host, crear el usuario live y autologuear a dicho usuario entre otras.

Además de estas herramientas se provee un web-frontend(12) para crear imágenes Live sin necesidad de instalar ningún paquete aunque la funcionalidad se ve reducida y no se pueden personalizar archivos específicos. Permite seleccionar el tipo de imagen (cd, netboot, usb), la distribución base, un conjunto de paquetes base, paquetes individuales, la arquitectura soportada, el boot-loader y si se incluirán los fuentes o no entre otras cosas.

Para personalizar la imagen tenemos dos categorías de opciones, las configuraciones en tiempo de construcción o Build-Time y configuraciones en tiempo de arranque o Boot-Time. Las opciones de Boot-Time se subdividen en aquellas que ocurren al principio del booteo y las aplica el paquete live-build y las que ocurren más adelante en el booteo que son aplicadas por el paquete live-config. Cualquier configuracion de Boot-Time puede ser modificada por el usuario antes de arrancar el sistema Live, especificándola en el prompt de arranque. Esto podría incluir la disposición del teclado, zonas horarias, persistencia, etc. Las configuraciones de Build-Time comprenden una muy extensa cantidad de opciones que al prcesarlas generan la estructura de archivos que posteriormente son usadas por live-build. Más adelante se datallan las que se utilizaron para construir el siste-

ma.

Una vez que todas las opciones están configuradas en la estructura de directorios se puede comenzar el proceso de construcción del sistema. Este proceso está dividido en etapas, donde las distintas personalizaciones son aplicadas en secuencia en cada una de las etapas. La primer etapa es la de bootstrap. Es la fase inicial cuyo objetivo es poblar la carpeta de chroot (raíz distinta para un comando o consola) con paquetes para obtener un sistema Debian mínimo. Paso seguido se hace el chroot y se completa la construcción de la carpeta de chroot poblándola con todos los paquetes listados en la configuración. La mayoría de las personalizaciones ocurren en esta etapa. La etapa final es en la que se construye la imagen live (binary stage), que construye una imagen booteable usando los contenidos de la carpeta chroot creada anteriormente como filesystem raíz del sistema Live y da como resultado el archivo .iso a ser escrito en un CD o la imagen para el USB o la imagen para el booteo por red. En cada etepa se aplican los comandos en una secuencia determinada para asegurar que las personalizaciones se solapen razonablemente.

Hay distintas formas de seleccionar qué paquetes serán instalados por live-build en la imagen que cubren cada necesidad. Se puede nombrar paquetes individuales para instalar en una lista de paquetes. También se pueden seleccionar listas de paquetes predefinidos o incluso utilizar tareas de APT. Por último, también se pueden usar archivos de paquetes de prueba o experimentales antes de que aparezcan en los repositorios oficiales simplemente copiando estos archivos directamente en la estructura de directorios config/.

Para agregar archivos particulares de manera que estén disponibles para el sistema Live mientras se está ejecutando se utiliza la estructura de directorios config/ recién mencionada. Si por ejemplo queremos que agregar un archivo para que sea el archivo /var/www/index.html debe colocarse en config/chroot-local-

includes/var/www/index.html. Este mecanismo también se utiliza para personalizar archivos de configuración dado que si el archivo existe en el sistema generado a partir de la instalación de un paquete se sobreescribe luego con el archivo que fue colocado en el chroot-local-includes.

Con esto concluye el análisis de las tecnologías involucradas. Se abordaron los temas con la amplitud y profundidad necesarias como para poder plantear un prototipo que se detalla en el capítulo siguiente. A forma de cierre se presenta un esquema del funcionamiento del sistema que incluye las tecnologías analizadas:

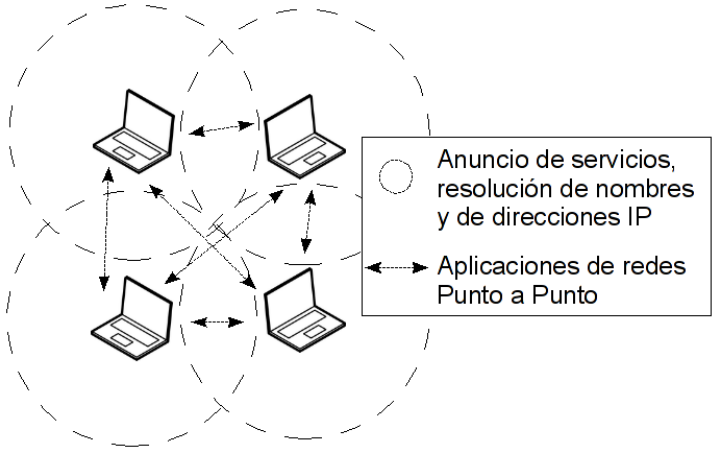


Figura 2.7: Esquema del sistema en funcionamiento

En la figura se pueden apreciar cuatro computadoras con placas inalámbricas que arrancaron el sistema, donde cada una está al alcance del resto. Cuentan con una dirección IP única y pueden transmitir a direcciones multicast información que debe estar disponible para todas las computadoras de la red. Por otro lado pueden transmitir a direcciones IP específicas paquetes de aplicaciones como clientes de chat, escritorios remotos, etc. Todas éstas comunicaciones se realizan sin la intervención de Access Points.

# Capítulo 3

# Implementación

Hasta aquí se han analizado las implicancias tecnológicas y las alternativas técnicas referidas al sistema propuesto. Se han balanceado las distintas alternativas y justificado las decisiones en base a cada argumento planteado. Ahora veremos como cada componente va ocupando su rol en un desarrollo específico, cómo en la práctica cada técnica implicada sirve al propósito específico y aplicado. En las siguientes secciones se detalla cada paso que se siguió en la construcción del sistema final. Se explica cómo pueden ser reproducidos o adaptadas según se considere necesario. Es importante aclarar que las pruebas descriptas en la próxima sección fueron indispensables para homogeneizar todas las técnicas planteadas.

## 3.1. Instalación de Debian Live

Para comenzar el desarrollo debemos instalar la herramienta de creación de LiveCD, en este caso instalaremos Debian Live en una distribución Ubuntu, bien podría ser otra distribución compatible con Debian, los requerimientos mínimos son: acceso como usuario root, un kernel de Linux (2.6.x) una consola compatible con POSIX (bash, dash, etc.) y deboostrap o cdbootstrap (herramienta para instalar debian en un subdirectorio de otro

sistema). Como primer paso editamos el repositorio de paquetes del sistema APT. Como derivaremos una versión de Debian Squeeze, agregamos el siguiente repositorio (como root):

```
echo 'http://ftp.debian.org/debian squeeze main' >> /etc/apt/solurces.list
```

Este paso no es necesario si ya estamos usando el sistema Debian Squeeze o superior. Luego actualizamos la lista de paquetes e instalamos el paquete live-build:

```
apt-get update
apt-get install live-build
```

También es posible instalar la herramienta bajando el código fuente y compilándolo en nuestro sistema a elección.

Como vimos la herramienta fundamental es "live-build" y trabaja en base a partir de un directorio base que se usará para guardar los archivos de configuración, los paquetes necesarios y la imagen final en el formato especificado. Los archivos base de configuración se generan con el comando "lb config", este comando toma parte de los parámetros de personalización como veremos en el siguiente punto. Si no se le pasa ningún parámetro se genera una estructura correspondiente a la versión estándar de Debian Live. Con el comando "lb build" se construye la imagen con las especificaciones de la subcarpeta config. Si deseamos hacer algún cambio para la versión siguiente de nuestro sistema debemos usar al comando "lb clean" antes de volver a invocar a "lb build". Resumiendo:

```
$ mkdir sistema
$ cd sistema
$ lb config
# lb build
```

Cuando este largo proceso termina deberíamos tener el archivo "binary-hybrid.iso" en nuestra carpeta sistema. Este archivo representa una imagen de una versión completa del sistema Debian Live. El mismo archivo puede ser grabado en un CD o en un disco USB. Recordemos que para una nueva versión (que sólo realiza cambios con respecto a la etapa binary) debemos hacer:

```
# lb clean --binary
```

## 3.2. Elección de los Paquetes

El próximo paso es seleccionar paquetes específicos que se incluirán en nuestra imagen personalizada. El comando "lb config" es el que se encarga de tomar los paquetes y hacerlos disponibles para la imagen. Si por ejemplo queremos incluir los paquetes Gnome simplemente hacemos

```
$ lb config  -p gnome
```

Y nuestro directorio sistema/config se poblará con los archivos correspondientes. "lb config" dispone de una amplia cantidad de parámetros de personalización y como necesitaremos incluir opciones más específicas usaremos el mecanismo "auto" para llamar a este comando. El mecanismo auto consiste en colocar en los archivos "sistema/auto/config", "sistema/auto/build", "sistema/auto/clean" scripts con los comandos y sus parámetros.

