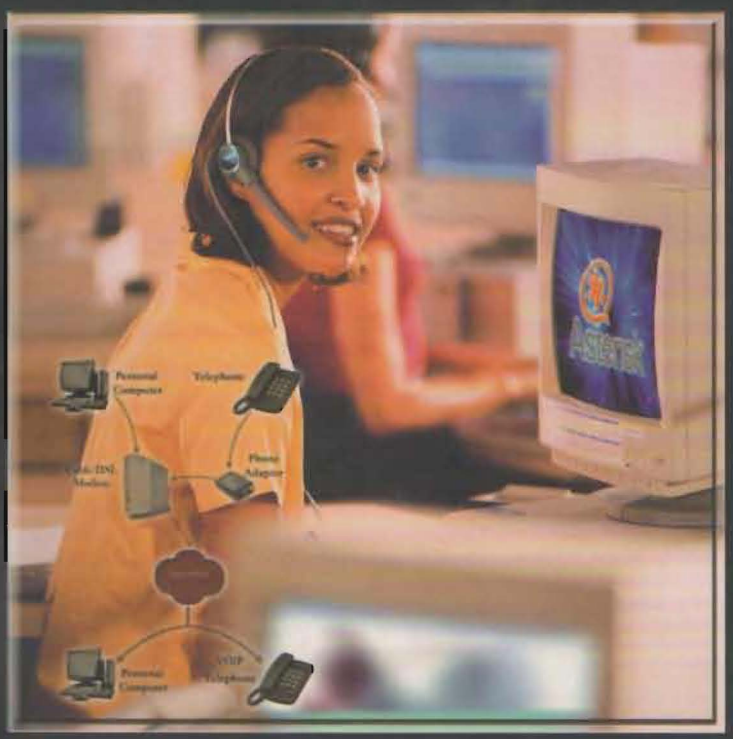
VoIP y Asterisk

Redescubriendo la telefonía



COORDINADORES:

Julio Gómez López

Francisco Gil Montoya

Alfaomega Ra-Ma®

VoIP y Asterisk

Redescubriendo la telefonía

Julio Gómez López

Francisco Gil Montoya

VoIP y Asterisk

Redescubriendo la telefonía

Desde que Mark Spencer escuchó el primer tono de línea en su servidor Linux hasta hoy día, el uso de la VoIP se ha consolidado enormemente en ámbitos empresariales, institucionales y académicos gracias a Asterisk. Hoy día se pueden confiar las comunicaciones de cualquier empresa a un sistema considerado por todos los especialistas como de los más estables y fiables. Y encima, gratis.

Con esta obra se pretende conocer de primera mano cómo es posible la comunicación telefónica mediante métodos basados en el uso de redes Ethernet. Para ello, Linux y Asterisk se convierten en un factor esencial, ya que van a proporcionar la plataforma ideal sobre la que desarrollar e implementar un amplísimo abanico de posibilidades que no se podían ni imaginar antes, con el uso de sistemas telefónicos convencionales y propietarios. Y encima, a un costo muy inferior...

En este libro el lector podrá encontrar una guía simple y muy práctica para implantar su sistema telefónico de voz sobre IP basado en Asterisk. Encontrará multitud de ejemplos, así como aplicaciones que facilitarán su instalación y configuración.

Temas incluidos:

- **La telefonía tradicional.** Sistemas analógicos, sistemas digitales, redes móviles y centralitas tradicionales.
- **VoIP – La nueva revolución.** Introducción, ventajas, arquitectura, protocolos de señalización y de audio.
- **La revolución se llama Asterisk.** Instalación, configuración básica, interconexión de centralitas, etc.
- **Lógica de marcado o Dialplan.** Administración de contextos, extensiones, buzón de voz, operadora automática, etc.
- **Gestión de Asterisk mediante interfaz web.** Instalación y configuración de FreePBX, configuración de trunks, extensiones, IVR, ring groups, etc.

Al final del libro se encuentran los siguientes anexos que complementan la obra: administración básica de GNU/Linux, aspectos básicos de redes, clientes de VoIP, distribuciones precompiladas de Asterisk, software de terceros para Asterisk y seguridad de un sistema VoIP.

www.alfaomega.com.mx

ISBN 978-607-7686-08-8

Alfaomega Grupo Editor

VoIP y Asterisk

Redescubriendo la telefonía

VoIP y Asterisk

Redescubriendo la telefonía

Coordinadores:

Julio Gómez López

Francisco Gil Montoya

Autores de capítulo:

(por orden alfabético)

Alfredo Alcayde García	Ingeniero Informático
Raúl Baños Navarro	Investigador de la Universidad de Almería
Jesús Camacho Rodríguez	Ingeniero Informático
Juan Antonio García Moreno	Consultor de Comunicaciones
Consolación Gil Montoya	Profesora de la Universidad de Almería
Francisco Gil Montoya	Profesor de la Universidad de Almería
María Dolores Gil Montoya	Profesora de la Universidad de Almería
Julio Gómez López	Profesor de la Universidad de Almería
Saúl Ibarra Corretgé	Consultor de tecnologías VoIP
Francisco José Méndez Cirera	Ingeniero Informático
David Prieto Carrellán	Ingeniero Informático

Almería, 2008

Alfaomega Ra-Ma®

Datos catalográficos
Gómez, Julio; Gil, Francisco
VoIP y Asterisk. Redescubriendo la telefonía
Primera Edición
Alfaomega Grupo Editor, S.A. de C.V., México
ISBN: 978-607-7686-08-8
Formato: 17 x 23 cm Páginas: 348

VoIP y Asterisk. Redescubriendo la telefonía
Julio Gómez López, Francisco Gil Montoya
ISBN: 978-84-7897-902-8, edición original publicada por RA-MA Editorial, Madrid, España

Primera edición: Alfaomega Grupo Editor, México, enero 2009

Pitágoras 1139, Col. Del Valle, 03100, México D.F.

Miembro de la Cámara Nacional de la Industria Editorial Mexicana
Registro No. 2317

Pág. Web: **http://www.alfaomega.com.mx**
E-mail: **atencionalcliente@alfaomega.com.mx**

ISBN: 978-607-7686-08-8

Edición autorizada para venta en México y todo el continente americano.

Impreso en México. Printed in Mexico.

Empresas del grupo:

México: Alfaomega Grupo Editor, S.A. de C.V. – Pitágoras 1139, Col. Del Valle, México, D.F. – C.P. 03100.
Tel.: (52-55) 5089-7740 – Fax: (52-55) 5575-2420 / 2490. Sin costo: 01-800-020-4396
E-mail: atencionalcliente@alfaomega.com.mx

Colombia: Alfaomega Colombiana S.A. – Carrera 15 No. 64 A 29 – PBX (57-1) 2100122, Bogotá, Colombia, Fax: (57-1) 6068648 – E-mail: scliente@alfaomega.com.co

Chile: Alfaomega Grupo Editor, S.A. – General del Canto 370-Providencia, Santiago, Chile
Tel.: (56-2) 235-4248 – Fax: (56-2) 235-5786 – E-mail: agechile@alfaomega.cl

Argentina: Alfaomega Grupo Editor Argentino, S.A. – Paraguay 1307 P.B. "11", Buenos Aires, Argentina, C.P. 1057 – Tel.: (54-11) 4811-7183 / 8352, E-mail: ventas@alfaomegaeditor.com.ar

Dedicado a María,
que trajo este libro bajo el brazo

ÍNDICE

INTRODUCCIÓN

Francisco Gil Montoya y Julio Gómez López

La telefonía de hoy en día, tal y como la hemos conocido siempre, toca a su fin. La era de las nuevas tecnologías, con Internet a la cabeza, está poniendo patas arriba bastantes nichos tecnológicos que han permanecido invariables e intocables durante mucho tiempo. El mundo de las comunicaciones por voz es uno de ellos.

Desde que Antonio Menucci[1] inventara el primer teléfono (existe cierta polémica acerca de quién inventó realmente el primer teléfono) allá por el año 1860, se han venido produciendo cambios y mejoras en los sistemas de telefonía que han permitido su expansión por todo el mundo, llegando a prácticamente todos los hogares y rincones.

No obstante, a día de hoy, estos sistemas siguen basándose en tecnologías de hace varias décadas, obsoletas y que no son óptimas en muchos sentidos. Piénsese, por ejemplo, en el uso que se hace de la línea telefónica cuando se establece una comunicación: desde un extremo hasta el otro, se necesita que exista una canal (habitualmente un par de cobre) constantemente abierto o dedicado, con el consiguiente despilfarro de recursos.

---

[1] http://es.wikipedia.org/wiki/Antonio_Meucci

Hoy en día ya no es necesario dedicar un recurso por completo para mantener una conversación. En la era de Internet, es posible que una conversación telefónica se pueda mantener entre cualesquiera dos puntos, ocupando una simple porción del espectro o ancho de banda de nuestra conexión de área local o hacia Internet. La voz se convierte en paquetes y pasa a denominarse **Voz sobre IP** (**VoIP**[2] en inglés).

Si la voz ya no viaja por un circuito dedicado y exclusivo, sino que ahora es parte de nuestras comunicaciones de datos, conseguimos otro de los grandes beneficios de las nuevas tecnologías, la **unificación.** La voz y los datos viajan por la misma red, y no sólo eso, además son tratados y gestionados de forma conjunta y coordinada. Este escenario permite un uso óptimo de los recursos.

Todo lo anterior implica que, a partir de ahora, la forma en que estamos acostumbrados a comunicarnos cambiará sustancialmente. Las grandes compañías ya lo saben y se están preparando para ello. Gracias al despliegue de nuevas redes de datos (VDSL, FTTH, etc.), se puede ofrecer ya una gran cantidad de servicios al usuario final, con una calidad óptima y además de manera centralizada: televisión, Internet y por supuesto, voz. Ya no será necesario una "línea telefónica" y sí una línea de datos.

Dentro de este contexto, la voz sobre IP está teniendo un auge vertiginoso. Fundamentalmente, aquellos países que disponen y/o se están dotando de redes de última generación y aquellos que realmente están invirtiendo en ello, disponen de ventaja en su uso y disfrute. Sin redes fiables que dispongan de un buen caudal en cuanto al ancho de banda, no es posible una migración seria y efectiva. Desde este punto de vista, los países o comunidades que no tengan claro este concepto perderán el tren digital y aumentarán aún más su déficit tecnológico (la famosa *brecha digital*).

La voz sobre IP está teniendo un gran auge actualmente, apoyada en dos pilares fundamentales: el protocolo SIP[3] y las aplicaciones GPL[4] (*OpenSource*) tipo Asterisk[5].

---

[2] http://es.wikipedia.org/wiki/Voip

[3] http://es.wikipedia.org/wiki/Session_Initiation_Protocol

[4] http://es.wikipedia.org/wiki/GPL

[5] http://www.asterisk.org

Sobre todo, este último está teniendo una gran aceptación en los entornos más dinámicos e innovadores como son las empresas de tecnología o de corte tecnológico.

Asterisk es una plataforma telefónica de código abierto (GPL) que pretende revolucionar el mundo de las comunicaciones IP frente a las tradicionales soluciones de grandes corporaciones como Cisco, Nortel, Ericsson, Siemens, etc., caracterizadas por su falta de transparencia hacia el usuario, sus protocolos propietarios y cerrados, así como su elevado coste. También se habla de Asterisk como una plataforma convergente por cuanto que unifica muchos de los servicios que, tradicionalmente, se ofrecían por separado y/o en sistemas no integrados.

Asterisk es una solución completamente software y corre bajo GNU/Linux. Esta configuración le confiere una robustez innata para desplegar servicios típicos de los sistemas tradicionales, pero aportando mucha más flexibilidad, control, creatividad y a muy bajo coste.

A los típicos servicios de buzón de voz, conferencias, colas, agentes, música en espera, parking de llamadas, etc., se le une toda la potencia de interacción con cualquier lenguaje de programación para realizar cualquier funcionalidad que se desee. Todo se hace vía software. Y se hace de una manera transparente, cumpliendo los estándares internacionales fijados de manera que pueda interoperar con otros sistemas o tecnologías de manera clara y cercana. Esto último sólo lo pueden soñar los sistemas propietarios (¡¡ y a qué precio oiga!!).

Mediante la presente obra se intenta mostrar un enfoque eminentemente práctico en el uso de la voz sobre IP a través del paquete **Asterisk**. Este enfoque es, fundamentalmente, inicial, básico y con un carácter eminentemente introductorio. No obstante, se incluyen los aspectos esenciales que ayudarán a tener una idea general y de conjunto, permitiendo al lector recorrer un camino personal en la construcción de un sistema completo de telefonía IP mediante herramientas de uso libre y código abierto.

Para una mejor comprensión, se emplean numerosos ejemplos en aquellos capítulos que se centran, mayoritariamente, en la descripción de la filosofía de trabajo de Asterisk. Es recomendable que el lector esté familiarizado con una administración básica de Linux, aunque esto no es del todo imprescindible.

El libro se estructura en los siguientes capítulos:

- ***Capítulo 1. La telefonía tradicional.*** Se proporciona una visión genéricá acerca de las tecnologías más usuales existentes en el campo de las comunicaciones y, más concretamente, en la Red Pública Conmutada RTC (PSTN en inglés).

- ***Capítulo 2. VoIP - La nueva revolución.*** En este capítulo se cubren los aspectos básicos y esenciales de la incipiente tecnología de Voz sobre IP (VoIP en inglés). Se detallan los protocolos más importantes y utilizados en la vida real, así como los elementos de hardware y software que hacen posible la comunicación.

- ***Capítulo 3. La revolución se llama Asterisk.*** En este capítulo se entra de lleno en el apasionante mundo de Asterisk. Se habla y detallan todos los aspectos de su potencial, instalación, lógica de trabajo y configuración. Todo lo anterior, aderezado de sugerentes y aclaradores ejemplos prácticos.

- ***Capítulo 4. Lógica de marcado o "Dialplan".*** Una vez conocida la rutina de trabajo de Asterisk, se pasa a detallar su sintaxis, las funciones y variables de trabajo, así como los diferentes módulos que lo componen y/o que pueden añadirse a conveniencia. También se presentan numerosos ejemplos prácticos alrededor de cada explicación.

- ***Capítulo 5. Gestión de Asterisk mediante interfaz Web.*** Se detalla cómo es posible manejar la configuración de Asterisk mediante diversas interfaces Web al objeto de automatizar el proceso de creación de usuarios, troncales de salida, IVR's, etc., así como acercar la complejidad y potencial que presenta el sistema al usuario menos experto o no iniciado.

Al final del libro se encuentran una serie de anexos que complementan la filosofía general del libro. En ellos se realiza una introducción general al mundo de GNU/Linux y las redes de datos. Así mismo, se presentan una serie de ejemplos de configuración de clientes para VoIP junto a una detallada descripción de software creado para trabajar conjuntamente con Asterisk. Se presentan unos ejemplos prácticos que ayudarán al lector a comprender mucho mejor la potencia del sistema, así como sus diversos usos en el mundo real. Por último, se realiza una breve introducción a los diferentes problemas de seguridad que pueden surgir en los sistemas de telefonía IP, proponiendo algunas soluciones a los mismos.

Además se pone a disposición del lector el uso de la Web *http://www.adminso.es* para completar información relacionada con Asterisk. Tras un proceso de registro, se tendrá acceso a diferente material electrónico como, por ejemplo, ficheros de configuración, diferente material didáctico, etc.

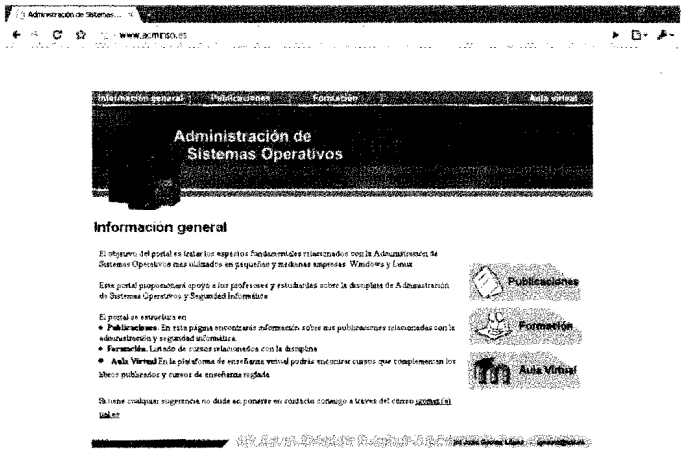


Figura 1. Portal www.adminso.es

Por último, reseñar que el libro está basado en la rama 1.4 de Asterisk, que a fecha de edición, es la rama estable y en desarrollo del proyecto. Existen además la rama 1.0 y 1.2 (en estado de mantenimiento, es decir, sólo se resuelven aspectos críticos o de seguridad), y la rama 1.6 (aún en estado beta).

CAPÍTULO 1

LA TELEFONÍA TRADICIONAL

Jesús Camacho Rodríguez y Francisco José Méndez Cirera

1 Sistemas analógicos

La red telefónica básica (**RTB**[1]) fue creada para transmitir la voz humana. Tanto por la naturaleza de la información a transmitir, como por la tecnología disponible en la época en que fue creada, es de tipo analógico. Hasta hace poco se denominaba **RTC** (Red Telefónica Conmutada), pero la aparición del sistema **RDSI** (digital pero basado también en la **conmutación de circuitos**) ha hecho que se prefiera utilizar la terminología **RTB** para la primitiva red telefónica (analógica), reservando las siglas **RTC** para las redes conmutadas de cualquier tipo (analógicas y digitales); así pues, la **RTC** incluye la primitiva **RTB** y la moderna **RDSI** (Red Digital de Servicios Integrados). RTB es en definitiva la línea que tenemos en el hogar o la empresa, cuya utilización ha estado enfocada fundamentalmente hacia

[1] La RTB es conocida en literatura inglesa como **PSTN**.

las comunicaciones mediante voz, aunque cada vez ha ido tomando más auge el uso para transmisión de datos como fax, Internet, etc.

Cada línea **RTB** tiene asignada una numeración específica (su dirección telefónica) y está físicamente construida por dos hilos metálicos (conocidos como par de cobre), que se extienden desde la central telefónica hasta la instalación del abonado (se conoce también como bucle de abonado). Cada central atiende las líneas de abonado de un área geográfica determinada. A su vez, las centrales telefónicas están unidas entre sí por sistemas más complejos y basados en tecnología digital. Esta unión de centrales constituye el sistema telefónico nacional que a su vez está enlazado con los restantes del mundo.

En los años 60 las centrales telefónicas, mayoritariamente analógicas, fueron transformando su tecnología a digital. Ello solventó diversos problemas, como los relacionados con la degradación de la señal de voz y la imposibilidad de manejar gran cantidad de llamadas. Del mismo modo, la intención fue también digitalizar el bucle local pero por motivos meramente económicos el bucle local continuó siendo analógico. Finalmente, la medida que se adoptó fue la de digitalizar la comunicación entre las centralitas telefónicas, manteniendo el bucle local analógico, y obteniéndose así los beneficios de la telefonía digital a un precio razonable. Esta medida dio lugar a lo que se conoce como RDI "Red Digital Integrada".

La situación actual para la **RTB** puede clasificarse como híbrida; lo normal es que la transmisión sea todavía analógica en los bucles de abonado de ambos extremos y digital en su tráfico entre centrales (esto requiere una doble conversión, analógico-digital y digital-analógico). Para su digitalización, la señal analógica es muestreada a 8.000 veces por segundo (8 Khz.). El valor de cada muestra puede ser un valor entre 0 y 255 (puede ser representado por 1 byte -octeto-) lo que supone un flujo de datos de 8 KB/s o 64 Kb/s, lo cual se denomina calidad de sonido telefónico.

Como hemos visto, se disponga por tanto de tecnología RDSI o analógica se requiere de un enlace desde nuestro hogar hasta la central telefónica asignada a nuestra zona. Es por ello que es de gran importancia conocer los dos tipos de conexiones telefónicas analógicas existentes, conocidas como FXS y FXO, es decir, los nombres de los puertos o interfaces usados por las líneas telefónicas y los dispositivos analógicos.

1.1 FXS

La interfaz "*Foreign eXchange Subscriber*" o FXS es el puerto por el cual el abonado accede a la línea telefónica, ya sea de la compañía telefónica o de la

central de la empresa. En otras palabras, la interfaz FXS provee el servicio al usuario final (teléfonos, módems o faxes).

Los puertos FXS son, por lo tanto, los encargados de:

- Proporcionar tono de marcado.
- Suministrar tensión (y corriente) al dispositivo final.

Para entender mejor el concepto, piense en el caso de un hogar tradicional. La interfaz FXS es el punto donde se conectan los teléfonos del hogar que quieren hacer uso de la línea. La interfaz FXS sería entonces la roseta de telefonía del hogar.

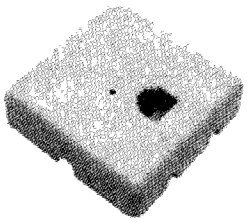


Figura 1-1. Roseta telefónica o PTR

1.2 FXO

La interfaz "*Foreign eXchange Office*" o FXO es el puerto por el cual se recibe a la línea telefónica. Los puertos FXO cumplen la funcionalidad de enviar una indicación de colgado o descolgado conocida como cierre de bucle.

Un ejemplo de interfaz FXO es la conexión telefónica que tienen los teléfonos analógicos, fax, etc. Es por ello que a los teléfonos analógicos se les denomina "dispositivos FXO".

A modo de resumen se quiere destacar que dos puertos se pueden conectar entre sí con la condición de ser de distinto tipo, es decir, FXO y FXS son siempre pareja (similar a un enchufe macho/hembra).

En la figura 1-3 se muestra el escenario de un hogar tradicional. Como podemos apreciar siempre se conectan entre sí interfaces de distintos tipos, es decir, FXS con FXO o viceversa. El teléfono posee una interfaz FXO como se

muestra en la imagen, el cual es conectado a la roseta de la compañía telefónica (FXS).

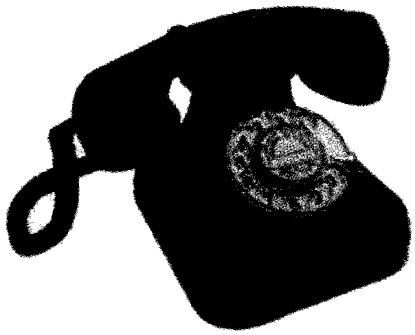


Figura 1-2. Dispositivo FXO

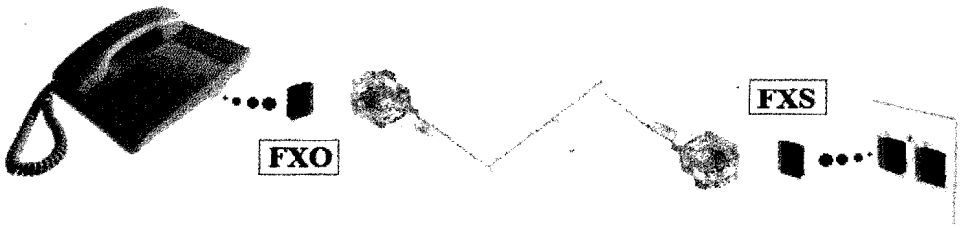


Figura 1-3. FXS /FXO sin centralita

2 Sistemas digitales

2.1 RDSI

Los trabajos de desarrollo de la Red Digital de Servicios Integrados (RDSI o ISDN, en inglés Integrated Services Digital Network) comenzaron en la década de los 80, pero ésta no sería comercializada hasta principios de los años 90. Se esperaba que la RDSI pudiera revolucionar la industria de las comunicaciones telefónicas como hoy día se espera que lo pueda hacer la VoIP. Sin embargo, y aunque las compañías telefónicas pusieron mucho empeño en extenderlo al mayor número de lugares posibles, muchos consideran la RDSI un fracaso debido a que

todo lo que prometía no se pudo llevar a cabo. Lo cierto es que la RDSI nunca terminó de despegar ya que cuando lo estaba haciendo surgió otra tecnología que tuvo una implantación mucho más barata y rápida, la *Asymmetric Digital Subscriber Line* (ADSL).

La RDSI permite que en una línea coexistan múltiples canales, pudiendo contener cada uno de ellos datos (canales B) o señalización (canales D). Pero además, la RDSI no se limita sólo a la transmisión de voz. Cada canal tiene un ancho de banda de 64 Kbps, de forma que pueden emplearse canales B y D para la transmisión de datos (éstos últimos siempre que no haya datos de señalización). Precisamente esta característica dota a la RDSI de una mayor flexibilidad que la que poseen las líneas de la RTB, ya que los canales pueden ser reconfigurados sobre la marcha para que transmitan voz o datos.

Tal y como se muestra en la figura 1-4, la línea RDSI básica, también conocida como BRI (Basic Rate Interface), tiene tres canales (dos canales B y un canal D), de forma que pueden realizarse dos llamadas telefónicas de forma simultánea en una única BRI. Los usuarios finales de este tipo de línea fueron, en principio, empresas relativamente pequeñas. Desafortunadamente, cuando esta versión de la RDSI fue lanzada al público, otros tipos de medios y servicios ya habían evolucionado de forma que ofrecían más ancho de banda sin la complejidad y el coste asociados a ésta. Todavía existen algunos usuarios de líneas BRI (emplean ésta principalmente para videoconferencia debido a su ancho de banda fijo), pero en la mayoría de los casos se encuentran en proceso de cambio hacia la ADSL, cable o algún tipo de tecnología inalámbrica.

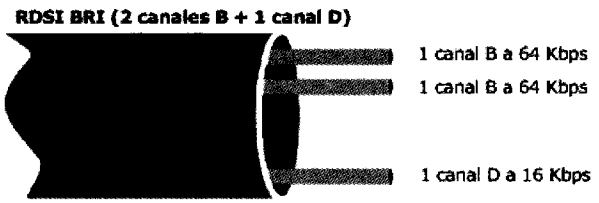


Figura 1-4. Arquitectura de un cable RDSI BRI

A diferencia de la versión BRI de RDSI, la PRI (Primary Rate Interface) posee dos versiones, una de 31 (30 canales B y 1 canal D) y otra de 24 canales (23 canales B y 1 canal D), por lo tanto, con ésta pueden realizarse 30 ó 23 llamadas

telefónicas al mismo tiempo[2], respectivamente. Su implantación ha sido mayor que la de la BRI y normalmente constituye la elección para instalaciones de un tamaño considerable. Además, sus costes son proporcionalmente menores que los asociados a la BRI.

2.2 E1/T1

Un T1 es un acceso digital que dispone de 24 canales, pudiéndose realizar en cada uno de ellos (menos uno) una llamada.

Mientras que el T1 es muy común en Estados Unidos y Japón, en Europa se emplea con mayor frecuencia el E1. A diferencia del T1, esta línea dispone de 32 canales en vez de 24.

Tanto los T1s como los E1s tienen que señalizar las llamadas de alguna manera. Esto se consigue mediante lo que se conoce como Señalización por Robo de Bit (*Robbed Bit Signaling*), es decir, que cada cierto tiempo se usa un bit de cada canal para así señalizar y enviar información a través de la línea (T1s), o mediante multiplexación del bit en un canal común, algo que se emplea sobre todo en Europa (E1s).

Usar T1s y E1s para proporcionar datos y voz a la vez es muy común. En esta ocasión, algunos de los canales de las líneas son asignados para ser usados para datos y otros son asignados para ser usados para voz. Incluso se puede dar el caso de que existan canales sin usar. Los proveedores de servicios pueden proporcionar en este caso precios más bajos de lo normal, ya que, por ejemplo, unos cuantos canales podrían ser para voz, otros para conectarse a Internet y un último grupo podría ser para conectarse de forma privada a otra oficina de la organización.

Por todo lo comentado, si necesita tener, por ejemplo, de 8 a 16 líneas así como conexión de datos, tanto un T1 como un E1 (dependiendo de la zona donde estemos) podrían constituir una buena elección.

---

[2] Los dos tipos de enlaces primarios se denominan E1 y T1. El primero de ellos es utilizado en Europa y Australia, mientras que el segundo se usa en Estados Unidos, Canadá y Japón, fundamentalmente.

2.3 OTROS

Además de las líneas mencionadas anteriormente, existen otros tipos de líneas digitales que son empleadas normalmente para realizar la comunicación de una red a otra red. Principalmente se trabaja con las siguientes:

- Las líneas T3s, que son proporcionadas a través de cable coaxial o enlace de microondas y que son capaces de transportar 28 T1s, o lo que es lo mismo, 672 canales. Esto hace que una T3 tenga un ancho de banda de 44,736 Mbps.

- Las líneas E3s, proporcionadas únicamente a través de cable coaxial. Son capaces de transportar 16 E1s, lo que hace un total de 512 canales. El ancho de banda de este tipo de líneas es de 34,368 Mbps.

- Las líneas T4s, proporcionadas tanto a través de cable coaxial como a través de enlace de microondas. Son capaces de transportar 168 T1s, es decir, 4.032 canales, por lo que su ancho de banda es de 274,176 Mbps.

- Por último, la *Synchronous Optical Network* (SONET) y la *Synchronous Digital Hierarchy* (SDH), proporcionadas a través de fibra óptica. La primera se emplea en Estados Unidos y Canadá, mientras que la segunda lo es en el resto del mundo. Los anchos de banda de transmisión de datos empleados en estas líneas varían desde los 51,840 Mbps hasta los 39,813 Gbps (aunque teóricamente se podrían alcanzar los 159,252 Gbps).

La relación anterior no es definitiva, ni mucho menos, ya que cada día salen al mercado nuevos estándares de mucha más capacidad, como por ejemplo el novedoso FTTH.

Para la señalización entre redes, además de emplear las técnicas mencionadas anteriormente en T1 y E1, se suele emplear también un método llamado *Signaling System 7* (SS7), conocido como C7 en los países europeos. Éste es un protocolo que aporta ciertas ventajas sobre los otros ya que está basado en conmutación de paquetes y la señalización no se realiza de forma intercalada en la línea de transmisión, sino que se realiza a través de paquetes que contienen toda la información necesaria al comienzo de la conexión. Esto provoca que toda la información sea enviada de manera más rápida.

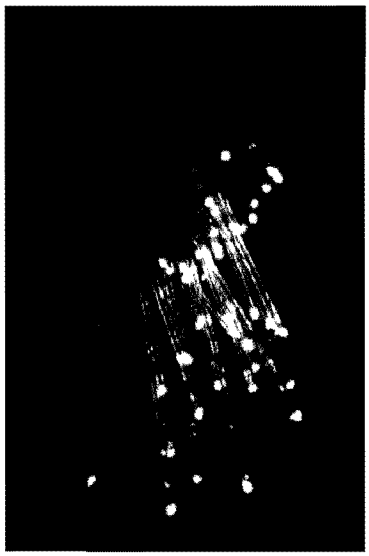


Figura 1-5. Cable de fibra óptica

3 Redes móviles

3.1 GSM (2G)

GSM (*Global System for Mobile communications*, proveniente en un principio de *Groupe Special Mobile*) es el estándar más popular y extendido para teléfonos móviles en todo el mundo. Se comenzó a trabajar en él en la década de los 80, pero no sería hasta 1991 cuando la primera red GSM fue lanzada, concretamente en Finlandia. Su promotor, la Asociación GSM, estima que el 82% del mercado global de teléfonos móviles lo emplea. Precisamente su ubicuidad hace que el *roaming*[3] internacional sea muy común entre los operadores de móviles, permitiendo a los usuarios suscritos a sus servicios emplear sus teléfonos en muchas partes del mundo. Los operadores móviles también se han visto favorecidos por esta implantación tan grande ya que les ha permitido elegir su equipamiento entre multitud de fabricantes de todo el mundo que emplean GSM en sus dispositivos.

---

[3] Roaming o itinerancia es un concepto relacionado con la capacidad de un dispositivo para moverse de una zona de cobertura a otra.

Figura 1-6. Logotipo GSM

GSM difiere de sus predecesores en que ambos canales, tanto el de señalización como el de voz, son en esta ocasión digitales. Por ello se considera a GSM como un sistema de telefonía móvil de segunda generación (2G). Además, con GSM comenzó a ser mucho más fácil integrar, en los teléfonos móviles, la posibilidad de establecer comunicaciones de datos.

GSM en una red celular para dispositivos móviles, lo que significa que los terminales se conectarán a ella buscando estaciones base (también conocidas como células o BTS, en inglés Base Telephone Station) en sus inmediaciones. GSM funciona principalmente en cuatro rangos de frecuencias: las bandas de frecuencia de 900 MHz y 1800 MHz son las más comunes, mientras que en algunos países americanos (como Estados Unidos o Canadá) se emplean las bandas de 850 MHz y 1900 MHz debido a que las anteriores se encontraban ya en uso para otras aplicaciones. También existen casos, aunque son poco frecuentes, en los que se emplean las bandas de frecuencia de 400 MHz y 450 MHz. Este hecho se produce por ejemplo en los países escandinavos, donde los dispositivos móviles de primera generación comenzaron empleando esos rangos de frecuencias y decidieron mantenerlos para su uso con GSM.

La red existente detrás de GSM (y que el usuario corriente no percibe) es bastante grande y compleja. De otra manera sería imposible proporcionar todos los servicios que el usuario final recibe de ésta.

GSM emplea varios codecs[4] de audio para comprimir el sonido transmitido a través de los terminales móviles. Al principio, fueron empleados dos codecs, *Half Rate* y *Full Rate*, que se llamaban así debido a la relación que éstos guardaban con la forma en la que usaban el canal de transferencia (de forma parcial o de forma completa, respectivamente) en el que eran empleados. Ambos codecs eran bastante

---

[4] *Codec* es una abreviatura de codificación-decodificación. Su uso se refiere a la capacidad de codificar y decodificar una señal de audio en un sistema concreto.

eficientes en cuanto a compresión, además de implementar la identificación de partes importantes de audio permitiendo la priorización y protección de dichas partes. A partir de 1997 comenzó a emplearse el codec *Enhanced Full Rate* (EFR), que mejoró el estándar y usaba el canal de transferencia completamente.

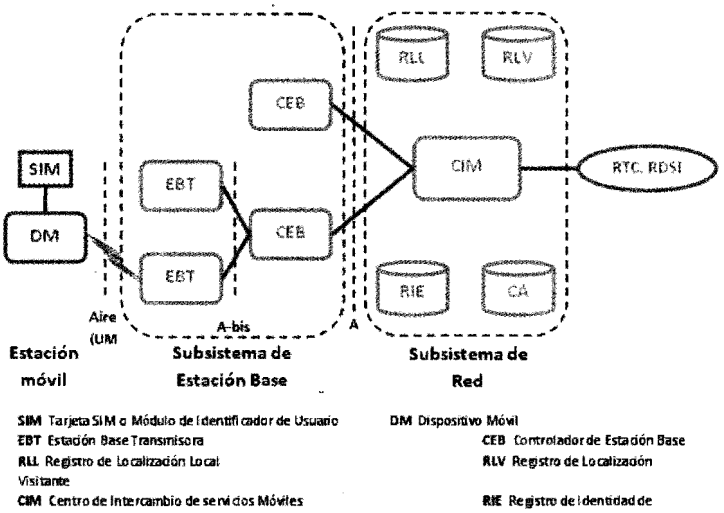


Figura 1-7. Esquema de una red tradicional GSM

El estándar GSM fue pionero al incluir una alternativa barata a las llamadas de voz a través de la red de telefonía, el mensaje de texto (*Short Messaging Service* o SMS), soportado hoy día por prácticamente la totalidad de estándares para móvil. También incluía el número de teléfono para emergencias (concretamente en Europa es el 112), que hacía muy fácil a los viajeros el poder contactar con los servicios de emergencias sin tener que conocer el número local de éstos.

Las nuevas versiones del estándar han sido retro-compatibles con los teléfonos GSM originales. En la especificación de 1997 el estándar añadió capacidad para transportar paquetes de datos a través del servicio *General Packet Radio Service* (GPRS), incluyendo entre otras cosas mensajes multimedia (*Multimedia Messaging Service* o MMS) o aplicaciones de Internet a través del *Wireless Application Protocol* (WAP). GPRS es comúnmente conocido como 2,5G, debido a que es una especificación que se encuentra entre la segunda y la tercera generación de telefonía móvil. En la especificación de 1999 se introdujo una mejora en la velocidad de transmisión de datos a través del uso del servicio *Enhanced Data rates for GSM Evolution* (EDGE).

3.2 UMTS (3G)

UMTS (*Universal Mobile Telecommunications System*) es una tecnología de tercera generación (3G) para telefonía móvil. Está estandarizado por 3GPP (3rd *Generation Partnership Program*), una colaboración entre grupos de telecomunicaciones de varios lugares del mundo para desarrollar una especificación de un sistema de telefonía aplicable globalmente y que cumpla las exigencias de ITU IMT-2000. Ese sistema está basado en una evolución de las especificaciones de GSM. 3GPP fue creado a finales de 1998, pero no sería hasta principios de 2000 cuando surgiría la especificación de la primera red UMTS.

Las bandas de frecuencia en las que opera UMTS varían dependiendo del país, aunque en el estándar original se definía el rango de frecuencias 1885–2025 MHz para la comunicación de móvil a estación base (*uplink* o enlace de subida) y el rango de frecuencias 2110–2200 MHz para la comunicación de estación base a móvil (*downlink* o enlace de bajada). El amplio espectro de frecuencias que emplea UMTS ha sido ampliamente criticado ya que retrasó el despliegue en algunos países en los que es lenta la asignación de frecuencias (como en Estados Unidos).

Para los operadores de GSM existentes, la migración de esta tecnología a UMTS constituye un camino simple pero costoso. Esto se debe a que una gran parte de la infraestructura de UMTS se comparte con GSM, pero el coste de obtener las nuevas licencias para el espectro de frecuencias así como implementar UMTS en las torres existentes es alto. Por otra parte, los mayores fabricantes de móviles 2G son hoy día también fabricantes de modelos 3G.

Además de la familia GSM de codecs de voz, con el desarrollo de UMTS, EFR dio paso a un codec de ratio variable llamado *AMR-Narrowband*. Éste tiene una calidad alta y es robusto contra interferencias cuando es usado empleando el canal de transferencia completamente, mientras que es menos robusto pero mantiene una calidad relativamente alta cuando es usado en buenas condiciones empleando el canal de transferencia de forma parcial.

UMTS proporciona una gran mejora en la transferencia de datos con respecto a sus predecesores, pudiendo alcanzar (eso sí, de forma teórica) hasta 14 Mbps. En la práctica se han llegado a alcanzar tasas de transferencia de bajada de 7,2 Mbps, una velocidad muy superior a los 9,6 Kbps que ofrecían los primeros canales de datos empleados en GSM. Esta velocidad de transferencia ha abierto la posibilidad de ejecutar aplicaciones y realizar acciones con nuestros terminales móviles que nos parecían impensables hace tan sólo unos años. A largo plazo, el proyecto *3GPP Long Term Evolution* planea que UMTS pueda alcanzar en una tecnología para móviles de cuarta generación (4G) velocidades de bajada de hasta 100 Mbps y de subida de hasta 50 Mbps.

Las primeras redes comerciales UMTS fueron lanzadas en 2002 y para promocionarlas se hizo especial énfasis en las posibilidades que éstas brindaban relacionadas con aplicaciones como la televisión por móvil o la videoconferencia. Poco a poco, las experiencias en Japón y otros lugares de temprana implantación mostraron que las videoconferencias no eran muy empleadas y que aplicaciones como la televisión por móvil no alcanzaban la demanda esperada, empleándose la alta velocidad de transferencia de datos de UMTS mayoritariamente para acceder a Internet. De esta forma, hoy día es común el uso de las redes UMTS para acceder a Internet, ya sea directamente desde un terminal móvil o bien desde un ordenador a través de Wi-Fi, Bluetooth, infrarrojos o USB.

4 Centralitas tradicionales PBX

4.1 INTRODUCCIÓN

Una **Centralita privada** o ***PBX***[5] es un dispositivo de telefonía que actúa como conmutador de llamadas en una red telefónica o de conmutación de circuitos.

La centralita es un dispositivo de telefonía que se suele utilizar en la mayoría de las medianas y grandes empresas, no así en los hogares, donde los terminales existentes son pocos y las exigencias no son importantes. Permite a los usuarios o abonados compartir un determinado número de líneas externas (analógicas o digitales) para hacer llamadas telefónicas entrantes o salientes, así como establecer comunicaciones internas entre todos los dispositivos que dependen de la PBX. Entre las muchas ventajas que ofrece, una PBX es una solución mucho menos cara que proporcionar a cada usuario de la empresa una línea telefónica externa. Así mismo, a una PBX se le pueden conectar teléfonos, máquinas de fax, módems y otros dispositivos de comunicación.

La PBX normalmente se instala en la propia empresa y conecta las llamadas entre los teléfonos situados e instalados en la misma. Habitualmente, hay un número limitado de líneas externas, también llamadas líneas troncales, para realizar y recibir llamadas externas a la empresa desde un origen externo que suele ser la RTC (PSTN).

Las llamadas realizadas a números de teléfono externos, mediante una PBX, se suelen realizar anteponiendo un dígito (habitualmente el 0) al número

---

[5] Private Branch Exchange en inglés.

externo en algunos sistemas, de forma que la PBX selecciona automáticamente una línea troncal saliente. Al contrario, las llamadas realizadas entre usuarios dentro de la empresa normalmente no necesitan el marcado de ningún número especial o el uso de una línea externa troncal. Esto se debe a que la PBX enruta o conmuta las llamadas internas entre teléfonos que están conectados físicamente a dicha PBX.

En la figura 1-8 se puede apreciar un esquema de ejemplo de conexión de varias PBX pertenecientes a la Universidad de Almería.

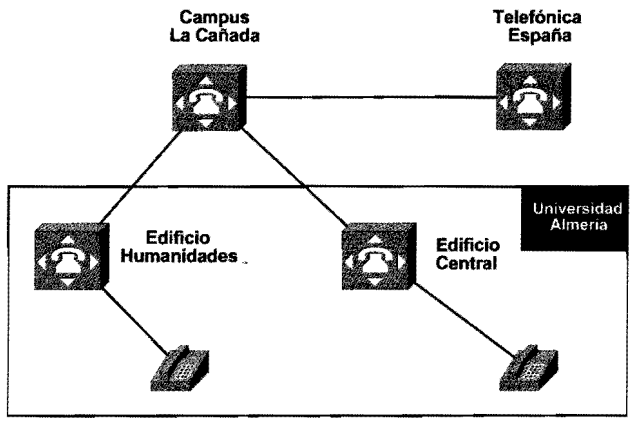


Figura 1-8. Esquemas de interconexión de centralitas

4.2 SISTEMAS COMERCIALES

Actualmente existe una gran diversidad de modelos de centralitas: centralitas con mayor o menor número de extensiones para pequeñas o grandes empresas, de más o menos prestaciones, con mayor o menor funcionalidad, totalmente analógicas, híbridas o completamente IP.