Se proveen scripts ejemplos con los comandos básicos en "/usr/share/live/build/examples/auto/*". Como punto de partida para usar el sistema auto los copiamos y nos aseguramos de que sean ejecutables:

```
$ cp /usr/share/live/build/examples/auto/* sistema/auto/
$ chmod 755 sistema/auto/*
```

luego de la copia el archivo "sistema/auto/config" tiene el siguiente contenido:

```
#!/bin/sh
lb config noauto \
    --package-lists "standard" \
    "${@}"
```

Aquí comenzamos a editar. Es fundamental conservar el parámetro noauto para que el script no vuelva a llamarse recursivamente. A continuación se muestra el archivo "auto/config" utilizado en el sistema y luego se detalla la función de cada parámetro y la finalidad de sus valores.

```
#!/bin/sh
lb config noauto \
 --distribution squeeze \
 --linux-flavours 686 \
 --debian-installer false \
 -b iso-hybrid \
 --packages-lists gnome-core \
 --bootappend-live "username=user locales=es_AR.UTF-8 keyboard-layouts=latam" \
 --mirror-bootstrap http://ftp.br.debian.org/debian \
 --mirror-chroot-security http://security.debian.org/ \
 --syslinux-splash /home/usuario/Descargas/cigarra.png\
 --packages "avahi-autoipd avahi-daemon avahi-discover avahi-utils avahi-ui-utils\
             iceweasel network-manager-gnome user-setup\
             less empathy vino openoffice.org-impress" \
"${@}"
```

**distribution squeeze** la distribución base será Debian Squeeze (estable).

**linux-flavours 686** usar código compilado compatible con Pentium Pro en adelante (686-bigmem si queremos soportar arquitectura intel con más de 4GB de RAM).

**debian-installer false** no incluir la opción de instalar Debian desde el sistema Live.

**b iso-hybrid** el formato de salida de la imagen será un .iso que también puede clonarse a un USB (usar USB-HDD para pendrive o NET para bootear por la red/PXE netboot).

**packages-lists gnome-core** incluir todos los paquetes base de Gnome.

**bootappend-live "param=valor"** pasar parámetros al kernel al momento de bootear la imagen Live. username especifica el nombre del usuario del sistema Live que se creará en el arranque, locales indica la internacionalización y codificación de caracteres que se usará y keyboard-layout especifica la distribución de teclas del teclado para Latinoamérica.

**mirror-bootstrap** repositorio de debian para descargar los paquetes al construir la imagen

**mirror-chroot-security** idem anterior pero para los paquetes de seguridad

**syslinux-splash** imagen personalizada para el menú de arranque de syslinux

**packages** especifica una lista de paquetes particulares a agregar.

Los paquetes particulares escogidos responden a los análisis previos sobre funciones o aplicabilidad. También se tuvo en cuenta que la instalación por defecto funcione en el sistema sin mayores configuraciones aunque esto no fue posible en muchos casos. En particular cada uno tiene la siguiente función:

**avahi-autoipd** demonio para la asignación de direcciones IP de forma descentralizada.

**avahi-daemon** demonio para anuncio de servicios, soporte para mDNS/DNS-SD.

**avahi-discover** interfaz de usuario del descubridor de servicios de avahi.

**avahi-utils** utilidades que interactúan con el demonio avahi, incluye avahi-discover y avahi-set-host-name entre otras.

**avahi-ui-utils** utilidades de interfaz gráfica para listar anuncios de servidores VNC o SSH (consola)

**iceweasel** navegador web basado en firefox.

**network-manager-gnome** herramienta de configuración de redes con interfaz gráfica Gnome incluida.

**user-setup** utilidad para crear usuarios cuando solo existen los del sistema.

**less** programa de paginado similar a more.

**empathy** cliente de chat multiprotocolo con soporte para avahi.

**vino** servidor VNC de Gnome para compartir escritorios por la red con soporte para avahi

**openoffice.org-impress** editor y visualizador de presentaciones. Análogo a PowerPoint.

## 3.3. Personalización de los archivos

Además de especificar qué paquetes se incluyen en el sistema es necesario cambiar la configuración por defecto que estos proveen. Para lograrlo se sobreescriben los archivos por defecto usando la carpeta "config/chroot-local-includes".

### 3.3.1. Creando la Conexión

Uno de los objetivos principales del sistema es que el modo ad-hoc quede preconfigurado. Si bien con Network-Manager el usuario puede acceder más fácilmente a las configuraciones de red el propósito es la mínima intervención del usuario en las configuraciones. Si establecemos un archivo de configuración tradicional en /etc/network/interfaces Network-Manager no administrará la placa de red correspondiente y en consecuencia se perderá flexibilidad. Por otro lado en este tipo de configuración o se asigna manualmente una dirección IP (lo que no es aceptable en nuestro caso) o se establece que la IP se obtendrá a partir de un servidor DHCP que tampoco es nuestro caso (para utilizar direcciones Link-Local con esta configuración se espera al timeout del DHCP que por defecto es de 30 segundos y recién en ese momento puede actuar avahi-autoipd). Una de las forma de preconfigurar una conexión con Network-Manager es agregar un archivo de conexión en /etc/Network-Manager/system-connections, de

esta forma dicha conexión estará disponible para todos los usuarios. Para el sistema se creó un archivo de conexión con todos los parámetros de la red y se colocó en sistema/config/chroot-local-includes/etc/Network-Manager/system-connections/conexion-adhoc para que sea incluido en la imagen final. A continuación se lista el contenido del archivo y se indica el objetivo de cada parámetro (que son bastante autoexplicativos):

```
[connection]
id=CIGARRA-NET
uuid=1fd0a12d-fbd2-4700-9a93-060ecef6df6e
type=802-11-wireless
autoconnect=true
timestamp=1

[ipv4]
method=link-local
ignore-auto-routes=false
ignore-auto-dns=false
dhcp-send-hostname=false
never-default=false

[ipv6]
method=ignore
ignore-auto-routes=false
ignore-auto-dns=false
never-default=false

[802-11-wireless]
ssid=67;73;71;65;82;82;65;78;69;84;
mode=adhoc
channel=0
rate=0
tx-power=0
mtu=0
```

[**connection**] Aquí se lista la meta información de la conexión. **id**: es el nombre visible para el usuario. **uuid**: es un identificador único de la conexión para el sistema, si se reconfigura el tipo, este número se recalcula. **type**: tipo de hardware específico de la conexión. **autoconnect**: indica que a Network-Manager que active la conexión cuando los recursos están disponibles. **timestamp**: indica la última vez que la conexión se activó satisfactoriamente. Este valor es 0 por default, en este caso se editó porque cuando es 0 la conexión no se muestra en la interfaz gráfica y sólo es accesible desde Redes Ocultas.

[**ipv4**] Aquí se lista la información del protocolo. **method**: al setearlo en link-local se asigna una IP en el rango 169.254.0.0/16, otros valores posibles podrían ser DHCP, manual, etc. **ignore-auto-routes**: se usa cuando quieren rutearse los paquetes con reglas específicas. **ignore-auto-dns**: si se activa se espera que se provean servidores DNS en forma manual. **dhcp-send-hostname**: se activa cuando los servidores asignan direcciones IP basados en el nombre proprocionado por el host. **never-default**: se activa si se quiere evitar que la conexión se use de forma automática para IPv4.

[**ipv6**] Información del protocolo IPv6. **method**: se configuró el método en ignore dado que por defecto IPv6 se habilita. El resto de los valors se colocó porque son obligatorios.

[**802-11-wireless**] Lista la información para la red inalámbrica. **SSID**: Como vimos en el identificador de la red inalámbrica podemos colocar un valor análogo a una dirección MAC. **mode**: se pone la placa en modo ad-hoc, el otro valor disponible obviamente es infrastructure. **channel**: si se especifica la conexión trabajará únicamente en dicho canal.**rate**: si se especifica se trabajará solamente en dicha velocidad, este parámetro es dependiente del driver. **tx-power**: si se especifica se transmitirá con la potencia designada, también depende del driver. **mtu**: si se especifica sólo se transmiten completos los paquetes de este tamaño o más pequeños, el resto se fragmenta en varias tramas.

Con este archivo la conexión puede ser utilizada por los usuarios simplemente eligiéndola entre las conexiones disponibles listadas por el Network-Manager. El único obstáculo presente es que por más que se edite el timestamp la conexión es percibida como que nunca fue utilizada. Esto hace que aparezca la primera vez en el apartado de conexiones ocultas y por tal motivo no se conecte automáticamente.

### 3.3.2. Configurar Avahi

Avahi funciona notablemente sin mayores complicaciones luego de la instalación con los parámetros por default. La personalización que se introdujo es con respecto a IPv6. Avahi trabaja con IPv4 e IPv6 simultáneamente, esto puede ser aprovechado por algunas aplicaciones pero en nuestro caso es suficiente con IPv4. Se decidió deshabilitar IPv6 principalmente porque cuando se listan los servicios anunciados por la red se obtiene entradas duplicadas correspondientes a cada una de las versiones del protocolo para cada servicio.

Para cambiar este comportamiento debemos cambiar el archivo /etc/avahi/avahi-daemon.conf, nuevamente debemos usar el estructura de directorios config. En el archivo sistema/config/chroot-local-includes/etc/avahi-daemon.conf se colocó en siguiente contenido:

```
[server]
#host-name=foo
#domain-name=local
#browse-domains=0pointer.de, zeroconf.org
use-ipv4=yes
use-ipv6=no
#allow-interfaces=eth0
#deny-interfaces=eth1
#check-response-ttl=no
#use-iff-running=no
#enable-dbus=yes
#disallow-other-stacks=no
#allow-point-to-point=no
#cache-entries-max=4096
#clients-max=4096
#objects-per-client-max=1024
#entries-per-entry-group-max=32
ratelimit-interval-usec=1000000
ratelimit-burst=1000

[wide-area]
enable-wide-area=yes

[publish]
#disable-publishing=no
#disable-user-service-publishing=no
#add-service-cookie=no
#publish-addresses=yes
#publish-hinfo=yes
#publish-workstation=yes
#publish-domain=yes
#publish-dns-servers=192.168.50.1, 192.168.50.2
```

```
#publish-resolv-conf-dns-servers=yes
#publish-aaaa-on-ipv4=yes
#publish-a-on-ipv6=no

[reflector]
#enable-reflector=no
#reflect-ipv=no

[rlimits]
#rlimit-as=
rlimit-core=0
rlimit-data=4194304
rlimit-fsize=0
rlimit-nofile=768
rlimit-stack=4194304
rlimit-nproc=3
```

El único cambio realizado con respecto al default fue pasar de use-ipv6=yes a use-ipv6=no. Varias personalizaciones se encuentran disponibles en este archivo, el inconveniente es que todas las estaciones de nuestra red usarán el mismo. Entonces si por ejemplo configuráramos host-name=un-nombre para que la PC se anuncie con un nombre personalizado, este parámetro lo tomarían todos los nodos y de nuevo todos tendrían el mismo nombre.

## 3.4. Creación de la Imagen

Luego de haber establecido qué paquetes se incluirán y qué archivos se agregarán, sólo resta crear la imagen. Este proceso toma su tiempo pero es automático, solamente debemos asegurarnos que nuestra pc cuente con acceso a Internet durante el armado de la imagen. Para iniciar la construcción de la imagen simplemente ejecutamos

```
# lb build
```

Es importante notar que este comando debe ser ejecutado como root. Si no se cuenta con permisos de root se permite establecer la opción lb config –root-command sudo pero no está asegurado que funcione siempre. En la práctica lb build es un comando que se compone de otros seis comandos de alto nivel y varias

decenas de comandos de bajo nivel que realizan los trabajos necesarios para instalar el sistema en una carpeta, luego cambiar al entorno generado y construir la imagen final en modo binario. Adicionalmente hay comandos de bajo nivel que manejan los paquetes fuentes y orígenes del software.

Una vez finalizado el proceso obtenemos la imagen binaria. Dependiendo que valor le hayamos dado al parámetro [opcion] en lb config –binary-images [opcion] podemos obtener los siguientes formatos:

```
iso        -> Imagen binaria para ser grabada a un CD
usb-hdd    -> Imagen para escribir un dispositivo USB
iso-hybrid -> Imagen híbrida para CD o USB
net        -> Imagen para arranque por red (precisa un servidor)
tar        -> Archivo con todos los fuentes
```

Como en nuestro caso estaba configurado en iso-hybrid el próximo paso es grabar este archivo a un CD con algún software apropiado. También podemos escribirlo en un pendrive ejecutando el siguiente comando:

```
# dd if=./binary-hybrid.iso of=/dev/sdb
```

El dispositivo USB puede llevar otra identificación distinta de /dev/sdb en otras computadoras. Debe prestarse mucha atención a esto porque podríamos terminar destruyendo el filesystem de nuestro disco rígido.

Una vez que tenemos nuestro CD o pendrive con la imagen debemos asegurarnos que la computadora en la que queremos usar el sistema pueda iniciar desde dicho dispositivo. Esta opción se especifica en la configuración de la BIOS, donde se establece en que orden se probarán los distintos medios para buscar un sistema operativo. Cualquier computadora medianamente moderna soporta el inicio desde dispositivos USB pero pero puede ocurrir que en algunos casos esta opción no esté disponible.

# Capítulo 4

# Pruebas

A continuación se detallarán las pruebas que se realizaron en los distintos pasos del desarrollo del sistema y sus resultados.

## 4.1. Web Frontend

**Debian-Live web-frontend**

Como primer intento y para probar el concepto se utilizó el web-frontend del proyecto Debian Live. Este sitio permite ejecutar el comando live-build remotamente generando una imagen en dicho servidor que luego puede ser descargada. Provee sólo algunas opciones para personalizar la imagen generada, las principales son el formato del binario generado, la distribución base, el conjunto de paquetes inicial y una lista de paquetes particulares. Aunque se proveen varias opciones más, no se pueden incluir archivos particulares personalizados que garanticen que la imagen final quede preconfigurada. Sí se pudo verificar la compatibilidad de los paquetes, que el hardware es detectado y que la funcionalidad ad-hoc y que la configuración de la red descentralizada es viable. Se utilizaron los siguientes parámetros:

```
BINARY_IMAGES = "usb-hdd"
DISTRIBUTION = "squeeze"
PACKAGES_LISTS = "gnome-core"
TASKS=""
PACKAGES = "avahi-daemon avahi-discover
```

```
 avahi-utils avahi-ui-utils avahi-autoipd avahi-dnsconfd iceweasel"
# Advanced chroot options
LINUX_FLAVOURS = "686"
```

Luego se descargó la imagen y se clonó a un pendrive que fue booteado en una netbook, el sistema booteó correctamente y detectó el hardware, también se listaron los procesos corriendo y se verificó que avahi había arrancado, la IP había sido seteada en 169.254.11.7 y se logró resolver correctamente el nombre debian.local (nombre del host y dominio por defecto) luego de ejecutar el comando ping.

Por otro lado se ejecutó el avahi zeroconf browser desde la interfaz gráfica y se verificó que la pc aparecía listada. Con esta prueba se pudo comprobar que la utilidad live-build era viable y que la lista de paquetes mínimos no generaban conflictos, además de comprobar el funcionamiento de la asignación de dirección IP de forma descentralizada. Con esta prueba no se pudo comprobar la funcionalidad ad-hoc y el testeo del sistema en más de una pc al mismo tiempo.

#### Slax web-frontend

Para tener una idea sobre otra herramienta se probó el web-frontend de Slax. En este sitio se pueden elegir los paquetes que se van a incluir en en la imagen Live pero no personalizar configuraciones específicas. Se buscaron los paquetes que pudieran incluir avahi o algún tipo de zeroconf y se agregaron al proyecto. Los paquetes elegidos fueron los de base y toda la funcionalidad compatible con zeroconf disponible: el cliente de chat pidgin, en cliente de VNC x11vnc y la librería libavahi-common3-0. Luego se descargó la imagen generada con dichos paquetes y se probó con virtualbox. El sistema final no proveía ningún soporte para autoconfiguración de direcciones IP ni resolución de nombres. Se hizo evidente que de elegir instalar esta herramienta, se deberían incluir paquetes por fuera de la distribución.