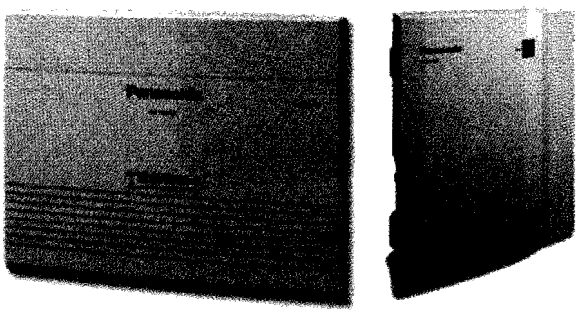


Figura 1-9. Centralitas tradicionales

En general, la mayoría de centralitas comerciales ofrece una serie de funciones muy importantes como la recepción de llamadas sin necesidad de comunicarse con la operadora, es decir, los llamantes pueden seleccionar el destino con el que desean hablar tecleando el número que tiene asignado. Un ejemplo sería *"Pulse uno para departamento de ventas, dos para departamentos de marketing..."*. También se integran funciones de buzón de voz en caso de que el destinatario no se encuentre disponible para responder a la llamada, mensajes en espera personalizados para cada una de las extensiones, desvío de llamadas, etc. Otras funcionalidades muy útiles en entornos empresariales son: conferencias, grupos de extensiones, restricción de llamadas, etc. Los grupos de extensiones permiten definir un conjunto de extensiones para cumplir entre todas una función específica, por ejemplo dar un servicio de atención al cliente. La restricción de llamadas es también fundamental en el entorno empresarial, la cual permite programar qué destinos para llamadas salientes están prohibidos o qué destinos están permitidos.

Las centralitas híbridas combinan las prestaciones de una central telefónica con la tecnología IP. A nivel empresarial esta integración con la tecnología IP ofrece grandes ventajas: los recursos humanos de la empresa pueden estar dispersos geográficamente manteniendo los recursos telefónicos centralizados, además de que aquellas empresas que desean utilizar su cableado de red para conectar teléfonos en lugares donde no siempre hay conectado un terminal telefónico, o bien trasladarse de un punto a otro de la red junto con su terminal telefónico (con todas sus prestaciones asociadas), sin tener que volver a configurar el terminal, resulta muy práctico.

En la figura 1-10 puede ver una centralita modelo "Panasonic TDA15":

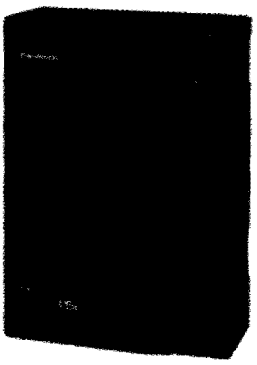


Figura 1-10. Centralita híbrida (Panasonic TDA15)

Otros fabricantes que se dedican a la comercialización son Alcatel, Ericsson, Avaya, Siemens, etc. Las prestaciones entre uno y otro fabricante son muy similares

CAPÍTULO 2

VOIP - LA NUEVA REVOLUCIÓN

Alfredo Alcaide García y Raúl Baños Navarro

1 Introducción a la VoIP

En la década de los 90, un grupo de personas perteneciente al entorno de la investigación, tanto de instituciones educativas como empresariales, comenzaron a mostrar un cierto interés por transportar voz y video sobre redes IP, especialmente a través de intranets corporativas e Internet. Esta tecnología es conocida hoy día como VoIP y es el proceso de dividir el audio y el vídeo en pequeños fragmentos, transmitir dichos fragmentos a través de una red IP y reensamblar esos fragmentos en el destino final permitiendo de esta manera que la gente pueda comunicarse.

La idea de la VoIP no es nueva, ya que hay patentes y publicaciones de investigaciones que datan de varias décadas. La VoIP ha tomado un papel central en la autopista de la información (o Internet) para que la red pueda interconectar cada hogar y cada negocio a través de una red de conmutación de paquetes. Fue la posibilidad de un despliegue masivo de Internet la que volvió a reabrir el interés en la VoIP a partir de esos años.

2 Evolución

En 1995, una pequeña compañía llamada *Vacoltec* anunció el lanzamiento del primer *teléfono software para Internet*. Este software era únicamente útil para entablar una comunicación de PC a PC y para ello necesitaba hacer uso de diversos requisitos hardware tales como micrófono, altavoces, tarjeta de sonido y módem. Básicamente el funcionamiento de este software es igual al de hoy día, transformar la señal de voz en paquetes IP una vez comprimida. Sin embargo, esta alternativa a la comunicación telefónica tradicional fue comercialmente un fracaso ya que las conexiones a Internet que se disponían ofrecían un ancho de banda muy escaso.

Durante los años siguientes, la tecnología asociada a las redes de datos y las comunicaciones continuó mejorando, para ser en 1998 cuando se dieron definitivamente los primeros pasos desde un punto de vista comercial. En este año diversas compañías lanzaron al mercado adaptadores que permitían hacer uso de los teléfonos tradicionales en un entorno VoIP. Ello facilitó el acercamiento a los clientes a la hora de poder hacer uso de la tecnología VoIP, por lo que algunas empresas importantes se lanzaron al mercado ofreciendo productos y servicios relacionados con esta tecnología. Durante el año 1998 la tecnología VoIP alcanzaba ya el 1% del tráfico total de voz: su carrera había comenzado.

En 1999, compañías dedicadas a las redes de datos tales como Cisco crearon las primeras plataformas destinadas a empresas capaces de tratar con tráfico VoIP. Esto supuso un nuevo impulso a la VoIP ya que comenzó a implantarse en muchas empresas. La consecuencia directa fue que la VoIP alcanzara en el año 2000 más del 3% del tráfico total de voz.

Las redes de datos siguieron mejorando en años venideros, y alrededor del año 2005 ya era fácil para cualquier persona de países desarrollados conseguir una conexión a Internet que cumpliera los requisitos mínimos para ofrecer una buena calidad de voz y una comunicación fiable a través de VoIP, reduciendo al mínimo las posibles interrupciones que se pudieran producir durante la conversación.

Esto supuso otro gran impulso a la VoIP y provocó que a día de hoy existan muchas soluciones que hacen uso de esta tecnología. Un ejemplo claro es Asterisk, una centralita telefónica de software libre que se distribuye bajo licencia GPL. Este producto, soportado comercialmente por Digium, se ha convertido en pocos años en una de las soluciones IP más extendidas en diversos ámbitos, como el empresarial o el educativo. Otro ejemplo destacable de producto VoIP es Skype, que fue creado por dos jóvenes universitarios en el año 2003. A diferencia de Asterisk, Skype hace uso de un protocolo privado que no está basado en un estándar, lo que a largo plazo se piensa que limitará a sus usuarios. A día de hoy Skype se puede emplear en multitud de plataformas y su uso se encuentra también ampliamente extendido.

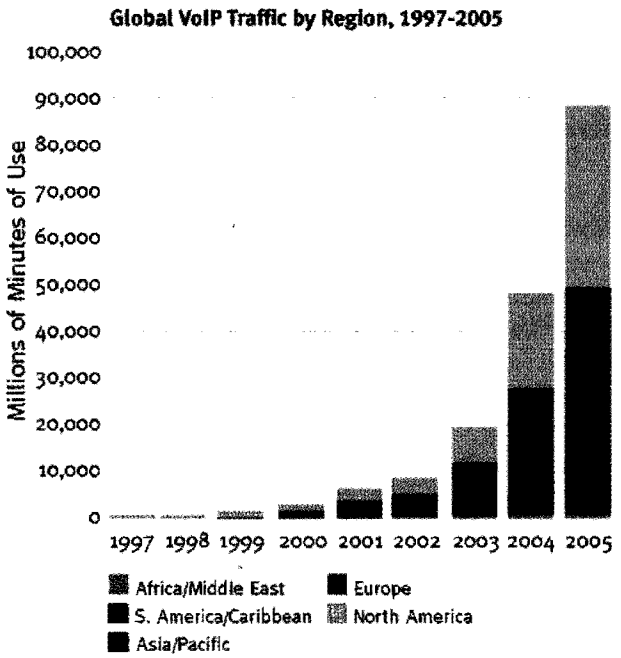


Figura 2-1. *Tráfico VoIP en todo el mundo (dividido por regiones)*

De un modo u otro, a finales del año 2008 se espera que el negocio relacionado con la VoIP llegue a la impresionante cifra de 5.000 millones de dólares. El bajo coste de las llamadas a distancia y las nuevas funcionalidades que se están implementando son sólo dos de los alicientes que están provocando esta revolución.

3 Ventajas

Aunque VoIP puede definirse de forma abreviada como una tecnología que aprovecha el protocolo TCP/IP para ofrecer conversaciones de voz, lo cierto es que es mucho más que esto. VoIP puede ser usada para reemplazar la telefonía tradicional en un entorno empresarial, en un pequeño negocio o en casa, o simplemente para añadir ventajas a un sistema de telefonía tradicional.

Consideremos hacer una llamada a una persona que se encuentra en la otra mitad del globo. ¡Lo primero en lo que pensaríamos, sin duda, sería la factura de teléfono que tendríamos que pagar! VoIP soluciona este problema y muchos otros.

VoIP tiene también algunas desventajas, sin embargo, las ventajas que puede aportar superan claramente a éstas. A continuación vamos a nombrar

algunos de los beneficios asociados al uso de VoIP y veremos cómo podría mejorar la comunicación por voz de nuestro negocio u hogar:

- **Ahorrar dinero.** Si no se usa VoIP para la comunicación por voz, entonces seguramente se esté utilizando la vieja línea de teléfono. En una línea RTC, tiempo significa dinero. Como VoIP emplea Internet como medio de transporte, el único coste que se tiene es la factura mensual de Internet a tu proveedor de servicio o ISP. Hoy día el servicio de Internet más común es una ADSL que se puede emplear de forma ilimitada y conlleva un coste fijo al mes. De esta forma, si el ADSL tiene una velocidad razonable, podrá hablar a través de VoIP con una buena calidad de llamada y el coste seguirá siendo siempre el mismo.

- **Más de dos personas.** En una línea de teléfono corriente, únicamente dos personas pueden hablar al mismo tiempo. Con VoIP, puedes configurar una conferencia que permite a un grupo de personas comunicarse en tiempo real. VoIP comprime los paquetes durante la transmisión, algo que provoca que se pueda transmitir una cantidad mayor de datos. Como resultado, se pueden establecer más llamadas a través de una única línea de acceso.

- **Hardware y software baratos.** Si eres un usuario de Internet que está deseando usar VoIP para comunicarse por voz, el único hardware adicional que necesitarás además de tu ordenador y tu conexión a Internet será una tarjeta de sonido, unos altavoces y un micrófono. Todo este material es a día de hoy bastante barato. Existen diferentes paquetes software descargables de Internet que emplean VoIP y que sirven para establecer comunicaciones por voz. Algunos ejemplos son aplicaciones tan conocidas como Skype o Net2Phone. Lo que debemos tener en cuenta es que para comenzar a emplear VoIP no necesitaremos un teléfono con todo el equipamiento asociado a éste, algo que podría resultar algo más caro. Además en la mayoría de los casos no será necesario hacer nuevas instalaciones de cableado telefónico, ya que VoIP se integra con la red de datos existente en la gran mayoría de empresas y hogares.

- **Prestaciones abundantes, interesantes y útiles.** Usar VoIP también significa beneficiarse de sus prestaciones abundantes, que pueden hacer la experiencia de emplear VoIP mucho más rica y sofisticada, tanto en tu hogar como en tu trabajo. En general, te encontrarás mejor equipado para la gestión de llamadas. Podrás, por ejemplo, hacer llamadas en cualquier lugar del mundo a cualquier destino del mundo únicamente empleando tu cuenta VoIP. De esta forma, la VoIP pasa a ser un servicio tan portable como el e-mail, es decir, no limita la movilidad del abonado. Otras prestaciones que ofrece VoIP son el reconocimiento de llamada,

posibilidad de crear números virtuales o el contestador automático, por poner algunos ejemplos.

- **Más que voz.** Al estar basada en una red de paquetes, VoIP puede manejar también otros tipos de datos además de la voz: podríamos transmitir imágenes, video o texto a la vez que la voz. De esta forma, puedes hablar con alguien a la vez que le envías archivos o incluso a la vez que te está viendo a través de una webcam.

- **Uso más eficiente del ancho de banda.** Se sabe que el 50% de una conversación de voz es silencio. VoIP rellena estos espacios de silencio con datos de forma que el ancho de banda de los canales de comunicación de datos no sean desaprovechados. La compresión y la posibilidad de eliminar la redundancia cuando se transmite voz serán también factores que elevarán la eficiencia del uso del ancho de banda de la conexión.

- **Esquema de red flexible.** La red que encontramos bajo VoIP no necesita tener un esquema o topología en concreto. Esto hace posible que una organización pueda hacer uso de la potencia de las tecnologías que elijan, como ATM, SONET o Ethernet.

  Cuando empleamos VoIP, la complejidad de la red inherente en las conexiones RTC es eliminada, creándose una infraestructura flexible que puede soportar muchos tipos de comunicación. El sistema estará más estandarizado, requerirá menos equipamiento y su tolerancia a fallos será mayor.

- **Teletrabajo.** Si trabajas en una organización que emplea una intranet o extranet, todavía podrás acceder a tu oficina desde casa a través de VoIP. Puedes convertir tu hogar en una parte de la oficina y usar remotamente la voz, el fax o los servicios de datos de tu lugar de trabajo a través de la intranet de la oficina. La naturaleza portátil de la tecnología VoIP está provocando que gane popularidad, ya que proporciona una gran cantidad de comodidades impensables hace unos años. La portabilidad tanto de hardware como de servicios se está convirtiendo cada día en algo más normal, y en ese contexto VoIP encaja perfectamente.

- **Fax sobre IP.** Los problemas de los servicios de fax sobre RTC son el alto coste que conllevan para largas distancias, la atenuación de la calidad en las señales analógicas y la incompatibilidad entre algunas máquinas cuando se comunican. La transmisión de fax en tiempo real sobre VoIP simplemente utiliza una interfaz de fax para convertir los datos en paquetes y asegura que éstos serán entregados completamente y de forma segura.

Otra ventaja de este sistema es que ni siquiera necesitaremos una máquina fax para enviar y recibir fax.

- **Desarrollo de software más productivo.** VoIP puede combinar diferentes tipos de datos, enrutándolos y señalizándolos de forma muy flexible y robusta. Como resultado de esto, los desarrolladores de aplicaciones de red encontrarán más fácil crear y desplegar aplicaciones que realicen comunicaciones de datos empleando VoIP. Además, la posibilidad de implementar VoIP en navegadores web y servidores proporciona un filón tanto productivo como competitivo a esta tecnología.

4 Arquitectura

Uno de los beneficios que aporta la VoIP es que la arquitectura, desde el punto de vista de su distribución, puede ser centralizada o distribuida. El enfoque centralizado es criticado porque al estar todo localizado en un mismo punto las futuras innovaciones tecnológicas se verán entorpecidas. Por otro lado la arquitectura distribuida es más compleja que la arquitectura centralizada. Sea partidario de un enfoque u otro, lo que la VoIP nos permite es una gran flexibilidad.

Sin entrar en debates sobre un enfoque u otro, en la figura 2-2 se muestra, a modo de ejemplo, un entorno VoIP.

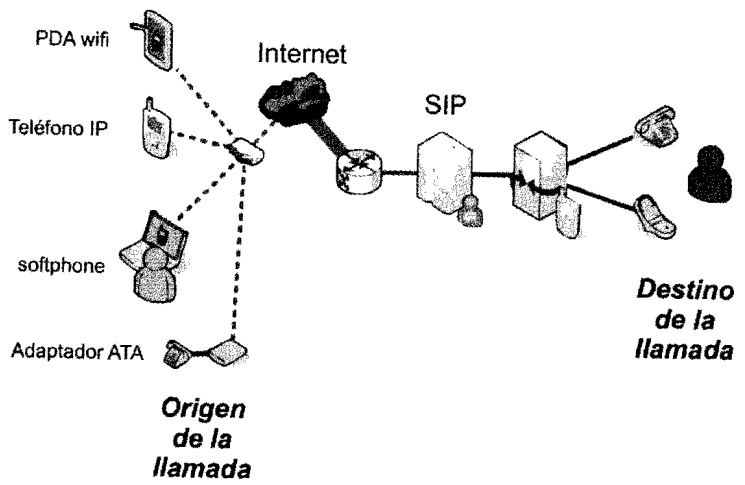


Figura 2-2. Arquitectura

En la figura 2-2 se muestra una arquitectura de VoIP muy general, donde podemos ver los distintos dispositivos que la compone:

- **TelefonoIP.** Es un teléfono similar a un teléfono tradicional con la diferencia que está adaptado para ser utilizado en entornos IP.
- **Softphone.** Es un teléfono similar al del punto anterior con la peculiaridad de que este es software.
- **Adaptador ATA.** Es un adaptador que permite conectar un teléfono convencional a una red IP.
- **SIP.** Es un protocolo usado por los proveedores de VoIP encargado de, entre otras funciones, iniciar y finalizar las llamadas VoIP.
- **B2BUA.** Es una entidad intermediaria encargada de procesar las comunicaciones VoIP y retransmitirlas a su destino.

A continuación se mostrará de manera más detallada los elementos más significativos de un entorno de VoIP.

4.1 TELÉFONOS IP

Los teléfonos IP son una parte importante de la arquitectura de la VoIP. Aunque se viene trabajando duro desde hace tiempo, hoy día se está consiguiendo avanzar a pasos agigantados en el desarrollo de estos terminales, y ya se empiezan a observar las posibilidades que estas unidades pueden llegar a brindar a nivel funcional. Pronto serán accesibles a una gran cantidad de usuarios equipos como videoteléfonos IP, soluciones de movilidad basadas en redes IP, sistemas multimedia "todo-en-uno" completamente flexibles o unidades capaces de videoconferencia con muchos usuarios. De hecho, la revolución no se producirá debido a la nueva forma que tenemos de conectar los teléfonos cuando se emplea VoIP, sino a la posibilidad que darán estos teléfonos de comunicarse de la forma que exactamente se desea.

En la actualidad, los teléfonos IP son, en su gran mayoría, muy similares al resto de teléfonos tradicionales. Sin embargo, si nos fijamos detenidamente en su aspecto exterior se puede apreciar que existen ciertas diferencias:

- Disponen de al menos un puerto de conexión RJ-45 en lugar del tradicional RJ-11.

- Suelen disponer de pantalla para mostrar información relevante.
- Incorporan varios botones programables que pueden usarse para diferentes funcionalidades.
- Conector de auriculares.

El puerto RJ-45 de los teléfonos IP es un puerto Ethernet con el cual se conectan dichos teléfonos a la red. A través de este puerto, éstos se comunican con cualquier otro dispositivo basado en IP que se encuentre en la red, como puede ser un *proxy* o enrutador para VozIP, otro teléfono IP, una puerta de enlace a la RTC (para realizar llamadas hacia la red telefónica tradicional) o el *router* que sabemos que establecerá la conexión con cualesquiera otros elementos IP de la red.

Ciertos modelos de teléfonos IP tienen varios conectores RJ-45 en lugar de uno. En estas ocasiones el teléfono tendrá un *switch* o incluso un *router* integrado, que permite conectar dispositivos como impresoras de red, ordenadores o incluso otros teléfonos IP.

Además de lo comentado anteriormente, algunos teléfonos tienen implementada la posibilidad de ser alimentados eléctricamente a través de la red de datos, es decir, la LAN proporcionará al teléfono la electricidad que necesita para funcionar. Esta tecnología se conoce con el nombre de *Power over Ethernet*[1] (PoE).

Los teléfonos IP se pueden encontrar a precios muy asequibles y cada vez bajan más, debido principalmente a que existen una gran cantidad de fabricantes, distribuidores y modelos. Dependiendo de sus características y posibilidades podríamos clasificar los teléfonos IP en tres categorías:

- **Gama baja**. Constituyen la mayoría y son aquellos que recuerdan más a los teléfonos tradicionales. Éstos proporcionarán un buen servicio para realizar llamadas por VoIP a otros terminales de la red o a través de *proxys*, aunque disponen de pocas funcionalidades extra. Entre las mismas cabe destacar el soporte para varios idiomas o la personalización de tonos de llamada y melodías.

---

[1] *Power over Ehternet*: alimentación eléctrica a través de la Red.

Figura 2-3. Teléfono de gama baja marca Pheenet

- **Gama media**. Son muy parecidos a los teléfonos IP básicos, pero añaden nuevas funcionalidades que los anteriores no poseen. Además suelen tener una pantalla más avanzada y grande, así como más conexiones hardware de las que tienen los básicos. Por ejemplo, es habitual que dispongan de pantalla retroiluminada, capacidades de VLAN e incluso la posibilidad de registrar varias líneas con operadores IP diferentes.

Figura 2-4. Teléfono IP de gama media marca Linksys

- **Gama Avanzada**. Estos teléfonos suelen incluir pantallas a color y muchas otras funciones extras como la posibilidad de configurar el acceso a un servidor LDAP de una organización o acceso Web a través de la pantalla del teléfono.

Figura 2-5. Teléfono IP Nortel IP Phone 2007, uno de los últimos modelos de este fabricante en salir al mercado, que incluye entre otras muchas cosas conectividad USB para ratón y teclado, puerto RJ-8 para conectar altavoces y pantalla táctil de 5,7"

Las funciones propias de los sistemas telefónicos tradicionales (rellamada, llamada en espera o llamada a tres por poner algunos ejemplos), además de muchas otras que no podremos encontrar en éstos, se encuentran implementadas en VoIP de dos formas distintas: como funciones del propio teléfono IP o a través de la red IP a la que está conectada dicho teléfono, por ejemplo con un servidor o un controlador telefónico. Mientras que con la telefonía tradicional todo este tipo de funciones sólo pueden ser accesibles mediante la compra de licencias adicionales, algo que supone un cambio radical con respecto al anterior enfoque y un enorme ahorro para los usuarios de todo este tipo de servicios.

4.2 GATEWAYS Y ADAPTADORES ANALÓGICOS

Un adaptador de teléfono analógico (normalmente conocido como *Analog Telephone Adaptor* o ATA) se puede describir brevemente como un dispositivo que convierte señales empleadas en las comunicaciones analógicas a un protocolo de VozIP. En concreto, estos dispositivos se emplean para convertir una señal digital (ya sea IP o propietaria) a una señal analógica (o viceversa) que pueda ser conectada a teléfonos o faxes tradicionales.

Existen diferentes versiones en función de que desee conectar un puerto FXO o un puerto FXS. Para más información véase el *Capítulo 1. La telefonía tradicional.*

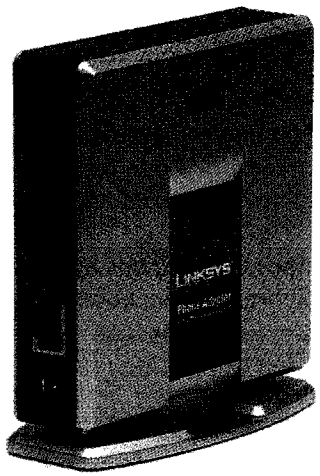


Figura 2-6. Adaptador telefónico para analógico y SIP (Linksys PAP2). Obsérvese que dispone de 2 puertos analógicos (puertos FXS) para conectar dos teléfonos y un puerto RJ-45 para conectividad ethernet

Estos adaptadores podrían ser descritos como *gateways*, ya que su función es justamente la de pasarela entre el mundo analógico y el IP. Sin embargo, el uso popular del término *gateway* de telefonía describiría mejor un adaptador telefónico multipuerto, generalmente con funciones de enrutamiento más complejas.

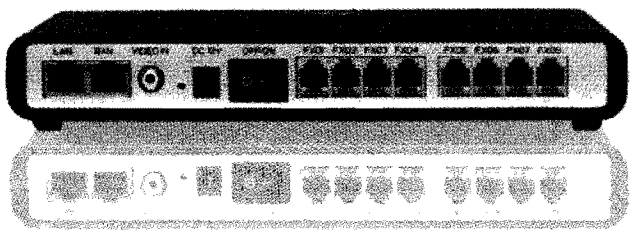


Figura 2-7. Adaptador analógico para interconexión con la RTC, marca Grandstream. Dispone de 8 puertos FXO y dos puertos RJ45 (incorpora router)

Aunque con estos adaptadores telefónicos no se puede disfrutar de todas las funciones y ventajas que ofrece la telefonía IP, éstos seguirán existiendo mientras exista la necesidad de conectar estándares incompatibles y viejos dispositivos a nuevas redes. Eventualmente, nuestra dependencia hacia esos viejos

dispositivos desaparecerá, como lo hizo, por ejemplo, nuestra dependencia hacia un dispositivo como el módem en su momento.

4.3 DISPOSITIVOS GSM/UMTS

Los teléfonos móviles son dispositivos electrónicos de pequeño tamaño empleados para realizar comunicaciones de voz o datos a través de una conexión a una estación base que pertenecerá a una determinada red de telefonía móvil. Éstos han supuesto una auténtica revolución en nuestra manera de comunicarnos.

Existen muchos tipos de teléfonos móviles, desde los más básicos hasta los teléfonos que ofrecen mayores funcionalidades, como los *smartphones* (teléfonos inteligentes), *musicphones* (teléfonos con posibilidad de reproducir música) o *cameraphones* (teléfonos con cámara integrada).

Desde el punto de vista de la VozIP se pueden encontrar dispositivos que integran ambas tecnologías, por ejemplo, SIP y GSM. Estos dispositivos permiten una mayor integración que la tecnología analógica tradicional ya que no dependen de conversiones intermedias a analógico y, por tanto, permiten enviar toda la señalización existente entre ambas redes, de forma transparente y fiable.

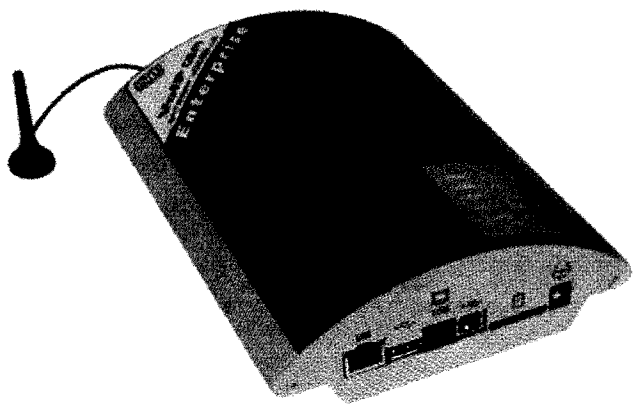


Figura 2-8. Gateway GSM-IP para interconexión directa de redes GSM e IP. Fabricante 2N, modelo VoiceBlue Enterprise (4 líneas GSM, Proxy SIP y H323)

4.4 SOFTPHONES

Los *softphones* son teléfonos implementados por software. Éstos proporcionarán a un dispositivo que no sea un teléfono, como un ordenador o una PDA, las funcionalidades de un teléfono VoIP. Para que esto sea posible, no es necesario que el dispositivo en cuestión sea muy potente. Simplemente se necesita un equipo de audio adecuado y alguna forma de conectarse a una red TCP/IP.

Se pueden encontrar modelos que funcionan bajo diferentes protocolos, aunque el más usado es el SIP. Entre ellos, el más conocido y usado es el X-lite (ver *Apéndice IV. Clientes VoIP*), aunque también existen muchos otros que presentan buenas funcionalidades. Existe una larga lista en la página wiki del proyecto Asterisk http://www.voip-info.org[2].

El concepto de teléfono está hoy día en constante evolución, lo que hace difícil en ocasiones diferenciar lo que es un *softphone* de lo que no lo es. La comunicación por VoIP está presente en programas de mensajería instantánea por poner un ejemplo, pero sólo el tiempo dirá si éstos pueden llegar a ser considerados *softphones*. A pesar de ello, cuando nosotros hablemos de un *softphone*, nos referiremos a un software ejecutable en ordenadores u otros dispositivos que tenga el aspecto de un teléfono, se use como cualquier teléfono y cuya funcionalidad principal sea la de realizar llamadas telefónicas.

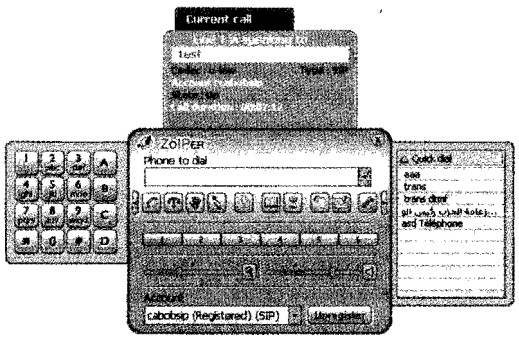


Figura 2-9. Softphone Zoiper: puede trabajar con el protocolo SIP o IAX, y dispone de una versión gratuita y otra comercial

[2] http://www.voip-info.org/wiki-VOIP+Phones

Figura 2-10. X-lite de CounterPath, uno de los softphones más empleados hoy día y se encuentra disponible para varios sistemas operativos

Los teléfonos VoIP hardware son una combinación entre un *softphone* y un hardware dedicado. Aunque la capacidad de computación de un ordenador o una PDA exceda de forma exultante a la de un teléfono IP, siempre debemos recordar que estos últimos son dispositivos especialmente creados para realizar llamadas y transportar voz.

La principal ventaja de un *softphone* sobre un teléfono VoIP hardware es el coste. En muchas empresas se debe instalar como mínimo en cada escritorio un ordenador y un teléfono. Si pudiéramos quitar el teléfono de todos esos escritorios se produciría un claro ahorro. Además, hay una gran cantidad de *softphones* y la mayoría de sistemas operativos ya vienen con alguno instalado por defecto.

Los *softphones* tienen por delante un largo camino hasta que puedan ser aceptados por la mayoría de usuarios. A la costumbre de usar un teléfono tradicional, se une el grave inconveniente de que al tener el teléfono en el ordenador se añaden complicaciones extras. Por ejemplo, si tenemos una caída de tensión o un corte eléctrico, el reinicio del PC siempre es mucho más problemático y lento que un softphone. A pesar de esto, siempre es una opción que se debe tener en cuenta al realizar una instalación, prestando sobre todo mucha atención al entorno en el que se vaya a emplear el teléfono y a que los usuarios no vayan a tener ningún problema a la hora de emplear un *softphone*.

4.5 PROXYS Y ENRUTADORES

Dentro de la arquitectura de VozIP es necesario el uso de ciertos elementos que permitan ordenar el tráfico telefónico y a la vez poner en contacto a los diferentes usuarios de las redes implicadas.

Tal y como trabajan los *routers* con los datos en general, recibiendo y enviando peticiones desde y hacia otras máquinas, los diferentes protocolos IP necesitan igualmente que alguien o algo encamine sus peticiones hacia los usuarios finales, a fin de establecer una conversación. Esta tarea la realizan los *proxys* o enrutadores, encargándose de rutar la señalización hacia los sitios adecuados en función de las indicaciones pertinentes que cada protocolo implementa. En la figura 2-11 puede verse un esquema de enrutado en un entorno SIP.

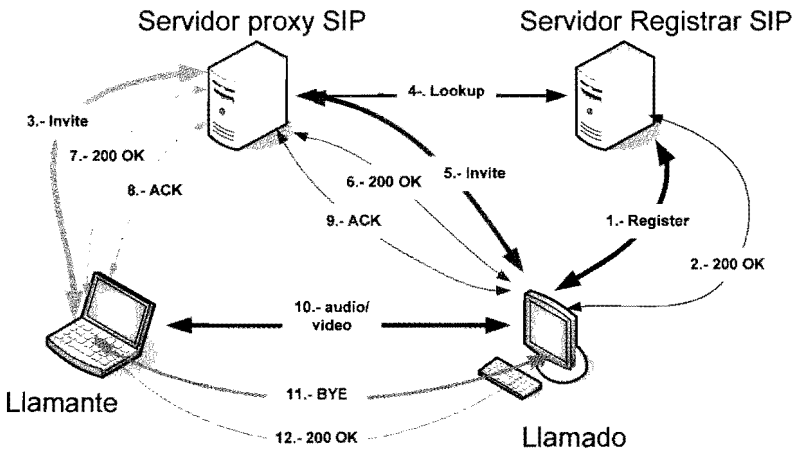


Figura 2-11. Esquema de enrutado en un entorno SIP

Existen multitud de implementaciones para servidores *proxy*, aunque entre las más usadas se encuentran las distribuciones de código abierto SER, Kamailio y OpenSIPS[3]. Estos paquetes son capaces de gestionar gran cantidad de peticiones por segundo, haciendo un gran trabajo de enrutado gracias a un diseño óptimo en su código base.

---

[3] http://www.iptel.org/ser http://www.kamailio.org/ http://www.opensips.org

5 Señalización y audio

5.1 PROTOCOLOS DE COMUNICACIÓN

La realización de una llamada entre dos teléfonos cualesquiera implica la utilización de diversos equipos electrónicos, los cuales deben comunicarse entre sí. Para poder garantizar que la comunicación entre los equipos se realiza adecuadamente, son necesarias diversas reglas y/o normas. Estas reglas y/o normas de las que se habla es lo que se conoce como protocolo de señalización.

En las redes analógicas o redes de conmutación de circuitos antes de que ambos extremos puedan comunicarse, se produce la reserva de recursos necesarios para que la comunicación tenga éxito. Si por cualquier circunstancia no puede llevarse a cabo esta reserva de camino entre ambos extremos se informa al emisor de este hecho. A la acción de "reservar un camino de recursos entre ambos extremos" es lo que se le conoce como **señalización**.

En la telefonía tradicional los protocolos de señalización se pueden clasificar en dos categorías:

- **Channel Associated Singnalling (CAS).** Tanto la información de señalización como los datos (voz) se transmiten por los mismos canales. Protocolos de señalización pertenecientes a esta categoría: G.732, E&M, etc.

- **Common Channel Signalling (CCS)** Aquí la información correspondiente a la señalización se transmite en un canal independiente al de los datos (voz). Protocolos de señalización pertenecientes a esta categoría es, por ejemplo, SS7.

En conmutación de paquetes los protocolos de señalización realizan acciones muy similares a los protocolos de señalización en conmutación de circuitos además de cuidar de que se cumplan ciertas garantías de calidad. Los protocolos de señalización más utilizados en conmutación de paquetes son: **SIP** y **H323**.

5.1.1 Session Initiation Protocool (SIP)

El protocolo SIP es un protocolo de señalización a nivel de aplicación encargado de la **iniciación, modificación y terminación de sesiones multimedia**, las cuales se llevan a cabo de manera interactiva. Por sesiones multimedia se refiere a aplicaciones de mensajería instantánea, aplicaciones de video, de audio, conferencias y aplicaciones similares.

SIP se definió en el RFC 2543 en marzo de 1999 por el grupo de trabajo MMSC perteneciente a IEFT. En junio de 2002, el IETF publicó una nueva revisión de SIP con el **RFC 3261**.

El protocolo SIP posee cuatro características que lo hacen muy recomendable para cumplir esta función:

Tabla 2-1. Características del protocolo SIP

Característica	Descripción
Localización del usuario	SIP posee la capacidad de poder conocer en todo momento la localización de los usuarios. De esta manera no importa en qué lugar se encuentre un determinado usuario. En definitiva la movilidad de los usuarios no se ve limitada.
Negociación de los parámetros	Posibilidad de negociar los parámetros necesarios para la comunicación: puertos para el tráfico SIP así como el tráfico Media, direcciones IP para el tráfico Media, codec, etc.
Disponibilidad del usuario	SIP permite determinar si un determinado usuario está disponible o no para establecer una comunicación.
Gestión de la comunicación	Permite la modificación, transferencia, finalización de la sesión activa. Además informa del estado de la comunicación que se encuentra en progreso.

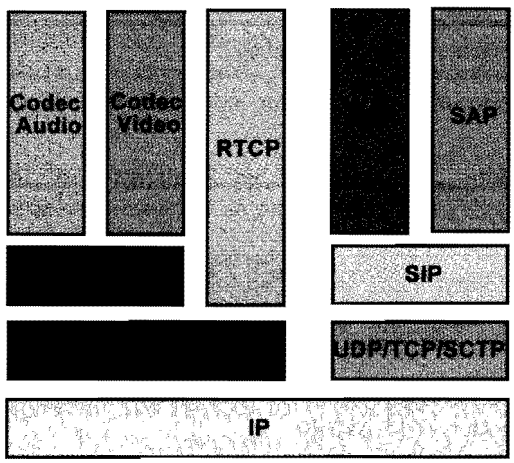


Figura 2-12. Arquitectura de protocolos SIP

El protocolo SIP es una parte de una arquitectura multimedia, ya que la única finalidad es la de gestionar las sesiones multimedia: iniciarlas, modificarlas, finalizarlas, etc. Sin embargo, se integra perfectamente con otros protocolos como **RVSP, RTP o RTSP.** Gracias al protocolo **SDP** se puede formar una completa arquitectura multimedia.

Conceptos básicos

El protocolo es similar a HTTP por la forma en que funciona (protocolo basado en texto) y es similar a SMTP en la forma en la que se especifican las direcciones SIP.

Las direcciones SIP identifican a un usuario de un determinado dominio. A estas direcciones SIP habitualmente se les llama **URI (Uniform Resource Identifier)**. Una URI se puede especificar de las siguientes maneras:

```
sip:usuario@dominio[:port]
sip:usuario@direcciónIP[:port]
```

El dominio representa el nombre del **proxy SIP** que conoce la dirección IP del terminal identificado por el usuario de dicho dominio. El puerto por defecto para SIP es 5060, aunque es posible especificar otros adicionales si es necesario.

En la tabla 2-2 se pueden ver algunos ejemplos de direcciones SIP.

Tabla 2-2. Ejemplos de direcciones SIP

Descripción	Dirección SIP
Usuario "200" perteneciente al dominio "ual.es"	200@ual.es
Usuario "200" perteneciente al dominio con dirección IP 192.168.1.120	200@192.168.1.120

Nota: *en la nomenclatura usuario@direcciónIP, la dirección puede referirse a la IP del usuario, en un momento determinado, o a su dominio.*

Es por tanto posible hacer uso de una dirección IP si no disponemos de un dominio registrado para este propósito.

Supongamos el escenario de la Universidad de Almería (ual.es). Cada uno de los trabajadores de la UAL dispone de una dirección SIP que lo identifica de manera unívoca en el dominio "ual.es" (véase la figura 2-13).

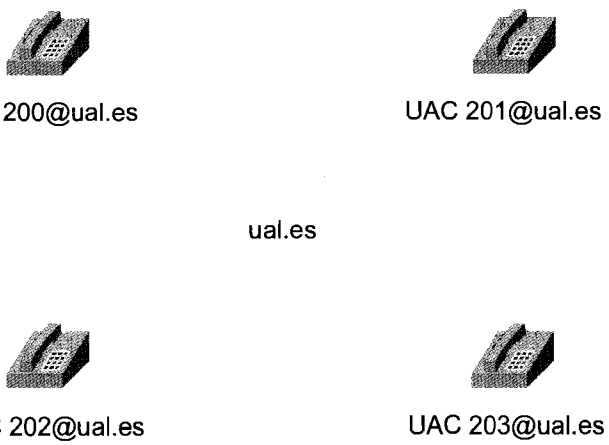


Figura 2-13. Ejemplo de escenario SIP

Si por ejemplo el usuario 200@ual.es desea comunicarse con el usuario 201@ual.es, el usuario 200 únicamente tendría que marcar en su teléfono el número 201. Al marcar el número 201, comienza la señalización SIP entre el terminal 200 y el 201 iniciando así una comunicación SIP entre ambos terminales para posteriormente establecerse una conexión *Media* entre ambos (RTP).

Este es un escenario en el que no es posible establecer la señalización SIP ya que el teléfono "200@ual.es" realmente desconoce la dirección IP en la que se encuentra el teléfono 201. Es por ello que para que la señalización SIP pueda llevarse a cabo, es necesario hacer uso de varios elementos intermediarios.

Para una comunicación SIP es necesaria la intervención de varios elementos, donde cada uno desempeña su papel. Los elementos de la comunicación son:

- **Los agentes de usuario (User agent),** o de manera abreviada **UA**, manejan la **señalización SIP**. Se pueden dividir en dos categorías:
  - **User agent client (UAC).** Es un elemento que realiza peticiones SIP y acepta respuestas SIP provenientes de UAS. Un ejemplo de UAC es un teléfono VoIP ya que realiza peticiones SIP.
  - **User agent server (UAS).** Es el elemento encargado de aceptar las peticiones SIP realizadas por el UAC y enviar a este la respuesta conveniente. Un teléfono VoIP también es un ejemplo de UAS, ya que acepta las peticiones de inicio de comunicación enviadas por

otro teléfono (UAC). Un servidor SIP o proxy también es un UAS, lo veremos a continuación.

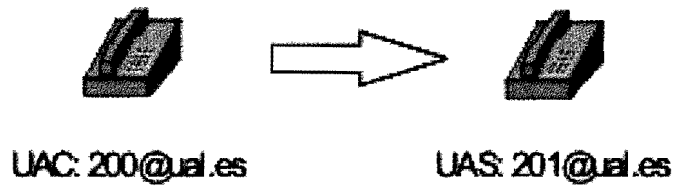


Figura 2-14. Ejemplo de llamada SIP (200 llama al 201)

- **Los intermediarios** necesarios para que la comunicación entre dos UA sea posible:
  - **Servidor Proxy**. Es el elemento encargado de reenviar las peticiones SIP provenientes de un UAC al UAS destino que corresponda, así como de encaminar las respuestas del UAS destino al UAC origen. Podemos hacer una similitud con el encaminamiento que realizan los router con los paquetes a nivel IP, es decir, sería como el encargado de enrutar los paquetes SIP. Para rutar, lo que hace es una traducción de la dirección de destino dada de la forma *usuario@dominio* a la forma *usuario@direcciónIP* donde dirección IP es la dirección IP que tiene en ese momento el teléfono de destino.

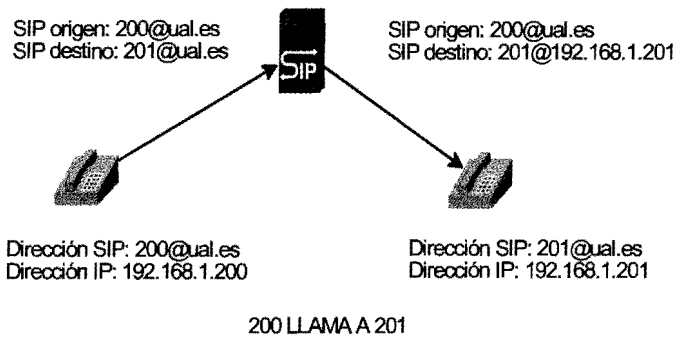


Figura 2-15. Ejemplo de llamada SIP con intermediarios (200 llama al 201)

En la figura 2-15 podemos ver la función más importante de un proxy SIP, la de enrutar los mensajes SIP estableciendo así la señalización SIP pertinente. El escenario mostrado en la figura no es

todavía funcional ya que el proxy SIP necesita conocer la dirección IP física asociada con cada usuario. En el ejemplo, el proxy SIP tiene que saber que el teléfono identificado como 201 de dominio "ual.es" tiene asignada la dirección IP real 192.168.1.201.

De esta nueva necesidad se encarga el servidor de registro y localización:

- **Registrar-Location server.** Acepta las peticiones de registro de los UAC, guardando toda la información referente a la localización física del UAC, para que si posteriormente llega una petición con destino el UAC, sea posible localizarlo (sea posible traducir su dirección a la forma *usuario@direcciónIP* donde dirección se refiere a la IP del usuario).