## 4.2. Instalación Debian live

El siguiente paso fue instalar la herramienta Debian Live. La primera instalación se realizó en una pc Pentium 4 con 512MB de RAM y un sistema operativo Debian 5 Lenny agregando los repositorios para Debian Squeeze que es la versión estable actualmente. Sin realizar ninguna personalización se ejecutó el comando lb build para confeccionar la imagen oficial estable de Debian Live. El comando se ejecutó por un tiempo, descargó algunos paquetes y terminó a raíz de un Segmentation Fault. Analizado esto se estableció que el comando debería ejecutarse desde un sistma más nuevo. Seguidamente se instalaron mismos paquetes en una netbook Acer Aspire One con procesador Intel Atom y 2 GB de RAM, placa wifi Atheros AR5B95 b/g/n con un sistema operativo Ubuntu 10.04. En este caso se personalizaron los paquetes incluyendo los mismos que se habían usado en el webfrontend y la imagen se confeccionó correctamente. Esta imagen se escribió sobre un pendrive y se booteó dicha netbook desde el dispositivo USB.

El sistema inició con normalidad pero se detectó que el paquete avahi-dnsconfd no permitía resolver nombres si se conectaba el sistema a un AP de una red wireless hogareña. Por lo tanto se decidió quitar dicho paquete de futuras versiones.

Seguidamente se usó la misma imagen para emular un CD en Virtual Box. En este caso el arranque del sistema se vió interrumpido, no se autologueaba en el entorno gráfico y solicitaba autenticación de usuario y contraseña. Al pasar a la consola se verificó que no había forma de ingresar con el usuario por defecto y que probablemente no se había creado ningún usuario "humano" para el sistema. Por este motivo se incorporó el paquete user-setup en futuras versiones por si volvía a ocurrir este inconveniente. Luego de esta modificación no volvió a presentarse el problema.

## 4.3. El sistema en funcionamiento

Con la información obtenida en las pruebas anteriores se confeccionó la imagen final con las configuraciones y personalizaciones que se detallaron en el capítulo correspondiente. Con el sistema ya creado se prosiguió a testear el funcionamiento en más de una computadora para verificar fehacientemente la funcionalidad de red.

Las computadoras utilizadas fueron la netbook especificada anteriormente y una notebook Lenovo 3000 c200 con un procesador Intel Celeron M520 1.6 Ghz y 1GB de RAM y placa de red inalámbrica Broadcom b/g miniPCI. Se arrancó el sistema satisfactoriamente en ambas computadoras. Al momento de testear la red inalámbrica se verificó que en la notebook Lenovo no había sido detectada la placa de red wireless. Esto se debe al firmware no-libre que precisa esta placa de red. Para solucionarlo se conectó la placa Ethernet a Internet, se agregó el repositorio "contrib" de Debian (que no estaba disponible en la imagen) y se descargó e instaló el paquete "firmware-b43-installer" para placas broadcom. Esta instalación hizo funcionar la placa inalámbrica y a partir de ese momento se comenzó a probar la funcionalidad de red del sistema. En ambas computadoras se seleccionó la conexión que fue personalizada en la imagen para acceder a la red. Seguidamente se verificaron las direcciones IP:

```
lenovo$ ip addr
 wlan0 inet 169.254.5.211

netbook$ ip addr
 wlan0 169.254.11.85
```

El demonio avahi-autoipd de cada computadora asignó correctamente direcciones distintas a las placas wireless dentro del rango especificado por el estándar. Con el comando ping se comprobó que se podían transmitir paquetes entre las estaciones sin estar conectadas a ningún AP.

Luego se abrió avahi browse en una interfaz gráfica y se observó que se listaban dos estaciones. Cómo se anunciaron con el mismo nombre (debian) una había cambiado automáticamente su nombre público a debian-2, se pudo forzar un nuevo nombre público con el siguiente comando

```
avahi-set-hostname lenovo

avahi-set-hostname netbook
```

También se verificó la resolución de nombres con:

```
ping lenovo.local

ping netbook.local
```

Y se resolvieron los nombres a las respectivas direcciones IP y cada placa respondió a la comunicación.

Una vez comprobada la conectividad se prosiguió con las aplicaciones. Primero se inició el cliente de chat Empathy, que por defecto al inicio solicita la creación de una cuenta con nombre, apellido, apodo y protocolo. Se creo una cuenta "gente cerca" en ambas máquinas, que es la que hace uso de avahi. Luego de abrir la aplicación en ambas máquinas fue posible chatear. También se pudo transmitir un archivo entre las computadoras simplemente arrastrándolo sobre el nombre del otro usuario. Adicionalmente se verificó que en avahi browse aparecía nueva información sobre el servicio de chat disponible en cada pc. La otra aplicación probada fue VNC. En una de las pc se estableció que compartiría el escritorio. Esto inició el servidor Vino y la disponibilidad del servicio se hizo visible también en avahi browse. En la otra pc se abrió la aplicación para examinar servicios VNC (Vinagre) y efectivamente apareció listado el servidor anunciado y luego de conectarlo fue posible visualizar el escritorio de la otra estación en una ventana nueva.

La otra prueba realizada fue desconectar la conexión personalizada y conectar un cable cruzado a las placas Ethernet de

cada PC. Seguidamente a través de Network Manager se estableció autoip para los parámetros de IPv4 de la conexión Ethernet y se pudo contar con la funcionalidad del sistema pero sobre una red cableada sin ningún servidor configurado.

Adicionalmente, y dejando de lado la funcionalidad de red también se comprobó que es posible montar exitosamente las particiones ext3 y ntfs del disco rígido de la pc a partir de la interfaz gráfica.

# Capítulo 5

# Apartado sobre Seguridad

Los objetivos propuestos y la aplicabilidad del sistema se basan en entornos donde existe confianza entre los usuarios, por lo tanto está fuera del alcance de la propuesta desarrollar garantías para que el entorno sea seguro.

Lo que se describe a continuación es un breve análisis de las implicancias en materia de seguridad sobre el sistema final y de porqué no debe transmitirse información sensible en este sistema. Se realiza esta aclaración porque en la actualidad existen gran cantidad de conexiones inalámbricas sin garantías de seguridad que aunque funcionan correctamente, los usuarios desconocen que se encuentran en un entorno no privado.

## 5.1. Cambiar de Red

Una de las primeras consideraciones es que las aplicaciones del sistema que se anuncian por la red podría no tener en cuenta a qué red estamos conectados. De este modo si cambiamos la conexión ad-hoc preconfigurada y conectamos el sistema a otra red las aplicaciones segurán anunciándose y esta información puede obtenerse por otros sistemas en la misma red y ser usada con fines maliciosos. Si se desactiva el servicio de Avahi igualmente esta información puede obtenerse con cualquier herramienta de

escaneo de puertos usadas para estos fines en otras redes.

## 5.2. Autenticación

Se mencionó que el modo ad-hoc no realiza autenticaciones de los nodos o los usuarios, por lo tanto cualquier persona podría configurar manualmente su placa wireless en modo ad-hoc, setear una IP en el rango 169.254.0.0/16 y quedaría dentro de la red pudiendo acceder a los recursos compartidos. Esto puede ser una ventaja un grupo de usuarios quieren conectar sus equipos pero no cuentan con copias del sistema pero permite también que otras personas ajenas al grupo accedan a la red. Este riesgo está presente en todas las redes ad-hoc, no solamente en el sistema descripto.

## 5.3. Cifrado de los Paquetes

Los mecanismos de cifrado WEP, WPA y WPA2 encriptan los paquetes antes de transmitirlos y se implementan en el hardware de red. Se ha demostrado que todos estos mecanismos son crakeables por distintos métodos desde una estación que pueda interceptar las transmisiones por el aire. Para implementar soluciones de seguridad más avanzadas es necesario nuevo hardware que soporte encriptaciones más seguras o encriptar los paquetes por software antes de solicitar que se envíen. Esta vulnerabilidad está presente en todas las redes inalámbricas independientemente del modo en que se coloquen. Diversos trabajos se están realizando para resolver estos inconvenientes en sistemas futuros y en sistemas actuales la solución WPA2 es considerada bastante robusta.

# Capítulo 6

# Manual de Usuario

El siguiente manual explica las funciones básicas del sistema y como usar las herramientas. Cualquier usuario familiarizado con alguna distribución Linux con entorno gráfico Gnome encontrará otras utilidades conocidas.

## 6.1. Iniciar el Sistema

El primer paso para iniciar el sistema es contar con una imagen del mismo. Este aimagen es un archivo .iso que puede ser grabado en un CD con cualquier programa de grabación provisto por el Sistema Operativo o Brasero / K3B para Linux y CDBurnerXP / Nero para Windows. En caso que se quiera crear un LiveUSB, debemos clonar el mismo archivo .iso al pendrive. Para esto el primer paso es conectar el pendrive a un puerto usb y luego debemos usar el comando dd en linux o el programa que mencionado más abajo para Windows. Es importante tener bien en claro la ubicación de nuestra imagen.iso y que nombre de dispositivo fue asignado al pendrive. En Linux suponiendo que nuestra imagen esta en /home/user/imagen.iso y nuestro pendrive fue detectaco como /dev/sdb ejecutamos en una consola el siguiente comando:

```
# dd if=/home/user/imagen.iso of=/dev/sdb bs=1M
```

Para Windows se puede usar el programa Image Writer for Windows que se puede descargar de `https://launchpad.net/win32-image-writer+download`. Luego de instalarlo lo ejecutamos y elegimos la ubicación de la imagen y la letra de unidad del pendrive.

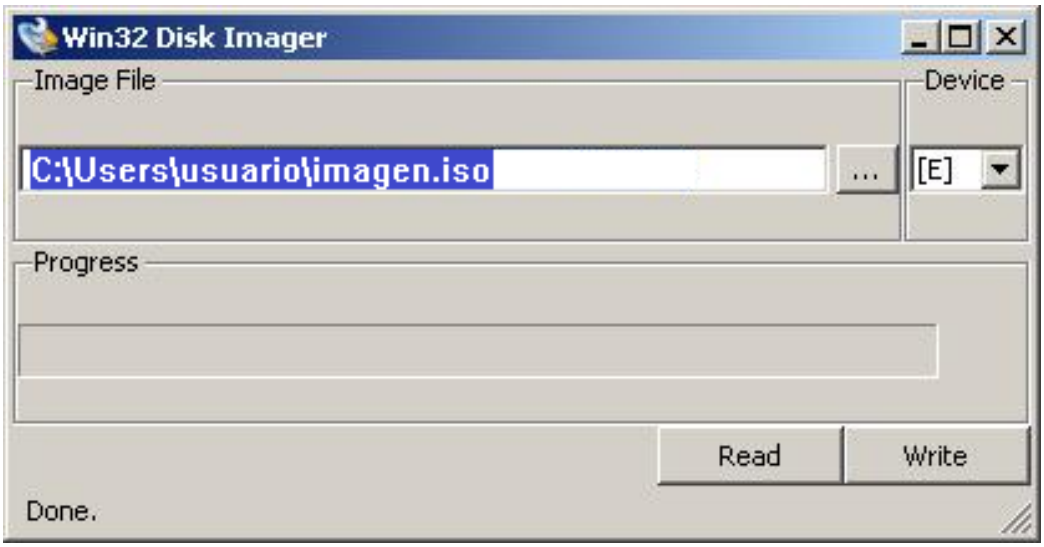


Figura 6.1: Grabando una imagen por USB en WinImageWriter

Luego iniciamos la computadora con el CD o el pendrive colocados. Si no se inicia el sistema se debe cambiar el orden de booteo de los dispositivos accediendo a la configuración de la BIOS de nuestra PC.

Cuando inicia el sistema presenta un menú donde se puede elegir entre las opciones Live, Live(fasilsafe) y Help.

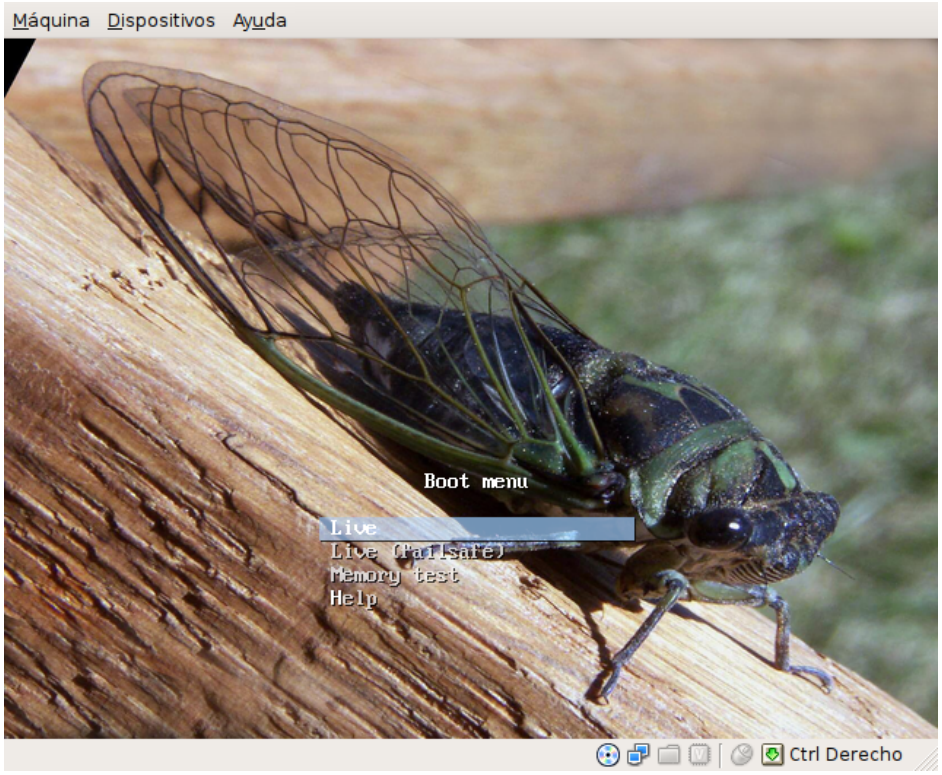


Figura 6.2: Inicio Syslinux

Live inicia el sistema y Live (failsafe) es por si alguna componente de hardware de nuestra computadora no permite iniciar Linux. Luego de esto el sistema inicia y presenta la interfaz gráfica lista para usar.

## 6.2. Conectando la Red

Para conectar el sistema a cualquier red se usa el applet de Network Manager ubicado en la parte superior derecha del escritorio. Allí aparecen listadas las redes disponibles. Para iniciar la red ad-hoc seleccionamos CIGARRA-NET.

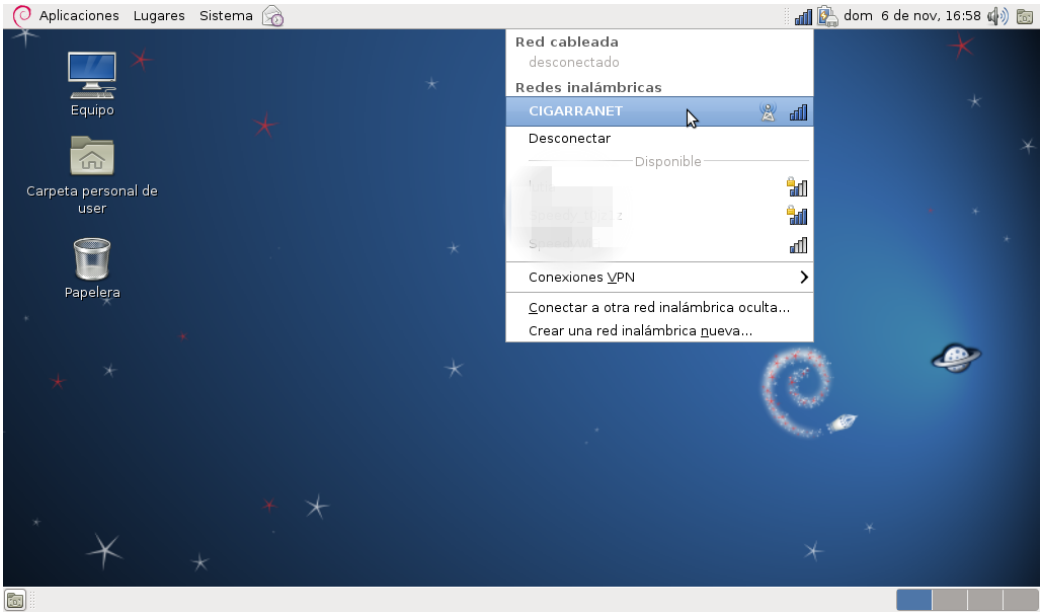


Figura 6.3: Conectar a la red ad-hoc

Si la conexión no aparece listada debemos buscarla en conectar a otra red inalambrica oculta.