Continuando con la figura 2-15, para que 200 llame al usuario 201 es necesario que previamente ambos teléfonos se hayan registrado en servidor de registro. Esto es necesariamente así porque el proxy SIP necesita conocer las dirección IP del teléfono 201 para enviarle la petición de inicio de conversación y del mismo modo necesita la dirección IP del teléfono 200 para que pueda rutarle las respuestas SIP generadas por el teléfono 201.

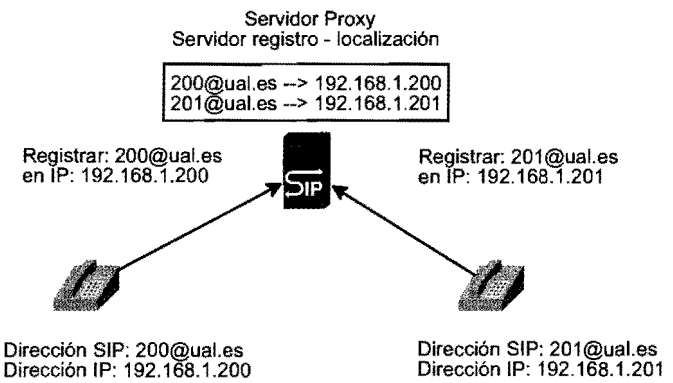


Figura 2-16. Proceso de registro

Una vez que los teléfonos se han registrado en el servidor de registro, estos pueden entonces realizar y recibir llamadas entre sí ya que el proxy SIP conoce sus direcciones IP físicas/reales (mediante consultas). Habitualmente, el proxy SIP y el servidor de "registro-localización" se encuentran juntos en el mismo software por lo general.

Ahora sí se puede afirmar que la llamada realizada en la figura 2-16 va a tener éxito.

- **Redirect Server.** Su funcionamiento es similar al servidor proxy anterior, con la diferencia que cuando este resuelve la dirección, esto es, realiza la traducción, informa al UAC que realizó la petición SIP para que sea este mismo el que la envíe hacia el UA destino. Un servidor de redirección actúa realmente como un UAS.

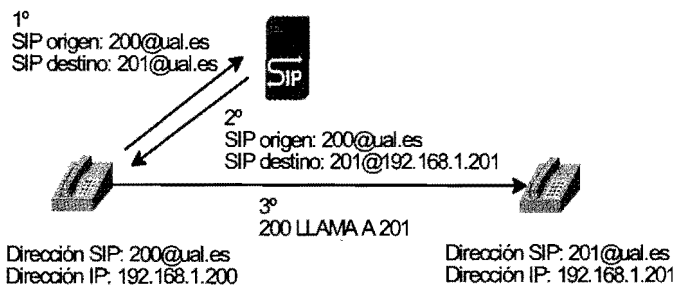


Figura 2-17. Registro completado

En la figura 2-17 ambos teléfonos se han registrado previamente en el proxy SIP. A partir de ahora cuando se hable de servidor proxy SIP se refiere a un servidor que contiene conjuntamente el proxy SIP y el servidor de registro-localización.

- **Back-to-back user agent (B2BUA).** Es una entidad intermediaria que procesa peticiones SIP entrantes comportándose como un UAS, y responde a estas actuando como un UAC regenerando por completo la petición SIP entrante en una nueva petición SIP que va a ser enviada.

Peticiones SIP

Hasta el momento se ha descrito a grandes rasgos cómo se lleva a cabo una comunicación SIP y los requisitos o elementos que se necesitan para ella. Sin embargo no se han indicado qué mensajes son intercambiados entre los distintos elementos durante una comunicación SIP. En primera instancia se verán los mensajes SIP para posteriormente tratar con ellos en diversos escenarios ejemplo.

En la tabla 2-3 se muestran las distintas peticiones SIP.

Tabla 2-3. Peticiones SIP

Petición SIP	Descripción
INVITE	Es la petición SIP que se envía a un usuario cuando queremos establecer con él una comunicación, una llamada.
ACK	Esta petición es enviada por el usuario origen que envió la petición INVITE para hacer saber al usuario destino que su respuesta 200 OK ha sido recibida. Es el momento en que ambos pueden empezar a enviar tráfico Media.
BYE	Para finalizar la conexión, la comunicación entre los dos usuarios establecida anteriormente con INVITE.
CANCEL	Se utiliza para cancelar una petición, por ejemplo INVITE, que se encuentra en progreso. Por ejemplo si el teléfono destino está sonando pero aún no ha sido descolgado y el teléfono origen cuelga, se envía un CANCEL a diferencia de un BYE que se enviaría si el teléfono destino hubiera sido descolgado previamente y por tanto la comunicación establecida unos instantes.
OPTIONS	Un UA puede enviar peticiones OPTIONS a un UAS para solicitar cierta información sobre este.
REGISTER	Un UAC envía peticiones REGISTER a un servidor de registro-localización para informar de la posición actual en la que se encuentra en un momento determinado. Esto hace posible que el UAC pueda ser localizado haciendo uso de su misma dirección user@dominio sin importar donde el UAC se encuentre físicamente.

Respuestas SIP

Cada petición SIP lleva asociada una respuesta (la que corresponda) enumerada con un código que la identifica. Estos códigos van desde el identificador 100 hasta el identificador 699, siendo además agrupadas en grupos de respuestas tales como: 1xx, 2xx, 3xx, 4xx, 5xx y 6xx:

- Las respuestas del grupo **1xx** indican el **estado temporal de la comunicación**. Estas se utilizan por ejemplo cuando se tiene en progreso el establecimiento de una comunicación mediante la petición INVITE.

- Las respuestas pertenecientes al grupo **2xx** corresponden a respuestas que informan del **éxito de una petición SIP**. Por ejemplo, cuando se establece con éxito el establecimiento de comunicación con la petición INVITE se envía una respuesta 200 OK informando al UAC origen de este hecho.

- Las respuestas que conforman el grupo **3xx** informan de que la **petición SIP ha de ser reenviada a otro UAS**. Un servidor de redirección nos enviaría una respuesta con código "302 Moved temporarily".

- Las respuestas pertenecientes al grupo **4xx** corresponden a errores en el cliente SIP.

- Las respuestas del grupo **5xx** corresponden a errores en el servidor SIP.

- Las respuestas pertenecientes al grupo **6xx** informan de errores generales.

A continuación en la tabla 2-4 se muestran las distintas respuestas SIP.

Tabla 2-4. Posibles respuestas del protocolo SIP

Tipo de respuesta	Identificador	Significado
Informan del estado provisional de la comunicación	100	Trying - Intentando
	180	Ringing - Sonando
	181	Call Being Forwarded - Llamada está siendo transferida
	182	Call Queued - Encolada
	183	Session Progress - Llamada en progreso
Informan del éxito de la comunicación	200	OK - OK
	202	Accepted - Aceptada
Informan del reenvío necesario de la petición SIP	300	Multiple Choices - Múltiples opciones
	301	Moved Permanently - Movido permanentemente
	302	Moved Temporarily - Movido temporalmente
	305	Use Proxy - Usar Proxy
	380	Alternative Service - Servicio alternativo
Informan de errores del cliente	400	Petición Bad Request - Mala petición
	401	Unauthorized - No autorizado
	402	Payment Required - Se requiere pago
	403	Forbidden - Prohibido
	404	Not Found - No encontrado
	405	Method Not Allowed - Método no permitido
	406	Not Acceptable - No es aceptable
	407	Proxy Authentication Required - Se requiere autentificación
	408	Request Timeout - Tiempo agotado para la petición
	410	Gone - Se ha marchado
	413	Request Entity Too Large - Petición demasiado grande

	414	Request URI Too Long - URI demasiado larga
	415	Unsupported Media Type - Tipo de media no soportado
	416	Unsupported URI Scheme - Esquema URI no soportado
	420	Bad Extension - Extensión incorrecta
	421	Extension Required - Se requiere extensión
	423	Interval Too Brief - Intervalo demasiado corto
	480	Temporarily Unavailable - No disponible temporalmente
	481	Call/Transaction Does Not Exist - No existe la llamada/transacción
	482	Loop Detected - Círculo vicioso detectado
	483	Too Many Hops - Demasiados Hops
	484	Address Incomplete - Dirección incompleta
	485	Ambiguous - Ambiguo
	486	Busy Here - Ocupado
	487	Request Terminated - Petición terminada
	488	Not Acceptable Here - No es aceptable aquí
	491	Request Pending - Petición pendiente
	493	Undecipherable - Indescifrable
Informan de errores del servidor	500	Server Internal Error - Error interno del servidor
	501	Not Implemented - No implementado
	502	Bad Gateway - Gateway incorrecto
	503	Service Unavailable - Servicio no disponible
	504	Server Time-Out - Tiempo agotado en el servidor
	505	Version Not Supported - Versión no soportada
	513	Message Too Large - Mensaje demasiado largo
Informan de errores generales	600	Busy Everywhere - Ocupado en todos sitios
	603	Declined - Rechazado
	604	Does Not Exist Anywhere - No existe en ningún sitio
	606	Not Acceptable - No aceptable

Conocidas tanto las peticiones SIP así como las respuestas SIP que podemos recibir en cada caso, se van a mostrar escenarios más detallados.

En la figura 2-18 se muestra cómo se lleva a cabo el registro del usuario *200@ual.es* ante un proxy SIP que requiere autenticación, es decir, se muestra cómo un terminal se registra en el proxy SIP.

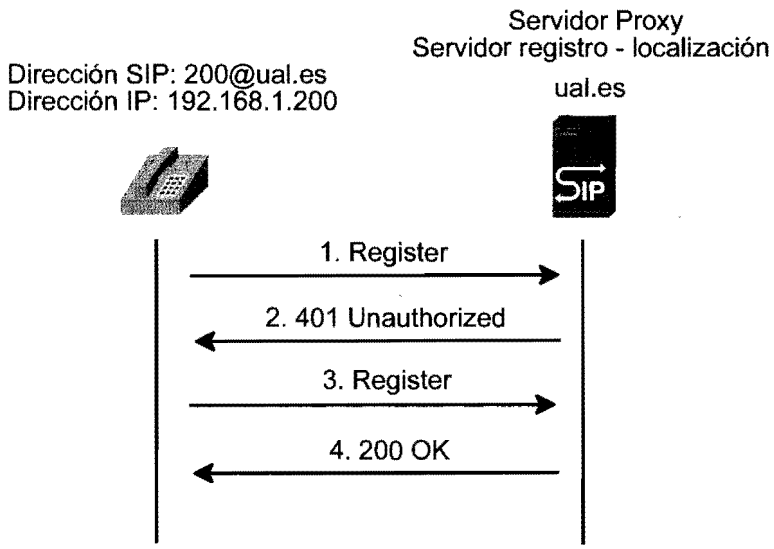


Figura 2-18. Proceso de registro de un usuario en un Proxy SIP

El teléfono envía la petición SIP Register al proxy SIP. Este al estar configurado para exigir autenticación envía una respuesta de vuelta indicando que el usuario *200@ual.es* no está autorizado por el momento y requiere por tanto mostrar los credenciales necesarios. El teléfono envía por tanto de nuevo la petición *Register* añadiendo a esta unos nuevos campos donde indica el password con el que se encuentra registrado en el proxy SIP de *ual.es*. Dado que la información de autenticación es correcta, el proxy SIP responde con una respuesta 200 OK indicando el éxito de la petición.

Suponiendo que tanto el usuario 200 como el 201 se encuentran registrados, en la figura 2-19 se muestra cómo se establecería una comunicación entre ellos.

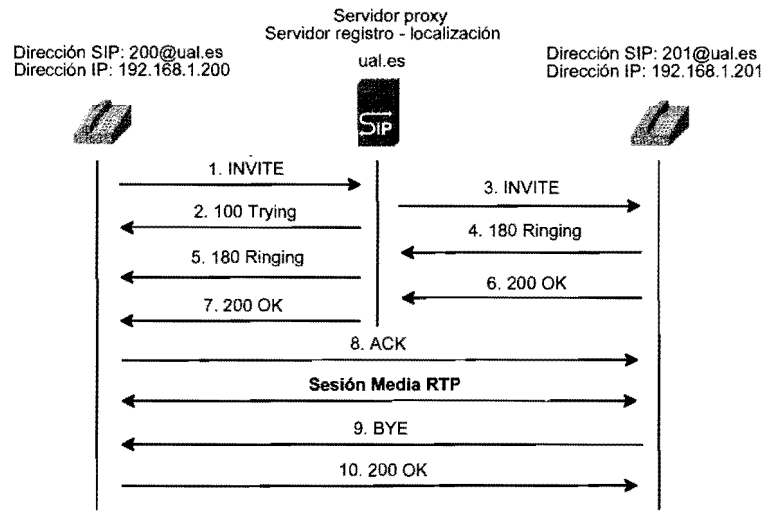


Figura 2-19. Proceso de establecimiento de llamada entre dos terminales

En la figura 2-19 se muestra cómo se completa la señalización *REGISTER* e *INVITE*. Estas son las peticiones más importantes, las cuales se requiere conocer detalladamente. Para ello vamos a describir a continuación el contenido de los mensajes intercambiados para ver el funcionamiento de SIP en mayor profundidad.

Descripción de los mensajes SIP

Conocidos los pasos en los que se lleva a cabo una comunicación entre dos teléfonos, cómo se registran ante un proxy SIP con autenticación, etc., se está en disposición de conocer el contenido de cada uno de los mensajes SIP que se intercambian. En la tabla 2-5 que se presenta a continuación se muestran cada uno de los campos de la cabecera SIP junto a su descripción.

Un ejemplo del contenido de los mensajes SIP intercambiados entre un teléfono y el proxy SIP durante el registro es el que se muestra en el listado 2. Antes, es necesario conocer la configuración del teléfono que va a registrarse:

Tabla 2-5. Campos de los mensajes SIP

Campo	Descripción
Via	En este campo se almacena cada uno de los elementos por los que va pasando la petición. Almacenar el camino que sigue la petición desde su origen al destino es muy útil para las respuestas, ya que estas simplemente tienen que seguir el camino inverso.

Max-Forwards	Número máximo de saltos permitidos a la petición para llegar a su destino. Ese valor será decrementado en cada uno de los saltos por los que va pasando.
From	En ella se indica la entidad origen que envió la petición SIP. Se especifica mediante la URI o mejor dicho mediante el AOR (Address of Record) que no es ni más ni menos que una URI global y pública que puede ser rutada desde cualquier punto.
To	Hace referencia a la URI de destino o AOR de destino de la petición. Hay que mencionar que esta no se utiliza para rutar el paquete hacia el próximo salto, sino que siempre mantiene el destino de la petición inicial.
Call-ID	Es un identificador único y global que se forma mediante combinación de una cadena aleatoria, el nombre de la máquina o la dirección IP del teléfono. La combinación de los tags que se indican en TO, FROM junto con el CALL-ID definen e identifican de manera unívoca un diálogo SIP entre dos extremos.
Cseq	Es un número de secüencia, donde cada nueva petición que se envía en un mismo diálogo incrementa en una unidad su valor. No es ni más ni menos que un contador de peticiones pertenecientes a un mismo diálogo.
Contact	En él se indica la SIP URI de la forma usuario@direcciónIP[:puerto] o usuario@dominio[:puerto] que representa la dirección de contacto directo con el emisor de la petición. La finalidad de esta dirección de contacto directo es la de que las futuras peticiones se puedan enviar directamente al emisor, evitando a la petición seguir el mismo camino, como sucede con las respuestas, que siguen el camino inverso dictaminado por las cabeceras VIA.
Content-type	Tipo del cuerpo del mensaje. No siempre tiene cuerpo el mensaje, pero si lo tuviera por lo general sería "application/SDP", esto es, el protocolo multimedia SDP.
Content-length	Tamaño del cuerpo del mensaje.

Tabla 2-6. Ejemplo del contenido de los mensajes

Atributo	Valor
Login	200
Password	200
Dirección IP	192.168.1.129
Proxy SIP	192.168.1.36:5060

Dominio	192.168.1.36 *Dirección del proxy SIP, pero podría ser otra*

El primer mensaje es el mensaje REGISTER enviado del teléfono al proxy SIP:

Listado 2-1. Mensaje 1 - 192.168.1.129:5060 -> 192.168.1.36:5060

```
REGISTER sip:192.168.1.36 SIP/2.0
Via: SIP/2.0/UDP
192.168.1.129:5060;rport;branch=z9hG4bK0B189FA6A0030AD1A61FA65
DB8174B61
From: 200 <sip:200@192.168.1.36>;tag=618484649
To: 200 <sip:200@192.168.1.36>
Contact: "200" <sip:200@192.168.1.129:5060>
Call-ID: 3C76206F9D406603585E2A12F49FACF6@192.168.1.36
CSeq: 42031 REGISTER
Expires: 1800
Max-Forwards: 70
User-Agent: X-Lite release 1105d
Content-Length: 0
```

En la comunicación hay que destacar dos cosas. La primera es que el teléfono quiere recibir las respuestas asociadas a esta petición que genere el proxy SIP, ya que incluye un registro con su dirección IP en el campo VIA. La segunda y muy importante es la información indicada en la cabecera Contact ya que es donde el teléfono de identificador 200 le indica al proxy SIP la dirección IP que tiene en este preciso instante. Esto permite al proxy SIP poder localizarlo para enviarle futuras peticiones.

Listado 2-2. Mensaje 2 - 192.168.1.36:5060 -> 192.168.1.129:5060

```
SIP/2.0 401 Unauthorized
Via: SIP/2.0/UDP
192.168.1.129:5060;rport=5060;branch=z9hG4bK0B189FA6A0030AD1A6
1FA65DB8174B61
From: 200 <sip:200@192.168.1.36>;tag=618484649
To: 200
<sip:200@192.168.1.36>;tag=329cfeaa6ded039da25ff8cbb8668bd2.dc
6e
Call-ID: 3C76206F9D406603585E2A12F49FACF6@192.168.1.36
CSeq: 42031 REGISTER
WWW-Authenticate: Digest realm="192.168.1.36",
nonce="48a3134d61dad5515f79f3f7363bda6aab8a1f90"
Server: OpenSER (1.3.2-notls (i386/linux))
Content-Length: 0
```

Continuando con el flujo de la figura 2-19, el proxy SIP envía la respuesta "401 Unauthorized" ya que en la petición REGISTER anterior el teléfono no indicó los credenciales, es decir, no indicó su password, y por tanto no está autorizado. La cabecera *WWW-Authenticate* indica al teléfono que debe autenticarse para el dominio 192.168.1.36.

El siguiente mensaje de la comunicación es el envío del mensaje REGISTER pero esta vez añadiendo el campo "Authorization" donde indica sus credenciales.

Listado 2-3. Mensaje 3 - 192.168.1.129:5060 -> 192.168.1.36:5060

```
Via: SIP/2.0/UDP
192.168.1.129:5060;rport;branch=z9hG4bK0F49CB5E56B3A4C8D02D235
CEBA14D44
From: 200 <sip:200@192.168.1.36>;tag=618484649
To: 200 <sip:200@192.168.1.36>
Contact: "200" <sip:200@192.168.1.129:5060>
Call-ID: 3C76206F9D406603585E2A12F49FACF6@192.168.1.36
CSeq: 42032 REGISTER
Expires: 1800
Authorization: Digest
username="200",realm="192.168.1.36",nonce="48a3134d61dad5515f7
9f3f7363bda6aab8a1f90",response="b7cc90ae907b4a7655aeec66df6c8
0d2",uri="sip:192.168.1.36"
Max-Forwards: 70
User-Agent: X-Lite release 1105d
```

Finalmente si los credenciales son correctos el proxy SIP le comunica al teléfono que la autenticación se ha llevado a cabo con éxito y ahora el teléfono 200 puede realizar y recibir llamadas.

Listado 2-4. Mensaje 4 - 192.168.1.36:5060 -> 192.168.1.129:5060

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP
192.168.1.129:5060;rport=5060;branch=z9hG4bK0F49CB5E56B3A4C8D0
2D235CEBA14D44
From: 200 <sip:200@192.168.1.36>;tag=618484649
To: 200
<sip:200@192.168.1.36>;tag=329cfeaa6ded039da25ff8cbb8668bd2.26
cb
Call-ID: 3C76206F9D406603585E2A12F49FACF6@192.168.1.36
CSeq: 42032 REGISTER
Contact: <sip:200@192.168.1.129:5060>;expires=1800
Server: OpenSER (1.3.2-notls (i386/linux))
Content-Length: 0
```

Aprovechando este ejemplo se van a introducir dos nuevos conceptos: transacción y diálogo:

- **Transacción.** Una transacción se lleva a cabo entre un UAC y un UAS. Comprende todos los mensajes desde la primera petición hasta la última respuesta asociada a esta (no se toman como respuesta válida para finalizar la transacción las 1xx). Si la petición es INVITE y la última respuesta no es del tipo 2xx, la transacción incluye el ACK como parte de esta. Si por el contrario la respuesta es del tipo 2xx, el mensaje ACK no se incluye en la transacción.

- **Diálogo.** Por lo general comienza con una petición INVITE y finaliza con la petición BYE. Un diálogo es identificado por las etiquetas (tag) de los campos FROM, TO y el campo Call-ID.

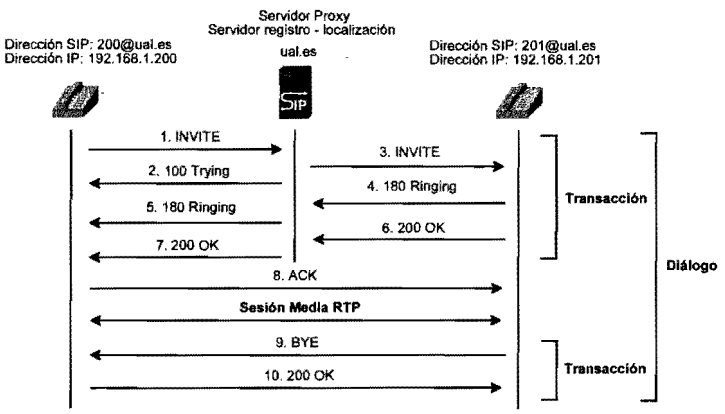


Figura 2-20. Petición invite

Ringing es una respuesta del tipo 1xx por lo que no se toma como respuesta válida para finalizar la transacción. Sin embargo la respuesta 200 OK sí que es una respuesta válida para finalizar la transacción.

Dado que la respuesta a la petición INVITE es del tipo 2xx, el ACK no se incluye en la transacción. Por último una nueva transacción se inicia con la petición BYE y finaliza con la respuesta asociada "200 OK".

5.1.2 H323

El protocolo H.323 fue diseñado por ITU, *International Telecommunication Union* en 1996. Fue diseñado para ser un estándar en la

transmisión de audio, video y datos a través de las redes IP en las cuales no existe garantía en la calidad del servicio. El estándar H.323 ofrece control y señalización de la llamada, control y transporte multimedia, control del ancho de banda punto a punto y conferencias.

La señalización de H.323 es muy rápida, sobre todo si las comparamos con la de SIP, la cual utiliza paquetes de gran tamaño. Esto es debido a que el formato de los mensajes en H.323 es binario, mientras que en los mensajes SIP el formato es texto plano. El diseño de H.323 está muy arraigado a la filosofía seguida en el diseño de la PSTN: simplicidad y alta disponibilidad.

H.323 es una suite de protocolos tanto de audio como de video, junto a los componentes necesarios para ofrecer comunicaciones multimedia en redes de conmutación de paquetes.

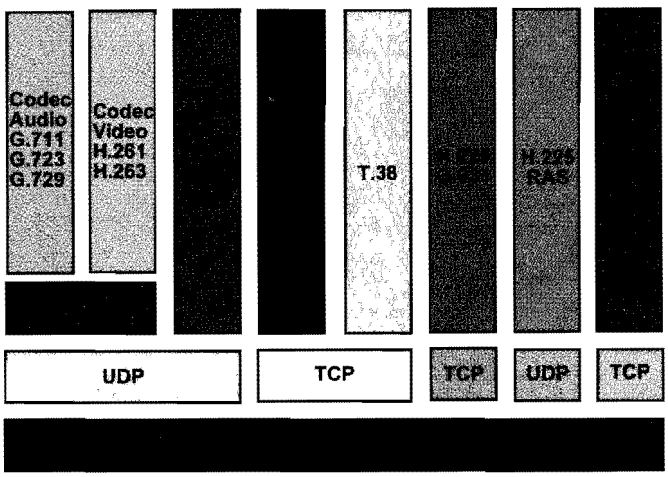


Figura 2-21. Protocolo H.323

Tal y como se muestra en la figura 2-21 el protocolo H.323 incluye el protocolo H.225 para empaquetar, sincronizar e iniciar llamadas mediante la señalización Q.931. Por otro lado H.245 se usa tanto para la negociación como para el manejo de los canales lógicos. T.120 y T.38 son utilizados para Datos y Fax.

A pesar de estar muy extendido, actualmente el auge de H.323 está descendiendo, ya que está siendo sustituido por SIP, el cual es modular y por tanto mucho más flexible.

5.1.3 Otros

El "Inter-Asterisk Exchange Protocol", o de manera abreviada IAX, es también un protocolo de señalización; y algo más. La principal diferencia entre IAX y SIP o H.323 es que IAX no utiliza RTP, sino que en su lugar implementa su propio mecanismo de transmisión de voz.

IAX es mucho más compacto que los dos anteriores ya que ha sido diseñado únicamente para aplicaciones telefónicas, a diferencia de H.323 y sobre todo de SIP, que pueden utilizarse en otros tipos de tráfico media.

IAX trabaja junto a UDP con una característica muy especial: todas las comunicaciones (registro, señalización de llamada, transmisión de voz) hacen uso de un único puerto UDP. Por lo tanto el NAT no supone un problema en IAX a diferencia de SIP, ya que tanto los datos de señalización como el audio viajan por el mismo puerto.

Un inconveniente de IAX es que no es un estándar, sino un protocolo independiente creado por Mark Spencer, creador de Digium. A pesar de ser un protocolo propietario es abierto y ha sido aceptado por la comunidad de VoIP.

La nueva revisión de IAX, IAX2, resulta ser un protocolo con muchas novedades respecto de su versión anterior pero con la característica de conservar aún su sencillez. Permite utilizar una gran cantidad de codecs y stream, lo que le permite aumentar su funcionalidad para dar soporte a aplicaciones no únicamente telefónicas.

En la tabla 2-7 se muestran las diferencias más importantes entre SIP e IAX.

Tabla 2-7. Comparativa entre SIP e IAX

	SIP	IAX	Conclusión
Tipos de mensajes	Los mensajes son en formato texto	Los mensajes son en formato binario	IAX consume menos ancho de banda
Señalización	Datos y señalización en puertos distintos	Datos y señalización por el mismo puerto	En SIP aparecen problemas de NAT
Señalización II	Al ir la señalización y audio por puertos distintos, el audio puede ir de extremo a extremos sin pasar por el servidor SIP	Al ir la señalización y audio por el mismo puerto, el audio pasa obligatoriamente por el servidor IAX	Consumo alto de recursos en el servidor IAX ante una gran cantidad de llamadas

Estándar	Estandarizados por la IETF hace tiempo	Aún está siendo estandarizado	SIP es soportado por la mayoría de equipos
Uso de puertos	1 señalización + 2 Media RTP (uno por sentido)	Un único puerto para señalización y audio	SIP requiere de más puertos libres

Existen otros protocolos que son utilizados generalmente por compañías telefónicas así como proveedores de servicios de VoIP: MGCP (Media Gateway Controller Protocol); MEGACO/H.248, cuya funcionalidad es la de conformar un estándar en la señalización de gateways media que físicamente se encuentran distribuidos; SIGTRAN (Signalling Transport) aparece como otro protocolo gracias al cual solventan algunas limitaciones de MGCP, etc.

5.2 PROTOCOLOS DE AUDIO

En el punto anterior se han mostrado los distintos protocolos que se tienen a disposición para establecer una comunicación entre dos extremos, sin embargo como ya se indicó anteriormente esto no es suficiente para establecer una comunicación media. Para establecer un flujo de comunicación media es necesario un protocolo que intercambie la información entre los extremos de dicha comunicación, es decir, que transporte la información entre un origen y su destino, además de proveer de las técnicas necesarias para enviar los problemas que se pueden presentar durante el intercambio, tales como: jitter[4], retardo, etc.

Los protocolos más utilizados para esta finalidad de transporte de audio y video en tiempo real son:

- **Real Time Protocol (RTP)**
- **Real Time Control Protocol (RTCP)**

RTP se encuentra definido en el RFC3550 y es el encargado de transportar tanto audio como vídeo en tiempo real. Utiliza UDP como protocolo de transporte, ya que el uso de TCP y su control de flujo y congestión darían lugar a un retardo elevado durante la comunicación a causa de las retransmisiones.

---

[4] jitter: variación del retardo.

El protocolo RTP, para llevar a cabo su función, hace uso de un número de secuencia, marcas de tiempo, envío de paquetes sin retransmisión, identificación del origen, identificación del contenido, sincronización, etc., lo que le permite en presencia de pérdidas, jitter o retardo poder continuar con la reproducción del flujo de paquetes. Por lo tanto no puede garantizar que la entrega de tráfico se haga en tiempo real, aunque sí garantiza al menos que lo hará de forma sincronizada.

El protocolo RTCP es el protocolo compañero de RTP. RTCP es el encargado de monitorizar el flujo de los paquetes RTP. Obtiene estadísticas sobre el jitter, RTT, latencia, pérdida de paquetes, etc. Fundamentalmente está relacionado con la calidad de servicio. El inconveniente es que, aunque realice una monitorización de la calidad de servicio de RTP, no se poseen mecanismos como reservar ancho de banda o control de la congestión para intentar paliar una situación en la que la calidad de la transmisión no es suficiente. Es por ello por lo que la utilización de RTCP es opcional, aunque también recomendable.

Al inicio del punto anterior *5.1. Protocolos de comunicación* se hizo referencia a que el protocolo SIP se integra perfectamente con protocolos de audio como RTP, RTSP (Real Time Streaming Protocol) gracias al protocolo SDP, que permite formar así una completa arquitectura multimedia. Por ello queda más que justificado el realizar una introducción del protocolo SDP.

SDP o Session Description Protocol, definido en el RFC4566, se utiliza durante la negociación que lleva a cabo SIP entre los dos agentes. Su función es la de detallar cómo se va a realizar el intercambio de comunicación posterior mediante protocolos como, por ejemplo, RTP. Para ello indica toda la información relacionada con el tráfico Media tal como IP y puerto donde espera cada agente recibir el audio, el codec a utilizar entre ambos, etc.

En SIP, la petición INVITE es la que contiene como cuerpo del mensaje la descripción de la sesión (el SDP) del agente llamémosle "llamante", mientras que la repuesta "200 OK" del agente "llamado" contiene la descripción de la sesión del agente "llamado". En la descripción de la sesión cada agente informa al otro extremo de los codec que soporta, de la dirección IP y puerto donde espera recibir el tráfico RTP además de cierta información adicional requerida por RTP.

2-5. Petición Invite - 192.168.1.130:5061 -> 192.168.1.130:5060

```
INVITE sip:200@192.168.1.130 SIP/2.0
Via: SIP/2.0/UDP
192.168.1.130:5061;rport;branch=z9hG4bK7F01146D747821A560544FB
9D08549
From: X-Lite <sip:201@192.168.1.130:5061>;tag=1084480144
To: <sip:200@192.168.1.130>
Contact: <sip:201@192.168.1.130:5061>
Call-ID: 1333EB07-D999-9148-107A-218A1CBB6B8E@192.168.1.130
```

```
CSeq: 21508 INVITE
Max-Forwards: 70
Content-Type: application/sdp
User-Agent: X-Lite release 1105d
Content-Length: 310
v=0
//IP donde el terminal 201 espera recibir el tráfico Media//
o=201 3462578896 3462578922 IN IP4 192.168.1.130
s=X-Lite
c=IN IP4 192.168.1.130
t=0 0
//Puerto donde el terminal 201 espera recibir el tráfico
Media//
m=audio 8000 RTP/AVP 0 8 3 98 97 101
a=rtpmap:0 pcmu/8000
a=rtpmap:8 pcma/8000
a=rtpmap:3 gsm/8000
a=rtpmap:98 iLBC/8000
a=rtpmap:97 speex/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-15
a=sendrecv
```

Listado 2-6. Respuesta 200 OK - 192.168.1.129:5061 -> 192.168.1.130:5060

```
SIP/2.0 200 Ok
Via: SIP/2.0/UDP
192.168.1.130:5060;branch=z9hG4bK1ad6ffa8;rport
From: "device" <sip:201@192.168.1.130>;tag=as312f6ed9
To: <sip:200@192.168.1.129:5061>;tag=1526648226
Contact: <sip:200@192.168.1.129:5061>
Call-ID: 61426ef41239ebf33a6e4d9361c7564a@192.168.1.130
CSeq: 102 INVITE
Content-Type: application/sdp
Server: X-Lite release 1105d
Content-Length: 310
v=0
//IP donde el terminal 200 espera recibir el tráfico Media//
o=200 3238555091 3238570131 IN IP4 192.168.1.129
s=X-Lite
c=IN IP4 192.168.1.129
t=0 0
//Puerto donde el terminal 200 espera recibir el tráfico
Media//
m=audio 8000 RTP/AVP 0 8 3 98 97 101
a=rtpmap:0 pcmu/8000
```