Si las conexiones inalámbricas directamente no aparecen listadas es porque el sistema no ha detectado nuestra placa wireless. En este caso lo más probable es que se precise un instalar un driver externo para soportar el hardware. Si este es el caso hay que recordar que todos los cambios que realicemos en el sistema se perderán al reiniciar.

### 6.2.1. Visualizar servicios

Una vez que accedimos a la red ad-hoc seguramente queremos ver que máquinas y que aplicaciones están disponibles. Para esto elegimos en el menú Aplicaciones > Herramientas del sistema > Navegador de Zeroconf de Avahi. Aquí se listan ordenado por aplicación los recursos disponibles. Al seleccionar uno se puede observar el detalle da la información compartida anunciada por la red.

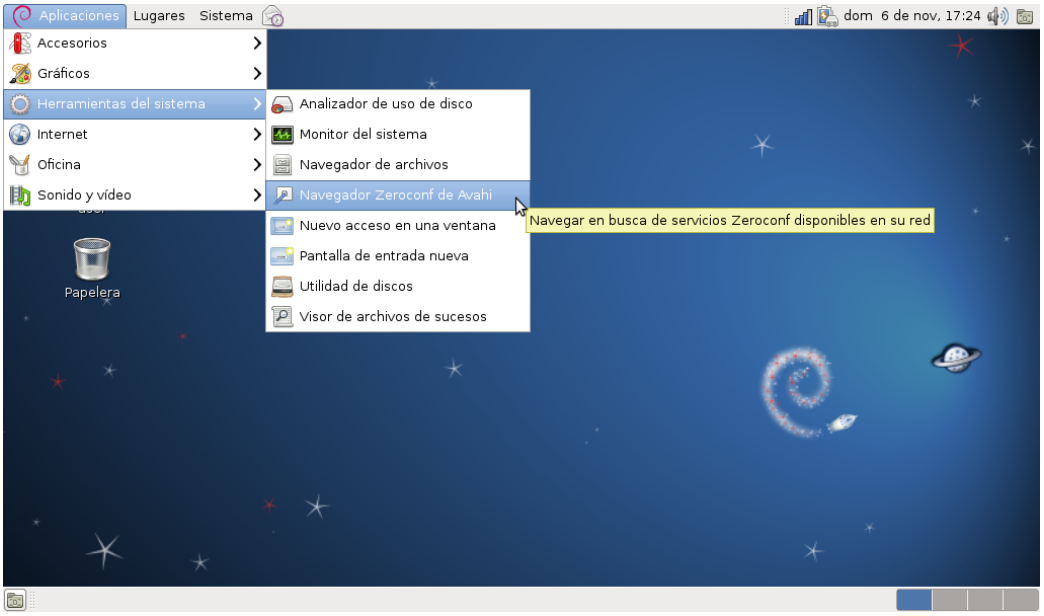


Figura 6.4: Navegador Zeroconf - Visualización de los Servicios

### 6.2.2. Establecer el nombre de la PC

Por defecto todas las máquinas del sistema llevan el nombre debian. A medida que se van conectando el nombre público de cada una se va estableciendo como debian-2, debian-3, etc. Si queremos cambiar este nombre debemos ir al menú Aplicaciones > Accsesorios > Terminal y en la consola que se abre escribir el siguiente comando:

```
avahi-set-hostname nombre
```

donde nombre es el nuevo nombre público que daremos a nuestra máquina. No es recomendable cambiar este nombre si se tiene otras aplicaciones abiertas porque podrían no notar el cambio y seguir anunciando el nombre anterior.

## 6.3. Aplicaciones

### 6.3.1. Empathy

Para poder chatear con el resto de las personas en la red usaremos el cliente de chat Empathy. Para acceder debemos ir a Aplicaciones > Internet > Cliente de mensajería instantánea Empathy. Este programa soporta múltiples protocolos de chat por lo tanto al ingresar nos pedirá crear una cuenta para especificar el protocolo. En este cuadro de diálogo debemos marcar No, por ahora solo quiero ver la gente conectada cerca y le damos aceptar.

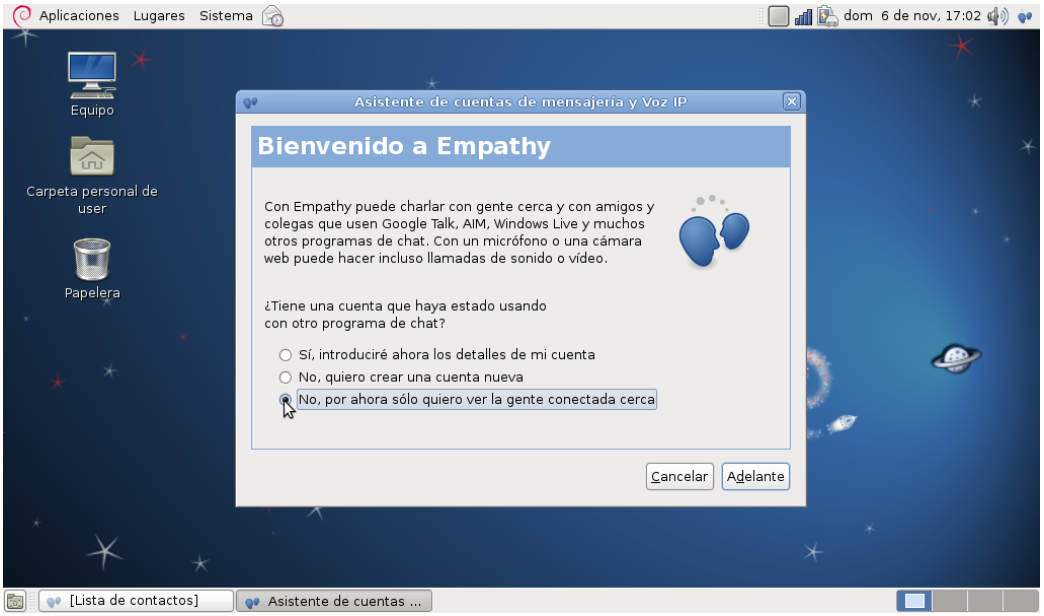


Figura 6.5: Cuenta nueva en Empathy

Seguidamente se presenta otro cuadro de diálogo indicando que introduzcamos nuestros datos personales. Esta información es la que estará disponible para las otras personas conectadas a la red, en especial el apodo.

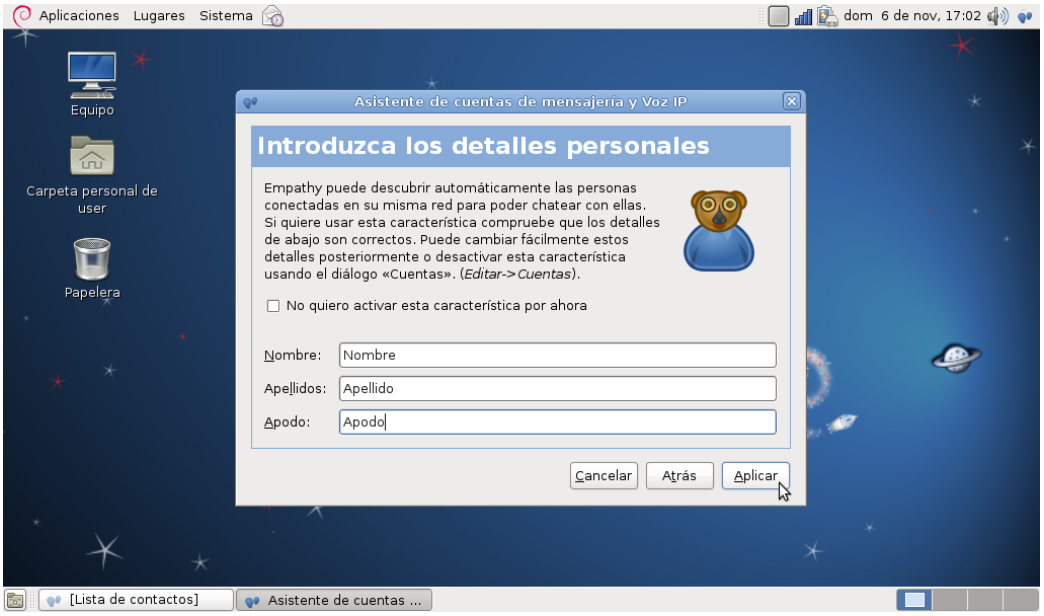


Figura 6.6: Datos personales en Empathy

Luego de introducir nuestros datos presionamos aplicar y a continuación la aplicación muestra una lista de los usuarios conectados. Haciendo click sobre alguno podemos comenzar una conversación. Si en vez de hacer click arrastramos un archivo sobre el nombre de un usuario se le enviará a este una petición para que acepte la transmisión de un archivo y al aceptarla se transferirá dicho archivo. Adicionalmente se puede establecer el estado de nuestra cuanta como Disponible, Ocupado, Invisible, etc. Cabe aclarar que los usuarios que no hayan iniciado el programa no estarán disponibles para chatear.

### 6.3.2. Escritorios Remotos

Otra funcionalidad provista es la capacidad de que otros usuarios puedan visualizar nuestra pantalla en su máquina (o nosotros las de ellos). Primero describiremos como permitir el acceso a ver nuestra pantalla desde otras computadoras. El primer paso es ir a Sistema > Preferencias > Escritorio Remoto.


Figura 6.7: Compartir el escritorio por medio de VNC

A continuación se presenta una ventana que permite controlar de que forma se compartirá nuestro escritorio. Al activar la primer casilla permitimos que los otros usuarios vean lo que hacemos. También podemos permitir que controlen nuestro mouse y nuestro teclado. En las otras opciones podemos configurar que se muestre una notificación cada vez que alguien nos está mirando, solicitar nuestra confirmación y solicitar una contraseña a quienes se quieran conectar. El último grupo de opciones maneja la visualización del ícono que representa el programa en nuestra barra de tareas.

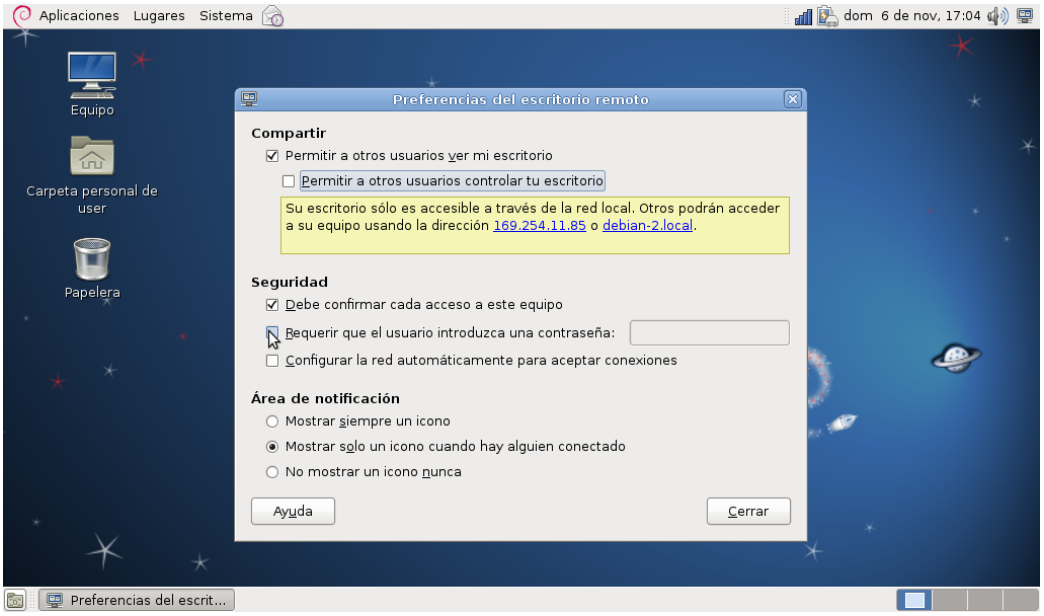


Figura 6.8: Configurar los parametros para compartir el escritorio por VNC

Para terminar de compartir el escritorio accedemos al mismo menú o al icono y se vuelve a abrir la misma ventana. Aquí desmarcamos Permitir a otros usuarios ver mi escritorio (que antes habíamos marcado).

Para poder ver los escritorios de otros debemos ir al menú Aplicaciones > Internet > Examinador de servidores de VNC de Avahi. Aqui se nos presenta una listado de todas las máquinas que están compartiendo sus escritorios en la que podemos elegir cual visualizaremos. Dependiendo de los parámetros que configuró el usuario del escritorio remoto nos dará acceso inmediatamente o deberemos esperar su confirmación. Adicionalmente nos podría solicitar una contraseña. Una vez que accedimos se abrirá una ventana que muestra el escritorio remoto. Esta ventana puede redimensionarse o minimizarse. Si se cierra se finalizará la visualización. También podremos controlar el teclado y el mouse remotos si el usuario lo configuró de ese modo.

# Capítulo 7

# Conclusiones

## 7.1. Conclusiones Generales

A continuación se hacen las consideraciones finales sobre las herramientas analizadas, el sistema desarrollado y los caminos a seguir ampliar o profundizar lo planteado en este trabajo.

Se analizaron varias herramientas para la creación de sistemas Live, se probaron en detalle algunas y se construyó exitosamente el sistema con Live Build del proyecto Debian Live. Las distintas herramientas cubren distintas necesidades y cada una establece una relación entre grado de personalización y profundidad de conocimientos necesarios para usarlas. Live Build es una herramienta avanzada que permite amplias opciones de personalización y es flexible. Además cuenta con el respaldo de la comunidad Debian y una amplia documentación.

Las conclusiones sobre el análisis de las tecnologías inalámbricas las podemos separar en varios puntos. A nivel de los estándares observamos que 802.11 cuenta con varias revisiones y evoluciones a/b/g/n entre otras y todavía no se ha establecido ninguno como principal o más aceptado como es el caso de Ethernet para redes cableadas. Esto está relacionado con el hecho de ser una tecnología que está en pleno proceso de masificación. Por otro lado hay dispositivos que no implementan toda la funciona-

lidad de los estándares o lo hacen de forma deficiente. Los bugs dominan el mundo de los drivers de placas de red inalámbricas y muchos fabricantes no liberan el firmware para sus dispositivos o solo proveen drivers para Windows reduciendo el soporte que puede haber para dichos dispositivos en otros sistema operativos.

La problemática específica planteada por la configuración de la red y la falta de servidores centralizados se solucionó satisfactoriamente incluyendo y configurando las herramientas Network Manager y Avahi en el sistema final. Avahi resuelve la asignación de direcciones IP, resolución de nombres y anuncio de servicios de forma descentralizada sin intervención del usuario y Network Manager permite que el usuario elija una conexión inalámbrica ad-hoc reconfigurada de una lista de conexiones en una interfaz gráfica y brinda flexibilidad para conectar el sistema a otras redes wireless o cableadas.

El trabajo de implementación dió como resultado un prototipo del sistema propuesto, una "prueba de concepto" que muestra que es posible combinar distintas tecnologías para alcanzar los objetivos establecidos. Se presenta un sistema Live flexible que provee una red ad-hoc inalámbrica preconfigurada con mínima intervención del usuario y un conjunto mínimo de aplicaciones de usuario para mostrar la funcionalidad y el potencial del sistema. Este sistema está disponible en formato .iso que puede ser grabado en un CD o en un pendrive USB.

Se comprobó que si se inicia el sistema Live es posible conectar de forma inalámbrica y sin Acces Points varias computadoras y chatear, intercambiar archivos y visualizar remotamente los escritorios, sin configuraciones por parte del usuario y sin modificar los sistemas originales de los dispositivos. También se verificó que son accesibles los archivos de estos sistemas originales o de medios de almacenamiento con los que cuente cada computadora.