```
a=rtpmap:8 pcma/8000
a=rtpmap:3 gsm/8000
a=rtpmap:98 iLBC/8000
a=rtpmap:97 speex/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-15
a=sendrecv
```

5.3 ALGORITMOS DE CODIFICACIÓN Y DECODIFICACIÓN DE VOZ (CODECS)

La información transportada por un protocolo de audio tal como RTP es tratada por un codec antes de añadirla en un paquete RTP.

Un codificador-decodificador o de manera abreviada "codec" es un algoritmo que traduce una señal analógica en una señal digital. Por lo tanto uno de los aspectos a valorar a la hora de elegir un codec es el tamaño al codificar la onda analógica. Si el tamaño es muy grande una vez finalizada la codificación, la compresión del codec será baja y por tanto se espera una buena fidelidad en sonido digital. Sin embargo, al tener un tamaño grande se requiere un mayor ancho de banda para transmitirlo. Por otro lado si el tamaño es pequeño tras la codificación, el ratio de compresión es alto y no se espera tanta fidelidad con el sonido analógico original como en el caso anterior. En este caso al ser el ratio de compresión alto, no requiere tanto ancho de banda para poder ser enviado por la red. Por tanto la elección de un codec de mayor o menor fidelidad hay que valorarlo, ya que no siempre es tan importante un alto grado de fidelidad. Por ejemplo, en el caso de los humanos nuestro oído tiene unos límites a partir de los cuales no percibe mejoras en la calidad del sonido y es entonces importante no seleccionar un codec con demasiada calidad pero sí con el menor consumo de ancho de banda posible manteniendo una calidad aceptable, lo que nos va a permitir tener un mayor número de comunicaciones VoIP simultáneamente.

Los codecs de audio para telefonía se pueden dividir en dos grupos: aquellos basados en "pulse code modulation" (PCM) y aquellos que reestructuran la representación digital de PCM en formatos más livianos. Ambos tipos se detallan mediante los distintos aspectos técnicos, a los cuales es posible realizar distintas configuraciones que modifican el funcionamiento del codec. A continuación vamos a ver de manera sencilla su funcionamiento.

En la figura 2-22 puede observarse cómo el codec va tomando consecutivamente las distintas partes que componen el flujo de audio que se tiene de entrada. A cada una de estas partes se le denomina muestra.

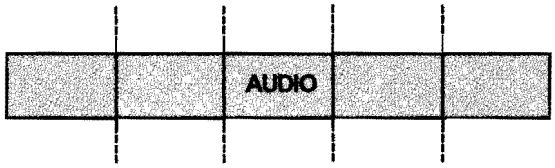


Figura 2-22. Flujo de audio

Cada codec tiene unos valores determinados en dos atributos relacionados con el tamaño y tiempo de la muestra: *sample size* o *tamaño muestra* y *sample interval* o *intervalo muestra*. El primero determina el número de bits de la muestra una vez codificada por el codec. El segundo atributo indica el intervalo de tiempo o cada cuánto tiempo se lleva a cabo este proceso, es decir, cada cuánto se va codificando una muestra.

Conocido el tamaño de cada muestra codificada y la frecuencia de muestreo, es posible calcular el ancho de banda que es necesario para poder establecer una comunicación de buena calidad entre dos puntos. Esta medida se conoce con el nombre de *Bit rate* o *tasa de bits*. Supongamos que el tamaño del sample size es 24 bytes y el intervalo de tiempo *sample interval* es cada 30 ms:

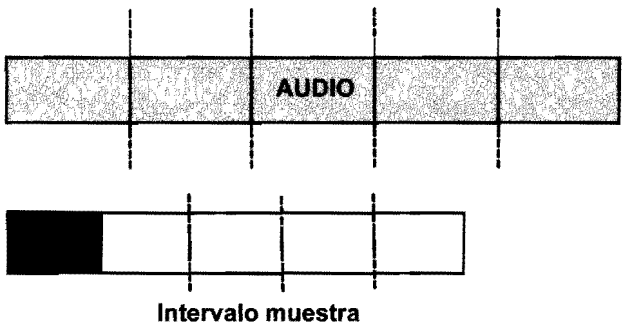


Figura 2-23. Intervalo de muestreo

$$(24 * 8) / (30 * 0.001) = 6400 \text{ bits/segundo}$$

Este sería el ancho de banda teórico necesario para poder establecer una buena comunicación. Sin embargo en la realidad se requiere bastante ancho de banda, ya que no sólo se envía el audio codificado, sino que también se envían cabeceras de los distintos paquetes que encapsulan el *sample* o *la muestra*.

Tal y como se puede ver en la figura 2-24, el siguiente paso consiste en que cada una de las muestras codificadas se incluyan como contenido en un paquete RTP, el cual se incluye a su vez en un paquete UDP, y por último en un paquete a nivel IP. Al final se genera un paquete de un tamaño mayor al sample size, y es ese el motivo por lo que se requiere más ancho de banda. A este tamaño extra, que se añade al *sample size,* y que es absolutamente necesario para poder rutar el paquete a través de la red, se le denomina *overhead* o *sobrecarga.*

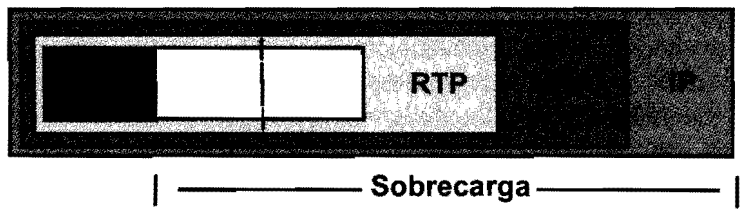


Figura 2-24. Estructura de un paquete RTP

Teóricamente se le añade un tamaño de cabecera de IP de 20 bytes, un cabecera UDP de 8 bytes y una cabecera RTP de 12 bytes, lo que hace un total de overhead de 40 Bytes (mínimo). Ahora sí podemos calcular el ancho de banda que teóricamente vamos a utilizar:

((40 + 24) * 8) / (30 * 0.001) = 17066 bits por segundo.

Como se puede apreciar, la necesidad de ancho de banda ha aumentado considerablemente de unos 6400 a 17066 bits por segundo. Como ya se ha comentado, esto es debido al *overhead.* Si observamos la figura 2-24 realmente se tiene demasiado overhead para un pequeño frame de audio codificado. La relación "audio codificado y overhead necesario para enviar ese audio codificado" no está bien equilibrada, ya que se tienen 24 bytes de audio útil por 40 bytes de las distintas cabeceras. Una medida para paliar esta proporción excesiva de overhead, es introducir más de un frame en el paquete RTP (véase la figura 2-25).

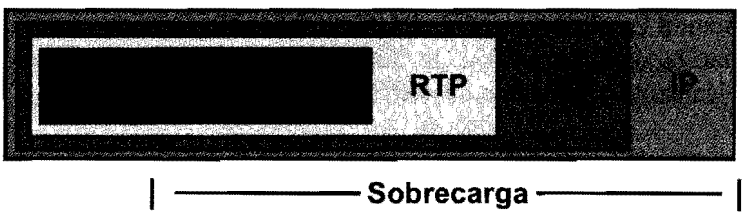


Figura 2-25. Estructura de un paquete RTP con varios frames

De esta manera la proporción de audio útil y overhead está mucho más equilibrada. Sin embargo esto tiene una consecuencia negativa, y es que la latencia aumenta. Si antes se genera una muestra codificada cada 30 ms y que la enviábamos rápidamente, ahora cada muestra se sigue generando también cada 30 ms pero son necesarias 3 (en este ejemplo). Ello aumenta el tiempo en que se envía cada paquete a 90 ms:

((40 + (3 * 24)) * 8) / (3 * (30 * 0.001)) = 9956 bits por segundo.

El ancho de banda necesario se ha visto reducido considerablemente respecto a la situación anterior pero aumenta la latencia, es decir, hay que esperar más tiempo a obtener el audio suficiente para generar 3 frames para poder completar un paquete. Esto provoca que el destinatario del paquete tarde más tiempo (aunque recibe más cantidad de audio) en llegar a destino, por lo que si se retrasa (*lag*) durante el trayecto se va a producir temporalmente un corte en la comunicación. Además ahora se tienen 72 bytes de audio útil frente a 40 bytes de overhead. Indirectamente se presenta otro problema además de la latencia, y es que la pérdida de un paquete con 72 bytes de audio es mucho más considerable que la pérdida de un paquete de 24 bytes de audio.

En definitiva y a modo de resumen, incluir más o menos frames de audio en el paquete tiene sus ventajas e inconvenientes. Unas consideraciones que pueden ayudarnos a decantarnos por una u otra forma son las siguientes:

- Si se dispone de una buena conexión a Internet, quizás lo más recomendable sería reducir la latencia todo lo posible, con la consecuencia de que ante una pérdida de un paquete los resultados no serían tan graves.

- Si por el contrario no dispone de una buena conexión, un término medio entre latencia y overhead es lo más recomendable.

- Si el *sample interval* del codec es muy pequeño, a pesar de que incluyamos varios frames de sonido en un mismo paquete, la latencia va a ser pequeña (recordar que la latencia es el resultado de multiplicar el número de frames de sonido en el paquete por el sample interval). De esta manera podemos enviar la mayor cantidad de audio posible, teniendo una baja latencia (lógicamente el consumo de ancho de banda aumentará).

- Si el *sample interval* del codec es muy grande, es recomendable no incluir más de un frame de sonido en el paquete ya que la latencia es elevada.

En definitiva, a la hora de elegir un codec hay que tener en cuenta la **calidad de sonido** y el **ratio de compresión** (directamente relacionado con el

ancho de banda y la latencia) y seleccionar aquel que más se adapte a nuestro requerimientos de voz.

En la tabla 2-8 se muestra un resumen con los codecs más utilizados actualmente y la configuración más equilibrada. El significado de cada campo de la tabla es:

- **Bit Rate** es, teóricamente, el ancho de banda requerido por un solo sentido de la comunicación y suponiendo un uso continuo del ancho de banda. Por tanto, para la práctica este valor ha de ser multiplicado por dos (ya que se tienen dos sentidos en la comunicación). Para algunos codecs, el consumo de ancho de banda real será menor al obtenido tras la multiplicación ya que durante una conversación se tienen silencios que son detectados por el codec y no hace por tanto un uso continuo del ancho de banda. Para aquellos que no detectan el silencio en ambos sentidos, el consumo real será mayor ya que hay que tener en cuenta que no se envía únicamente datos de voz, sino también datos como cabeceras, que suponen un "overhead" en el tamaño del paquete de voz y requiere por tanto un ancho de banda adicional.

- **Audio útil** (ms) no es ni más ni menos que la "cantidad de voz útil", la cantidad de voz real que representa el total del paquete. Es un detalle que afecta de manera notable en el funcionamiento del codec desde el punto de vista del *lag* y del consumo de ancho de banda.

- **Ancho de banda ethernet estimado** indica el consumo de ancho de banda esperado en una conversación.

- **Latencia** o tiempo transcurrido entre envío y envío de los paquetes.

- **Calidad general** del codec (valor del 1=muy mala al 5=excelente).

Tabla 2-8. Comparativa de los codecs

Nombre	Bit rate (kbps)	Audio útil (Bytes)	Ancho estimado (kbps)	Latencia (ms)	Observaciones	Calidad General
G.711	64	240	74.6	30	PCM. Existen dos versiones "u-law" (US, Japan) y "a-law" (Europa).	4.1

G.723.1	6.4	24	17	30	Utiliza Multipulse Maximum Likelihood Quantization (MP-MLQ) o Algebraic-Code-Excited Linear-Prediction (ACELP). Alta compresión manteniendo una buena calidad de sonido.	3.8-3.9
G.726	32	80	48	20	ADPCM. Sustituye a los codecs G.721 y G.723	3.85
G.728	16	60	26.6	30	Utiliza Code-Excited Linear-Prediction (CELP) para codificar.	3.61
G.729	8	20	24	20	G729: codec original. G729A menos complejo que G729 pero menor calidad (compatible con G729). G729B es como G729 pero con supresión de silencios (no es compatible con las anteriores). Por último G729AB es un G729A con supresión de silencios y únicamente compatible con G729B.	3.92

Nombre	Bit rate (kbps)	Audio útil (Bytes)	Ancho estimado (kbps)	Latencia (ms)	Observaciones	Calidad General
GSM 06.10	13.2	33	29.2	20	Utiliza Regular-Pulse Excitation Long-Term Predictor (RPE-LTP). Usado por la tecnología celular GSM. Es soportado por gran cantidad de plataformas hardware y software.	3.8
LPC10	2.4	7	16.7	22.5	Linear predictive codec (LPC). La voz suena un poco "robótica".	

Speex	11.2	28	27.2	20	El bitrate es variable. Además detecta el silencio.	
ILBC	15.2	57	25.8	30	Reciente, por lo que su soporte en dispositivos comerciales es muy reducido. Requiere un importante procesamiento del sonido.	4.14

6 Conclusiones

La tecnología de VoIP dispone de todo lo necesario y fundamental para comenzar su expansión: teléfonos IP, centralitas software así como hardware, estándares que definen el funcionamiento, líneas de datos con buen ancho de banda, etc. Las ventajas respecto a la telefonía tradicional resultan muy claras, y es por ello por lo que la mayoría de las empresas telefónicas están comenzando a ofrecer servicios de VoIP, y el motivo no es otro que la tecnología VoIP es la telefonía del futuro.

CAPÍTULO 3

LA REVOLUCIÓN SE LLAMA ASTERISK

Saúl Ibarra Corretgé y David Prieto Carrellán

1 Introducción

Hoy en día se puede afirmar sin lugar a dudas que las comunicaciones están cambiando. Al principio sólo disponíamos de telefonía fija, ahora tenemos telefonía móvil, y además Internet.

En las últimas décadas la tecnología ha avanzado de una manera imparable, y desde hace poco estos avances han llegado al mundo de las comunicaciones empresariales.

Todo comenzó cuando un chico joven (Mark Spencer) decidió montar una empresa para dar soporte sobre temas relacionados con GNU/Linux, a la que llamó Linux Support Services. Para captar clientes y meterse en el mercado Mark quería dar un servicio de atención las 24 horas del día, de forma que alguien podría llamar a LSS, dejar un mensaje y su incidencia sería atendida lo antes posible.

Esta idea derivó en la necesidad de un sistema telefónico, pero tras varias consultas de precios, Mark se dio cuenta de que no podría pagarlo (LSS se fundó con 4000$ de capital social). Cualquier otra persona habría pensado en robar un banco, pedir prestado el dinero o simplemente abandonar su idea, pero Mark tenía esa mentalidad hacker: "¿por qué no hacerlo yo mismo?".

Así, emprendió la aventura de programar una PBX software desde cero, algo inexistente hasta el momento. Asterisk supone un cambio radical en los sistemas de comunicaciones, ya que estos están basados en hardware, y Asterisk es software, por lo que ofrece una mayor flexibilidad y escalabilidad.

Para tener un producto completo a Asterisk le faltaba interactuar con las líneas analógicas y digitales existentes en la telefonía tradicional, y en busca de esta interoperabilidad, el proyecto Asterisk se encontró con el proyecto *Zapata Telephony*, iniciado por Jim Dixon. En este punto, Asterisk ya era capaz de unir ambos mundos: la telefonía analógica/digital tradicional y la VoIP.

Finalmente, en Linux Support Services se dieron cuenta de que lo más importante de su negocio no era el negocio en sí, sino Asterisk, por lo que su nombre cambió a Digium y el enfoque de la empresa cambió radicalmente para enfocarse al desarrollo de Asterisk.

La clave del éxito de Asterisk es que es Software Libre, aunque la decisión de hacerlo libre fue simple casualidad, ya que Mark tenía experiencia en el mundo *Open Source* con proyectos como cheops, l2tpd y Gaim. Más tarde se dio cuenta de que la decisión de hacer libre Asterisk fue lo que le ha dado tanto éxito, ya que contrasta totalmente con el mundo cerrado de las PBX tradicionales.

Todo esto ha llevado a que Asterisk se expanda muy rápidamente, y haya captado todo tipo de público: usuarios de Asterisk, empresas que basan su modelo de negocio en Asterisk y los hackers de Asterisk. Todos ellos han formado el ecosistema de la telefonía *Open Source*, que ha supuesto esta revolución imparable en el mundo de las comunicaciones.

2 Arquitectura

Tal y como puede verse en la figura 3-1, Asterisk fue diseñado de manera modular, de manera que cada usuario pueda seleccionar qué partes de Asterisk o módulos desea utilizar. Esto hace de Asterisk una aplicación realmente escalable y extensible:

- **Escalable.** Es posible desactivar los módulos no utilizados para instalar Asterisk en dispositivos embebidos de pocos recursos.

- **Extensible.** Para programar un nuevo módulo de Asterisk no es necesario conocer todo el código de Asterisk.

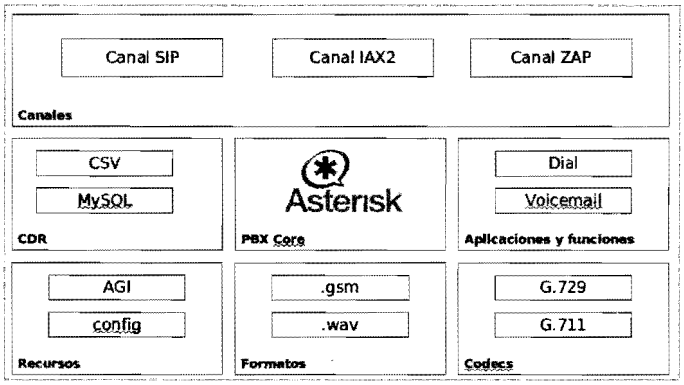


Figura 3-1. Estructura modular de Asterisk

Esta arquitectura permite al usuario construir su sistema Asterisk como si de un "Lego" se tratase, seleccionando los módulos que va a necesitar. Estos módulos se dividen en 7 categorías:

- **Core.** Se trata del núcleo de Asterisk, que incluye las funciones más básicas y posibilita la carga de módulos.

- **Recursos.** Aportan funcionalidades adicionales al core, como la posibilidad de leer ficheros de configuración (res_config), música en espera (res_musiconhold), etc.

- **Canales.** Permiten a Asterisk manejar un dispositivo de una determinada tecnología. Por ejemplo, para manejar dispositivos SIP se utiliza el módulo chan_sip, para IAX2 chan_iax y para canales analógicos/digitales chan_zap.

- **Aplicaciones y funciones.** Estos módulos conforman la "caja de herramientas" de Asterisk, ya que son los módulos que aportan las distintas herramientas para configurar nuestro sistema Asterisk.

- **CDR.** Estos módulos controlan la escritura del registro telefónico generado por Asterisk a diferentes formatos, por ejemplo a un fichero CSV, una base de batos MySQL, etc.

- **Codecs.** Para que Asterisk pueda codificar y decodificar la información de audio/vídeo que tiene que enviar y recibir dispone de distintos codecs.

- **Formatos.** Estos módulos posibilitan a Asterisk "entender" y manejar ficheros en distintos formatos, como mp3, alaw, ulaw, etc.

Se pueden definir los módulos que Asterisk cargará en fichero *modules.conf*, y consultar cuáles se encuentran cargados ejecutando *module show* desde el CLI de Asterisk.

3 Instalación

Una vez que estamos dispuestos a instalar Asterisk, mucha gente se hace la eterna pregunta: ¿qué distribución elegir? Esta pregunta da lugar a interminables charlas, y la respuesta siempre suele ser la misma: la distribución con la que te sientas más cómodo.

No obstante, las instrucciones que se darán en este libro están orientadas a Debian y a derivados de RedHat como Fedora y Centos, aunque no resulta difícil encontrar instrucciones precisas para otros sistemas.

¿Qué paquetes instalar?

En la web de Asterisk podemos encontrar 4 paquetes: Asterisk, Asterisk-Addons, Zaptel y LibPRI. Dependiendo de la instalación que vayamos a hacer necesitaremos unos u otros. Si nuestra instalación va a ser puramente VoIP, sin interacción con líneas analógicas o digitales, sólo necesitaremos el paquete "asterisk". Si vamos a utilizar tarjetas de comunicaciones analógicas, también necesitaremos el paquete "zaptel". Si se van a utilizar tarjetas de comunicaciones digitales, será necesario instalar "libpri".

El paquete "asterisk-addons" puede ser instalado en cualquiera de los escenarios de configuración, ya que contiene módulos adicionales que no se han incluido en el paquete principal por temas relacionados con las licencias de uso.

Dependencias necesarias para instalar Asterisk

Para compilar Asterisk se necesita el compilador *gcc* así como la utilidad *make*, además de las cabeceras de C. También son necesarias las librerías de

desarrollo de *OpenSSL*, ya que Asterisk las utiliza en su propia librería criptográfica. Para instalar estas dependencias en Debian basta con ejecutar:

```
# apt-get install build-essential libncurses5-dev libssl-dev
```

Si además vamos a instalar Zaptel, necesitaremos las cabeceras del Kernel (o las fuentes, si es un Kernel personalizado) y también se recomiendan las librerías *newt*, para disponer de la utilidad *zttool*. Para instar estas dependencias ejecutamos:

```
# apt-get install linux-headers-$(uname -r) libnewt-dev
```

Llegado este punto, nuestro sistema ya está listo para que comencemos a compilar Asterisk.

Descargar los paquetes necesarios

Procedemos a descargar los paquetes necesarios de la web de Asterisk:

```
# mkdir -p /usr/src/asterisk
# cd /usr/src/asterisk
# wget http://downloads.digium.com/pub/libpri/libpri-1.4-current.tar.gz
# wget http://downloads.digium.com/pub/zaptel/zaptel-1.4-current.tar.gz
# wget http://downloads.digium.com/pub/asterisk/asterisk-1.4-current.tar.gz
# wget http://downloads.digium.com/pub/asterisk/asterisk-addons-1.4-
current.tar.gz
# for pkg in *.tar.gz; do tar -zxvf $pkg; done
```

Ahora ya tendremos todos los paquetes descomprimidos en */usr/src/asterisk*, así que ¡a compilar!

Compilar libPRI

LibPRI es la librería encargada de dar soporte a señalización de primario (E1/T1) a Zaptel. No es necesario instalarla si no se van a utilizar tarjetas de primario.

```
# cd /usr/src/asterisk/libpri-1.4-current
# make
# make install
```

Compilar Zaptel

El paquete Zaptel contiene los módulos del Kernel para hacer funcionar las tarjetas de comunicación analógicas y digitales. Además, contiene varias utilidades de configuración y diagnóstico.

```
# cd /usr/src/asterisk/zaptel-1.4-current
# ./configure
# make menuselect
# make
# make install
# make config
```

A la hora de compilar Zaptel, primero ejecutamos *./configure*, de manera que se chequea nuestro sistema para comprobar si tenemos las dependencias necesarias instaladas. Posteriormente, al hacer *make menuselect*, se nos muestra un menú donde es posible elegir qué módulos y utilidades de Zaptel serán compilados. Si un módulo va a ser compilado, se mostrará un '*' a su lado, mientras que si no está disponible por la falta de alguna dependencia, se mostrará una 'X'. Una vez hemos elegido las opciones que queremos compilar, pulsamos 's', para salvar estas opciones.

Tras seleccionar los módulos que serán compilados, se ejecuta *make*, que los compilará, y *make install*, que los instalará. Si queremos que Zaptel se añada al arranque del sistema, ejecutamos *make config*, y se creará un script de inicio en */etc/init.d/*.

Compilar Asterisk

La compilación de Asterisk es muy similar a la de Zaptel, pero en este caso, como Asterisk tiene muchos módulos habrá muchos que se marquen con una 'X', ya que nos falta alguna librería para poder compilarlo. Cuando un módulo está marcado con una 'X', en la parte inferior de la pantalla se muestra la librería que necesita, por lo que si queremos que se compile no tenemos más que instalar esa librería, y tras ejecutar de nuevo *./configure*, saldrá marcado con un '*' en el *menuselect*.

```
# cd /usr/src/asterisk/asterisk-1.4-current
# ./configure
# make menuselect
# make
# make install
# make confg
# make samples
```

Al igual que en Zaptel, se ejecuta *make config*, para meter a Asterisk en el arranque, y automatizar así el proceso de arranque del servidor Asterisk. Además, si queremos que se generen los ficheros de ejemplo en */etc/asterisk/* ejecutamos *make samples*, y obtendremos una configuración por defecto muy limitada, pero funcional, de Asterisk.

Compilar Asterisk-Addons

El paquete Asterisk-Addons contiene diversas utilidades que no han podido ser incluidas en el paquete principal por temas relacionados con el licenciamiento de aplicaciones. Entre ellas está la aplicación que maneja el formato mp3, el módulo que escribe el CDR en MySQL y más. Su compilación es muy similar a la de Asterisk:

```
# cd /usr/src/asterisk/asterisk-addons-1.4-current
# ./configure
# make
# make install
```

Instalar voces para Asterisk en castellano

Al hacer *make menuselect* del paquete Asterisk, podréis comprobar que es posible instalar el juego oficial de sonidos de Asterisk en español. Este conjunto de sonidos tiene bastante acento inglés, y por tanto, utilizaremos otro conjunto de sonidos creado por Alberto Sagredo.

```
# cd /usr/src/asterisk
# wget http://www.voipnovatos.es/voces/voipnovatos-core-sounds-es-alaw-1.4.tar.gz
# wget http://www.voipnovatos.es/voces/voipnovatos-extra-sounds-es-alaw-1.4.tar.gz
# for pkg in voipnovatos*.tar.gz; do tar -zxvf $pkg -C /var/lib/asterisk/sounds; done
```

Tras ejecutar todos los pasos anteriores, obtendremos una instalación muy básica pero funcional de Asterisk, que podemos modificar y con la que podemos empezar a aprender.

4 Estructura de directorios

Al compilar e instalar Asterisk, se crean muchos directorios, y cada uno contiene una parte de Asterisk. A continuación se muestran todos los directorios relacionados con Asterisk, así como su función:

- */etc/asterisk/*. El directorio más importante para Asterisk, contiene los ficheros de configuración, así como el fichero *asterisk.conf*, donde se indica la ubicación de los demás directorios.

- */usr/lib/asterisk/modules/*. Contiene los ficheros binarios de los módulos de Asterisk que han sido compilados.

- */var/lib/asterisk/*. Contiene diversos ficheros importantes para Asterisk en distintos subdirectorios, además del *astdb*, la Base de Datos de Asterisk (Berkeley DB2), donde se guarda la información de registro de usuarios, etc.

    - */var/lib/asterisk/agi-bin/*. Directorio que contiene los scripts AGI que pueden ser ejecutados desde el dialplan con las aplicaciones AGI, EAGI, FastAGI o DeadAGI.

    - */var/lib/asterisk/firmware/*. Contiene ficheros de firmware necesarios para la comunicación de Asterisk con otros dispositivos como el IAXy.

    - */var/lib/asterisk/images/*. Contiene imágenes, que pueden ser transmitidas por canales que lo soporten.

    - */var/lib/asterisk/keys/*. Asterisk soporta autenticación mediante RSA en IAX2. En caso de configurar enlaces IAX2 con este tipo de autenticación, las claves pública y privada se almacenarán aquí.

    - */var/lib/asterisk/moh/*. Este directorio contiene los ficheros que serán utilizados como música en espera.

    - */var/lib/asterisk/sounds/*. Este directorio contiene los distintos sonidos que Asterisk es capaz de reproducir. Al utilizar aplicaciones como *Playback* o *Background*, si no se indica la ruta absoluta al fichero, se busca en este directorio.

    - */var/lib/asterisk/static-http/*. En caso de haberlo instalado, contiene los ficheros del Asterisk-GUI, además de un ejemplo de uso de *AJAM* (Asynchronous Javascript Asterisk Manager).

- */var/spool/asterisk/*. El directorio de spool de Asterisk contiene diversos subdirectorios, relacionados con la entrada/salida de ficheros:
    - */var/spool/asterisk/dictate/*. En este directorio se sitúan los ficheros generados por la aplicación *Dictate*.
    - */var/spool/asterisk/meetme/*. Contiene los ficheros de audio de las conferencias *MeetMe* que hayan sido grabadas.
    - */var/spool/asterisk/monitor/*. Contiene los ficheros de audio con las grabaciones realizadas con las aplicaciones *Monitor* y *MixMonitor*.
    - */var/spool/asterisk/outgoing/*. Asterisk lee periódicamente este directorio en busca de callfiles, ficheros que permiten generar llamadas automáticamente.
    - */var/spool/asterisk/system/*. Si utilizamos la aplicación System, Asterisk guarda los posibles ficheros temporales generados en esta carpeta.
    - */var/spool/asterisk/tmp/*. Contiene ficheros temporales que Asterisk puede necesitar antes de mover un fichero de un sitio a otro.
    - */var/spool/asterisk/voicemail/*. Asterisk utiliza este directorio para almacenar todos los ficheros con los mensajes de los buzones de voz.
- */var/run/*. Este directorio del Sistema Operativo contiene los ficheros con el identificador de proceso (PID) de los procesos activos, incluido el de Asterisk, tal y como se indica en el fichero *asterisk.conf*.
- */var/log/asterisk/*. Contiene los ficheros de log, así como el CDR en formato CSV (valores separados por comas).

5 Puesta en marcha de Asterisk

Como se ha visto en el punto anterior, al hacer un *make config* en el directorio de fuentes de Asterisk, se instalará el script de inicio en */etc/init.d/* de manera que Asterisk arranque automáticamente al iniciar el sistema operativo. Ahora vamos a ver cómo arrancar manualmente Asterisk, y cómo conectarnos a una instancia de Asterisk que ya se encuentra en ejecución.

Arrancar Asterisk

Para arrancar Asterisk basta con ejecutar lo siguiente como root:

```
# asterisk
```

De esta manera, Asterisk arrancará en segundo plano, es decir, no se mostrará ningún mensaje al usuario, y éste podrá seguir trabajando tranquilamente, mientras Asterisk está funcionando.

Si, por el contrario, queremos arrancar Asterisk en primer plano, utilizaremos el modificador -c y se nos mostrará el CLI (*Command Line Interpreter*):

```
# asterisk -c
CLI>
```

Si hemos arrancado Asterisk en primer plano, la única manera de salir del CLI es parando Asterisk, ejecutando *stop now* desde el CLI.

Conectarse a una instancia arrancada

Como se ha visto arriba, si se arranca con -c, Asterisk arrancará en segundo plano. Para conectarse al CLI de esa instancia de Asterisk, utilice el modificador -r:

```
# asterisk -r
Connected to Asterisk 1.4.18 currently running on asterisk14
(pid = 3231)
asterisk14*CLI>
```

Para salir del CLI si nos hemos conectado con -r, no es necesario ejecutar *stop now*, basta con hacer *exit*, y saldremos del CLI, pero sin parar el servicio Asterisk.

Si nos intentamos conectar a una instancia arrancada de Asterisk, pero no habíamos arrancado Asterisk previamente, se mostrará el siguiente mensaje:

```
Unable to connect to remote asterisk (does
/var/run/asterisk.ctl exist?)
```

Añadir información de verbose y debug

Tanto cuando arrancamos Asterisk como si nos conectamos a una instancia con -r, podemos hacerlo con las opciones de verbose y debug, para que se nos muestre más información. El *verboose* aporta información detallada del

funcionamiento de Asterisk, mientras que el *debug* muestra información a más bajo nivel, que puede resultar de utilidad al diagnosticar fallos en Asterisk.

Para añadir verbose sólo hay que arrancar Asterisk con -v y para añadir debug con -d. Cuantas más uves y des se añadan más información se mostrará, por ejemplo:

```
# asterisk -vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvddddddddd
```

6 Consola de comandos (CLI)

Al arrancar Asterisk en primer plano (-c) nos encontraremos con su intérprete de comandos o CLI (*Command Line Interpreter*). A través de esta línea de comandos es posible obtener ayuda, habilitar información de debug, originar llamadas, etc.

A continuación se muestran los usos más frecuentes del CLI, aunque también se irán comentando más opciones en capítulos posteriores.

Mostrar información sobre las aplicaciones:

```
CLI> core show applications
CLI> core show application <nombre de la aplicación>
```

Mostrar información sobre las funciones:

```
CLI> core show functions
CLI> core show function <nombre de la función>
```

Cargar, descargar y mostrar los módulos de Asterisk:

```
CLI> module show
CLI> module load <nombre del módulo>
CLI> module unload <nombre del módulo>
```

Mostrar los usuarios SIP/IAX:

```
CLI> sip show peers
CLI> iax2 show peers
```

Mostrar los canales activos:

```
CLI> core show channels
CLI> sip show channels
CLI> iax2 show channels
CLI> zap show channels
```

Parar y reiniciar Asterisk:

```
CLI> stop now
CLI> stop when convenient
```

```
CLI> stop gracefully
CLI> restart now
CLI> restart when convenient
CLI> restart gracefully
```

7 Configuración básica

A continuación se van a mostrar ejemplos de configuración de los distintos tipos de canales que Asterisk es capaz de gestionar. En concreto, se examinará la configuración de los dispositivos SIP, IAX y ZAP, así como un dialplan básico.

Todos estos conceptos, serán ampliados más tarde en el *Capítulo 4. Lógica de marcado o dialplan.* Pero al finalizar este capítulo el lector ya será capaz de realizar llamadas simples entre extensiones de distintas tecnologías.

7.1 CANALES SIP

7.1.1 Protocolo SIP

El protocolo SIP (*Session Initiation Protocol*) puede ser considerado el estándar de facto para la VoIP. La actual versión, SIP v2, fue publicada como el RFC3261 en 1996. Desde entonces, su crecimiento ha sido inmenso, dado que el protocolo se diseñó de manera que fuera extensible y modular, manteniendo un núcleo funcional pequeño.

SIP es un protocolo basado en texto, fuertemente inspirado en HTML y el paradigma IP. Al estar basado en texto y ser similar a HTML, es legible por el ser humano, y es más fácil de diagnosticar.

Como su nombre indica (Session Initiation Protocol) SIP sólo sirve para gestionar la sesión del usuario. SIP es un protocolo de nivel de aplicación, que transporta únicamente señalización, y sólo sirve para el establecimiento, mantenimiento y cierre de sesión, en este caso de comunicaciones de audio. Es importante tener claro que SIP no transporta audio.

Bajo estas líneas se muestra lo que se conoce como el "SIP trapezoid", indicando que SIP sólo transporta señalización, y que el flujo multimedia es transportado usando RTP (*Realtime Transport Protocol*) no teniendo que coincidir necesariamente con el camino seguido por la señalización.

Gracias al protocolo SIP, los terminales se registran en el servidor, indicándole su IP, para que el servidor sea capaz de enviarles llamadas.

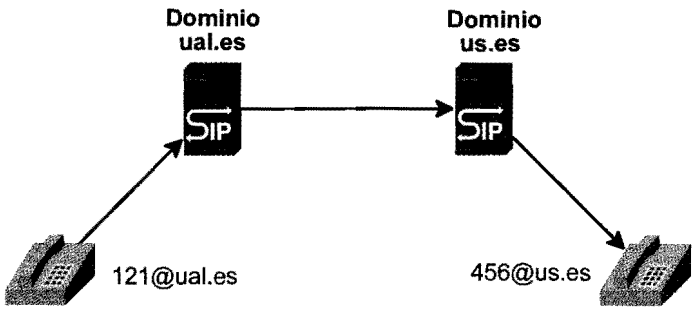


Figura 3-2. SIP trapezoide

7.1.2 Configuración de canales SIP

La configuración de dispositivos SIP se realiza en el fichero *sip.conf*, y al igual que en casi todos los ficheros de configuración, existe una sección que se aplica a todos los dispositivos definidos, la sección [general]:

```
[general]
parametro1=valor1
parametro2=valor2
[bob]
parametro3=valor3
[alice]
parametro4=valor4
```

Como se puede observar en el ejemplo, todos los dispositivos tendrán fijados los parámetros 1 y 2, el usuario Bob tendrá fijado el parámetro 3, y Alice el parámetro 4.

A la hora de definir usuarios SIP, lo más importante es definir el tipo de usuario. Asterisk define 3 tipos de usuarios:

- **peer.** Los peers son los usuarios a los que Asterisk manda llamadas, es decir, Asterisk llama A un peer.
- **user.** Los users son los usuarios de los que Asterisk recibe llamadas, es decir, Asterisk recibe llamadas DE un user.
- **friend.** Los friends son la agrupación de los 2 conceptos anteriores, es decir, un friend, es un peer y un user a la vez.

Realmente, el concepto de *user* no tiene demasiado sentido, ya que lo más habitual es utilizar Asterisk para realizar llamadas, por lo que utilizaremos el tipo *friend* para los teléfonos y el tipo *peer* para los proveedores de VoIP.

Veamos un ejemplo más completo de configuración de un usuario SIP:

```
[bob]
type=friend
secret=1234
context=desde-usuarios
callerid=Bon <2100>
host=dynamic
```

Como se puede apreciar en el ejemplo, hemos definido a Bob como un *friend*, pero hemos añadido varios parámetros más:

- **secret.** Indica la contraseña que se utilizará para la autenticación.
- **context.** Indica el contexto que se aplicará a este usuario. El concepto de contexto será explicado más detalladamente en el siguiente capítulo.
- **callerid.** Fija el identificador del llamante para el usuario definido, es decir, cuando bob llame a alguien, este verá "Bob <2100>" en la pantalla de su terminal.
- **host.** Indica la IP del usuario. Lo habitual es poner 'dynamic', ya que así requerimos que el usuario se registre, obteniendo así su IP.

Cuando realizamos un cambio en el fichero *sip.conf*, es necesario recargar la configuración SIP de Asterisk, para que los cambios tengan efecto. Para ello, ejecutamos *sip reload* en CLI:

```
PBX*CLI> sip reload
 Reloading SIP>
  == Parsing '/etc/asterisk/sip.conf': Found
  == Parsing '/etc/asterisk/sip_notify.conf': Found
PBX*CLI>
```

Para consultar los usuarios presentes en el sistema, así como su estado, podemos ejecutar *sip show peers* y *sip show users* desde el CLI:

```
PBX*CLI> sip show peers
Name/username     Host            Dyn Nat ACL Port     Status
alice/alice       192.168.1.48     D          5060
Unmonitored
bob/bob           192.168.1.31     D          5060
Unmonitored
laura/laura       192.168.1.48     D          5060
Unmonitored
```

```
3 sip peers [Monitored: 1 online, 0 offline Unmonitored: 3
online, 0 offline]
PBX*CLI> sip show users
Username      Secret  Accountcode  Def.Context      ACL  NAT
alice         1234                 desde-usuarios   No
RFC3581
bob           1234                 desde-usuarios   No
RFC3581
laura         1234                 desde-usuarios   No
RFC3581
PBX*CLI>
```

Para probar el funcionamiento de la configuración anterior, crearemos un pequeño dialplan de ejemplo:

```
[general]

[desde-usuarios]
exten => 2000,1,Dial(SIP/alice)
exten => 2001,1,Dial(SIP/bob)
exten => 2002,1,Dial(SIP/laura)
```

Con este dialplan, podremos llamar a Alice marcando el 2000, a Bob marcando el 2001 y a Laura marcando el 2002.

Para probar lo anterior, podemos configurar un teléfono SIP o un softphone. En los siguientes ejemplos, se utiliza Zoiper, un softphone gratuito con soporte para SIP e IAX.

7.2 PROTOCOLO IAX

Además de soportar los protocolos más extendidos de VoIP (SIP, H.323...), Asterisk introdujo un protocolo propio, orientado inicialmente a la interconexión de centralitas Asterisk. Este protocolo se denominó IAX (*Inter-Asterisk eXchange protocol*). Actualmente se utiliza la versión 2 del protocolo (IAX2), estando la primera versión ya en desuso, por lo que al hablar de IAX, estaremos normalmente refiriéndonos a IAX2.

Las principales características de IAX son:

- Todo el tráfico de datos se realiza a través de un único puerto UDP. Esto incluye tanto la señalización como el tráfico de voz. En otros protocolos se usa un puerto distinto para la señalización y otro, negociado dinámicamente, para el tráfico de voz. Esto introduce problemas con los cortafuegos y con el NAT que, aunque se pueden resolver con distintos

mecanismos, no dejan de ser complicados. En IAX no nos encontramos con estos problemas.

- Un mismo flujo de datos puede contener información de varias conversaciones al mismo tiempo (trunking). Esto se consigue enviando en un solo paquete UDP información de señalización y de voz de una o más llamadas. Gracias a esto se minimiza el ancho de banda "desperdiciado" por las cabeceras de los paquetes UDP.

- Es un protocolo binario, a diferencia de SIP, por ejemplo, que es un protocolo basado en texto. De esta manera se consigue minimizar el tráfico de datos de señalización.

Pero no todo son ventajas:

- El tráfico siempre pasa por el servidor. Al ir señalización y voz por el mismo canal, forzosamente hay que pasar por el servidor. Con otros protocolos, como SIP, dos teléfonos pueden comunicarse enviando la señalización a través del servidor, pero el tráfico de voz puede ir directo de un teléfono al otro sin pasar por el servidor, lo que en determinados escenarios puede suponer un enorme ahorro de ancho de banda.

- No está muy extendido. Actualmente sólo Asterisk y unos pocos productos más soportan IAX2. Tampoco es normal que proveedores de telefonía IP nos permitan conectar con ellos a través de este protocolo.

Por otra parte, es interesante comentar que aunque el protocolo se diseñó inicialmente para interconectar dos centralitas Asterisk, también existen teléfonos IP y softphones que pueden conectar con un servidor Asterisk a través de este protocolo.

7.2.1 Configuración de Canales IAX

El fichero de configuración en el que definiremos los parámetros de los canales IAX2 en nuestro servidor Asterisk es *iax.conf*.

Este fichero tiene una estructura similar a la que hemos visto anteriormente en el *sip.conf*, para los canales SIP, con una sección *[general]* con la información de configuración que afectará a todos los canales y parámetros por defecto, y después distintas secciones para cada uno de los dispositivos con los que nuestro Asterisk se va a comunicar mediante IAX2.

Aquí también diferenciaremos entre equipos que van a realizar llamadas a través de Asterisk (*users*), equipos que van a recibir llamadas procedentes de Asterisk (*peers*) y equipos que van a hacer y recibir llamadas *(friends).*

7.2.2 Definición de extensiones IAX2

Para los más impacientes, aquí tenemos nuestro primer *iax.conf* con una configuración mínima para conectar dos extensiones. En la sección [general] definiremos el idioma en el que deseamos que se reproduzcan los mensajes del sistema para los usuarios IAX2. Seguidamente definimos dos usuarios (Juan y María).

```
[general]
language = es

[juan]
type = friend
host = dynamic
secret = passwordDeJuan
context = desde-usuarios
callerid = Juan <2003>

[maria]
type = friend
host = dynamic
secret = passwordDeMaria
context = desde-usuarios
callerid = Maria <2004>
```

Dentro del CLI de Asterisk forzamos la carga del módulo *chan_iax2.so* para que se lea de nuevo esta configuración:

```
PBX*CLI> module reload chan_iax2.so
```

Podemos comprobar que en Asterisk ya se han definido los dos usuarios, ejecutando *iax2 show peers* e *iax2 show users* en el CLI:

```
PBX*CLI> iax2 show users
Username          Secret                 Authen
Def.Context       A/C    Codec Pref
maria             passwordDeMaria        000000000000003  desde-
usuarios    No      Host
juan              passwordDeJuan         000000000000003  desde-
usuarios    No      Host
PBX*CLI>
PBX*CLI> iax2 show peers
Name/Username     Host                   Mask             Port
Status
```

```
maria                (Unspecified)    (D)  255.255.255.255  0
Unmonitored
juan                 (Unspecified)    (D)  255.255.255.255  0
Unmonitored
2 iax2 peers [0 online, 0 offline, 2 unmonitored]
```

Para probar estas cuentas podemos usar un softphone que soporte el protocolo IAX2. Zoiper (http://www.zoiper.com) es una buena opción para esto, ya que además de IAX2 soporta SIP, por lo que será el softphone que utilizaremos como ejemplo.

La configuración de una cuenta IAX2 en Zoiper es bastante simple. Basta con introducir el nombre/dirección IP del servidor Asterisk y nuestro usuario y contraseña. Después de esto ya podemos conectar con el servidor.

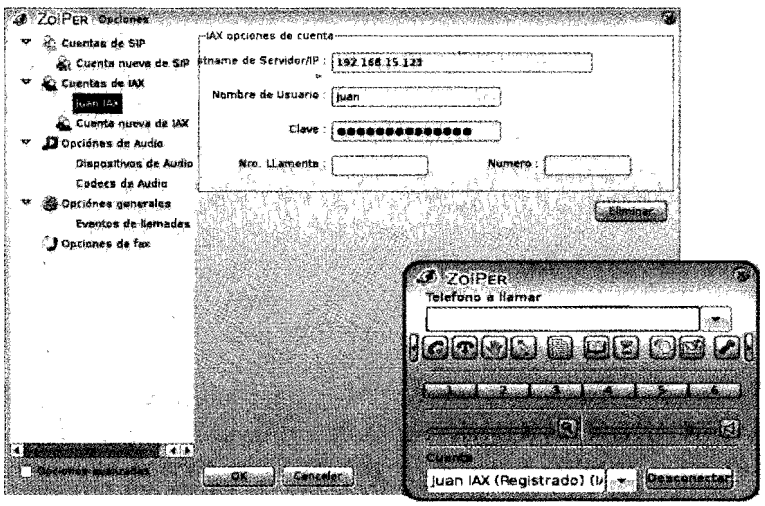


Figura 3-3. Configuración de ZoIPer

Para que nuestros usuarios Juan y María puedan llamarse, necesitaremos un pequeño dialplan. En el siguiente capítulo se profundizará sobre este tema, pero podemos empezar a hacer alguna prueba ampliando un poco el *extensions.conf* que habíamos generado para probar las extensiones SIP:

```
[general]

[desde-usuarios]
exten => 2000,1,Dial(SIP/alice)
exten => 2001,1,Dial(SIP/bob)
exten => 2002,1,Dial(SIP/laura)
exten => 2003,1,Dial(IAX2/juan,30)
```

```
exten => 2004,1,Dial(IAX2/maria,30)
```

Hemos añadido las extensiones 2003 y 2004. Al marcar la extensión 2003, llamaremos al usuario Juan, que usa el protocolo IAX2. Con la extensión 2004 llamaremos a María, que usa también IAX2.

Ahora que tenemos usuarios que usan terminales SIP y usuarios que usan terminales IAX2 podemos probar una de las capacidades más interesantes de Asterisk, y es que Bob puede llamar a Juan simplemente marcando su número de extensión. Bob, que usa un teléfono SIP, no sabe qué tecnología estará usando Juan, y no tiene que hacer nada especial para poder hablar con él, aunque el teléfono de Bob sólo soporte SIP y el teléfono de Juan solamente funcione con IAX2. Asterisk "*hace magia*" con los canales y realiza la traducción de protocolos de forma transparente. En la figura 3-4 se muestra un esquema básico de comunicación entre un usuario SIP y un usuario IAX2.

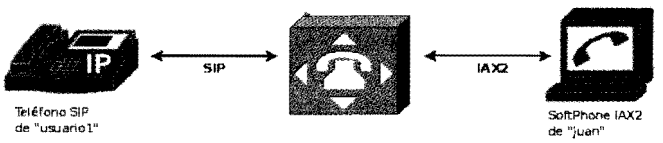


Figura 3-4. Llamada entre un usuario SIP y un usuario IAX2

7.2.3 Interconexión de dos Asterisk mediante IAX2

Acabamos de comprobar lo sencillo que es definir extensiones IAX2 para usar con Asterisk. Sin embargo, donde de verdad podemos obtener más ventajas usando el protocolo IAX2 es en la interconexión de dos centralitas Asterisk, de forma que los usuarios de una y otra instalación puedan hablar entre ellos, o que podamos rutar las llamadas al exterior a través de la instalación por la que resulten más económicas.

Imaginemos el siguiente escenario: una empresa tiene su sede central en Madrid y una sucursal en Caracas. En ambas sucursales disponen de un servidor Asterisk.

Enlazar estas dos centralitas a través de IP nos va a dar la posibilidad de que los usuarios de las dos sucursales puedan llamarse entre ellos y que estas llamadas no pasen a través de la red telefónica, lo que supone un gran ahorro para la empresa. También va a permitir que las llamadas que se realicen desde Caracas a clientes o proveedores de España sean enviadas a través de IP hasta la centralita de

Madrid, y desde allí salgan a la red telefónica, con lo que la llamada no sería facturada como una llamada internacional.

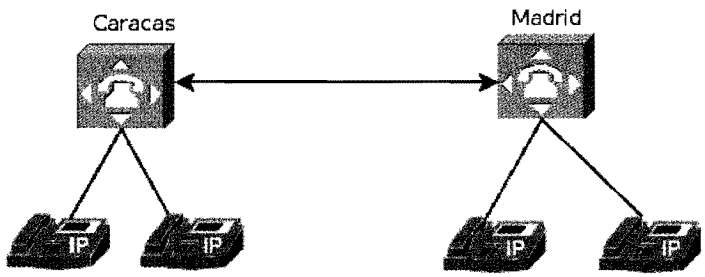


Figura 3-5. Interconexión de dos Asterisk mediante IAX2

Pero además de los beneficios económicos, la interconexión de centralitas ofrece grandes beneficios operativos: un usuario de Madrid que quiera hablar con un usuario de Caracas podría hacerlo simplemente marcando su número de extensión, sin tener que pasar por la operadora. Los usuarios que viajan entre las dos sucursales pueden recibir llamadas o acceder a servicios como su buzón de voz desde cualquier terminal de las dos instalaciones, etc.

El esquema que vamos a seguir va a ser la creación de un "*user*" y un "*peer*" en cada servidor Asterisk, para gestionar las llamadas entrantes (del "*user*") y salientes (hacia el "*peer*") a la otra sucursal. Vamos a suponer que el servidor de Madrid dispone de una dirección IP fija, mientras que la conexión a Internet de la oficina de Caracas tiene dirección IP dinámica.

También vamos a suponer que en Madrid las extensiones tienen una numeración del tipo: 1xxx (cuatro cifras y empiezan por 1), mientras que en Caracas las extensiones son de tres cifras y empiezan por 2 (2xx).

Configuración del servidor "Madrid":

```
iax.conf:

[general]
language = es

[caracas]
type = peer
host = dynamic
secret = passwordSecreta
trunk = yes
disallow = all
```

```
allow = gsm

[caracas]
type = user
host = dynamic
secret = passwordSecreta
context = desde-caracas
trunk = yes
disallow = all
allow = gsm
```

Como comentábamos, hemos definido un *user* y un *peer* para la sucursal de Caracas. En ambos hemos incluido el parámetro *host=dynamic* para indicar que la dirección IP de Caracas no es conocida.

También se ha definido el parámetro *trunk = yes*, para aprovechar la posibilidad de agregar información de varias llamadas en un mismo paquete UDP, lo que conllevará un ahorro de ancho de banda cuando haya varias llamadas concurrentes[1].

También para minimizar el ancho de banda consumido por las conversaciones hemos deshabilitado el uso de cualquier otro codec de audio *(disallow=all)* y habilitado únicamente el codec "GSM" (*allow=gsm*).

En la configuración del "*user*" Caracas, hemos indicado que el contexto en el que entrarán las llamadas provenientes de esta sucursal será "desde-caracas". En la configuración del dialplan ("*extensions.conf*") de Madrid debemos crear un contexto llamado "desde-caracas" para gestionar estas llamadas:

```
extensions.conf:

[general]

[desde-caracas] ; Llamadas provenientes de la sucursal de
Caracas
include => a-extensiones ; pueden llamar a extensiones locales

[a-caracas]  ; Llamadas con destino a extensiones de Caracas
```

---

[1] Para activar el "trunking" en IAX2 necesitamos disponer de una fuente de tiempo proporcionada por el driver de Zaptel. Si no disponemos de tarjetas de telefonía que usen este driver, será necesario cargar al menos el módulo "ztdummy", que proporcionará esta fuente de tiempo.

```
exten => 3XX,1,Dial(IAX2/caracas/${EXTEN},30)
exten => 3XX,n,Hangup

[a-extensiones] ; llamadas a extensiones locales
exten => 2XXX,1,Dial(SIP/${EXTEN},20)
exten => 2XXX,n,Hangup

[desde-extensiones] ; llamadas provenientes de extensiones
locales
include => a-extensiones ; permitimos que llamen a otras
extensiones
include => a-madrid      ; permitimos llamadas a extensiones
de Caracas
```

Configuración del servidor "Caracas":

```
iax.conf:

[general]
language = es
register => caracas:passwordSecreta@ip_servidor_madrid

[madrid]
type = friend
host = ip_servidor_madrid
secret = passwordSecreta
context = desde-madrid
trunk = yes
disallow = all
allow = gsm
```

La primera diferencia que encontramos con respecto a la configuración del servidor de Madrid es la línea:

```
register => caracas:passwordSecreta@ip_servidor_madrid
```

La finalidad de esta línea es que el servidor de Caracas se registre en el servidor de Madrid, identificándose como usuario "*caracas*" y la contraseña "*passwordSecreta*". De esta manera el servidor de Madrid podrá conocer la dirección IP que tenga asignada en ese momento y así podrá enviarle llamadas.

En este caso hemos optado por definir al servidor de Madrid como "*friend*", en vez de separar la configuración de "*user*" y "peer". El resultado es el mismo.

```
extensions.conf:

[general]

[a-madrid]
```

```
exten => 2XXX,1,Dial(IAX2/madrid/${EXTEN},30)
exten => 2XXX,n,Hangup

[desde-madrid] ; llamadas que llegan de Madrid
include => a-extensiones ; permitimos que llamen a extensiones
locales

[a-extensiones]  ; llamadas con destino a extensiones locales
exten => 3XX,1,Dial(SIP/${EXTEN},20)
exten => 3XX,n,Hangup

[de-extensiones] ; llamadas provenientes de extensiones
locales
include => a-extensiones ; permitimos que llamen a otras
extensiones
include => a-madrid      ; permitimos llamadas a extensiones
de Madrid
```

Una vez configurados los dos servidores, los usuarios de ambas sucursales podrán hablar entre sí, marcando simplemente el número de extensión de la otra persona.

7.2.4 Aumentando la seguridad

En el ejemplo anterior hemos utilizado contraseñas para que un servidor se identifique ante el otro. Para aumentar la seguridad, Asterisk admite el uso de claves RSA para identificar a los sistemas.

Para usar este mecanismo de identificación, debemos crear un par de claves (clave pública y clave privada) mediante el comando *ast-genkey*. Imaginemos que el servidor "Caracas" va a usar un par de claves RSA para identificarse ante el servidor "Madrid":

```
caracas# astgenkey -n caracas
```

Este comando va a generar un par de claves (*caracas.key* y *caracas.pub*) que debemos mover al directorio de claves de Asterisk (normalmente */var/lib/asterisk/keys*). Las claves no estarán protegidas por contraseña (para que el servidor pueda cargarlas al reiniciarse). La clave privada (.key) debe permanecer secreta, mientras que la clave pública (.pub) debemos copiarla a todos aquellos equipos con los que queramos usar esta identificación. En nuestro caso, copiaremos el archivo *caracas.pub* al directorio de claves del servidor "Madrid".

Ahora es necesario hacer algunos cambios en el *iax.conf* de ambos servidores. En primer lugar, el servidor "Caracas" debe usar estas claves tanto para registrarse ante el servidor Madrid como para identificarse al enviarle las llamadas:

```
iax.conf del servidor "Caracas":

[general]

; sustituimos la contraseña por el nombre del fichero de clave
register => caracas:[caracas]@192.168.15.123

[madrid]
type = friend
host = 192.168.15.123
secret = passwordSecreta
context = desde-madrid
disallow = all
allow = gsm
outkey = caracas  ; nombre de la clave a usar para enviar
llamadas
                  ; a madrid
```

Y por su parte, el servidor "Madrid" debe reconocer que este par de claves pertenece al servidor Caracas:

```
iax.conf del servidor "Madrid":

[general]

[caracas]
type = peer
host = dynamic
secret = passwordSecreta
trunk = yes
disallow = all
allow = gsm

[caracas]
type = user
host = dynamic
auth = rsa
inkeys = caracas  ; indicamos el nombre del fichero que
contiene
                  ; la clave publica del servidor remoto
context = desde-caracas
trunk = yes
```

```
disallow = all
allow = gsm
```

Es importante aclarar que este par de claves RSA sólo se utiliza para identificar a los servidores. El tráfico de voz no circulará cifrado aunque se usen claves RSA para la identificación.

7.3 CANALES ZAP

Aunque Asterisk sea un software capaz de actuar como centralita IP, no podemos olvidarnos de que la telefonía tradicional sigue existiendo y tenemos que interactuar con ella.

Es muy probable que Asterisk no hubiera llegado donde ha llegado de no haberse aliado con Jim Dixon y su proyecto Zapata Telephony, más conocido como Zaptel. Este proyecto benefició a ambos, ya que Asterisk no disponía de conectividad con las líneas analógicas/digitales, y Zaptel carecía del software necesario para funcionar.

7.3.1 Canales analógicos

En el *Capítulo 1. La telefonía tradicional,* se ha hecho una pequeña introducción al funcionamiento de la telefonía analógica, aunque queda fuera del alcance de este libro, pero para poder configurar las tarjetas Zaptel correctamente con Asterisk hay ciertos conceptos básicos que hay que tener en cuenta. Recordemos:

- Puerto FXO (Foreign eXchange Office): se trata del puerto que recibe el tono, es decir, se conecta a central (la línea telefónica). Utiliza señalización FXS.

- Puerto FXS (Foreign eXchange Station): este puerto es el que genera el tono de marcado, el que se conecta al terminal (teléfono). Utiliza señalización FXS.

***Nota:** si se conecta una línea telefónica a un FXS podría dañarse el hardware, ya que ambos extremos generan voltaje.*

Tal y como muestra la figura 3-6, a nivel físico los módulos FXS se pueden identificar por su color verde, y los módulos FXO son rojos.

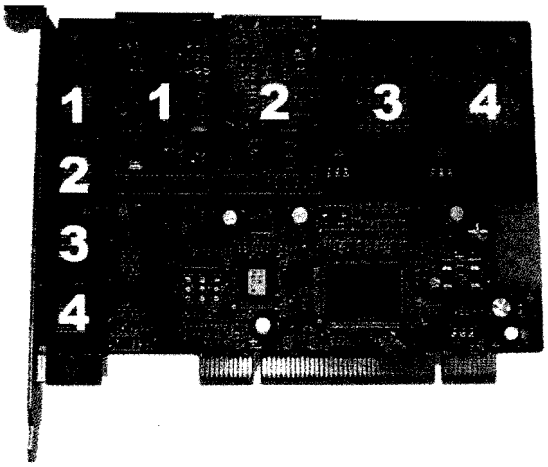


Figura 3-6. Tarjeta Zaptel

Configuración de un canal FXO

La configuración de los dispositivos Zaptel se realiza en dos pasos, por un lado se configura el subsistema Zaptel, y por otro lado Zapata, el canal de Asterisk para Zaptel.

Configuración del hardware Zaptel

Para configurar Zaptel es necesario editar el fichero */etc/zaptel.conf* e indicar el tipo de señalización y la zona geográfica.

A continuación se muestra un ejemplo de configuración para un FXO:

```
fxsks=2
loadzone=es
defaultzone=es
```

La primera línea indica el tipo de señalización que será utilizada, en este caso para el canal 2.

Señalización analógica

En la telefonía analógica, existen tres tipos de señalización, que se utiliza para solicitar tono de marcado y así poder establecer llamadas:

- **GroundStart (gs).** En este tipo de señalización el tono se solicita conectando a una toma de tierra uno de los polos. Se empezó a utilizar para evitar la situación en que ambos extremos intentan establecer una llamada casi al mismo tiempo, aunque actualmente no se utiliza mucho.

- **LoopStart (ls).** El tono de marcado se obtiene cerrando el circuito DC, y esto continuará así hasta que se cuelgue la línea. Este tipo de señalización es el más utilizado en Norte América.

- **KewlStart (ks).** Es el sistema más utilizado actualmente y es prácticamente igual que el LoopStart, pero con supervisión de desconexión remota. En un circuito no supervisado, normalmente la detección de desconexión no es un problema, pero en aplicaciones como el buzón de voz, si el circuito no es supervisado, no es posible detectar nada, podría grabarse silencio.

El parámetro *loadzone* permite configurar el canal para que utilice las indicaciones de tonos como el de marcado, ocupado, etc. para el país indicado. En caso de no indicar el loadzone para algún canal, se aplicará el valor del parámetro *loadzone=us*.

Configuración de Zapata

La configuración del canal de Asterisk para manejar el hardware Zaptel se realiza en el fichero *zapata.conf*, que, por defecto, se encuentra situado en */etc/asterisk*. En el fichero *zapata.conf*, además, se configuran diversos parámetros como el cancelador de eco, la llamada en espera, etc.

A continuación se muestra un ejemplo de configuración del fichero *zapata.conf*:

```
[trunkgroups]

[channels]
usecallerid=yes
hidecallerid=no
callwaiting=yes
threewaycalling=yes
transfer=yes
echocancel=yes
context=entrantes-analogicas
signalling=fxs ks
channel => 2
```

La configuración de los canales Zaptel se realiza dentro de la sección [channels]. El fichero *zapata.conf* es ligeramente diferente a los demás, ya que la configuración se aplica hasta que se encuentra la opción *channel=>*, luego es posible variarla para la configuración de otro canal. Veámoslo con un ejemplo:

```
opcion1=valor1
channel => 3
opcion1=valor2
channel => 4
opcion2=valor3
channel = > 5
```

Para el canal 3, estará definida la opción 1, con el valor 1. Para el canal 4, estará definida la opción 1, pero esta vez con el valor 2. Y para finalizar, el canal 5 tendrá definidas las opciones 1 y 2, con los valores 2 y 3 respectivamente.

Analicemos los parámetros del ejemplo anterior:

- **Usecallerid.** Indica si se utilizará o no el identificador de llamada.
- **Hidecallerid.** Indica si debe ocultarse el identificador de llamadas.
- **Callwaiting.** Habilita o deshabilita la llamada en espera.
- **Threewaycaling.** Permite habilitar/deshabilitar el servicio de llamada a tres.
- **Transfer.** Habilita o deshabilita la posibilidad de realizar transferencias.
- **Echocancel.** Permite habilitar la cancelación de eco.
- **Context.** Indica el contexto donde se enviarán las llamadas entrantes por el canal configurado.
- **Signalling.** Indica el tipo de señalización. En este caso es fxs_ks, ya que al ser una FXO, utiliza señalización FXS, y de tipo KewlStart (ks) que es el más común y el utilizado en España.
- **Channel.** Indica los canales a los que afectará la configuración anterior, en este caso el 2.

Configuración de un canal FXS

Como se ha comentado anteriormente, los canales FXS nos van a permitir conectar teléfonos analógicos, para poder usarlos con Asterisk. La configuración es muy similar a la de las interfaces FXO, como se podrá apreciar por los ejemplos mostrados.

Partiendo de la configuración anterior, vamos a añadir lo necesario para que el módulo FXS funcione con Asterisk.

Configuración del hardware Zaptel:

```
fxoks=1
fxsks=2
loadzone=es
defaultzone=es
```

El único cambio introducido es la primera línea, indicando que el canal 1 será configurado con señalización FXO de tipo *KewlStart*, recordemos que la señalización FXO se debe a que la interfaz es una FXS.

Configuración de Zapata

Para poner en funcionamiento la FXS tendremos que añadir lo siguiente en el fichero *zapata.conf* que ya teníamos en el ejemplo anterior:

```
immediate=no
context=desde-usuarios
signalling=fxo_ks
channel => 1
```

Las FXS tienen dos modos de funcionamiento: por un lado está el modo habitual, en el que el usuario descuelga y espera escuchar un tono de invitación a marcar, tras lo cual se cursará la llamada. El otro modo de funcionamiento es conocido como 'el teléfono rojo', y consiste en la marcación automática de un número, o la realización de diversas acciones justo al descolgar el teléfono, sin que el usuario tenga que marcar nada.

El modo de funcionamiento se configura mediante la opción *immediate*. Si está a 'no', el funcionamiento es el habitual de un teléfono, mientras que si lo ponemos a 'yes', nada más descolgar se ejecutará la extensión 's' del contexto definido, posibilitando así la creación de funcionalidades como el *teléfono rojo*.

El resto de opciones mostradas tienen el mismo significado que en la configuración de la interfaz FXO:

- **Context.** Indica el contexto desde el que se realizarán las llamadas.
- **Signalling.** Indica el tipo de señalización de la interfaz, en este caso, por ser una FXS le corresponde la señalización FXO.
- **Channel.** Indica el canal que se está configurando.

Dialplan para probar la configuración de Zaptel

Aunque se realiza un análisis más exhaustivo del diaplan en el *Capítulo 4. Lógica de marcado o dialplan,* se comenta en detalle en el siguiente capítulo, a continuación se muestra un ejemplo funcional para probar que la configuración ha sido la correcta:

```
[entrantes-analogicas]
exten => s,1,Noop(Llamada entrante por la línea analógica)
exten => s,n,Dial(SIP/alice)

[desde-usuarios]
exten => 2000,1,Dial(SIP/alice)
exten => 2001,1,Dial(SIP/bob)
exten => 2002,1,Dial(SIP/laura)
exten => 2003,1,Dial(ZAP/1)
```

Con el dialplan del ejemplo, al recibir una llamada por la línea analógica, esta será enviada a la extensión de Alice, y será posible llamar al terminal analógico marcando el número 2003.

7.3.2 Canales digitales

Los detalles del funcionamiento de la telefonía digital quedan fuera del alcance del libro, pero, al igual que en la telefonía analógica, es necesario tener claros algunos conceptos (consulte el *Capítulo 1. La telefonía tradicional* para más información).

Actualmente Zaptel sólo soporta accesos primarios (ISDN PRI), por lo que en este apartado trataremos exclusivamente con conexiones T1/E1.

Las diferencias entre los canales T1 y E1 son las siguientes:

- T1 tiene un ancho de banda de 1544Mbps, frente a los 2048Mbps de E1.
- T1 dispone de 24 canales, y E1 de 32.

- Utilizando señalización ISDN, T1 tiene 23 canales de voz, y E1 30.
- T1 puede tener muchos tipos de señalización ISDN, mientras que en E1 se utiliza EuroISDN únicamente.
- T1 se utiliza sobre todo en los EEUU, Canadá y Japón (J1), mientras que E1 se utiliza a nivel mundial.

Sincronización

Los circuitos T1/E1 necesitan una fuente de reloj, para poder mantener la sincronización. Este es un factor a tener en cuenta a la hora de configurar nuestra tarjeta Zaptel, ya que en función de a dónde conectemos el enlace T1/E1 tendremos que configurar la sincronización de diferente manera:

- Si enlazamos con un proveedor de telefonía, el proveedor nos dará la señal de reloj.
- Si enlazamos con otra PBX mediante un cable cruzado, tendremos que generar nosotros la señal de reloj.

Segmentación

En enlaces T1/E1, el ancho de banda total disponible se divide en distintos *time-slots* (canales) que serán utilizados para transmitir voz o datos. Dependiendo del tipo de enlace (T1 o E1) disponemos de distintas formas de segmentación:

- T1
  - D4
  - ESF (Extended Super Frame)
- E1
  - CAS (Channel Associated Signalling)
  - CCS (Common Channel Signalling)

El tipo de señalización a utilizar lo determina el operador. En España, los enlaces primarios son de tipo E1, con señalización CCS.

Codificación

La codificación indica cómo se envían los datos a través del enlace y también se utiliza para mantener la sincronía cuando los canales no se encuentran en uso. Tenemos 3 tipos de codificación disponibles:

- High Density Bipolar 3 (HDB3)
- Alternate Mark Inversion (AMI)
- Bipolar with 8 Zeros Substitution (B8ZS)

Al igual que la señalización, la codificación a utilizar en el enlace vendrá determinada por el operador. En España, la codificación utilizada es hdb3.

Configuración de Zaptel

Para configurar una tarjeta Zaptel de primarios, es necesario definir un "*span*" por cada uno de los primarios en el fichero *zaptel.conf*, de la siguiente manera:

```
span=número,timing,LBO,señalización,codificación
```

A la hora de configurar el *timing*, indicaremos un 1 si Asterisk va a recibir el timing del proveedor, o un 0 si Asterisk va a PROVEER el *timing* en el *span* definido.

El último parámetro que no se ha comentado hasta ahora es el LBO (*Line Build Out*), que sirve para compensar la longitud del cable de conexión del enlace. Lo habitual es fijar este valor a 0, aunque si la distancia es elevada, podemos ir aumentando su valor.

Teniendo en cuenta lo anterior, la configuración típica de una línea de primario en España sería:

```
span=1,1,0,ccs,hdb3,crc4
```

Nótese que hemos añadido la opción crc4 al final. Esto indica que se realizarán comprobaciones CRC, y para saber si debemos incluirlo o no tendremos que hablar con el operador.

Una vez hemos configurado el *span* tenemos que configurar los canales del enlace, que para un E1 serán 30 canales de voz (canales B) y un canal de datos (canal D). Una vez añadido esto tendremos esta configuración para el primario en el fichero *zaptel.conf*:

```
span=1,1,0,ccs,hdb3,crc4
bchan=1-15,17-31
dchan=16
```

Como se puede apreciar, se han definidos los canales B del 1 al 15 y del 17 al 31, dejando el canal 16 como canal D, que suele ser lo habitual.

Configuración de Zapata

Una vez tenemos configurada la tarjeta a nivel de Zaptel, tenemos que configurar el fichero *zapata.conf* (situado en */etc/asterisk*), para configurar cómo interactuará Asterisk con la tarjeta.

```
[trunkgroups]

[channels]
usecallerid=yes
hidecallerid=no
callwaiting=yes
threewaycalling=yes
transfer=yes
echocancel=yes
group=1
context=entrantes-pri
switchtype=euroisdn
signalling=pri cpe
channel => 1-15,17-31
```

Como se puede observar, la configuración no varía demasiado con respecto a la de las tarjetas analógicas, aunque hay algunas cosas que cambian:

- **switchtype**. Indica el protocolo ISDN utilizado. En España se utiliza el EuroISDN.
- **signalling.** En accesos primarios, la señalización puede ser *pri_cpe* o *pri_net*, dependiendo del rol que esté tomando Asterisk:
    - **pri_cpe.** Asterisk se conecta con la red del proveedor, Asterisk es un cliente (*cpe = customer premise equipment*).
    - **pri_net.** Asterisk actúa como si fuera un operador (network). Utilizaremos este tipo de señalización cuando nos conectemos con otra PBX mediante un cable cruzado.

Dialplan para probar la configuración Zaptel:

```
[entrantes-pri]
exten =>  9XXXXXXXX,1,Noop(Llamada entrante al DID: ${EXTEN})
exten =>  9XXXXXXXX,n,Dial(SIP/alice)


[desde-usuarios]
exten => 2000,1,Dial(SIP/alice)
exten => 2001,1,Dial(SIP/bob)
exten => 2002,1,Dial(SIP/laura)
exten => _9XXXXXXXX,1,Dial(ZAP/g1/${EXTEN})
```

El dialplan para probar las conexiones de primario es muy similar al de las conexiones analógicas, pero tiene algunas diferencias significativas ya que las llamadas entrantes no entran a la extensión 's' del contexto indicado. En lugar de esto, entran con el DID completo, de manera que en dialplan de prueba planteado recogeríamos todos los DIDs y mandaríamos la llamada a Alice.

7.3.3 Grupos de canales en Zaptel

En Zaptel es posible definir grupos de canales, y así llamar a un grupo, en lugar de a un canal en concreto, y Zaptel se encargará de llamar a un canal concreto dentro del grupo siguiendo una de las estrategias definidas. Esto es particularmente útil en la configuración de primarios, ya que es posible que se pretenda que dos primarios sean enviados a un contexto, y un tercer primario sea enviado a otro contexto diferente.

Para definir un grupo basta con añadir *group=X* en el fichero *zapata.conf* a la hora de definir los canales. Si ponemos *group=1*, todos los canales definidos a partir de ese momento pertenecerán al grupo 1, hasta que introduzcamos otra directiva group. Veamos un ejemplo:

```
group=1
channel => 1-15,17-31
```

En este caso, tanto los canales del 1 al 15 como los del 17 al 31 pertenecen al grupo 1.

```
group=1
channel => 1-15
group=2
channel => 17-31
```

En este segundo ejemplo, por el contrario, los canales del 1 al 15 pertenecen al grupo 1, pero los del 17 al 31 pertenecen al grupo 2.

Para llamar a un grupo en lugar de hacer un Dial a ZAP/1, lo haríamos a ZAP/g1/ indicando que es el grupo 1, aunque hay varias opciones:

- **g.** Utiliza el canal de menor número disponible del grupo especificado.
- **G.** Utiliza el canal de mayor número disponible del grupo especificado.
- **r.** Sigue una estrategia Round-Robin[2] de menor a mayor, selecciona el primer canal disponible, haciéndolo de manera ascendente.
- **R.** Sigue una estrategia Round-Robin de mayor a menor, selecciona el primer canal disponible, haciéndolo de manera descendente.

7.3.4 Aplicando la configuración de Zaptel

Anteriormente, hemos visto cómo configurar las tarjetas Zaptel, editando el fichero *zaptel.conf*. Una vez editado el fichero es necesario aplicar la configuración que este contiene, y para ello utilizaremos el comando *ztcfg* de la siguiente manera:

```
ztcfg -vv
```

Tras ejecutarlo, el sistema indicará que la configuración de los canales se ha realizado, o error en caso de que algo sea incorrecto.

7.4 BUZONES DE VOZ

El buzón de voz es una de las funcionalidades que Asterisk incluye "de serie". El módulo de buzón de voz permite definir distintos buzones para cada usuario, así como buzones compartidos por varios usuarios (un buzón general para toda la empresa, o para cada departamento). Este módulo permite también cierto grado de personalización, pudiendo el usuario configurar los mensajes que se reproducirán cuando alguien desee dejar un mensaje en "mi buzón". Otra característica interesante es que el servidor nos enviará un correo electrónico avisándonos de que tenemos un mensaje nuevo. Además, en este mismo correo electrónico puede enviarse el mensaje como archivo adjunto.

---

[2] Round-Robin. Estrategia de marcado consistente en realizar el marcado de forma rotatoria en los canales ocupados.

La configuración de los buzones se realiza en el fichero *voicemail.conf*, que siguiendo con la estructura habitual de los ficheros de configuración de Asterisk contiene una sección *[general]* y distintas secciones para distintos contextos[3].

En la sección *[general]* podremos definir parámetros como el formato en el que se van a grabar los mensajes (wav, gsm...) o el formato de los correos electrónicos de aviso de nuevos mensajes. Para definir el formato de los correos, disponemos de las siguientes variables que serán sustituidas en el momento de generar el mensaje:

- **VM_NAME**. Nombre del propietario del buzón.
- **VM_DUR**. Duración del mensaje.
- **VM_MSGNUM**. Número de orden de este mensaje.
- **VM_MAILBOX**. Buzón de voz.
- **VM_CALLERID**. Nombre y número de la persona que ha llamado.
- **VM_CIDNUM**. Número de la persona que ha llamado.
- **VM_CIDNAME**. Nombre de la persona que ha llamado.
- **VM_DATE**. Fecha y hora del mensaje.

```
[general]

format = wav49|gsm

emailsubject=[PBX]: Tienes un nuevo mensaje en el buzón
${VM_MAILBOX}

emailbody=Estimado ${VM_NAME}:\n\n\tCon fecha ${VM_DATE} has
recibido una llamada de ${VM_CALLERID}, quien te ha dejado un
mensaje de ${VM_DUR} duración (es el número ${VM_MSGNUM})\n en
el buzón número ${VM_MAILBOX}.\n Recuerda que puedes oir los
```

[3] El uso de distintos contextos de buzón de voz puede resultar interesante en algunos casos, pero generalmente es suficiente con tener un único contexto "default" con todos los buzones.

```
mensajes de tu buzón de voz llamando al número *98.
Gracias!\n

emaildateformat=%A, %d-%m-%Y a las %H:%M:%S

attach=yes
```

Los buzones de voz se definirán dentro de un contexto. Estos contextos no tienen relación con los contextos definidos en el fichero *extensions.conf*, sino que son una forma de separar buzones.

El formato que usaremos para definir cada buzón será:

```
extension => contraseña, nombre usuario, email usuario,
busca_usuario, opciones
```

Normalmente definiremos solamente los tres primeros parámetros para cada buzón (contraseña, nombre y e-mail), pudiendo definir también para cada usuario una dirección de un "buscapersonas" a la que se le enviarían los mensajes y también opciones particulares para cada buzón que modificarían las opciones por defecto (número máximo de mensajes en el buzón, si se debe o no enviar el fichero adjunto con el mensaje, etc.).

```
[default]
1001 => 3342,Juan
Perez,juanperez@miempresa.com,,attach=yes|maxmsg=100
1002 => 8745,Maria,mariaperez@miempres.com

[departamentos]
1 => 111,Ventas,ventas@miempresa.com
2 => 222,Compras,compras@miempresa.com
```

En la definición de las cuentas de usuario en *sip.conf* y en *iax.conf* se puede hacer referencia a estos buzones, de manera que los usuarios recibirán también en su terminal una notificación de los mensajes que tienen en su buzón:

```
sip.conf:

[juan]
type=friend
mailbox=1001@default
...

[maria]
type=friend
mailbox=1002@default
...
```

Para poder probar la funcionalidad del buzón de voz, vamos a ampliar el dialplan que definimos anteriormente:

```
extensions.conf:

...
[a-extensiones]  ; llamadas con destino a extensiones locales
exten =>  2XXX,1,Dial(SIP/${EXTEN},20)
exten =>  2XXX,n,VoiceMail(${EXTEN})
exten =>  2XXX,n,Hangup

[servicios]
exten => *98,1,VoiceMailMain()

[desde-extensiones] ; llamadas provenientes de extensiones
locales
include => a-extensiones ; permitimos que llamen a otras
extensiones
include => servicios
...
```

Aunque estos ejemplos se verán con detalle en el *Capítulo 4. Lógica de marcado o dialplan*, lo que hemos conseguido con estos cambios es que al llamar a una extensión 2xxx, si no contesta o está ocupada o no conectada, nos saltará su buzón de voz para que podamos dejarle un mensaje. Por otra parte, los usuarios podrán llamar al número *98 para escuchar sus mensajes.

CAPÍTULO 4

LÓGICA DE MARCADO O DIALPLAN

Saúl Ibarra Corretgé y David Prieto Carrellán

1 Introducción

Asterisk es una "caja de herramientas" con la que podemos construir nuestros sistemas de telefonía. En realidad Asterisk es sólo eso, un conjunto de herramientas, ya que las funcionalidades que el sistema final tenga dependen de lo bien que se utilicen esas herramientas.

El dialplan es el corazón de Asterisk. Cada dígito que se marque en un terminal recorrerá el dialplan, buscando "qué hacer", por lo que de una manera básica, podríamos comparar el dialplan con una tabla de enrutado: el usuario marca un número, y el dialplan contiene las acciones a realizar para ese número que se ha marcado.

2 Contextos, extensiones y prioridades

El dialplan de Asterisk se encuentra en el fichero *extensions.conf* (aunque es posible dividirlo en varios ficheros, como se verá más adelante) y está compuesto por contextos, extensiones y prioridades.

Las extensiones son números (o caracteres alfanuméricos) que el usuario es capaz de marcar, y es importante no confundir esto con los terminales en sí, ya que esta asociación puede conducir a errores. Estas extensiones contienen acciones asociadas, que son ejecutadas de manera secuencial, por orden de prioridad.

Los contextos son agrupaciones lógicas de extensiones, y se utilizan para dividir el dialplan en diversos entes lógicos. Esta división es necesaria para disponer de un dialplan mantenible, escalable y con posibilidades de ofrecer diversos entornos de marcado aislados.

Al definir un usuario en el *sip.conf* o *iax.conf*, le asociamos un contexto, por lo que ese usuario sólo podrá marcar las extensiones incluidas en su contexto.

```
[prueba]
exten => 1234,1,Noop(Esto es una prueba)
exten => 1234,n,Noop(Esto es otra prueba)
```

En el ejemplo de arriba se muestra un sencillo contexto con una extensión con 2 prioridades. Siempre es necesaria la prioridad 1, pero para las siguientes prioridades se puede usar la 'n', que hará que se incremente automáticamente en una unidad, evitando así tener que hacerlo manualmente.

3 Sintaxis

La sintaxis para las extensiones de un contexto es la siguiente:

```
exten => numero de extensión, prioridad,
aplicación(argumentos)
```

Seguido de la palabra clave *exten*=> se expresa el número de extensión o el patrón de la misma. Las extensiones pueden ser expresadas como patrones, ya que resultaría demasiado engorroso escribir los número del 2000 al 2999, 2XXX es mucho más elegante.

A la hora de definir nuestras extensiones podemos hacerlo de una manera numérica o alfanumérica, como en los ejemplos vistos hasta el momento:

```
exten => 2000,1,Dial(SIP/bob)
exten => prueba,1,Dial(SIP/alice)
```

En lugar de expresar siempre las extensiones, podemos utilizar patrones y variables, para disponer de un dialplan más claro y mantenible. Los patrones comienzan con un guión bajo (_) y utilizan los siguientes caracteres especiales:

- **X**. Cualquier dígito del 0 al 9.
- **Z**. Cualquier dígito del 1 al 9.
- **N**. Cualquier dígito del 2 al 9.
- **[]**. Cualquier dígito que se encuentre entre los corchetes, por ejemplo [123] implica el 1, el 2 o el 3.
- **. (punto)**. Cualquier cosa, por ejemplo _*9*. implica cualquier número que empiece por 9, sin tener en cuenta el 9 en sí mismo.
- **!**. Carácter de desambigüación. Indica que el procesamiento ha de ser detenido tan pronto como se haya encontrado un patrón adecuado.

Utilizando los patrones es posible simplificar el dialplan y en la práctica se emplean patrones como los siguientes:

- **_[789]XXXXXXXX.** Números fijos nacionales (empiezan por 7, 8 ó 9 y tienen 9 dígitos).
- **_6XXXXXXXX.** Números de móvil (empiezan por 6 y tienen 9 dígitos).
- ...

4 Aplicaciones y funciones

Mientras la llamada va "caminando" por el dialplan, se van ejecutando diversas aplicaciones y funciones.

Se denominan aplicaciones aquellos módulos que realizan algún tipo de acción sobre algún canal, y funciones aquellos que realizan otro tipo de procesamiento que no afecte directamente a un canal.

Al marcar una extensión se ejecuta la aplicación asociada a la prioridad correspondiente, mientras que las funciones sólo pueden ser utilizadas dentro de las aplicaciones. Veamos un ejemplo:

```
exten => 2000,1,Dial(SIP/bob)
```

Al marcar la extensión 2000 se ejecutará la aplicación Dial, a la que indicamos que llame al usuario Bob, que es un usuario SIP. Indicamos primero la tecnología (SIP/) y después el nombre del usuario que hemos definido en el fichero correspondiente, en este caso el *sip.conf*.

Las funciones, en cambio, se ejecutan dentro las aplicaciones, por ejemplo:

```
exten => 1234,1,Noop(Me estas llamando desde el:
${CALLERID(num)})
```

En este caso se ejecuta la aplicación *Noop*, que únicamente imprime el mensaje que se le pasa como argumento por pantalla, y en su interior se ejecuta la función CALLERID, que devuelve el número del llamante.

5 Prioridades y etiquetas (labels)

Como hemos visto, al marcar una extensión se van ejecutando las aplicaciones correspondientes de manera secuencial, de acuerdo a su prioridad:

```
exten => 1234,1,Aplicación 1
exten => 1234,2,Aplicación 2
...
```

El uso de prioridades numéricas puede dificultar la modificación del dialplan a futuro, ya que si se desea insertar una acción en medio, sería necesario modificar la prioridad de todas las siguientes.

Para evitar esto podemos utilizar la prioridad "n", que simplemente indican la siguiente prioridad:

```
exten => 1234,1,Aplicación 1
exten => 1234,n,Aplicación 2
...
exten => 1234,n,Aplicación 13
```

Como se puede observar, de esta manera es mucho más sencillo insertar nuevas aplicaciones en medio, sin necesidad de preocuparnos de cambiar los números de prioridad.

Un problema que nos puede surgir a la hora de sustituir las prioridades numéricas por la prioridad "n" es la imposibilidad de realizar saltos a zonas concretas de dialplan con aplicaciones como *Goto*:

```
exten => 1234,1,Playback(tt-monkeys)
exten => 1234,2,Goto(5)
...
exten => 1234,5,Noop(Soy la prioridad 5!)
```

Al prescindir de las prioridades numéricas, no sabríamos a dónde queremos realizar el salto. Para solucionar esto, es necesario utilizar etiquetas. Las etiquetas nos sirven para marcar o etiquetar una prioridad de una extensión, y posteriormente referirnos a ella de una manera sencilla:

```
exten => 1234,1,Playback(tt-monkeys)
exten => 1234,n,Goto(empezar)
...
exten => 1234,n(empezar),Noop(Soy una prioridad etiquetada!)
```

En el ejemplo se ha utilizado la etiqueta "empezar", de manera que puede estar situada en cualquier punto de la extensión, y el salto seguirá funcionando.

6 Un dialplan sencillo

Cuando se ha comentado la configuración de dispositivos SIP e IAX2, se ha utilizado un pequeño dialplan de ejemplo del que no hemos hablado hasta ahora. A continuación se muestra un dialplan básico, para que el lector tenga una visión más global de lo que un dialplan representa, antes de comenzar a añadir más servicios que aumentarán su complejidad:

```
[a-extensiones]
exten => 2000,1,Dial(SIP/bob,60,Tt)
exten => 2001,1,Dial(SIP/alice,60,Tt)


[a-servicios]
exten => 1234,1,Playback(tt-monkeys)
exten => 1235,1,Dial(IAX2/guest@pbx.digium.com/s@default)

[desde-usuarios]
include => a-extensiones
include => a-servicios
```

En el ejemplo hemos definido tres contextos: el contexto *a-extensiones* incluye las extensiones 2000 y 2001, que hacen uso de la aplicación dial para llamar a Bob y a Alice respectivamente. El contexto *a-servicios* contiene 2 servicios: 1234, que reproduce una locución de ejemplo (el sonido de unos monos) y 1235, que realiza una llamada a Digium a través de IAX2.

El contexto *desde-usuarios* incluye los otros 2 contextos, y es el que asignaremos a las extensiones en el *sip.conf*, de manera que sean capaces de marcar todas las extensiones incluidas por este contexto. Podría haberse realizado todo en un solo contexto, pero eso derivaría en un dialplan muy poco mantenible, como se explica más adelante en este capítulo.

7 Buzón de voz

Hasta ahora hemos visto cómo un usuario puede llamar a otro, pero si la llamada no llega a establecerse porque la otra persona está ocupada o no contesta, no queda otro remedio que llamarle más tarde.

Para poder dejar un mensaje en un buzón de voz disponemos de la aplicación *Voicemail,* que ya nombramos en el *Capítulo 3. La revolución se llama Asterisk*, al explicar la configuración de los buzones de voz. Adaptemos nuestro dialplan para que la llamada salte al buzón de voz en caso de que no se llegue a establecer la comunicación:

```
[a-extensiones]
exten => 2000,1,Dial(SIP/alice)
exten => 2000,n,Voicemail(2000)

exten => 2001,1,Dial(SIP/bob)
exten => 2001,n,Voicemail(2001)

exten => 2002,1,Dial(SIP/laura)
exten => 2002,n,Voicemail(2002)

exten => 2003,1,Dial(IAX2/juan,30)
exten => 2003,n,Voicemail(2003)

exten => 2004,1,Dial(IAX2/maria,30)
exten => 2004,n,Voicemail(2004)
```

Con este dialplan, cuando llamemos a la extensión 2001, el sistema intentará llamar al usuario Bob mediante SIP durante 20 segundos, y si no es posible la comunicación, avanzará al siguiente paso del dialplan para la extensión 2001, en el que ejecutará la aplicación *Voicemail*, pasando como parámetro el número de la extensión como número del buzón (2001).

La aplicación Voicemail acepta una serie de parámetros, además del número del buzón en el que deseamos dejar el mensaje:

- **u.** Reproduce el mensaje de "no disponible".
- **b.** Reproduce el mensaje de "ocupado".
- **s.** Comienza la grabación del mensaje sin reproducir antes las instrucciones.

Para aplicar estos parámetros podemos hacer uso del valor de la variable *${DIALSTATUS}*, que nos indica la razón por la que falló la aplicación *Dial*:

```
[a-extensiones]
exten => 2000,1,Dial(SIP/alice,20)
exten => 2000,n,Goto(2000-${DIALSTATUS},1)

; si está ocupado, se reproduce el mensaje correspondiente
exten => 2000-BUSY,1,VoiceMail(2000,b)
exten => 2000-BUSY,n,Hangup

; si está no contesta, se reproduce el mensaje de "no
disponible"
exten => 2000-NOANSWER,1,VoiceMail(2000,u)
exten => 2000-NOANSWER,n,Hangup

; en cualquier otro caso, hacemos lo mismo que si no contesta.
exten => 2000-.,1,Goto(2000-NOANSWER)
```

Ahora, para que nuestros usuarios puedan escuchar los mensajes que reciben en su buzón de voz desde cualquier terminal, vamos a añadir una extensión especial al contexto de servicios. La aplicación que nos permite escuchar los mensajes de voz es *VoiceMailMain*, y recibe como parámetro el número del buzón al que queremos acceder.

Si no pasamos el número del buzón como parámetro, la aplicación nos pedirá que lo tecleemos. Esto puede ser interesante para que un usuario pueda consultar su buzón de voz desde cualquier terminal, y no sólo desde el suyo.

Por otra parte, si añadimos el parámetro "s" no se nos pedirá la contraseña para poder oir los mensajes. En el siguiente ejemplo vamos a configurar la extensión *98 para que los usuarios puedan escuchar sus mensajes desde cualquier terminal (el sistema les pedirá que marquen su número de extensión y su contraseña), y la extensión *97 para escuchar los mensajes del buzón de la extensión desde la que estamos llamando, sin necesidad de introducir contraseñas.

```
[servicios]
; Marcando *98 desde cualquier terminal accedemos al buzón de
voz
; nos pedirá el número del buzón y la contraseña
exten => *98,1,VoiceMailMain()
exten => *98,n,Hangup

; Marcando *97 escucho "mis" mensajes sin poner contraseña.
exten => *97,1,VoiceMailMain(${CALLERID(num)},s)
exten => *97,n,Hangup
```

8 Macros

En muchas ocasiones nos encontraremos con fragmentos del dialplan que son idénticos, y en los que solamente cambia un pequeño detalle. En el dialplan anterior, el siguiente código tendríamos que copiarlo para la extensión 2001, 2002, 2003, modificando en cada fragmento el protocolo a usar (SIP o IAX), nombre del usuario y el número de extensión. Esto, además de ser bastante trabajoso, complicará el día de mañana cualquier modificación.

```
[a-extensiones]
exten => 2000,1,Dial(SIP/alice,20)
exten => 2000,n,Goto(2000-${DIALSTATUS},1)

; si está ocupado, se reproduce el mensaje correspondiente
exten => 2000-BUSY,1,VoiceMail(2000,b)
exten => 2000-BUSY,n,Hangup

; si está no contesta, se reproduce el mensaje de "no
disponible"
exten => 2000-NOANSWER,1,VoiceMail(2000,u)
exten => 2000-NOANSWER,n,Hangup

; en cualquier otro caso, hacemos lo mismo que si no contesta.
exten => 2000-.,1,Goto(2000-NOANSWER)
```

Para solucionar este problema, podemos definir Macros, que son el equivalente a "funciones" o "subrutinas" de los lenguajes de programación. Al igual que las funciones, una macro recibirá unos parámetros que podremos utilizar dentro de su definición.

Las macros se definen en el fichero *extensions.conf*, como un contexto más, teniendo en cuenta una serie de condiciones:

- El nombre del "contexto" empezará por "macro-" para identificar que se trata de una Macro, y no de un contexto normal.

- Las macros siempre empezarán por la extensión "s". No podemos hacer un "goto" desde otra parte del Dialplan a un punto concreto de la macro, que no sea a su inicio.

- Dentro de la definición de la Macro, los parámetros recibidos son accesibles mediante *${ARG1}*, *${ARG2}*... Y la extensión desde la que se ha llamado a la macro la podemos obtener en *${MACRO_EXTEN}*.

Para ejecutar una macro, en el dialplan tenemos disponible la aplicación "Macro":

```
exten => 1234,1,Macro(nombre macro, parametro1, parametro2,
parametro3..)
```

Como la mejor forma de entender las cosas es mediante un ejemplo, vamos a convertir a una macro el código que teníamos para llamar a la extensión 2000, saltando al buzón de voz si está ocupado, etc. A esta macro la llamaremos "extensionVM":

```
[macro-extensionVM]
; esta macro recibirá dos parámetros
; el primero indica el dispositivo que hay que llamar
; el segundo es el número del buzón de voz al que desviar la
llamada

exten => s,1,NoOp(llamada para ${ARG1})
exten => s,n,Dial(${ARG1},20)
exten => s,n,Goto(s-${DIALSTATUS},1)

; si está ocupado, se reproduce el mensaje correspondiente
exten => s-BUSY,1,VoiceMail"(${ARG2},b)
exten => s-BUSY,n,Hangup

; si está no contesta, se reproduce el mensaje de "no
disponible"
exten => s-NOANSWER,1,VoiceMail(${ARG2},u)
exten => s-NOANSWER,n,Hangup

; en cualquier otro caso, hacemos lo mismo que si no contesta.
exten => s-.,1,Goto(s-NOANSWER,1)
```

Y adaptamos el contexto *[a-extensiones]* para que use esta macro para llamar a los usuarios:

```
[a-extensiones]
exten => 2000,1,Macro(extensionVM,SIP/alice,2000)
exten => 2001,1,Macro(extensionVM,SIP/bob,2001)
exten => 2002,1,Macro(extensionVM,SIP/laura,2002)
exten => 2003,1,Macro(extensionVM,IAX2/juan,2003)
exten => 2004,1,Macro(extensionVM,IAX2/maria,2004)
```

Como podemos ver, esto facilita muchísimo el mantenimiento del *dialplan*, y lo hace al mismo tiempo mucho más legible.

9 Guardando la información en la base de datos

Asterisk mantiene determinada información sobre su estado actual en una base de datos, conocida como *AstDB*[1]. La información que se almacena en esta base de datos incluye los datos de registro de los usuarios, estado de las colas, etc. Esto permite que el sistema pueda recuperar esta información en caso de tener que reiniciarse.

La información en la AstDB se organiza en familias y se identifican mediante una clave que será única dentro de la familia. Para cada familia y clave se puede almacenar un valor.

Desde el CLI disponemos de distintos comandos con los que podemos acceder a la información de la AstDB, pudiendo tanto leer como escribir en ella:

- **database show <familia> <clave>.** Muestra el contenido de la base de datos, pudiendo filtrar por familia y clave:

```
PBX*CLI> database show
/IAX/Registry/caracas                    :
192.168.15.209:4569:60
/IAX/Registry/juan                       :
192.168.15.2:4570:60
/SIP/Registry/bob                        :
    192.168.15.2:5060:3600:bob:sip:bob@192.168.15.2

PBX*CLI> database show iax
/IAX/Registry/caracas                    :
192.168.15.209:4569:60
/IAX/Registry/juan                       :
192.168.15.2:4570:60

PBX*CLI> database show iax registry/juan
/IAX/Registry/juan                       :
192.168.15.2:4570:60

PBX*CLI> database showkey iax/registry/juan
/IAX/Registry/juan                       :
192.168.15.2:4570:60
```

- **database del <familia> <clave>.** Permite borrar un valor de la base de datos.

---

[1] Internamente esta base de datos tiene el formato de Berkeley DB y se encuentra por defecto en el archivo /var/lib/asterisk/astdb.

- **database deltree <familia> <arbol-de-claves>.** Borra todo un árbol de claves de la base de datos (por ejemplo, puede borrar todo el árbol "Registry" de la familia "IAX").

- **database put <familia> <clave> <valor>.** Permite introducir información en la base de datos.