Luego de considerar algunos puntos expuestos se establece que no se debe usar el sistema para transmitir información sensible dado que las garantías de seguridad se ven reducidas.

Finalmente se creó el proyecto de Software Libre "Linux Cigarra" para que esté a dispocición de toda la comunidad, para su uso como sistema final y para que los interesados puedan realizar aportes o mejoras. El proyecto está disponible en `http://sourceforge.net/p/linuxcigarra`.

## 7.2. Trabajo a Futuro

La posibilidad de agregar aplicaciones de usuario de forma flexible a la imagen del sistema abre un abanico de personalizaciones disponibles. Con el objetivo de presentar un sistema de circulación masiva se podría trabajar en la usabilidad del sistema y personalización de la interfaz gráfica como por ejemplo agrupar las utilidades en nuevos menús o cuestiones estéticas. Otra propuesta sería desarrollar un browser de servicios más amigable. Se podría pensar que cada estación cuente con un servidor web con soporte para scripts que listen las estaciones en la red y al acceder a la dirección correspondiente se listen las aplicaciones disponibles de dicha estación con links a las herramientas específicas. Para continuar ampliando la funcionalidad se cuenta con todas las aplicaciones de los repositorios de Debian o compatibles, esto permitiría ofrecer distintas versiones o flavours del mismo sistema con aplicaciones específicas para cada ámbito.

También sería interesante poder medir la escalabilidad del sistema, el área de cobertura máxima en modo ad-hoc de distintas placas de red y la pérdida de paquetes y colisiones al aumentar la cantidad de nodos en la red. Para realizar estos estudios debe disponerse del hardware o herramientas de simulación adecuadas y software de medición apropiado.

En base a las conclusiones sobre la seguridad del sistema final, un planteo de mejoras en este punto y sus posibles soluciones podrían ampliar el alcance del uso de la plataforma desarrollada.

Por otro lado sería un aporte importante analizar los mecanismos para hacer de este sistema una instalación permanente en una computadora. Esto podría basarse en la incorporación del instalador de Debian a la imagen y/o analizar las opciones de persistencia en medios USB o discos rígidos provistas por Debian Live.

Por último cabe mencionar que el modo ad-hoc de por sí no permite hacer forwarding de los paquetes. Esto implica que si dos computadoras A y B no están al alcance una de otra pero A sí está al alcance de una tercer computadora C y B también está al alcance de C entonces A y B no pueden usar C para comunicarse entre sí. Para resolver esta situación se emplea otro tipo de redes llamadas redes MESH. Se podría pensar como adaptar el sistema para usar MESH o soportar algún tipo de ruteo distribuido como Optimized Link State Routing protocol (OSLR). Esto permitiría ampliar el alcance de la red y compartir las conexiones a otras redes con las que podrían contar algunos de los nodos (Internet por ejemplo).

# Bibliografía


[1] http://www.infobae.com/notas/nota.php?Idx=274519&IdxSeccion=0 - Nota periodística donde se brindan datos de la venta de dispositivos móviles en la Argentina.

[2] http://www.conectarigualdad.gob.ar - Política de inclusión digital de alcance federal que plantea distribuir 3 millones de netbooks en el período 2010-2012.

[3] http://www.gnu.org/philosophy/free-sw.es.html - Definición de Software Libre de la Free Software Foundation.

[4] http://distrowatch.com - Sitio especializado en rastrear las distintas distribuciones de Linux.

[5] ISO/IEC 8802-11 Second edition: 2005/Amendment 6 2006: IEEE STD 802.11i-2004 - Estándar internacional que en su parte 11 describe la capa física y MAC para el intercambio de información en sistemas de área local y metropolitana en redes LAN inalámbricas. Capítulos 5 para modos de funcionamiento, 6 y 9 para capa MAC y 14 y 15 para caopa PHY.

[6] Este documento intitulado "Modos de funcionamiento 802.11 o Wi-Fi" de Kioskea.net está puesto a disposición bajo la licencia Creative Commons. Puede copiar, modificar bajo las condiciones puestas por la licencia, siempre que esta nota sea visible.

[7] http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Linux.Wireless.modem.html - Descripción de la capa física en redes WLAN escrita por Jean Tourrilhes, el desarrollador de Wireless Extensions para Linux, parte del documento "Linux Wireless LAN Howto".

[8] http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Linux.Wireless.mac.html - Parte del documento mencionado en la cita anterior dedicado a la capa de acceso al medio.

[9] http://www.science.uva.nl/research/air/wiki/ZeroconfTechnologies - Wiki del grupo de ingeniería en redes y sistemas de la Univ. de Amsterdam. Describe las líneas generales de Zeroconf.

[10] IETF RFC3927 - Rqequest For Comments para direcciones IP autoasignables (Link-Local). S. Cheshire, B. Aboba, E. Guttman. Mayo 2005.

[11] IETF RFC5227 - Request For Comments para detectar conflictos en la asignación de direcciones IP. S. Cheshire. Julio 2008.

[12] http://www.demolinux.org/ - Una de las primeras distribuciones Live de Linux.

[13] http://live-build.debian.net/cgi-bin/live-build - Web frontend para la ceración de imagenes Live de Debian Live.

[14] http://linuxwireless.org/en/developers/Documentation/Glossary#SoftMAC - Definiciones del Wiki oficial de Linux Wireless.

[15] http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Linux.Wireless.Extensions.html - Parte del

documento ya citado donde se describen las Wireless Extensions para Linux.

[16] http://linuxwireless.org/en/developers/Documentation/Wireless-Extensions - Documentación del futuro reemplazo de Wireless Extensions para Linux.

[17] Cisco Wireless LAN Security escrito por Krishna Sankar, Sri Sundaralingam, Andrew Balinsky y Darrin Miller y publicado por Cisco Press en 2004. Se encuentra disponible en http://my.safaribooksonline.com/book/networking/wireless/1587051540.

[18] "Tecnologías Wireless y Movilidad en IPv4/IPv6," presentado por el Ing. Luis Marrone, Lic. Andrés Barbieri y Mg. Matías Robles en el XVII Congreso Argentino de Ciencias de la Computación realizado en el año 2011.

[19] http://en.wikipedia.org/wiki/Comparison_of_open_source_wireless_drivers - Lista de la compatibilidad de los distintos chipsets wireless y su soporte con drivers para Linux.

[20] http://www.wi-fiplanet.com/tutorials/article.php/10724_1451421_2/Understanding-Ad-Hoc-Mode.htm - Análisis del modo ad-hoc realizado por Jim Geier en 2002 para el sitio wi-fiplanet.com especializado en Wi-Fi Libre.

[21] Zero Configuration Networking: The Definitive Guide escrito por Stuart Cheshire y Daniel H. Steinberg y publicado por O'Reilly Media, Inc. en 2005 ISBN. Los capitulos 3 al 5 decriben como se resuelve la asignación IP, la resolución de nombres y el anuncio de servicios respectivamente. Se encuentra disponible en http://my.safaribooksonline.com/book/networking/network-management/

0596101007/introduction-to-bonjour-and-zeroconf/bonjour-chp-1.

[22] IEEE Std 802.11-2007: IEEE Standard for Information technology Telecommunications and information exchange between systems. - El capitulo 8 del estandar citado se refiere al modo ad-hoc.

[23] Mecanismo IEEE 802.11 RTS/CTS - Descripto en el estándar SO/IEC 8802-11 citeado anteriormente. Capitulo 7 sección 2.1 Control frames y anexo C Introduction to the MAC formal description.

[24] http://www.zeroconf.org/ZeroconfAndUPnP.html - Comparación de Zeroconf con UPnP realizada por Stuart Cheshire, responsable de Apple para Zeroconf y partícipe de la redacción final del estandar.

[25] https://launchpad.net/win32-image-writer - Página del programa para Windows "Image Writer for Windows" usado para grabar imagenes en pendrives.

[26] http://www.linux-live.org - Web de la herramienta para la creación de imagens live "Linux Live Scripts".

[27] http://en.wikipedia.org/wiki/Remastersys - Web de la herramienta para la creación de imagens live "Remastersys".

[28] http://bkhome.org/woof/index.html - Web de la herramienta para la creación de imagens live "Woof" de la distribución "Puppy Linux".

[29] http://live.debian.net/manual/es/html/about-project.html - Manual del proyecto Debian Live, usado intensivamente durante el desarrollo de la imagen y para documentar la herramienta.