```
PBX*CLI> database put familia clave valor
Updated database successfully

PBX*CLI> database show
/IAX/Registry/caracas                              :
192.168.15.209:4569:60
/IAX/Registry/juan                                 :
192.168.15.2:4570:60
/SIP/Registry/bob                                  :
192.168.15.2:5060:3600:bob:sip:bob@192.168.15.2
/familia/clave                                     : valor
```

Leer la información que Asterisk guarda en la base de datos puede resultar interesante para determinadas aplicaciones, pero lo verdaderamente interesante de esta base de datos es que nosotros podemos también guardar y leer información propia, para utilizarla desde el *dialplan*. Para esto disponemos de una serie de funciones:

- Guardar un valor en la base de datos:

```
exten => s,1,Set( DB(familia/clave) = valor )
```

- Leer un valor de la base de datos:

```
exten => s,1,Set( var = ${DB(familia/clave)})
```

- Comprobar si existe un valor para una familia y clave:

```
exten => s,1,GotoIf($[${DB_EXISTS(familia/clave)} =
0]?s,llamar)
```

En caso de existir un valor para la familia/clave, *DB_EXISTS* guarda este valor en la variable *${DB_RESULT}*.

Para ver un ejemplo de uso de estas funciones, vamos a añadir a nuestro *dialplan* la posibilidad de que los usuarios activen un "no-molestar" en sus extensiones cuando no quieran recibir llamadas:

```
; Marcando *78 activamos el "no-molestar".
; guardamos un valor en la base de datos, bajo
DND/num extension
exten => *78,1,NoOp(activando No Molestar para
${CALLERID(num)})
exten => *78,n,Set(DB(DND/${CALLERID(num)})=1)
```

```
exten => *78,n,Playback(beep)
exten => *78,n,Hangup

; Marcando *79 lo desactivamos, borrando esa clave de la astdb
exten => *79,1,NoOp(desactivando No Molestar para
${CALLERID(num)})
exten => *79,n,DBDel(DND/${CALLERID(num)})
exten => *79,n,Playback(beep)
exten => *79,n,Hangup
```

Podemos probar a llamar al número *78 y después ejecutar *database show* en el CLI. Veremos que se ha guardado un registro en la base de datos, de la forma: *DND/num_extension*.

Ahora actualizaremos nuestra macro extensionVM para que tenga en cuenta si el usuario tiene activado el "no-molestar":

```
[macro-extensionVM]
; esta macro recibirá dos parámetros
; el primero indica el dispositivo que hay que llamar
; el segundo es el número del buzón de voz al que desviar la
llamada

exten => s,1,NoOp(llamada para ${ARG1})

; si NO tiene activado el no-molestar, saltamos a "llamar"
exten => s,n,GotoIf($[${DB EXISTS(DND/${MACRO EXTEN})} =
0]?s,llamar)
exten => s,n(DND),Playback(vm-extension)
exten => s,n,Playback(vm-isunavail)
exten => s,n,Hangup
exten => s,n(llamar),Dial(${ARG1},20)
exten => s,n,Goto(s-${DIALSTATUS},1)

; si está ocupado, se reproduce el mensaje correspondiente
exten => s-BUSY,1,VoiceMail(${ARG2},b)
exten => s-BUSY,n,Hangup

; si está no contesta, se reproduce el mensaje de "no
disponible"
exten => s-NOANSWER,1,VoiceMail(${ARG2},u)
exten => s-NOANSWER,n,Hangup

exten => s-NOMOLESTAR,1,Playback(vm-extension)
exten => s-NOMOLESTAR,n,Playback(vm-isunavail)
exten => s-NOMOLESTAR,n,Hangup
```

```
; en cualquier otro caso, hacemos lo mismo que si no contesta.
exten => s-.,1,Goto(s-NOANSWER,1)
```

Otro de los usos más comunes de la AstDB es guardar la información sobre desvío de llamadas. Al activar esta funcionalidad, todas las llamadas recibidas en la extensión del usuario serán desviadas a otro número (que podrá ser otra extensión o un número externo).

Adaptemos nuestro dialplan para añadir esta funcionalidad:

```
[servicios]
; *72xxx... desvío mis llamadas al xxx...
exten =>  *72X.,1,NoOp(activando desvio para ${CALLERID(num)}
al ${EXTEN:3})
exten =>  *72X.,n,Set(DB(CF/${CALLERID(num)})=${EXTEN:3})
exten =>  *72X.,n,Playback(beep)
exten =>  *72X.,n,Hangup

; *73: desactiva el desvío
exten => *73,1,NoOp(desactivando desvío para ${CALLERID(num)})
exten => *73,n,DBDel(CF/${CALLERID(num)})
exten => *73,n,Playback(beep)
exten => *73,n,Hangup
```

Nuestra macro extensionVM la adaptamos también para que haga efectivo este desvío[2]:

```
[macro-extensionVM]
; esta macro recibirá dos parámetros
; el primero indica el dispositivo que hay que llamar
; el segundo es el número del buzón de voz al que desviar la
llamada

exten => s,1,NoOp(llamada para ${ARG1})

; si NO tiene activado el no-molestar, saltamos a "check-
desvio"
exten => s,n,GotoIf($[${DB EXISTS(DND/${MACRO EXTEN})} =
0]?s,check-desvio)
exten => s,n(DND),Playback(vm-extension)
exten => s,n,Playback(vm-isunavail)
```

---

[2] En esta macro no estamos comprobando que no se produzcan bucles: que el usuario 2001 no desvíe su teléfono a sí mismo, o que Bob desvíe su extensión a Laura y Laura desvíe la suya a Bob.

```
exten => s,n,Hangup

; Si no tiene desvío, saltamos a "llamar"
exten => s,n(check-
desvio),GotoIf($[${DB EXISTS(CF/${MACRO EXTEN})} =
0]?s,llamar)
exten => s,n,Dial(Local/${DB RESULT}@desde-usuarios,20)
exten => s,n(llamar),Dial(${ARG1},20)
exten => s,n,Goto(s-${DIALSTATUS},1)

; si está ocupado, se reproduce el mensaje correspondiente
exten => s-BUSY,1,VoiceMail(${ARG2},b)
exten => s-BUSY,n,Hangup

; si está no contesta, se reproduce el mensaje de "no
disponible"
exten => s-NOANSWER,1,VoiceMail(${ARG2},u)
exten => s-NOANSWER,n,Hangup

exten => s-NOMOLESTAR,1,Playback(vm-extension)
exten => s-NOMOLESTAR,n,Playback(vm-isunavail)
exten => s-NOMOLESTAR,n,Hangup

; en cualquier otro caso, hacemos lo mismo que si no contesta.
exten => _s-.,1,Goto(s-NOANSWER,1)
```

10 Colas y agentes

Las colas permiten manejar de manera eficiente las llamadas entrantes. Cuando se envía una llamada a una cola a través del dialplan, esta llamada se queda en espera hasta el momento en el que pueda ser atendida por un operador. Mientras esta persona está en espera, estará oyendo una música, y se pueden reproducir determinados mensajes periódicamente (*"todos nuestros agentes están ocupados, su llamada será atendida en breves momentos..."*) para evitar que desista si la espera dura más de la cuenta.

Las llamadas que entran en una cola serán distribuidas por orden de llegada entre los miembros de esta cola. Para decidir a qué usuario se envía cada llamada, el sistema usará una de las distintas estrategias de distribución de que dispone:

- **ringall.** Esta es la estrategia por defecto y envía la llamada a todos los miembros de la cola (aquellos que estén disponibles).

- **roundrobin**. Esta opción está obsoleta en Asterisk 1.4, y va enviando la llamada por orden a los miembros de la cola. Siempre en el mismo orden, y empezando por el primero.

- **leastrecent**. Envía la llamada al usuario que lleva más tiempo sin recibir llamadas.

- **fewestcalls**. Envía la llamada al usuario que ha atendido menos llamadas de esta cola.

- **random**. Selecciona el destino de forma aleatoria.

- **rrmemory**. Igual que roundrobin, pero "con memoria": intentará enviar la llamada uno a uno a todos los miembros de la cola, hasta que alguno la atienda. Pero la siguiente llamada no volverá a empezar con el primer usuario, sino que continuará el turno por donde paró en la anterior llamada.

Cada cola tendrá uno o varios miembros, que son los usuarios que van a atender las llamadas. Estos miembros pueden ser estáticos (*SIP/laura*, *IAX/juan*, *ZAP/2*, *ZAP/g1/912345678*, *Local/2001@a-extensiones*), o pueden añadirse y eliminarse de forma dinámica a la cola mediante las aplicaciones *AddQueueMember* y *RemoveQueueMember* del dialplan.

También tenemos en Asterisk el concepto de *Agentes*. Los agentes son usuarios que se conectan al sistema para atender llamadas. Para empezar a recibir llamadas, el agente debe identificarse en el sistema mediante su número de agente y contraseña. A partir de ese momento, el agente permanecerá conectado, escuchando una música hasta que haya alguna llamada para atender[3].

Los agentes se definen en el fichero *agents.conf*. En este fichero se indican los parámetros globales para todos los agentes (categoría de música en espera, si se grabarán las llamadas, etc.), y por último, se definen los agentes:

```
agents.conf

[agents]
```

[3] También existe la posibilidad de que el agente se identifique en el sistema y cuelgue, y que después el sistema le envíe las llamadas a aquella extensión en la que se encuentra. Para esto disponemos de la aplicación AgentCallbackLogin. Esta opción, aunque es muy útil, está obsoleta, pero se puede simular mediante las aplicaciones AddQueueMember y RemoveQueueMember.

```
autologoff=15 ; Si no atiende la llamada en 15 segundos,
desconectamos a este agente
wrapuptime=0 ; tiempo para volver a llamarle
musiconhold=default

group=1
agent => 101,1234,Agente 1
agent => 102,5678,Agente 2
agent => 103,9090,Agente 3
```

Las colas se definen en el fichero *queues.conf*. Este fichero también tiene una sección *[general]* en el que se definen los parámetros globales para el sistema de colas, seguida de la definición de cada una de las colas.

```
[general]
language = es
persistentmembers=yes ; los miembros conectados dinámicamente
se guardan en la AstDB
autofill=yes ; no espera a que algún agente atienda la primera
llamada de la cola para empezar a intentar conectar las
siguientes.

[ventas]
strategy=ringall ; llamaremos a todos los miembros a la vez
timeout=15 ; durante 15 segundos
retry=5    ; esperamos 5 segundos antes de reintentar
maxlen=0
periodic-announce=queue-periodic-announce
periodic-announce-frequency=20
member=>SIP/bob
member=>SIP/alice
member=>Agent/101

[contabilidad]
strategy=ringall ; llamaremos a todos los miembros a la vez
timeout=15 ; durante 15 segundos
retry=5    ; esperamos 5 segundos antes de reintentar
maxlen=0
periodic-announce = queue-periodic-announce
periodic-announce-frequency=20
member=>SIP/bob
member=>IAX2/juan
```

En este ejemplo se han definido dos colas: *ventas* y *contabilidad*. Las dos colas seguirán la estrategia *ringall*. Es decir: cada llamada que entre en la cola se enviará a todos los miembros de la misma. En ambas colas se han definido qué mensaje se reproducirá periódicamente (cada 20 segundos) a la persona que está en

espera. Y por último, se indican los miembros estáticos que atenderán las llamadas de cada una de las colas.

Vamos a añadir un par de extensiones en el contexto *[servicios]* para añadir/eliminar miembros dinámicamente de las colas:

```
[servicios]
; 100* Para añadir al usuario que llama a la lista de miembros
de la cola "ventas"
exten => 100*,1,AddQueueMember(ventas)
exten => 100*,n,Playback(beep)
exten => 100*,n,Hangup

; 100** Para eliminar a este usuario de la lista de miembros
de la cola "ventas"
exten => 100**,1,RemoveQueueMember(ventas)
exten => 100**,n,Playback(beep)
exten => 100**,n,Hangup
```

Hemos utilizado las aplicaciones *AddQueueMember* y *RemoveQueueMember*. Como se comentó anteriormente, estas aplicaciones añaden y eliminan miembros a una cola dinámicamente. Podemos comprobar desde el CLI el efecto de estas aplicaciones:

```
PBX*CLI> queue show ventas
ventas       has 0 calls (max unlimited) in 'ringall' strategy
(0s holdtime), W:0, C:0, A:1, SL:0.0% within 0s
   Members:
      SIP/alice (Invalid) has taken no calls yet
      Agent/2001 (Unavailable) has taken no calls yet
   No Callers

PBX*CLI>
```

Ahora realice una llamada desde el terminal de Bob al número 100*. Seguidamente comprobamos de nuevo el estado de la cola "ventas":

```
PBX*CLI> queue show ventas
ventas       has 0 calls (max unlimited) in 'ringall' strategy
(0s holdtime), W:0, C:0, A:1, SL:0.0% within 0s
   Members:
      SIP/bob (dynamic) (Not in use) has taken no calls yet
      SIP/alice (Invalid) has taken no calls yet
      Agent/2001 (Unavailable) has taken no calls yet
   No Callers

PBX*CLI>
```

Como puede comprobar, SIP/bob ha sido añadido como miembro de esta cola, y desde este momento recibirá también las llamadas que entren en la cola "ventas".

Hasta ahora ya se encuentran definidas las colas, y cada una tiene uno o más miembros dispuestos a atender llamadas, pero... ¿Cómo entran las llamadas en la cola? Para esto tenemos la aplicación "*Queue*" del dialplan.

```
exten =>
....,n,Queue(queuename[|options[|URL] [|announceoverride] [|time
out] [|AGI])
```

La aplicación *Queue* es parecida a la aplicación *Dial*, con la diferencia de que en vez de enviar la llamada directamente al canal (o canales) que se indique en el primer parámetro, la llamada es enviada a una cola.

Podemos definir una extensión para hacer una llamada a una cola:

```
[desde-usuarios]
exten => 100,1,Queue(ventas,50,t)
```

Sin embargo, donde vamos a ver realmente el potencial de las colas es al usarlas junto a un IVR.

11 Interactive Voice Response (IVR)

Con *IVR* nos referimos a los menús con los que el usuario puede interactuar mediante pulsaciones DTMF[4]. Las empresas suelen usar estos menús, reproduciendo un mensaje de bienvenida cuando reciben una llamada, y ofreciéndonos después las típicas opciones de *"pulse 1 para hablar con el departamento de ventas; 2 para hablar con contabilidad..."*. Esta sería la forma más simple de IVR, conocido también como *Operadora Automática*.

Pero también habremos usado alguna vez IVRs más complejos, como algunos de banca telefónica, en los que llamamos a un número de teléfono del banco, nos identificamos pulsando nuestro número de DNI y contraseña y podemos realizar operaciones como activar una tarjeta, o escuchar el saldo de nuestra cuenta.

---

[4] DMTF: tonos de diferente frecuencia que son generados por un teléfono al pulsar una tecla del mismo.

Asterisk nos ofrece todas las herramientas necesarias para crear desde el IVR más simple hasta los sistemas más complejos.

Lo primero que necesitaremos para crear un IVR son los sonidos que se van a reproducir. Estos sonidos deben encontrarse en el directorio de sonidos de Asterisk (normalmente */var/lib/asterisk/sounds/*) y tener un formato reconocido por éste. Junto con Asterisk se distribuye un juego de sonidos estándar en inglés. Como vimos en el capítulo sobre la instalación de Asterisk, existen juegos de voces de gran calidad en castellano, pero normalmente necesitaremos algunos sonidos personalizados (*'gracias por llamar a la empresa xxxxx', 'pulse 1 para hablar con el departamento de ventas...'*). Aunque es posible usar la aplicación *Record* de asterisk para grabar los mensajes, para obtener la mejor calidad se aconseja grabarlos con una aplicación especializada (Audacity, por ejemplo). En este caso, debemos grabar los archivos a 8KHz 16 bits.

Para grabar los mensajes desde el dialplan se utiliza la aplicación *Record*. La sintaxis de la aplicación es:

```
Record(nombre.formato|silencio|maxDuracion|opciones)
```

Si dentro del nombre del fichero añadimos los caracteres *%d*, éstos serán sustituidos por un número secuencial, para evitar que sobrescribamos una grabación existente. En este caso, la variable *${RECORDED_FILE}* contendrá el nombre final del fichero. La aplicación *Record* grabará el mensaje hasta que el usuario pulse la tecla #, o bien hasta que se llegue a la duración máxima indicada como parámetro o se detecten tantos segundos de silencio como se hayan indicado.

```
[servicios]
exten => *77,1,Record(sonido-%d.alaw)
exten => *77,n,Playback(beep)
exten => *77,n,Playback(${RECORDED_FILE})
exten => *77,n,Hangup
```

Para programar nuestros IVRs disponemos de algunas aplicaciones interesantes para el dialplan:

- **Playback(sonido).** Reproduce un sonido.
- **WaitExten(tiempo).** Espera a que el usuario teclee una opción (o un número de extensión).
- **Background(sonido).** Reproduce un sonido, pero el usuario puede interrumpir la reproducción, tecleando un número de opción. Sería equivalente a utilizar las aplicaciones *Playback* y *WaitExten* de forma simultánea.

- **GotoIfTime(hora|dias_semana|dias_mes|año?si_cierto:si_falso).**Realiza un salto a otro punto del dialplan dependiendo de la fecha y hora. Resulta muy útil para actuar de manera distinta si estamos en horario de oficina o no.

Para que no se entre en un bucle infinito, se suelen fijar dos tipos de retardo: tiempo inter-dígito y el tiempo de respuesta:

```
Set(TIMEOUT(digit)=3)
Set(TIMEOUT(response)=9)
```

Una vez que el usuario empieza a teclear una opción, el tiempo máximo permitido entre dígitos será *TIMEOUT(digit)*. Si pasa este tiempo desde el último dígito tecleado, el sistema considera que el usuario ha terminado de teclear el número de la opción y saltará a esa extensión (u opción). Si no existe, se saltará a la extensión 'i'.

Por otra parte, si el usuario no empieza a teclear una opción en el tiempo indicado por *TIMEOUT(response)*, el IVR saltará a la extensión 't'.

```
; IVR al que llegarán las llamadas entrantes
; El mensaje de Bienvenida dice "Gracias por llamar a nuestra
empresa. Para hablar con el departamento de Ventas pulse 1,
para hablar con Contabilidad pulse 2. Si conoce la extensión
de la persona con la que desea hablar, márquela ahora"

[entrada]
exten => s,1,Set(TIMEOUT(digit)=3)
exten => s,n,Set(TIMEOUT(response)=9)
; Comprobamos si estamos en horario de oficina
exten => s,n,GotoIfTime(09:00-19:30|mon-
fri|*|*?dentro horario)
exten => s,n,Playback(estamos-cerrados-deje-mensaje)
exten => s,n,VoiceMail(999,s)
exten => s,n,Hangup
exten => s,n(dentro horario),Background(bienvenida)
exten => s,n,WaitExten(10)
exten => s,n,Goto(operadora)

; Si elige la opción 1, mandamos al usuario a la cola de
Ventas
exten => 1,1,Queue(ventas,t,60)
exten => 1,n,Hangup

; Si elige la opción 2, mandamos al usuario a la cola de
Contabilidad
exten => 2,1,Queue(contabilidad,t,60)
exten => 2,n,Hangup
```

```
; Si elige una opción incorrecta, volvemos a darle las
instrucciones
exten => i,1,Goto(s,1)

; Si no selecciona ninguna opción, lo mandamos a la cola de
operadores
exten => t,1,Queue(operadora)

; Al incluir el contexto a-extensiones, permitimos que el
usuario teclee un número de extensión directamente
include => a-extensiones
```

12 Salas de conferencias

Las salas de conferencia permiten que varios usuarios mantengan una conversación entre ellos, como si estuvieran reunidos en una sala[5]. Asterisk ofrece muchas posibilidades para la creación de salas de conferencias, como por ejemplo crear salas en las que una persona puede hablar mientras todos los demás escuchan, o bien que todos puedan intervenir al mismo tiempo; enmudecer o expulsar a un usuario de una sala durante la conferencia, bloquearla para que no entren más usuarios, grabarla, etc.

Podemos definir una serie de salas de forma estática, en el fichero *meetme.conf*. Este fichero contiene una sección *[general]* con un único parámetro *audiobuffers* para indicar la cantidad de buffers que usará la aplicación para amortiguar el jitter. Después de la sección *[general]* hay una sección *[rooms]* con la definición de las distintas salas de conferencias:

```
meetme.conf:

[general]
audiobuffers=5

[rooms]
; Sala de conferencias n° 100, sin PIN
conf => 100
; Sala de conferencias n° 101 con PIN 1234
conf => 101,1234
```

---

[5] Asterisk necesita una fuente de tiempo externa para mezclar correctamente los canales en la sala de conferencias. El driver Zaptel provee esta fuente de tiempo, por lo que si no disponemos de hardware de Digium o compatible gestionado por Zaptel, tendremos que cargar el módulo ztdummy.

```
; Sala de conferencias nº 102, con PIN 1234 y PIN de
administrador 5544
conf => 102,1234,5544
```

Aunque también es posible crear las salas dinámicamente, por parte de los mismos usuarios.

La aplicación que usaremos en el dialplan para acceder a las salas de conferencias es *Meetme*:

```
exten => xxxx,n, Meetme(numero_de_sala[|opciones[|PIN]])
```

Las opciones que podemos especificar son las siguientes:

- **a.** Entrar en modo Administrador. El administrador puede bloquear la conferencia para que no entren más participantes. (Ver opción 's').
- **A.** Entrar en modo "marcado". Cuando este usuario salga de la conferencia, ésta finaliza.
- **b.** Ejecuta el script AGI indicado en ${MEETME_AGI_BACKGROUND}. *Por defecto: conf-background.agi* sólo funciona si todos los canales de la conferencia son ZAP.
- **c.** Se reproduce un aviso indicando cuántos usuarios hay en la conferencia.
- **d.** Crea la conferencia dinámicamente.
- **D.** Crea la conferencia dinámicamente, y le asigna un PIN.
- **e.** Selecciona una conferencia vacía, de las que están definidas en *meetme.conf.*
- **E.** Selecciona una conferencia vacía de las que están definidas CON PIN *en meetme.conf.*
- **F.** Reenvía los tonos DTMF en la conferencia.
- **i.** Reproduce un aviso cada vez que un nuevo usuario entra o abandona la conferencia.
- **I.** Reproduce un aviso cada vez que un nuevo usuario entra o abandona la conferencia (sin review').
- **l.** Entrar en modo "sólo escucha" en la conferencia.

- **m.** Entrar en modo "silencio". Posteriormente el usuario podrá pulsar '*' para des-enmudecerse.
- **M.** Activa la música en espera cuando sólo hay un usuario en la conferencia.
- **o.** Sólo se mezcla el sonido de los usuarios que están hablando. (Reduce el ruido de fondo y la carga de CPU).
- **p.** El usuario puede abandonar la conferencia pulsando #.
- **P.** Pide el PIN de la conferencia, aunque éste se haya indicado en la llamada a Meetme.
- **q.** Modo "silencioso" (no se reproducen avisos cuando un usuario entra/sale de la conferencia).
- **r.** Grabar la conferencia. La grabación se realizará sobre el fichero definido en *${MEETME_RECORDINGFILE}* usando el formato *${MEETME_RECORDINGFORMAT}*. El nombre por defecto es *meetme-conf-rec-${CONFNO}-${UNIQUEID}* y el formato por defecto wav.
- **s.** El usuario puede pulsar '*' durante la conferencia para activar un menú, en el que dispone de las siguientes opciones:
  - **1.** Enmudecerse/desenmudecerse (activar/desactivar mute propio).
  - **4/6.** Baja/sube el volumen del audio que le llega.
  - **7/9.** Baja/sube el volumen de su propia voz.
  - Además, si el usuario es administrador de esta conferencia, tiene la opción de bloquearla para no permitir que entre nadie más en ella, pulsando "3".
- **t.** Modo "talk-only": sólo puede hablar, sin oir nada.
- **T.** Asterisk enviará notificaciones a través del "*manager*" indicando qué usuario es el que está hablando en cada momento. También desde el CLI, mediante "*meetme list*" podemos conocer esta información.
- **w[(<secs>)].** Espera a que el "marcado" entre en la conferencia.
- **x.** Cierra la conferencia cuando salga el último usuario 'marcado'.

- **X.** Permite al usuario salir de la conferencia marcando una extensión válida de un solo dígito durante la conferencia. La extensión debe pertenecer al contexto *${MEETME_EXIT_CONTEXT}* o al contexto actual, si dicha variable no está definida.

- **1.** No reproduce el mensaje cuando entra la primera persona en la conferencia.

Ampliemos nuestro contexto *[servicios]*, con una extensión para crear y unirse a una conferencia:

```
[servicios]
exten => 800,1,Meetme(,DMIsr)
exten => 800,n,hangup
```

Cuando un usuario llame a la extensión 800, el sistema le pedirá que introduzca un número de conferencia, terminando con #. El usuario puede asignar el número que desee (con cualquier cantidad de dígitos). Seguidamente tendrá que asignar un PIN a esta conferencia, y decir su propio nombre. De esta forma se crea una nueva conferencia dinámicamente, con el número asignado por el usuario (parámetro D).

El usuario que ha creado la conferencia se quedará conectado, escuchando una música en espera hasta que se conecte un segundo usuario (parámetro M).

Para que los demás usuarios se puedan unir a la conferencia solamente tienen que marcar la extensión 800, teclear el número de conferencia que creó el primer usuario y el mismo PIN.

Cada usuario que se una a la conferencia deberá decir su nombre. Asterisk usará este nombre para reproducir un aviso a los demás usuarios, indicando el nombre de quien se une a la conferencia y de quien sale de la misma. (Parámetro I).

Durante la conferencia, cada usuario puede marcar la tecla * para enmudecer su terminal o adaptar los niveles de volumen del audio que envía/recibe (parámetro s).

Y por último, la conferencia se grabará en un archivo con el nombre *meetme-conf-rec-<nº conferencia>-<ID unico>.wav* en el directorio */var/lib/asterisk/sounds/* (parámetro r).

También existe una aplicación para administrar salas de conferencias desde el mismo Dialplan. Esta aplicación es "*MeetmeAdmin*" y su sintaxis es:

```
exten => xxx,n,MeetmeAdmin(nº_conferencia|comando[|usuario])
```

Esta aplicación, junto con la opción "X" de *Meetme*, que permite que el usuario pueda salir de la conferencia tecleando un número de extensión de un solo dígito, nos será muy útil para dar al administrador de la conferencia el poder de expulsar o silenciar a aquellos usuarios molestos en la conferencia:

La aplicación *MeetmeAdmin* admite las siguientes opciones:

- **e.** Expulsar al último usuario que ha entrado en la conferencia.
- **k.** Expulsar a un usuario (indicado en el tercer parámetro).
- **K.** Expulsar a todos los usuarios.
- **l.** Desbloquear la conferencia.
- **L.** Bloquear la conferencia – no podrá unirse nadie más.
- **M.** Silenciar a un usuario.
- **m.** Desactivar silenciar a un usuario.
- **N.** Silenciar a TODOS los usuarios no administradores.
- **n.** Desactiva silenciar a todos los usuarios no administradores.
- **r.** Resetear los niveles de volumen de un usuario.
- **R.** Resetear los niveles de volumen de todos los usuarios.
- **s.** Bajar el volumen de la voz de toda la conferencia.
- **S.** Subir el volumen de la voz de toda la conferencia.
- **t.** Bajar el volumen de la voz de un usuario.
- **T.** Subir el volumen de la voz de un usuario.
- **u.** Bajar el volumen del audio que llega a un usuario.
- **U.** Subir el volumen del audio que llega a un usuario.
- **v.** Bajar el volumen del audio que llega a todos los usuarios.
- **V.** Subir el volumen del audio que llega a todos los usuarios.

Esta es una de esas aplicaciones que si no se pone un ejemplo, no se ve cómo usarlas, así que vamos manos a la obra. Crearemos una extensión (899) para administrar salas de conferencias. Cuando el administrador entre en la extensión 899 se le pedirá una contraseña; porque no desea que cualquier usuario pueda administrar las salas de conferencias, ¿verdad? Seguidamente, el número de la sala que desea administrar. Después de esto, le pedimos el número de la conferencia que desea administrar y saltaremos a un contexto especial (*admin-conferencias*) en el que definimos las opciones que podrá usar.

```
[servicios]
...
; Extension 899 para administrar salas de conferencias.
exten => 899,1,Authenticate(123123) ; Para continuar debo
introducir la contraseña
exten => 899,n,Read(CONFNO,conf-getconfno); y el número de
conferencia  que voy a administrar
exten => 899,n,Goto(admin-conferencias,s,1)

[admin-conferencias]
; Entramos en la conferencia en modo administrador (a) y
activando
; la posibilidad de teclear una opción (X)
exten => s,1,Meetme(${CONFNO},aX)
exten => s,n,Hangup

; pulsando 1 silenciamos a todos los usuarios -no
administradores-
exten => 1,1,MeetmeAdmin(${CONFNO}|N)
; y volvemos a la conferencia
exten => 1,n,Goto(s,1)

; pulsando 2 devolvemos la voz a todos los usuarios
exten => 2,1,MeetmeAdmin(${CONFNO}|n)
exten => 2,n,Goto(s,1)

; pulsando 8 expulsamos al último usuario que entró en la
conferencia
exten => 8,1,MeetmeAdmin(${CONFNO}|e)
exten => 8,n,Goto(s,1)

; pulsando 9 expulsamos a un usuario
; deberemos introducir el número de usuario
; que es el número de orden en el que entró en la conferencia
exten => 9,1,Read(NUM,vm-extension)
exten => 9,n,MeetmeAdmin(${CONFNO}|k|${NUM})
exten => 9,n,Goto(s,1)
```

Aunque como vemos, es posible realizar determinadas operaciones sobre una sala de conferencias desde el dialplan, lo cierto es que esto no es muy práctico, ya que el administrador debe conocer el número de orden en el que han ido entrando los usuarios para poder operar sobre ellos. Para gestionar las salas de forma más operativa existen aplicaciones como *WebMeetMe* que se conectan con Asterisk a través del AMI y ofrecen bastante más facilidad al administrador.

13 Haciendo un dialplan mantenible

En este capítulo, se ha ido completando la instalación de una típica PBX con los servicios elementales, añadiendo las extensiones correspondientes. Estas extensiones han sido añadidas en diversos contextos, y no precisamente de manera arbitraria.

Es importante que un contexto represente un único concepto, es decir, el contexto *a-extensiones* define la forma de llamar a los terminales, no a teléfonos fijos nacionales. Esta forma de dividir el dialplan en contextos posibilita que luego estos sean combinados como se quiera mediante la sentencia *include*. Así, podemos crear contextos como *desde-usuarios* y *desde-usuarios-vip*, donde el segundo contexto permita realizar llamadas a destinos que *desde-usuarios* no puede:

```
[desde-usuarios]
include => a-extensiones
include => a-moviles
include => a-fijos

[desde-usuarios-vip]
include => a-extensiones
include => a-moviles
include => a-fijos
include => a-internacionales
```

Si hubiéramos incluido todo en el mismo contexto sería necesario repetir mucho código, y cada vez que quisiéramos cambiar algo habría que hacerlo en varios sitios, por lo que no resultaría fácilmente mantenible.

CAPÍTULO 5

GESTIÓN DE ASTERISK MEDIANTE INTERFAZ WEB

Juan Antonio García Moreno

1 Introducción

Con la creación de los Departamentos de Sistemas de la Información en las empresas, y la incorporación de técnicos informáticos cualificados, se empieza a demandar la posibilidad de gestionar las centralitas telefónicas sin acudir a personal externo.

A pesar de la cualificación de dicho personal, los sistemas tradicionales han limitado la adquisición de los conocimientos mínimos de administración de las PBXs. Por tanto, esta desventaja se traduce en que el personal no esté preparado para programar a nivel de código los nuevos dispositivos de VoIP como Asterisk. Tampoco es objetivo formar expertos en Asterisk para realizar una administración básica como crear Extensiones, Ring Group, etc., esporádicamente.

Es aquí donde surge la necesidad de un software de administración para Asterisk con el que el personal técnico de la empresa sea capaz de realizar esta "Administración Básica" de Asterisk.

2 Gestores web

En este capítulo hablaremos de varios de los "Gestores WEB" que existen para la Administración de Asterisk, principalmente basados en software libre, y nos centraremos de manera especial en FreePBX, el más extendido, tanto por instalaciones independientes junto a Asterisk como por las distintas distribuciones precompiladas que lo incorporan y utilizan.

2.1 FREEPBX

Es uno de los interfaces de gestión Web para Asterisk con mayor difusión en el mundo de la VoIP y Asterisk.

FreePBX es un desarrollo *Open Source* que se encuentra bajo licencia GNU GPL y ha sido desarrollado principalmente en Perl y PHP por Phillipe Lindheimer.

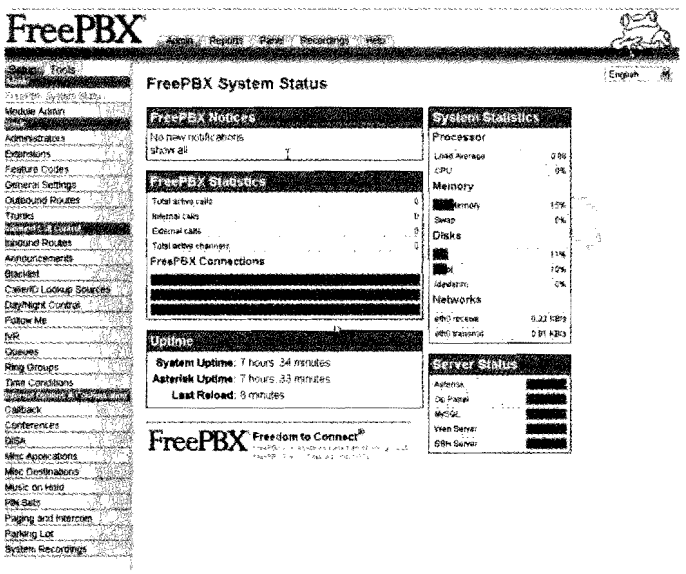


Figura 5-1. Página principal de FreePBX

Detrás de *FreePBX* existe una importante comunidad desarrollando y dando soporte a este proyecto (http://www.freepbx.org).

FreePBX utiliza Apache como servidor Web y MySQL como servidor de base de datos. A groso modo, FreePBX es una interfaz Web que guarda la

configuración de Asterisk en la bases de datos, reescribe los archivos de configuración de Asterisk con dicha información.

Cada vez que creamos una configuración o la cambiamos con FreePBX, los datos se guardan en la base de datos, y a la hora de actualizar, se vuelven a crear automáticamente algunos de los ficheros de texto de la configuración de Asterisk.

La filosofía de trabajo de FreePBX implica que no se debe modificar manualmente los ficheros de configuración de Asterisk, ya que serán reescritos por FreeBX cada vez que se realice algún cambio en la interfaz Web.

Hay que tener muy en cuenta este tema en el caso de que tengamos un servidor Asterisk en explotación y decidamos posteriormente montar FreePBX. Lo más probable es que perdamos toda nuestra configuración.

Para evitar estas situaciones, FreePBX cuenta con unos ficheros en los que el usuario puede añadir sus propias configuraciones interactuando estas con las que introduzcamos mediante este software de gestión sin ser modificados.

A parte del propio gestor de administración FreePBX también instala otros tres paquetes que ayudan a la gestión de la centralita:

- **Call Detail Record (CDR).** Proporciona un detalle de las llamadas entrantes y salientes en Asterisk.

- **Asterisk Flash Operator Panel (FOP).** Software que visualiza estados y con el que se pueden gestionar las llamadas, extensiones, colas, salas de conferencia, etc.

  Este software puede ser instalado independientemente de FreePBX y toda la información y soporte puede ser encontrado en la web "*www.asternic.org*".

- **Asterisk Recording Interface (ARI).** Este software gestiona, entre otras funcionalidades, los buzones de voz y grabaciones de los usuarios desde un entorno gráfico.

  Fue desarrollado por Dan Littlejohn de Littlejohn Consulting.

2.2 ASTERISKGUI

AsteriskGUI es el gestor oficial de DIGIUM para Asterisk (véase la figura 5-2).

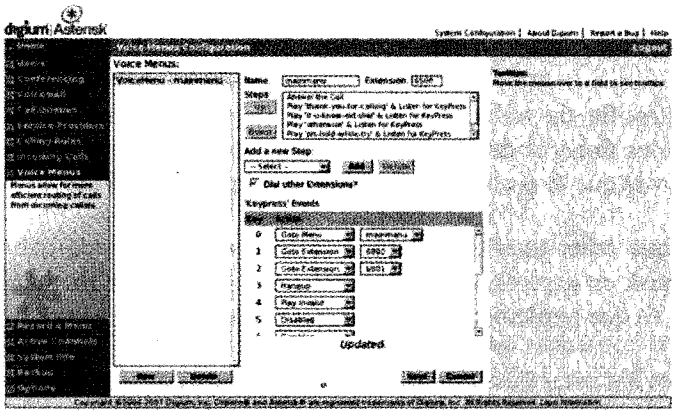


Figura 5-2. AsteriskGUI

Aun siendo el gestor oficial de Digium, no tiene la potencia que tienen otros gestores existentes actualmente, aunque siguen trabajando en él, mejorándolo día a día.

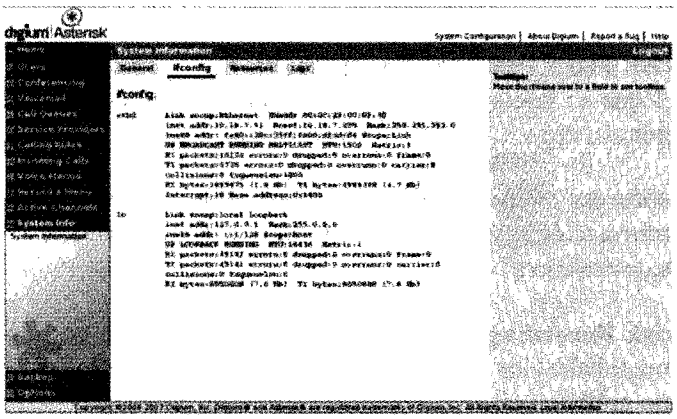


Figura 5-3. AsteriskGUI - Ifconfig

2.3 OTROS

Existen multitud de interfaces que permiten gestionar Asterisk, aunque no gozan de tanta popularidad como los anteriores. Entre ellos, *VoiceOne* está

teniendo un gran auge últimamente. En las figuras 5-4 y 5-5 se puede ver un detalle del gestor *VoiceOne*.

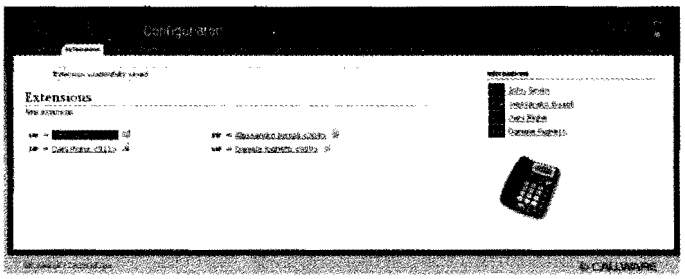


Figura 5-4. VoiceOne

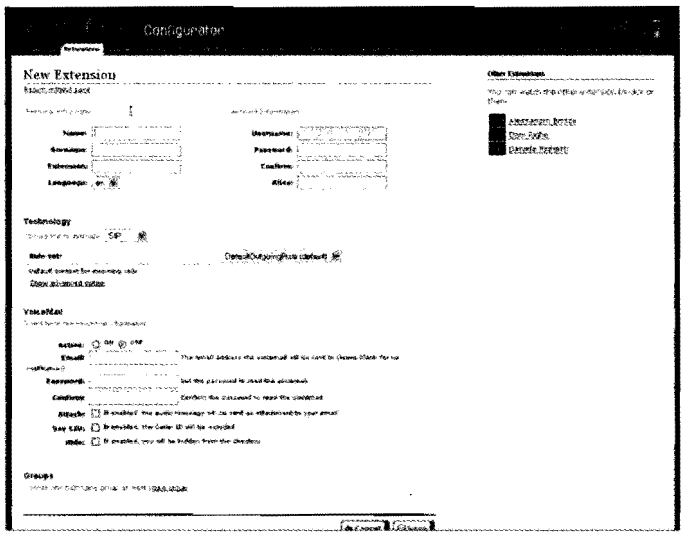


Figura 5-5. VoiceOne (Extensions)

3 Instalación de FreePBX

Antes de iniciar la instalación de FreePBX, es preciso que el servicio Asterisk esté instalado y configurado correctamente tal y como se ha visto en capítulos anteriores.

Dos de las distribuciones más utilizadas son Fedora y Debian, por lo que a lo largo del capítulo se irán comentando las diferencias que existen en la instalación de las mismas.

En primer lugar, y para comenzar con la instalación de FreePBX, hay que descargar la última versión de este software en el directorio */usr/src/*:

```
# cd /usr/src
# wget http://mirror.freepbx.org/freepbx-2.4.0.tar.gz
```

A continuación lo descomprimimos:

```
# tar xfvz freepbx-2.4.0.tar.gz
```

3.1 DEPENDENCIAS

Para el correcto funcionamiento de FreePBX, necesitamos instalar una serie de paquetes, entre ellos, el servidor *Apache*, servidor de correo *Sendmail*, PHP, MySQL y otras tantas librerías.

Para instalar las dependencias en CentOS 5.1, ejecutamos el siguiente comando:

```
# yum -y install e2fsprogs-devel keyutils-libs-devel krb5-
devel libogg libselinux-devel libsepol-devel libxml2-devel
libtiff-devel gmp kernel-devel ncurses-devel audiofile-devel
libogg-devel openssl-devel zlib-devel perl-DateManip sendmail-
cf
# yum -y install php-pear php-pear-DB php-gd php-mysql php-pdo
mysqlmysql-devel mysql-server httpd
```

Otro de los paquetes a instalar y compilar en CentOS 5.1 es Lame, y lo haremos de la siguiente forma:

```
# cd /usr/src
# wget
http://easynews.dl.sourceforge.net/sourceforge/lame/lame-
3.97.tar.gz
# tar zxvf lame-3.97.tar.gz
# cd lame-3.97
# ./configure
# make
# make install
```

Nota: *debe tener instalado el compilador "gcc" y sus dependencias antes de realizar esta tarea.*

Para instalar las dependencias en Debian, ejecute el siguiente comando:

```
# apt-get install -y apache2 php5 php5-cli mysql-server-5.0
php-pear php5-mysql php-db libapache2-mod-php5 php5-gd php5-
curl
```

3.2 INSTALACIÓN Y CONFIGURACIÓN DE MySQL

Tal y como se vio al principio del capítulo, FreePBX utiliza MySQL para almacenar el "dialplan", usuarios, etc., y posteriormente generar los ficheros de configuración que usará Asterisk para operar.

Una vez instalado, a continuación se va a crear y configurar la base de datos.

En Debian, los servicios instalados se inician automáticamente, pero en CentOS 5.1 se debe iniciar y configurar manualmente el sistema para que arranquen automáticamente cada vez que se inicie el sistema.

Para ello, en CentOS debe realizar la siguiente tarea antes de seguir configurando. Para configurar que se inicien los servicios al iniciar el sistema:

```
# chkconfig httpd on ; Para iniciar automáticamente Apache.
# chkconfig mysqld on      .; Para iniciar automáticamente
MySQL.
# chkconfig --list   ; Para comprobar que están configurados
; para iniciarse.
```

Esta tarea también podemos realizarla utilizando la utilidad *ntsysv*.

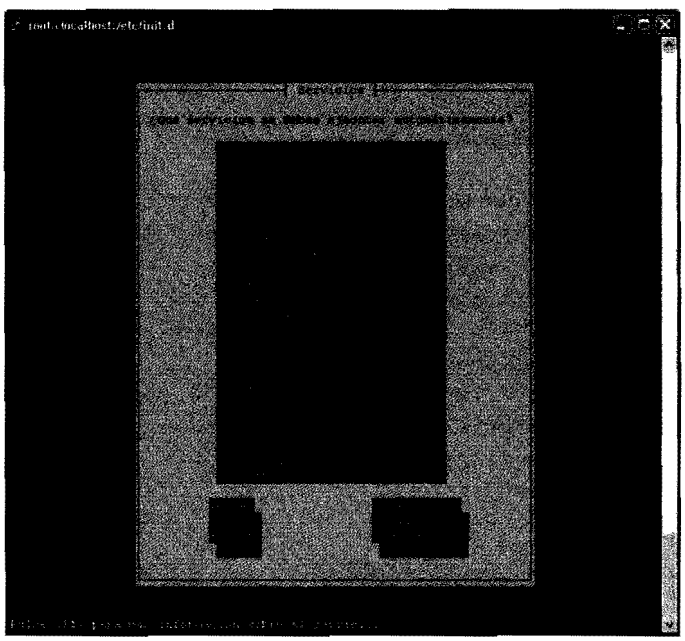


Figura 5-6. ntsysv

Para arrancar los servicios en CentOS debe ejecutar:

```
# service httpd start
# seviche mysqld start
```

Una vez arrancado el servidor MySQL, procederemos a la creación de las bases de datos necesarias:

- **asterisk**. Almacena la configuración que se usará para generar los ficheros que utilizará Asterisk para funcionar.

- **.asteriskcdrdb**. Almacena los datos de las llamadas realizadas, tiempos, canales por los que han salido, en definitiva, lo que se conoce como CDR (Call Detail Record).

Acceda al shell de MySQL:

```
# mysql -u root
```

Cree las bases de datos:

```
mysql> create database asteriskcdrdb;
Query OK, 1 row affected (0.00 sec)
mysql> create database asterisk;
Query OK, 1 row affected (0.01 sec)
```

Las acciones anteriores también puede realizarlas sin entrar en MySQL, simplemente ejecutando desde la consola los siguientes comandos:

```
# mysqladmin asteriskcdrdb
# mysqladmin asterisk
```

Ahora compruebe que se han creado correctamente las bases de datos:

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information schema |
| asterix            |
| asteriskcdr        |
| mysql              |
| test               |
+--------------------+

5 rows in set (0.01 sec)
```

Salga de MySQL ejecutando el comando *quit*:

```
mysql> quit;
#
```

A continuación cargue la estructura de ambas bases de datos desde los ficheros descargados para FreePBX. Realicemos las tareas que se detallan a continuación:

```
# cd /usr/src/freepbx-2.4.0
# mysql asterisk < SQL/newinstall.sql
# mysql asteriskcdrdb < SQL/cdr_mysql_table.sql
```

Entre en MySQL para comprobar que se han creado bien las estructuras de ambas bases de datos:

```
# mysql -u root
mysql> use asterisk;
mysql> show tables;
+-------------------------+
| Tables in asterisk      |
+-------------------------+
| admin                   |
| ampusers                |
| cronmanager             |
| devices                 |
| extensions              |
| featurescodes           |
| freepbx_log             |
| globals                 |
| iax                     |
| incoming                |
| module_xml              |
| modules                 |
| notifications           |
| sip                     |
| users                   |
| zap                     |
| zapchandids             |
+-------------------------+

17 rows in set (0.01 sec)

mysql> use asteriskcdrdb;
mysql> show tables;
+-------------------------+
| Tables in asteriskcdr   |
+-------------------------+
| cdr                     |
+-------------------------+

1 row in set (0.01 sec)
```

A continuación se establecen los permisos sobre las tablas creadas. El usuario y contraseña que configure a continuación debe configurarlos posteriormente en FreePBX para garantizar el acceso a estas bases de datos. Ejecute los siguientes comandos sin salir de la consola de MySQL:

```
mysql> GRANT ALL PRIVILEGES ON asteriskcdrdb.* TO
asteriskuser@localhost IDENTIFIED BY 'amp109';
Query OK, 0 rows affected (0.01 sec)

mysql> GRANT ALL PRIVILEGES ON asterisk.* TO
asteriskuser@localhost IDENTIFIED BY 'amp109';
Query OK, 0 rows affected (0.00 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.00 sec)

mysql> quit;
```

Por último, establezca la contraseña de acceso a las bases de datos para el "*root*":

```
# mysqladmin -u root password 'La Password'
```

3.3 INSTALACIÓN Y CONFIGURACIÓN DE APACHE

En el apartado *Dependencias* se instaló el servidor Apache, por lo que sólo queda configurarlo aquí.

Para ello, hay que asegurarse que tenemos creado el usuario *asterisk* y el grupo a*sterisk.*

Si no están creados, los creamos de la siguiente forma:

```
# groupadd asterisk
# useradd -c "asterisk PBX" -d /var/lib/asterisk -g asterisk
asterisk
```

A continuación cambie el usuario y grupo por defecto de ejecución del Apache por el *asterisk* ejecutando:

DEBIAN:

- Edite el fichero "/etc/apache2/apache2.conf".
- Busque las siguientes líneas:

```
User www-data
Group www-data
```

- Y se modifican por:

```
user asterisk
group asterik
```

- Crea el directorio y asigne los permisos al usuario *asterisk*:

```
# mkdir /var/lib/php5/session
# chown asterisk /var/lib/php5/session
```

- Para establecer el directorio de origen de los documentos, edite el fichero */etc/apache2/sites-available/default* y lo modifica de la siguiente manera:

```
DocumentRoot /var/www/html/
<Directory /var/www/html/>
```

- Reinicie el servidor Apache:

```
# /etc/init.d/apache2 restart
```

CentOS:

- Edite el fichero */etc/httpd/conf/httpd.conf*.

- Busque las siguientes líneas:

```
User apache
Group apache
```

- Y se modifican por:

```
user asterisk
group asterik
```

- Asigne los permisos al usuario *asterisk*:

```
# chown asterisk /var/lib/php/session
```

- Reinicie el Servidor Apache:

```
# /etc/init.d/httpd restart
```

En estos momentos lo único que queda es la instalación de FreePBX.

3.4 INSTALACIÓN DE FREEPBX (AMPORTAL)

Para realizar la instalación de FreePBX, es necesario que Asterisk se esté ejecutando. Para ello, y si no lo está ya, puede ejecutar:

```
# asterisk
```

En CentOS es necesario tener deshabilitado SELINUX, al objeto de evitar incompatibilidades y errores inesperados. Puede comprobar si SELINUX se está ejecutando con el siguiente comando:

```
# selinuxenabled && echo $?
```

Y pararlo sobre la marcha ejecutando:

```
# setenforce 0
```

Para hacer esto, en CentOS, edite el fichero */etc/selinux/config* y lo modifica de la siguiente forma:

```
SELINUX=disabled
SELINUXTYPE=targeted
```

Una vez hecho lo anterior, debe reiniciar el equipo para que tome los cambios anteriores para poder instalar el gestor.

Otro de los temas a tener en cuenta para evitar interferencias en la instalación es deshabilitar el firewall *iptables*, para lo que debe ejecutar:

```
# iptables -F
# iptables-save >/etc/sysconfig/iptables (Para guardar la
configuración en CentOS)
# iptables-save > /etc/iptables.up.rules (Para guardar la
configuración en DEBIAN)
```

***Nota:** una vez terminado todo el proceso de instalación y configuración, vuelva a activar el firewall.*

Configure el directorio de trabajo de Asterisk editando el fichero */etc/asterisk/asterisk.conf* y cambiando *astrundir => /var/run* por *astrundir => /var/run/asterisk*.

A continuación instale el gestor WEB:

```
# cd /usr/src/freepbx-2.4.0
# ./install amp --username=root --password='La Password' (Sin
las comillas)
```

El sistema realiza una serie de preguntas relacionadas con los datos que ha ido introduciendo hasta ahora. Conteste con los datos que se han ido planteando (usuarios, passwords, etc.).

Al finalizar la instalación debe aparecer un texto como el que se presenta a continuación sin haber mostrado errores:

```
Please update your modules and reload Asterisk by visiting
http://xx.xx.xx.xx/admin
```

```
**************************************************************
*
```

3.5 MODIFICACIONES PREVIAS AL INICIO DE FREEPBX

Antes de iniciar FreePBX, debe realizar una serie de modificaciones tanto de permisos de usuarios, como de passwords y algún que otro cambio de rutas de acceso.

3.5.1 Permisos en directorios

En este momento, tiene que cambiar los permisos al contenido de algunos directorios:

```
# chown asterisk:asterisk -R /var/lib/asterisk/
# chown asterisk:asterisk -R /etc/asterisk/
# chown asterisk:asterisk -R /etc/amportal.conf
# chmod 770 -R /var/lib/asterisk/
# chmod 775 -R /etc/asterisk/
# chmod 770 -R /etc/amportal.conf
```

3.5.2 Rutas del FOP y permisos para la IP de Administración

Debemos realizar unas modificaciones en el fichero de configuración del *Flash Operator Panel (FOP)*, para ello, edite el fichero */var/www/html/admin/views/panel.php* y elimine "*../*" que hay delante de la palabra "*panel*" de la siguiente línea:

```
'<iframe width="97%" height="600" frameborder="0" align="top"
src="../../panel/index_amp.php?context='.$deptname.'"></iframe
>'.
```

Y debe quedar de la siguiente forma:

```
'<iframe width="97%" height="600" frameborder="0" align="top"
src="../panel/index_amp.php?context='.$deptname.'"></iframe>'.
```

Y a continuación modifique la configuración de los equipos que se podrán conectar al *FOP* para la administración. En este caso, edite el fichero */etc/amportal.conf* y modifique la variable *AMPWEBADDRESS* de la siguiente forma para no limitarlo a ningún PC concreto:

```
AMPWEBADDRESS=
```

Si en vez de dejar en blanco escribe una IP, sólo se permitirá la gestión desde ese equipo.

3.5.3 Permisos y cambio Password al módulo Manager

Por seguridad, también debe cambiar la *password* por defecto del *Manager de Asterisk*.

Para ello, edite el fichero */etc/amportal.conf* y modifique la variable *AMPMGRPASS*, por ejemplo, con la siguiente *password*:

```
AMPMGRPASS = password
```

A continuación, edite el fichero */etc/asterisk/manager.conf* y modifique la variable *secret* de la sección *[admin]* con la misma password:

```
[admin]
secret = password
```

En este momento, con la configuración actual, se permite la conexión al *Manager* a procesos originados por el equipo en el que se está ejecutando:

```
deny=0.0.0.0/0.0.0.0
permit=127.0.0.1/255.255.255.0
```

Si desea que se conecten al *Manager* desde procesos iniciados en otro equipo, por ejemplo el *192.168.0.200*, tendíamos que añadir la línea correspondiente *permit* o comentar la línea actual *deny*:

```
deny=0.0.0.0/0.0.0.0
permit=127.0.0.1/255.255.255.0
permit=192.168.0.200/255.255.255.0
```

Por último, y para que surtan efecto todos los cambios anteriores, debe reiniciar FreePBX:

```
# amportal restart
```

4 Utilización de FreePBX

FreePBX está formado por multitud de módulos que pueden ser cargados según las necesidades de la instalación, muy en la línea de modularidad de Asterisk.

Parte de estos módulos pueden encontrarse en modo local y pueden ser instalados sin la necesidad de que se descarguen a través de Internet, y otros muchos se descargarán desde Internet previamente a su instalación.

4.1 INICIO DE FREEPBX

Para iniciar FreePBX por primera vez, debe ejecutar el siguiente comando:

```
# amportal start
```

Para que FreePBX se inicie automáticamente, debe incluir en *rc.local* el comando anterior, para ello, ejecute:

```
# echo "/usr/local/sbin/amportal start" >> /etc/rc.local
```

***Nota:** en Debian se puede dar la situación de que al iniciar el servidor no se inicie amportal porque el fichero rc.local contenga "exit 0" antes que "amportal start". En ese caso, habrá que editar el fichero y poner la línea de amportal delante de esta última.*

Una vez iniciado FreePBX, debe conectarse al mismo iniciando un navegador y poniendo la siguiente dirección:

```
http://<Dirección IP del Servidor>/admin
```

Lo primero que debe hacer al conectarse a FreePBX es aplicar los cambios realizados. Para ello, al conectarse aparecerá una barra naranja en la parte superior izquierda de la ventana del navegador (véase la figura 5-7).

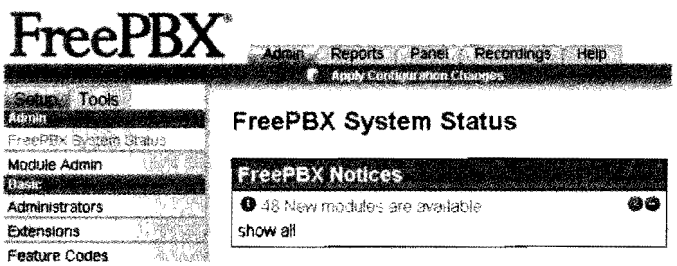


Figura 5-7. FreePBX (System Status)

Esta barra aparece cada vez que se realiza un cambio en FreePBX, y debe pulsar sobre ella para aplicar los cambios en la configuración. Al pulsar en *Apply Configuration Changes* el sistema solicita confirmación (véase la figura 5-8).

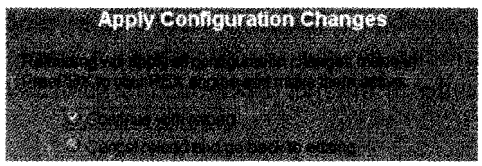


Figura 5-8. FreePBX (Aplicar configuración)

Una vez confirmados los cambios, se vuelve a la misma pantalla inicial, pero habrá desaparecido la barra naranja (véase la figura 5-9).

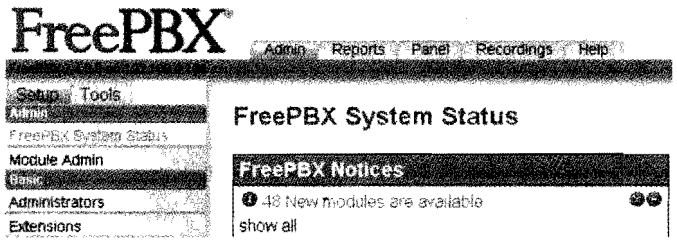


Figura 5-9. FreePBX (Con la configuración aplicada)

En estos momentos ya puede empezar a configurar la centralita.

4.2 ADMINISTRACIÓN DE FREEPBX

Como se ha comentado anteriormente, FreePBX consta de una serie de módulos base instalados para realizar la configuración de la PBX (extensiones, rutas de entrada y de salida, etc.), con los cuales, se podría montar una configuración básica para el funcionamiento de esta.

Adicionalmente, existen una serie de módulos, propios y de terceros, que puede añadir para administrar otras funcionalidades de Asterisk.

Estos módulos pueden estar instalados en local, es decir, vienen en el paquete de FreePBX que ha instalado, o se pueden descargar de Internet.

Los que están en local, están en el servidor pero no se han instalado en FreePBX aún, por lo que puede hacerlo sin necesidad de tener acceso a Internet, al contrario que los otros, para lo que necesita acceso a Internet.

En este apartado *FreePBX Notices* aparece si existen actualizaciones de los módulos instalados en FreePBX.

La mejor forma de ver la administración de la PBX, es plantear una situación real, de una implantación de una PBX en una empresa con unas necesidades reales.

De esta forma, el escenario a diseñar y configurar sería el siguiente:

Una empresa con 8 extensiones repartidas de la siguiente forma:

- 1 x Secretaria
- 1 x Dirección
- 2 x Departamento Técnico (Técnico + Responsable Departamento)
- 2 x Departamento Comercial (2 Técnicos)
- 2 x Departamento Atención al Cliente (2 Técnicos)

Cada Extensión tendrá su Buzón de Voz correspondiente.

Disponen de 2 líneas de entrada desde la PSTN y 2 Enlaces GSM o FCT (Fixed Cellular Terminal) para móviles. Todo se montará en una tarjeta Analógica de 4 puertos. Los 2 primeros canales serán las líneas directas de la RTB y los 2 últimos los de los enlaces GSM.

El horario de oficina es de 08:00 a 14:00 y de 16:00 a 18:00, de Lunes a Viernes.

- El flujo de las llamadas entrantes será el siguiente:

  1. Las llamadas entrantes fuera del horario de oficina serán atendidas por una locución que proporcionará el horario de atención al público y serán enviadas al buzón de voz de la secretaria.

  2. Las llamadas entrantes en horario de oficina serán atendidas por una locución, dependiendo si es mañana o tarde, y enviadas a un IVR (Interactive Voice Response), o menú interactivo de voz, que le permitirá al llamante ponerse en contacto con uno de los departamentos en cuestión, pulsando una opción, o esperar y que la llamada sea atendida por la secretaria. Si no es atendida por la secretaria, saltará el buzón de voz de ésta.

  3. El IVR constará de tres opciones, una para ponerse en contacto con cada uno de los departamentos.

Si pulsa la opción 1, se pasará la llamada a la extensión del técnico del departamento técnico. Si no es atendida, pasará al Buzón de Voz de éste.

Si pulsa la opción 2, se pasará la llamada a las 2 extensiones del departamento comercial. Si no es atendida la llamada, será transferida a la secretaria. Si tampoco fuese atendida por esta, pasaría a su buzón de voz.

Si pulsa la opción 3, se pasará la llamada a las 2 extensiones del departamento de atención al cliente. Si no es atendida por ninguna de las extensiones, pasará la llamada al buzón de voz de la primera de las extensiones de este departamento.

- El flujo de las llamadas salientes será el siguiente:

  1. Las llamadas nacionales saldrán por los canales 1 y 2 de nuestra tarjeta analógica, los canales están conectados directamente a las 2 líneas de la PSTN.

  2. Las llamadas a móviles serán enviadas a través de los puertos 3 y 4 de nuestra tarjeta analógica, los canales están conectados a los FCT o enlaces GSM.

  3. Las llamadas internacionales serán enviadas a través de una conexión ADSL hacia un proveedor de voz sobre IP, debido a lo económico de sus precios.

Asignaremos la numeración de las extensiones con 4 dígitos, dejando extensiones libres entre las asignadas, para futuras ampliaciones. Las extensiones quedarían de la siguiente forma:

- Secretaria (2010)
- Dirección (2000)
- Departamento técnico (técnico 01 (2020) + responsable departamento (2029))
- Departamento comercial (técnico 01 (2030) + técnico 02 (2031))
- Departamento atención al cliente (técnico 01 (2040) + técnico 02 (2041))

Nota: *no se empieza a numerar a partir de la 1000 por existir coincidencia con algunos números de teléfonos existentes en el plan de numeración de algunos proveedores, como el 1004, etc.*

Nota: *se ha propuesto el uso de una tarjeta analógica en este sistema porque suele ser el caso más habitual, pero igualmente podría haber sido una RDSI con 4 BRI's. o Primario. A efectos de FreePBX sólo cambiaría la forma de crear los enlaces con el exterior o Trunk, que cambiaría el formato.*

4.2.1 Instalación de módulos

Los módulos que vienen instalados por defecto en FreePBX son los siguientes:

- **Administrators**. Usado para la creación y administración de permisos de usuarios de FreePBX.
- **Extensions**. Utilizado para la creación de extensiones y su administración.
- **Feature Codes**. Utilizado para la administración de algunas de las características/aplicaciones de Asterisk que por defecto vienen configuradas en FreePBX.
- **General Settings**. Se utiliza para la administración de parámetros generales a toda la centralita, como a quién enviar un correo en caso de que exista una actualización, tiempo máximo que van a estar sonando las extensiones antes de pasar al buzón o colgar la llamada, etc.
- **Outbound Routes**. Utilizado para configurar las rutas que tomarán las llamadas salientes según ciertos patrones.
- **Trunks**. Se utiliza para la administración de los distintos caminos/enlaces existentes con el exterior (canales RDSI, canales analógicos, canales IP...).
- **Inbound Routes**. Utilizado para la administración de las llamadas entrantes, es decir, qué hará una llamada entrante dependiendo de una serie de parámetros establecidos en este apartado.
- **Zap Channel DIDs**. En una llamada analógica entrante, no podemos disponer del número al que han llamado (DID - Direct Inward Dialing), simplemente podemos ver por qué canal de nuestra tarjeta analógica ha

entrado, dato que sí tenemos en RDSI. Pues con este módulo, podremos asignar un número de teléfono (DID) según el puerto por el que entre la llamada, y después podrá ser administrada esa llamada desde el módulo "Inbound Routes" según ese número. Esto sólo se usa para el entorno de administración de FreePBX y sólo tiene sentido para facilitar la gestión de llamadas entrantes si queremos tener controlada la línea por la que está entrando la llamada.

- **Music On Hold.** Esto se utiliza para administrar nuestra propia Música en Espera, o la que viene preinstalada.

- **System Recordings.** Desde aquí podremos crear nuestras propias grabaciones, bien desde una extensión, bien haciendo una descarga de alguna grabación que hayamos creado previamente por otros medios y queramos utilizarla en nuestra PBX.

Como se ha podido apreciar, con los módulos precargados, perfectamente se puede poner en funcionamiento una centralita con las funcionalidades más básicas, pero no cubriría los requerimientos del supuesto.

Para disponer de todas las funcionalidades necesarias en nuestro escenario, se tienen que instalar una serie de módulos antes de empezar a configurar la centralita.

Toda la parte de administración/configuración de centralita se realizará dentro de la pestaña *Admin* de FreePBX.

Para añadir los módulos necesarios, dentro del apartado "*Admin*", pulse en *Module Admin* (véase la figura 5-10).

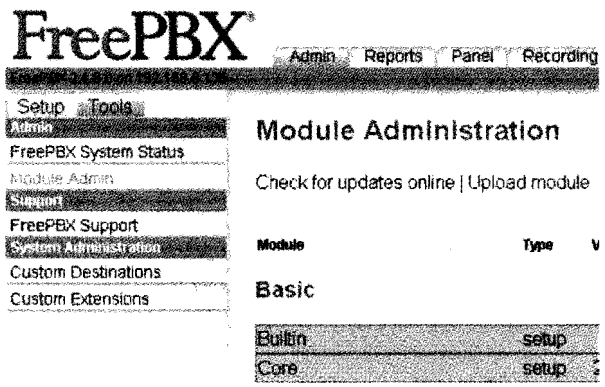


Figura 5-10. FreePBX (Administración de módulos)

Pulse en *Check for updates online* para que muestre los módulos instalados, pendientes por instalar y actualizaciones. A continuación pulse sobre los siguientes módulos y seleccionamos *Download and Install*:

- **Announcements.** Para poder administrar las locuciones de entrada de nuestra PBX (Saludos, Horarios de Trabajo, etc.).

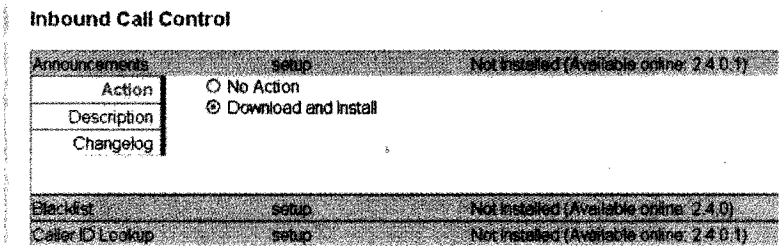


Figura 5-11. FreePBX (Inbound Call Control)

- **IVR.** Para administrar el menú de entrada de nuestra PBX.
- **Ring Groups.** Para administrar los grupos de extensiones.
- **Time Conditions.** Para administrar los horarios de entrada de las llamadas.
- **Call Forward.** Para administrar los desvíos de llamadas desde la PBX.
- **Call Waiting.** Para administrar las llamadas en espera desde la PBX.
- **Do-Not-Disturb (DND).** Para administrar el "No Molesten" desde la PBX.
- **Parking Lot.** Para administrar el "aparcamiento" de llamadas desde la PBX.

Para temas de gestión y administración del sistema, instalaremos los siguientes módulos:

- **Asterisk Logfiles.** Para la visualización de los Logs de Asterisk.
- **Asterisk CLI.** Para la ejecución de comandos de Asterisk en el CLI.
- **Asterisk Info.** Para ver el estado e información de la PBX (Extensiones, Canales Activos, etc.).

- **Bachup & Restore.** Para realizar copias de seguridad programadas de la configuración de la PBX.

- **Java SSH.** Para conectarse mediante un Cliente SSH al Servidor de Asterisk.

Por último, vaya al final de la página y pulse sobre el botón *Process* situado a la derecha para comenzar con la instalación de los módulos. Aparecerá una pantalla (véase la figura 5-12) con la lista de módulos que ha seleccionado para instalar, y pulse *Confirm*.

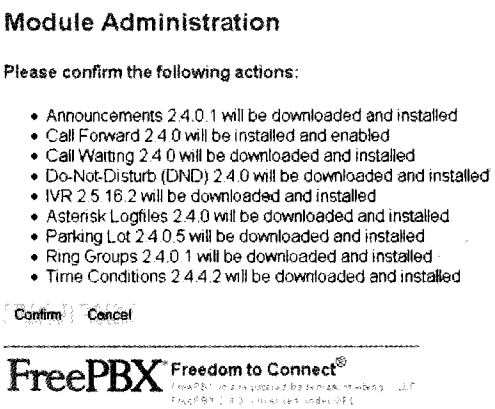


Figura 5-12. FreePBX (Instalación de módulos)

En estos momentos, y tal como muestra la figura 5-13, aparece una nueva ventana con la información de los procesos que se están realizando para la instalación de los módulos y si su resultado ha sido correcto o no.

Si todo ha ido bien, pulse sobre *Return*.

En la siguiente pantalla aparece la barra naranja, que comentamos al principio de este punto, informándonos de que debe aplicar los cambios realizados. Realizamos el proceso ya comentado para aplicar los cambios.

En este momento ya tiene todos los módulos necesarios para configurar la PBX según el escenario expuesto.

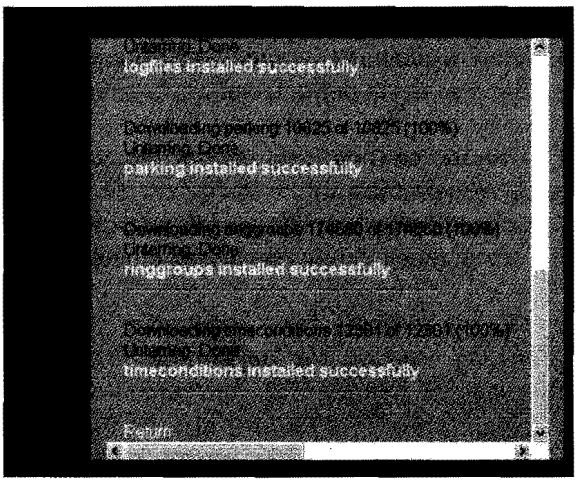


Figura 5-13. FreePBX (Información del proceso de la instalación)

Es importante tener en cuenta el orden en que se deben configurar los módulos, ya que unos hacen uso de otros.

Según los módulos que hay que configurar, el orden de configuración será el siguiente:

1. Trunks.
2. Extensions.
3. System Recordings.
4. Ring Groups.
5. IVR.
6. Announcements.
7. Time Conditions.
8. Zap Channel DIDs.
9. Inbound Routes.
10. Outbound Routes.
11. General Settings.

Para realizar la configuración del sistema, se mostrarán los cambios en cada uno de los módulos, teniendo otras muchas opciones que no se comentarán, ya que no son necesarias y se dejan por defecto.

El usuario interesado en todas estas opciones cuenta con una herramienta potentísima de ayuda. Al situarnos sobre cualquiera de las descripciones de las distintas opciones, nos muestra una ayuda en la que nos informa de para qué se utiliza esa opción y, en algunos casos, hasta nos muestra un ejemplo (véase la figura 5-14).

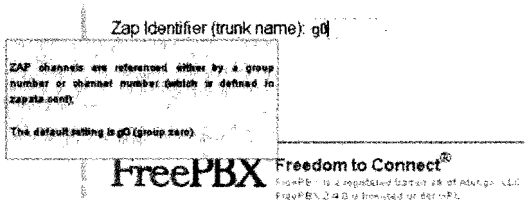


Figura 5-14. FreePBX (Ayuda emergente)

4.2.2 Configuración de Trunks

Las configuraciones de algunos tipos de *trunks* dependerán de la configuración establecida previamente en Asterisk, como por ejemplo los *Trunks Zap*, que depende de la configuración que establecimos en el fichero *Zaptel.conf* y el *Zapata.conf*.

En este ejemplo, se va a suponer que ha configurado en el fichero *Zapata.conf* los canales 1 y 2 asociados al *grupo 0*, y los canales 3 y 4 al *grupo 1*.

Es decir, en nuestro *zapata.conf* debe tener algo como:

```
.
.
.
group = 0
channel => 1,2
.
.
.
group = 1
channel => 3,4
```

A continuación se definen dos *Trunks Zap*, uno para las 2 líneas analógicas y otro para las 2 líneas que están conectadas a los FCT GSM.

Como ya se comentó, las líneas analógicas están conectadas a los canales 1 y 2 de la tarjeta, por lo que se corresponderá con el *grupo 0*, y los enlaces GSM en los canales 3 y 4, asociados al *grupo 1*.

En el menú del lateral izquierdo, pulse sobre la opción *Trunks*, y puede ver que, por defecto, FreePBX crea el *Trunk ZAP/g0*, donde "*g0*" se corresponde con *grupo 0*, que en el ejemplo será necesario para las 2 líneas analógicas, por lo que sólo hay que crear el *trunk* de los enlaces GSM.

Para ello, pulse sobre *Add ZAP Trunk* y nos desplazamos hasta *Zap Identifier (trunk name):*, donde escriba "*g1*", para que se corresponda con el "grupo 1" definido en nuestro *Zapata.conf*. A continuación pulse sobre *Submit Changes* para aceptar los cambios en esta pantalla y nos retornará a la pantalla anterior, donde ahora nos muestra los dos *Trunks Zap* definidos.

Estos son los dos *Trunks* que se utilizarán para las *llamadas nacionales* y las *llamadas a móviles*. Ahora queda por crear el Trunk para las *llamadas internacionales*.

Como ya se comentó, las llamadas internacionales saldrán a través de un enlace que crearemos contra un *Operador de VoIP (Voz sobre IP)*. Este enlace se realizará con el protocolo *SIP*, que es una condición que pone el operador, por lo que el trunk que se creará será un *Trunk SIP*.

Previamente a la creación de este trunk, debe hacer algunos cambios en el sistema para que nuestra PBX sea visible al proveedor desde Internet.

Modifique el fichero *sip_general_custom.conf* y añada las siguientes líneas:

```
nat = yes
externip = 80.80.80.80              ; Nuestra IP Externa Fija
localnet = 192.168.1.0/255.255.255.0     ; Nuestra Red Interna
de Asterisk.
```

En caso de no disponer de *Dirección IP Externa Fija*, hay que dar de alta el *host* en un DNS Público, como *DynDNS*, y añadir las siguientes líneas, en vez de las anteriores:

```
nat = yes
exterhost = host.dyndns.org         ; Nuestro Host en DynDNS
localnet = 192.168.1.0/255.255.255.0     ; Nuestra Red Interna
de Asterisk.
```

A parte, debemos abrir y direccionar los siguientes puertos UDP en el router hacia nuestro Asterisk:

```
Puerto UDP: 5060                    ; Señalización SIP
```

```
Rango UDP: 10000 al 20000   ; Audio RTP
```

Estos puertos son los definidos por defecto y pueden ser modificados, según necesidades, mediante los ficheros *rtp.conf* y *sip_general_custom.conf*.

Teniendo hecho todo lo anterior, se va a crear el *Trunk SIP*, por lo que se tiene que pulsar sobre *Add SIP Trunk* y crear el *trunk* de la siguiente forma:

```
Outgoing Settings
Trunk Name: Proveedor-SIP          ;(Podemos darle cualquier
nombre)
PEER Details:
 disallow = all
 allow = gsm&ulaw&alaw
 canreinvite = no
 context = from-zaptel
 dtmfmode = rfc2833
 fromdomain = 123.123.123.123      ;(IP ó Dominio del Proveedor
VoIP)
 fromuser = USUARIO          ;(El usuario creado por el
Proveedor VoIP)
host = 123.123.123.123
;(IP del Proveedor VoIP)
 insecure = very
 nat = yes
 qualify = yes
 rtptimeout = 30
 secret = PASSWORD ;(Password del USUARIO creado por el
Proveedor VoIP)
 type = peer
 username = USUARIO  ;(El usuario creado por el Proveedor
VoIP)
Registration
Register String: USUARIO:PASSWORD@123.123.123.123
```

***Nota:** todos los parámetros expuestos anteriormente han sido explicados en capítulos anteriores para la configuración de Asterisk sin asistente gráfico.*

Finalmente pulse en *Submit Changes* para aceptar los cambios que hemos añadido.

Es interesante resaltar que en este módulo hay opciones muy interesantes y que nos pueden ayudar en muchas situaciones que se nos pueden plantear. Por ejemplo, podemos configurar que sólo puedan salir por el Trunk en cuestión llamadas que vayan dirigidas a números que cumplan un determinado patrón, o quitarle dígitos al número destino antes de sacarlo por este enlace, o añadirle

prefijos, etc. Las 2 primeras opciones las podremos hacer mediante la opción "*Dial Rules*" y la última mediante "*Outbound Dial Prefix*".

En la pantalla principal de *Trunks* que se muestra en la figura 5-15 aparecen los tres enlaces que se han creado.

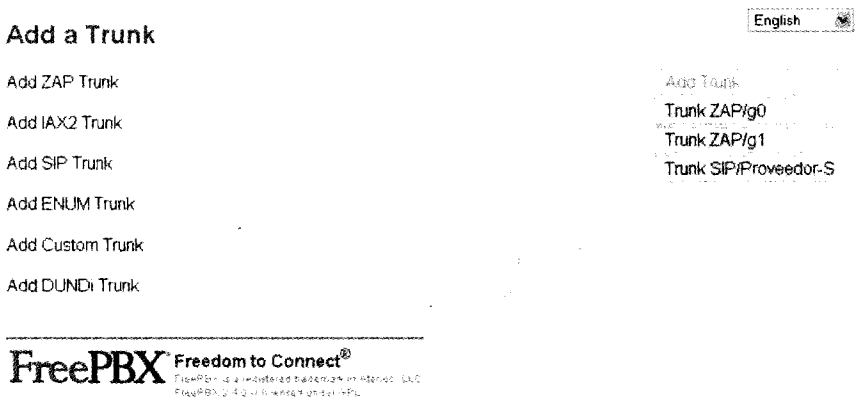


Figura 5-15. FreePBX (Trunks)

En este momento se ha terminado de añadir los enlaces con el mundo exterior, tan sólo queda aplicar los cambios en la configuración pulsando sobre la barra naranja *Apply Configuration Changes* y seguir los pasos de siempre.

Por ejemplo, ahora puede ver si se nos ha registrado el enlace SIP con el proveedor. Para ello, pulse en *Tools* y a continuación en el módulo *Asterisk CLI* en el apartado *System Administration*.

En la ventana que aparece, escriba en el campo *Command:* el comando de Asterisk *sip show peers* y pulse *Execute*. Si todo ha ido bien, nos debe aparecer algo similar a lo que aparece a continuación:

```
Name/username     Host    Dyn Nat ACL Port     Status
```

Esto significa que nuestro enlace SIP (*Proveedor-SIP*) se ha registrado con el usuario (*USUARIO*) correctamente (*OK (58 ms)*) en el puerto *5060* con el host *123.123.123.123*.

4.2.3 Configuración de *Extensions*

La creación y configuración de extensiones se realiza mediante el módulo *Extensions* de FreePBX. Para configurar las extensiones del sistema pulse sobre el

módulo *Extensions*, en la pestaña *Setup*, y que se encuentra dentro del apartado *Basic* de FreePBX, para añadir cada una de las 8 Extensiones de nuestro escenario.

Las extensiones que puede crear son de varios tipos: *SIP*, *IAX2* y *Zap*; pero en el ejemplo serán todas SIP.

En *Device* seleccione *Generic SIP Device* y pulse *Submit*.

Para crear la primera extensión debe escribir los siguientes campos:

```
Extensión 2000

Add Extension
User Extension: 2000
Display Name: Direccion

Device Options
secret: 0000

Voicemail & Directory
Status: Enabled
Voicemail Password: 0000
Email Address: <email@email.com>   ;Correo al que se enviará el
                                   ;mensaje.
Email Attachment: yes              ;Para que mande por correo
                                   ;el mensaje.
Play CID: yes                      ;Reproducir el Caller ID
                                   ;antes del mensaje.
Play Envelope: yes                 ;Reproducir la Fecha/Hora
                                   ;antes del mensaje.
```

Pulse *Submit* para guardar los cambios en esta Extensión.

A continuación debe realizar el mismo proceso con las siete extensiones restantes. Utilice los mismos valores de la primera extensión, a excepción de los dos primeros, que serán sustituidos por los siguientes:

```
Extensión 2010
Add Extension
User Extension: 2010
Display Name: Secretaria
Email Address: <email@email.com> ;Correo al que se enviará el
                                 ;mensaje.

Extensión 2020
Add Extension
User Extension: 2020
Display Name: Tecnico 01
```

```
Email Address: <email@email.com> ;Correo al que se enviará el
                                 ;mensaje.

Extensión 2029
Add Extension
User Extension: 2029
Display Name: Responsable Tecnico
Email Address: <email@email.com> ;Correo al que se enviará el
                                 ;mensaje.

Extensión 2030
Add Extension
User Extension: 2030
Display Name: Comercial 01
Email Address: <email@email.com> ;Correo al que se enviará el
                                 ;mensaje.

Extensión 2031
Add Extension
User Extension: 2031
Display Name: Comercial 02
Email Address: <email@email.com> ;Correo al que se enviará el
                                 ;mensaje.

Extensión 2040
Add Extension
User Extension: 2040
Display Name: Atencion Cliente 01
Email Address: <email@email.com> ;Correo al que se enviará el
                                 ;mensaje.

Extensión 2041
Add Extension
User Extension: 2041
Display Name: Atencion Cliente 02
Email Address: <email@email.com> ;Correo al que se enviará el
                                 ;mensaje.
```

Nota: *para que el sistema envíe el mensaje de voz al correo electrónico, previamente debe instalar un servidor de correo, como puede ser Sendmail o Postfix.*

El formato del correo, la dirección de origen, el cuerpo del mensaje, etc. se configuran en los ficheros:

```
/etc/asterisk/vm email.inc
/etc/asterisk/vm_general.inc
```

Finalmente aplique los cambios de configuración pulsando en la barra naranja.

Si a continuación intenta modificar cualquiera de las extensiones que se han creado, observará que aparece una gran cantidad de opciones, en el apartado *Device Options*, que no aparecían en la ventana cuando estaba creando esta extensión. De esta manera, puede cambiar opciones de Asterisk, comentadas en capítulos anteriores, como *port, callgroup, pickupgroup, codecs permitidos, etc.*

Esta es la parte más tediosa, ya que dependiendo del número de extensiones a añadir, debido a lo monótono que es, se hace interminable.

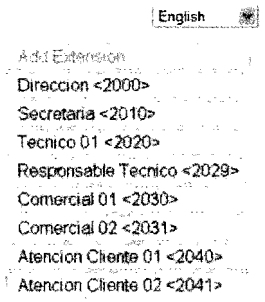


Figura 5-16. FreePBX (Extensiones)

Ahora ya puede configurar los diferentes dispositivos y registrarlos en la PBX.

Al igual que cuando se crea los *Trunks*, se accede al módulo *Asterisk CLI* y ejecute el comando de Asterisk *sip show peers*. Ahora debe aparecer, además, cada una de las extensiones que ha creado y los dispositivos que ha configurado deben aparecer registrados:

```
Name/username            Host            Dyn Nat ACL Port     Status
Proveedor-SIP/USUARIO    123.123.123.123      N       5060     OK (90 ms)
2041                     (Unspecified)   D    N       0        UNKNOWN
```

4.2.4 Configuración de *System Recordings*

Hay que crear cuatro locuciones para el escenario planteado:

- **Horario.** Locución que se reproducirá cuando entre una llamada fuera de horario habitual de oficina.

- **Mañana.** Locución que reproduciremos cuando nos entre una llamada en horario de mañana. Lo denominaremos *Manana* para omitir la "***ñ***" y que no nos dé error en el nombre al grabar la locución.

- **Tarde.** Locución que se reproducirá cuando entre una llamada en horario de tarde.

- **Menu.** La locución que nos dirá las distintas opciones que hay que marcar para ponerse en contacto con cada departamento.

Lo primero que hay que hacer es grabar las locuciones para después utilizarla desde los módulos correspondientes.

La grabación la puede realizar desde el mismo módulo *System Recordings* o bien con medios externos y posteriormente pasar las grabaciones a la PBX mediante este módulo.

A continuación se va a realizar primero una grabación de una locución desde el sistema. Esto se realiza grabando la locución desde una de las extensiones que hay creadas y registradas ya en la centralita.

Acceda al módulo *System Recordings* e indique que va a realizar la grabación desde la extensión, por ejemplo, 2030 y pulse *Go*.

Desde la extensión 2030 marque ***77** para grabar la primera de las locuciones. Una vez terminada la grabación, cuelgue la llamada.

Si desea escuchar la grabación, desde la misma extensión de grabación, pulse ***99** y se nos reproducirá lo que ha grabado.

A continuación puede colgar y repetir el proceso si no ha quedado la grabación bien o colgar y seguir adelante.

Para finalizar con el proceso de la grabación de la primera locución, en el campo *Name this Recording* escriba el nombre de la grabación *Horario* y pulse en *Save* para finalizar con la grabación de la primera locución.

Para grabar la siguiente locución *(Mañana)*, hay que indicar desde qué extensión se va a realizar la grabación. Si la va a realizar desde la misma extensión que la anterior, sólo habrá que marcar el ***77**, realizar la grabación, colgar y continuar con el proceso, poniendo el nombre *Manana* pulsando *Save* para que quede en nuestra PBX.

Para la locución *Tarde* debe realizar la misma tarea descrita en el párrafo anterior, evidentemente, cambiaremos el nombre.

Por último, grabe la locución *Menu*, pero esta vez, se realizará con otro método, es decir, se realiza la grabación con medios externos y después se guarda en la PBX.

Si se realiza esta grabación con medios externos, lo puede hacer, por ejemplo, en formato *WAV PCM Encoded, 16 Bits, at 8000Hz*.

A continuación, desde el módulo *System Recordings* pulse en *Examinar*, en la ventana que aparece, seleccione el fichero que acabamos de guardar, pulse *Abrir* y finalmente, pulse *Upload* para transferir el fichero a nuestra PBX.

Tras haber pulsado *Aceptar* en la ventana informativa que aparece, escribimos *Menu* como nombre a la locución y pulse *Save* para terminar con el proceso.

Tal y como puede ver en la figura 5-17, ya tiene todas las locuciones que va a necesitar grabadas en nuestra centralita. Si pulsa sobre alguna de ellas, puede añadirle algún comentario, eliminar la locución, añadirle más ficheros a la misma locución, etc.

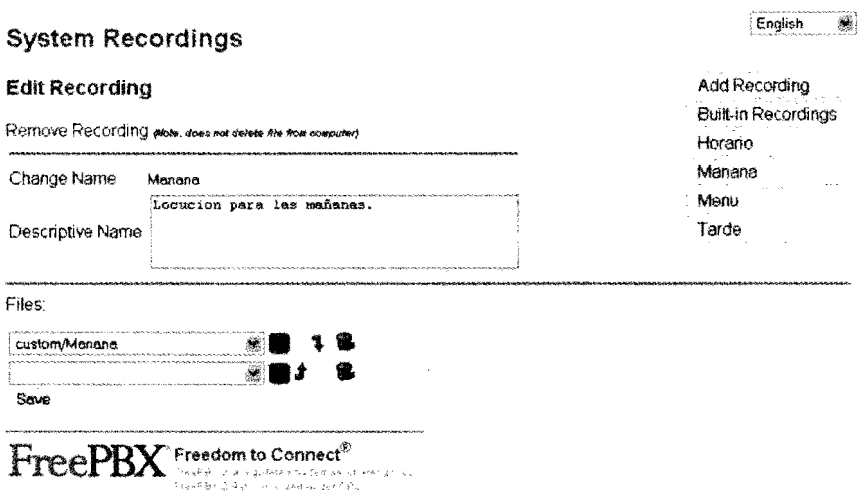


Figura 5-17. FreePBX (System Recordings)

***Nota:** hasta aquí, el orden de los módulos configurados no tendría que haber sido este, ya que ninguno depende de otro de ellos para configurarse, pero los que vienen a continuación sí que dependen de los anteriores para ser configurados.*

4.2.5 Configuración de *Ring Groups*

A continuación, se van a configurar las agrupaciones de extensiones necesarias para realizar el comportamiento en el escenario planteado para algunas situaciones.

Las situaciones en cuestión son las siguientes:

1. Si se pulsa la opción 2, se pasará la llamada a las 2 extensiones del departamento comercial. Si no es atendida la llamada, será pasada la llamada a la secretaria.

2. Si se pulsa la opción 3, se pasará la llamada a las 2 extensiones del departamento de atención al cliente. Si no es atendida por ninguna de las extensiones, pasará la llamada al buzón de voz de la primera de las extensiones de este departamento.

Para ambas situaciones, hay que crear dos grupos de extensiones, uno para cada situación. El módulo *Ring Groups* de FreePBX se utiliza para realizar estas tareas.

A continuación, para la *Situación 1*, modifique los siguientes campos:

```
Add Ring Group
Ring-Group Number: 203
Group Description: Dto Comercial
Extension List: 2030
                2031

Destination if no answer:
Extensions: <2010> Secretaria
```

Se ha creado el grupo de extensiones *203* que lo forman las extensiones *2030* y *2031* del *departamento comercial*, y cuando entre una llamada, si no se contesta se envía a la *extensión de la secretaria*.

Hay otras tantas opciones que se dejan por defecto, y con las que el usuario puede jugar, como pueden ser *Ring Strategy*, con el que se puede configurar en la forma y orden que sonarán las extensiones, o puede hacer que salte una locución cuando la llamada llegue a este grupo, con la opción *Announcement*, locución que debe haber sido grabada con anterioridad según el método ya comentado en la configuración del módulo *System Recordings*. Todas las opciones disponen de una ayuda bastante clara al situarse el ratón sobre las etiquetas.

Hay que comentar que el orden en la *Lista de la Extensiones* interviene en el caso que se escoja otra estrategia a la elegida por nosotros, "que suenen todas a la vez".

A continuación, para la *Situación 2*, modifique los siguientes campos:

```
Add Ring Group
Ring-Group Number: 204
Group Description: Dto Atencion Cliente
Extension List: 2040
                2041

Destination if no answer:
Voicemail: <2040> Atencion Cliente 01 (busy)
```

En este caso hemos creado el grupo de extensiones 204 que lo forman las extensiones *2040* y *2041* del departamento atención al cliente, y cuando entre una llamada, si no se contesta, se envía al buzón de la primera extensión (2040).

Como se acaba de ver, en estos módulos se han utilizado configuraciones previas de otros módulos. Por ejemplo, no se puede configurar los grupos de llamadas si antes no se hubieran creado las extensiones.

4.2.6 Configuración de *IVR*

En este punto se va a configurar el *IVR* o *Menú Vocal*, también llamada *Recepcionista Digital*, con las distintas opciones que se plantean en el escenario.

Acceda al módulo *IVR* y pulse en *Add IVR* y realice los siguientes cambios:

```
Change Name: Menu-00
Enable Directory: DESMARCADA
(Con esta opción se evita que se pueda acceder al directorio,
si lo hubiese, de la compañía)
Enable Direct Dial: MARCADA
(Con esta opción se permite que se pueda teclear directamente
la Extensión si se conoce)
Announcement: Menu
```

A continuación se van a crear las distintas opciones planteadas para el menú del escenario:

1. Si se pulsa la opción 1, se pasará la llamada a la extensión del técnico del departamento técnico.

2. Si se pulsa la opción 2, se pasará la llamada a las 2 extensiones del departamento comercial.

3. Si se pulsa la opción 3, se pasará la llamada a las 2 extensiones del departamento de atención al cliente.

4. Esperar y que la llamada sea atendida por la secretaria.

Por defecto, el sistema permite la posibilidad de introducir tres opciones en nuestro *IVR*, pero puede añadir más o eliminar según sus necesidades:

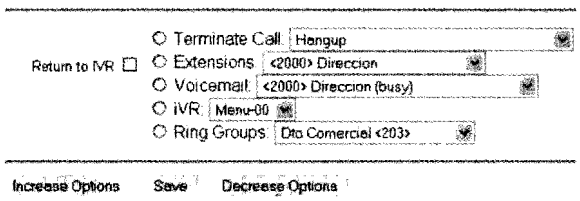


Figura 5-18. FreePBX - IVR

Tal y como muestra la figura 5-18, en el recuadro que vemos debajo de la opción *Return to IVR* es donde introducimos el número que hay que pulsar para que se ejecute la acción que seleccionemos de la derecha.

Comentado esto, las tres primeras opciones las realizamos con los siguientes datos:

```
Opción 1
Número a pulsar: 1
Acción: Extensions: <2020> Tecnico 01

Opción 2
Número a pulsar: 2
Acción: Ring Groups: Dto Comercial <203>

Opción 3
Número a pulsar: 3
Acción: Ring Groups: Dto Atencion Cliente <204>
```

Para crear la cuarta opción, pulse en *Increase Options* y aparecerá una nueva opción en blanco al final. La realizamos con los siguientes datos:

```
Opción 4
Número a pulsar: t
Acción: Extensions: <2010> Secretaria
```

Con la opción "*t*" está indicando que si se pulsa la "*t*" pasará la llamada a la secretaria, sino que si expira el tiempo establecido en *Timeout*, que por defecto se ha dejado en 10 segundos, pase la llamada a la secretaria. Como ya sabe, por capítulos anteriores, "*t*" es una de las extensiones especiales usadas por Asterisk.

También puede utilizar la extensión "*i*" en el IVR si quiere realizar alguna acción en concreto si la pulsación que se produce es incorrecta.

Y por último, para guardar el IVR que ha creando, pulse en *Save* y posteriormente aplique la configuración a Asterisk, como siempre, hay que actualizar los cambios en FreePBX.

4.2.7 Configuración de *Announcements*

Los *anuncios* o *Announcements* reproducen una locución y a continuación ejecutan una acción. Esto lo usaremos para varios casos en el supuesto, por ejemplo, para que las llamadas que se realizan fuera del horario se les reproduzca la locución y a continuación pase la llamada al buzón de voz de la secretaria, y para dar la bienvenida por la mañana o por la tarde, dependiendo de la hora de la llamada, y que después pase al Menú Vocal.

Para ello, acceda al módulo *Announcements* de FreePBX.

Para el caso de que la llamada entre fuera de horario de oficina, introduzca los siguientes datos en los distintos campos:

```
Add Announcement
Description: Horario de Oficina
Recording: Horario
Repeat: Disable

Destination after playback:
Voicemail: <2010> Secretaria (unavail)
```

Y a continuación pulse *Submit Changes* para grabar este *Anuncio*.

Con esto, se reproduce la grabación *Horario*, que ya se grabó desde el módulo *System Recordings*, y posteriormente, pasa la llamada al buzón de voz de la secretaria, dando la locución de que no está disponible y que se deje un mensaje.

Ahora sólo queda grabar el *Anuncio* para las llamadas que entren por la mañana y por la tarde.

Los datos a cumplimentar para ambos anuncios son los siguientes:

- Llamadas en Horario de Mañana.-

```
Add Announcement
Description: Horario de Manana
Recording: Manana
Repeat: Disable

Destination after playback:
IVR: Menu-00

Submit Changes
```

- Llamadas en Horario de Tarde.-

```
Add Announcement
Description: Horario de Tarde
Recording: Tarde
Repeat: Disable

Destination after playback:
IVR: Menu-00

Submit Changes
```

Aplicamos los cambios y hemos terminado con la configuración de los "*Anuncios*".

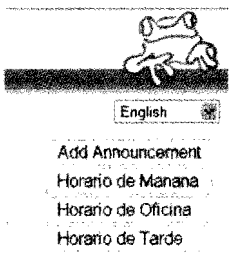


Figura 5-19. FreePBX - Anuncios

Como puede ver, el orden de configuración de los módulos tiene su importancia, ya que, por ejemplo, si no ha definido ningún *IVR*, no llegaría ni a aparecer la opción de enviar la llamada al *IVR* después de reproducir la locución.

4.2.8 Configuración de *Time Conditions*

Definidos ya los anuncios, se van a definir ahora los horarios de atención de llamadas según nuestro caso particular.

El horario que se ha planteado es el siguiente:

- **Mañana:** 08:00 a 14:00
- **Tarde:** 16:00 a 18:00
- **Días:** Lunes a Viernes

En la figura 5-20 se puede ver un diagrama del orden o secuencia de comprobación del supuesto.

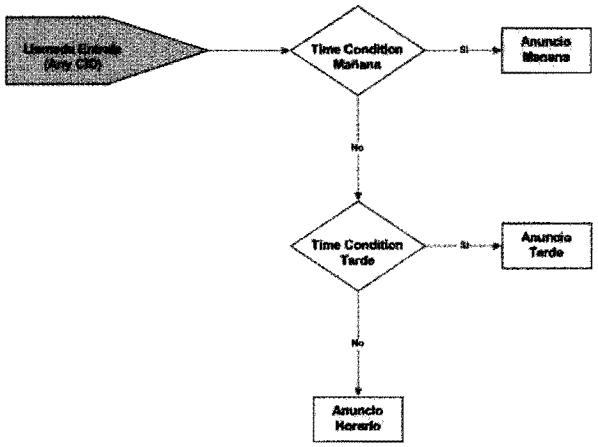


Figura 5-20. Secuencia de comprobación de anuncios

Es decir, cuando entra una llamada se comprueba si se encuentra en horario de mañana y se reproduce el anuncio de mañana. Si no se encuentra en horario de mañana comprueba si es el horario de tarde y si es el horario de tarde, pasa la llamada el anuncio de la tarde; sino, quiere decir que está fuera de horario de oficina, ya que no está ni en el horario de mañana ni en el de tarde, por lo tanto, pasa a la llamada al anuncio *Horario de Oficina.*

Para realizar la configuración de las condiciones, se empieza por la última condición, ya que desde la primera la tendremos que mandar a la segunda condición, tal y como se muestra en el diagrama de la figura 5-20.

Acceda al módulo *Time Conditions* de FreePBX, pulse en *Add Time Condition* y rellene los campos de la segunda condición con los siguientes datos:

```
Add Time Condition
Time Condition name: Horario Tarde
Time to start: 16:00
Time to finish: 18:00
Week Day Start: Monday
Week Day finish: Friday
Month Day start: 1
Month Day finish: 31
Month start: January
Month finish: December

Destination if time matches:
Announcements: Horario de Tarde

Destination if time does not matches:
Announcements: Horario de Oficina

Submit Changes
```

Para añadir la primera condición, rellene los campos con los siguientes datos:

```
Add Time Condition
Time Condition name: Horario Manana
Time to start: 08:00
Time to finish: 14:00
Week Day Start: Monday
Week Day finish: Friday
Month Day start: 1
Month Day finish: 31
Month start: January
Month finish: December

Destination if time matches:
Announcements: Horario de Manana

Destination if time does not matches:
Time Conditions: Horario Tarde

Submit Changes
```

Por último aplique la configuración a Asterisk pulsando en *Apply Configuration Changes* y ya tendrá las condiciones creadas tal y como muestra la figura 5-21.

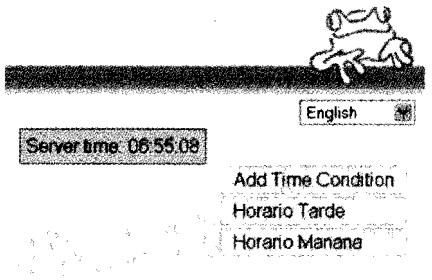


Figura 5-21. FreePBX – Aplicar configuración de anuncios

4.2.9 Configuración de *Zap Channel DIDs*

Tal y como se expuso en la descripción del supuesto, se dispone de los siguientes elementos hardware: una tarjeta analógica con 4 puertos FXO, dos de ellos conectados directamente a dos líneas telefónicas, con dos números distintos, y los otros dos FXO estarán conectados a dos enlaces FCT GSM.

Supongamos los números:

- Línea 1 -----> 954xxxx01
- Línea 2 -----> 954xxxx02

En las llamadas analógicas entrantes no se dispone de la información del número llamado (DID), es decir, si nos entra una llamada en nuestra PBX, no es posible conocer el número, *954xxxx01 ó 954xxxx02*, al que el remoto llamó; como mucho, puede averiguar por qué puerto FXO de nuestra tarjeta entró la llamada.

Por lo tanto, resulta imposible discriminar las llamadas entrantes según el número DID al que han llamado. Lo que sí se puede hacer es distinguir por qué puerto entra la llamada. Para ello se asocia un valor al DID según el puerto FXO por el que entra la llamada, y posteriormente se discrimina por este número DID. Es decir, si la llamada entra por el puerto ***FXO 01***, la *Línea 1*, el DID que se asigne para esa llamada será *954xxxx01* y si entra por el ***FXO 02***, la *Línea 2*, el DID será *954xxxx02*. A partir de aquí, podría gestionar las llamadas según el DID.

Estas tareas las puede realizar mediante el módulo *Zap Channel DIDs* de FreePBX.

Para configurar estos detalles, acceda al módulo y rellene los campos según los datos que se detallan a continuación:

```
Add Channel
Channel: 1
```

```
Description: Linea 1
DID: 954xxxx01

Submit Changes

Add Channel
Channel: 2
Description: Linea 2
DID: 954xxxx02

Submit Changes
```

Los enlaces GSM se asignan de la forma:

```
Add Channel
Channel: 3
Description: GSM 1
DID: 6xxxxxx01

Submit Changes

Add Channel
Channel: 4
Description: GSM 2
DID: 6xxxxxx02

Submit Changes
```

El último paso, como siempre, es aplicar la configuración a Asterisk y, tal como muestra la figura 5-22, podrá ver los canales creados en el sistema.

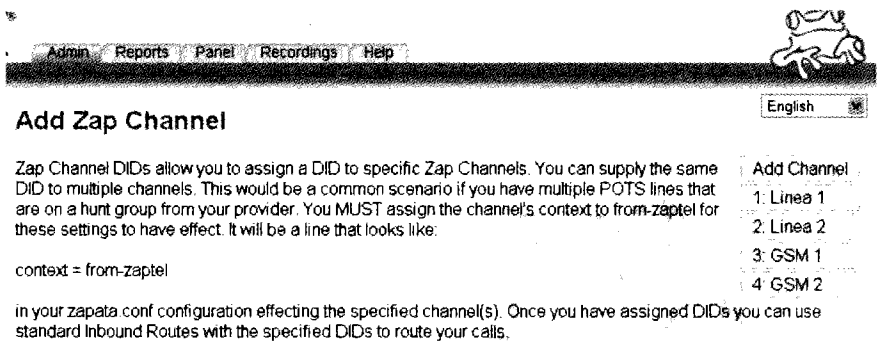


Figura 5-22. FreePBX – Canelez Zap

Sólo nos queda un detalle para poder utilizar esta configuración, y es que, en el fichero *zapata.conf* o *zapata-channels.conf*, los canales deben estar asignados al contexto *from-zaptel*, es decir, debe aparecer algo similar a lo siguiente:

```
context=from-zaptel
channel => 1-4
```

En estos momentos ya podemos hacer discriminaciones según el número al que está llamando la gente.

4.2.10 Configuración de *Inbound Routes*

Ahora únicamente queda configurar el nexo entre las líneas externas y toda la configuración, para que siga el proceso de actuación que hemos venido definiendo.

Esta tarea se realizará a través del módulo *Inbound Routes*. Acceda a este módulo y configure el sistema con los siguientes datos:

```
Add Incoming Route
Description: Linea 1
DID Numbre: 954xxxx01

Set Destination
Time Conditions: Horario Manana

Submit Changes

Add Incoming Route
Description: Linea 2
DID Numbre: 954xxxx02

Set Destination
Time Conditions: Horario Manana

Submit Changes

Add Incoming Route
Description: GSM 1
DID Numbre: 6xxxxxx01

Set Destination
Time Conditions: Horario Manana

Submit Changes
```

```
Add Incoming Route
Description: GSM 2
DID Numbre: 6xxxxxx02

Set Destination
Time Conditions: Horario Manana

Submit Changes
```

Como puede observar, todas las llamadas siguen el mismo camino, cuando entran van a parar al módulo *Time Conditions Horario Manana*, y desde aquí comienza toda la secuencia.

En esta fase, el usuario puede, por ejemplo, realizar alguna discriminación según la línea por la que entra una llamada, mandarla directamente a dirección o al departamento técnico, etc.

Si no ha pensado realizar estas discriminaciones, puede añadir una única *Inbound Route* y que todo lo que llegue sea enviado al módulo *Time Conditions Horario Manana*, sin tener que añadir las cuatro anteriores.

Si lo desea así, los datos para la única "*Inbound Route*" que habría que añadir son los siguientes:

```
Add Incoming Route
Description: TODO

Set Destination
Time Conditions: Horario Manana

Submit Changes
```

En este caso, al realizar el *Submit Changes* nos avisa de que esta ruta afectará a todas las llamadas entrantes.

Un campo que puede ser interesante es *Caller ID Number*, que es el número del llamante. Puede definir alguna ruta de entrada que compruebe el número desde el que se está llamando, y si coincide con el dado, pasarlo directamente a dirección, o al departamento técnico, etc.

Finalmente aplique la configuración a Asterisk.

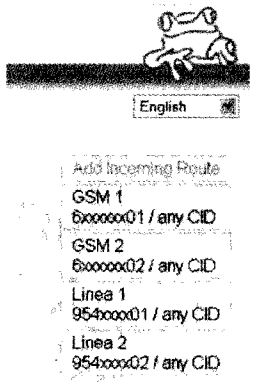


Figura 5-23. FreePBX – Incomming Route

Este es el momento la centralita está totalmente operativa para recibir llamadas desde el exterior y que sigan la operatoria que se ha definido.

No se pueden realizar llamadas salientes aún, ya que no están aún configurados los caminos de salida para estas llamadas.

4.2.11 Configuración de *Outbound Routes*

En este punto se va a configurar el flujo de las llamadas salientes. Las llamadas nacionales deben salir por los dos primeros canales FXO de la tarjeta, las llamadas a móviles salen por los dos últimos FXO que es donde están conectados los 2 FCT GSM, y las llamadas internacionales, por el proveedor de VoIP.

Por defecto, FreePBX crea una ruta de salida **0** llamada *9_outside*. Acceda al módulo *Outbound Routes* y eliminémosla pinchando sobre ella y a continuación pulse en *Delete Route 9_outside*". Aplique la configuración para que sea reconocida por Asterisk.

Por lo tanto, ahora hay que añadir las tres rutas de salida. Acceda de nuevo al módulo *Outbound Routes* y cumplimente los campos de la siguiente forma:

```
Ruta Nacionales
Route Name: Nacionales
Dial Patterns:
010
012
061
083
```

```
091
092
112
118XX
1XXX
[12345789]XXXXXXXX
(O cualquier otro que nos venga bien)

Truk Sequence: ZAP/g0

Submit Changes

Ruta Móviles
Route Name: Moviles
Dial Patterns:
6XXXXXXXX
(O cualquier otro que nos venga bien)

Truk Sequence: ZAP/g1

Submit Changes


Ruta Internacionales
Route Name: Moviles
Dial Patterns:
00.
(O cualquier otro que nos venga bien)


Truk Sequence: SIP/Proveedor-SIP
Submit Changes
```

Tal y como muestra la figura 5-24, puede ver los códigos que puede utilizar en el *Dial Pattern;* puede verlos situándose sobre la etiqueta *Dial Patterns*.

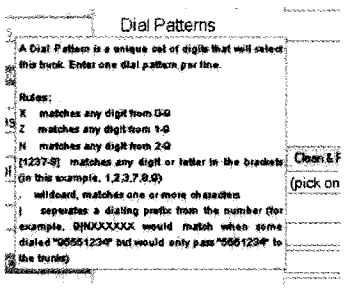


Figura 5-24. FreePBX – Códigos de Dial Patterns

Por último, aplique los cambios para que sean reconocidos por Asterisk.

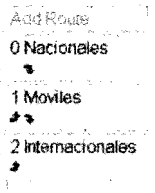


Figura 5-25. FreePBX - Routes

Ahora sí tiene totalmente configurada la centralita PBX en cuanto a llamadas entrantes y salientes. Ya puede ponerla en pruebas e ir afinando configuraciones según sus necesidades.

4.2.12 Configuración de *General Settings*

Por último, existe un módulo en FreePBX que ayuda a configurar parámetros generales de la PBX denominado *General Settings*.

Con la configuración de este módulo, puede configurar parámetros como:

- Tiempo que suena una extensión antes de ser enviada la llamada al buzón.
- Si se permite o no la transferencia de la llamada al que llama.
- Si se permite o no la transferencia de la llamada al que es llamado.
- Si se permiten llamadas SIP entrantes anónimas.
- Configuración regional.
- Formato de hora.
- Etc.

Para el ejemplo, sólo tiene que cambiar un par de parámetros. Para ello acceda al módulo *General Settings* y cambie los siguientes campos:

```
Dialing Options
Asterisk Outbound Dial command options: T

International Settings
Country Indications: Spain

Online Updates
Update Email: <Nuestro Email>
```

La anterior configuración permite que las llamadas salientes se puedan transferir, además de establecer las indicaciones para España y configurar una cuenta de correo a la que mandar la información de actualizaciones disponibles para ser instaladas.

Hasta aquí ha llegado la configuración de nuestra PBX mediante el interface gráfico FreePBX, quedando totalmente operativa y en producción.

4.2.13 Otros módulos interesantes

Se puede encontrar en la situación de que el cliente quiera tener la opción de modificar la secuencia de entrada de llamadas, es decir, si un día se queda alguien en la oficina fuera del horario habitual, tenga la posibilidad de que entren las llamadas a la extensión 2010 de la secretaria en vez de saltar el buzón, y retornarlo a su situación normal cuando se marchen.

Bien, FreePBX dispone para estas situaciones de un módulo, que no se instala por defecto, llamado *Day/Night Control* y que se utiliza para permitir estas situaciones. Este módulo es una especie de "interruptor" para activar la configuración especial o dejar la habitual simplemente con realizar una pulsación desde el teléfono.

```
Day / Night Mode Control
Day/Night Feature Code: 0
Description: Activa Secretaria
Current Mode: Day

Day
Time Conditions: Horario Manana

Night
Extensions: <2010> Secretaria

Save
```

donde *Day/Night Feature Code* es el índice y puede tener hasta 10 índices o interruptores.

A continuación hay que cambiar las Inbound Routes y direccionar las llamadas a:

```
Day / Night Mode: (0) Activa Secretaria
```

en vez de "*Time Conditions*" al que se dirigía la llamada entrante.

Para activar o desactivar la *Modalidad Día / Noche* hay que pulsar "**28<indice>*" en cualquier extensión. De esta forma, si ha creado el índice 0 (*Day/Night Feature Code: 0*), hay que pulsar "**280*" para Activar / Desactivar esta configuración, y se reproduce una locución indicando si la hemos activado o desactivado.

Habrá que acordarse siempre de retornar el "*interruptor*" a su estado de operatoria habitual cuando nos marchemos de la oficina.

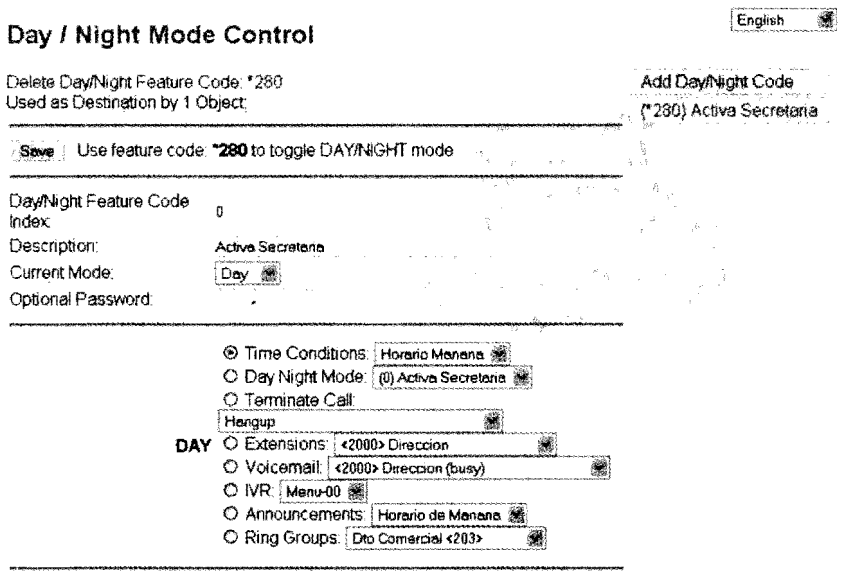


Figura 5-26. FreePBX – DayNight Code

Otra de las situaciones que nos podemos encontrar es que en un momento dado se necesite agregar una extensión, por ejemplo, o cambiar algún dato de las existentes, o cosas triviales que podría hacer un usuario avanzado sin la necesidad de que tenga que intervenir el administrador.

Si se permite el acceso a la configuración de FreePBX a cualquier usuario se corre el riesgo de que la PBX deje de funcionar por una manipulación indebida.

FreePBX cuenta con el módulo *Administrators*, el cual permite agregar usuarios y otorgar los permisos sobre las acciones que pueden realizarse en FreePBX.

En primer lugar, y antes de configurar este módulo, hay que permitir esta opción editando el fichero */etc/amportal.conf* y cambiando la variable *AUTHTYPE=none* por *AUTHTYPE=database*, como se indica al entrar en el módulo *Administrators*.

A continuación, y después de reiniciar FreePBX, verá que el sistema ahora le pide un nombre de usuario y contraseña. Por defecto, FreePBX crea un usuario *admin* con la contraseña *admin* y que tiene todos los permisos.

Al entrar en el sistema acceda al módulo *Administrators* y cambie la contraseña por defecto.

A continuación puede crear los usuarios que desea y establezca los permisos que estimemos oportunos, por ejemplo, para añadir un usuario en *Admin Access* seleccione la opción *Extensions*, y manteniendo la tecla Control pulsada, pulse también el *Apply Changes Bar*, y aplique la configuración. Con esto, ha creado un usuario que puede cambiar la configuración de las extensiones, pero nada más. Por lo que la configuración del *Dialplan* estará a salvo.

A continuación se detallan otros paquetes también instalados con FreePBX.

4.3 REPORTS

Uno de los paquetes que incorpora la instalación de FreePBX se puede encontrar en la pestaña *Reports* y permite ver toda la información de las llamadas recibidas y enviadas.

Esta aplicación ha sido desarrollada por *Coalescent Systems Inc* bajo licencia GPL.

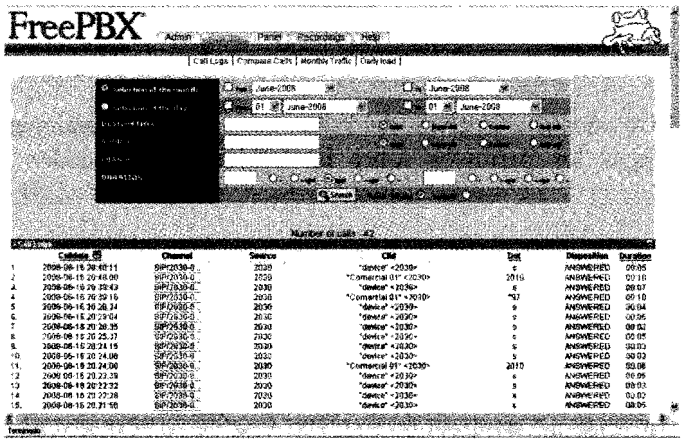


Figura 5-27. FreePBX - Reports

Este paquete es de gran utilidad si desea realizar informes de llamadas con gráficos, por meses, entre fechas, etc.

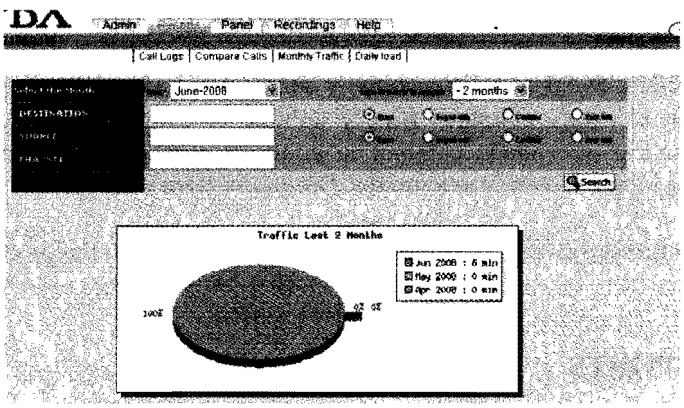


Figura 5-28. FreePBX – Reports (Traffic)

Además, puede obtener los listados en formato .PDF o .CSV y tratarlos posteriormente.

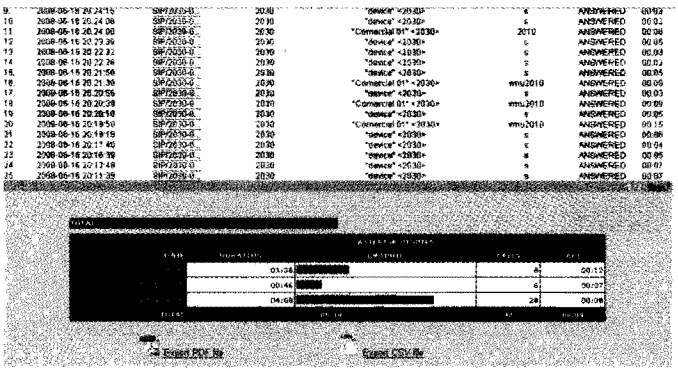


Figura 5-29. FreePBX – Reports (Tipos de listados)

4.4 VOICEMAIL & RECORDINGS (ARI)

Otra de los paquetes interesantes que se instalan junto con FreePBX es el que se sitúa bajo la pestaña *Recordings* de FreePBX.

Este paquete es conocido como *ARI*, y está desarrollado por *Dan Littlejohn* de *Littlejohn Consulting*, bajo licencia GPL. Con él, el usuario puede administrar su buzón de voz y alguna que otra prestación interesante sobre su extensión en particular.

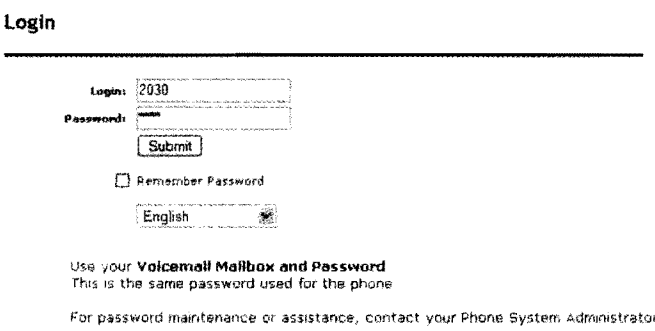


Figura 5-30. Voicemail Mailbox

Una vez dentro de la aplicación, se puede ver un listado con los mensajes de voz, antiguos (carpeta *Old)* y nuevos (carpeta *INBOX)*. Si pulsa sobre una determinada carpeta puede ver los mensajes que contiene (véase la figura 5-31).

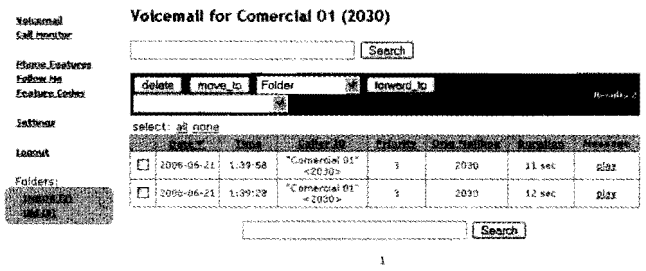


Figura 5-31. Ejemplo de buzón de voz de un usuario

Para reproducir un mensaje hay que pulsar sobre el enlace *play* que aparece en la columna *Message* en la línea del mensaje que deseamos oir.

Desde esta misma pantalla, puede marcar un mensaje, o varios, y borrarlos o moverlos a otra carpeta.

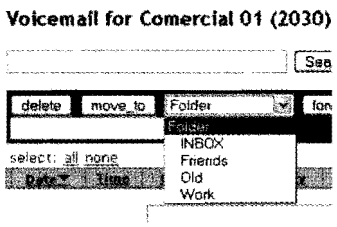


Figura 5-32. Voicemail (Mover mensajes)

Por otra parte, también puede seleccionar uno o varios mensajes y reenviárselos a otra extensión.

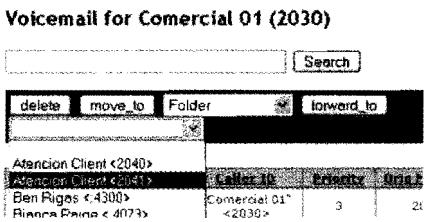


Figura 5-33. VoiceMail (Reenvío de mensajes)

Desde la opción *Call Monitor* se puede ver el reporte de toda la actividad de su extensión, buscar llamadas, ordenarlas, etc.

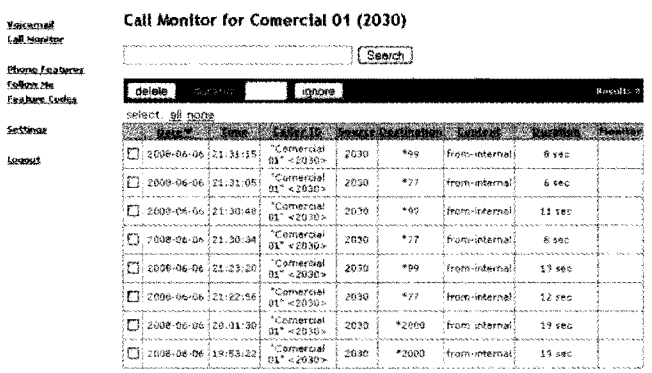


Figura 5-34. VoiceMail (Monitor de llamadas)

En la opción *Phone Features* el usuario puede configurar algunas opciones de su extensión, como pueden ser desvíos, llamada en espera, DND, etc.

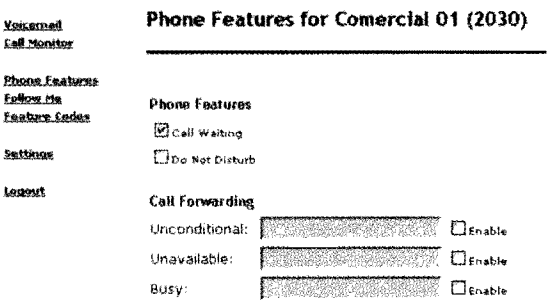


Figura 5-35. VoiceMail (Configuración de la extensión)

Si pulsa en la opción *Feature Codes* le aparece al usuario una lista con los *Códigos de Servicio* disponibles para esa extensión.

Finalmente, en la opción *Settings* podrá configurar opciones del buzón de voz, la contraseña, la dirección de correo de las notificaciones, el formato del fichero de audio, además de opciones de monitorización como la grabaciones de llamadas entrantes y salientes.

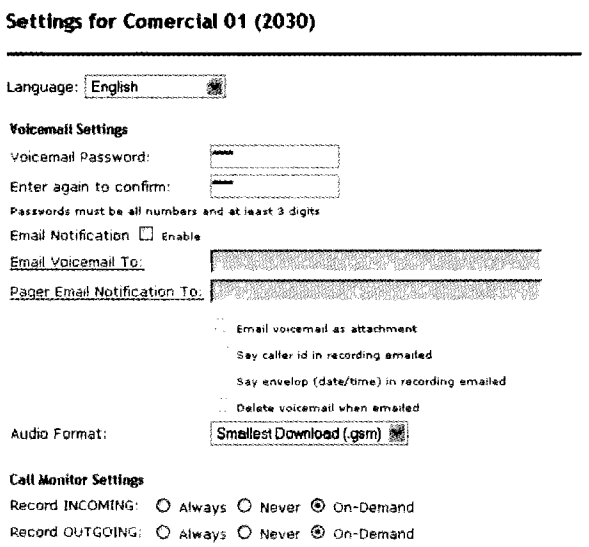


Figura 5-36. VoiceMail (Settings)

4.5 FLASH OPERATOR PANEL (FOP)

Otra de las aplicaciones que se comentará y que se instala por defecto junto con FreePBX es el *Flash Operator Panel (FOP)*.

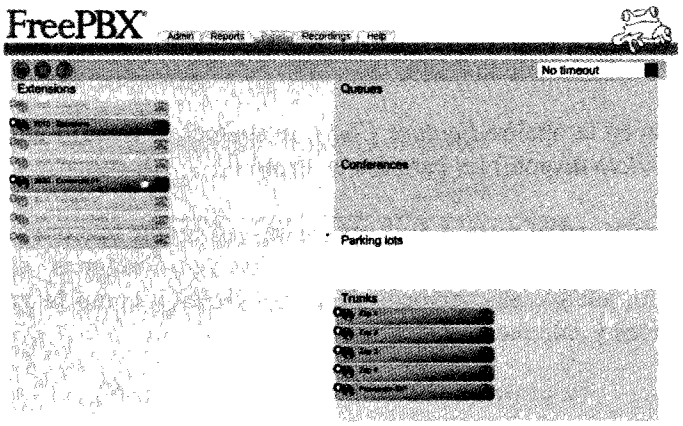


Figura 5-37. Flash Operator Panel

El FOP es un panel de operadora en el que podemos visualizar el estado de las extensiones, trunks, colas, etc.

Esta aplicación se enlaza con Asterisk a través del *puerto 5038* del *manager*, siendo esto configurable en */etc/asterisk/manager.conf*, como se detalla en la instalación de FreePBX, y en */var/www/html/panel/op_server.conf.*

En la figura 5-37, se puede apreciar que aparecen dos extensiones que resaltan sobre el resto. Esto indica que esas extensiones, la 2010 y la 2030, están registradas y el resto no.

En dicha figura, también se puede apreciar, abajo a la derecha, el listado de los *Trunks* que hay definidos en la centralita PBX.

Cuando entra una llamada a una extensión, el icono del teléfono, que aparece a su derecha, comienza a vibrar.

En el caso de que alguna extensión, o trunk, esté ocupado *FOP* mostrará el icono que hay a su izquierda en rojo.

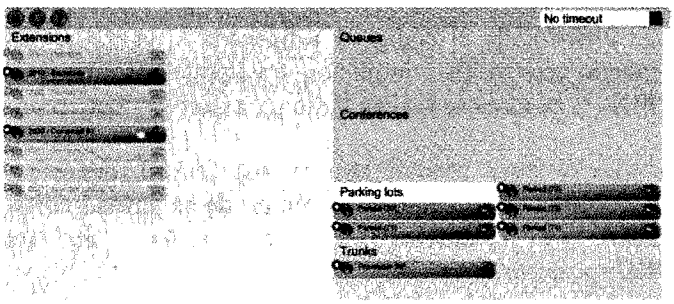


Figura 5-38. Flash Operator Panel gestionando llamadas

Por ejemplo, en la imagen de la figura 5-38, puede ver que la extensión 2030 está hablando con la extensión 2010 y que esta última tiene "aparcada" una llamada en la posición 71.

Además de visualizar los estados, podremos realizar distintas tareas propias de una operadora.

Para poder interactuar con la PBX desde el FOP, debe tener acceso introduciendo la clave configurada.

Si observa en la figura 5-38, en la parte superior izquierda, nos aparece el icono de un candado cerrado. Esto implica que el FOP está protegido y no puede realizar en él ninguna tarea sin introducir la clave.

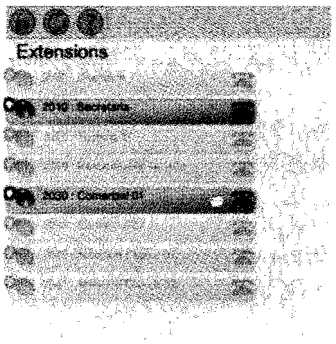


Figura 5-39. Flash Operator Panel (Protegido)

Para desbloquearlo, pulse sobre el icono y le pide el *código de seguridad.* Por defecto, este código es *passw0rd*, teniendo en cuenta que la "0" es un CERO.

Una vez introducido el código y refrescada la pantalla, nos aparecerá el icono de un candado abierto. Esto implica que ya podemos interactuar con nuestra PBX desde el FOP. Esta tarea se realiza tanto para desbloquear como para bloquear el FOP.

Es muy recomendable cambiar el código de seguridad editando el fichero */etc/amportal.conf* y cambie la contraseña en la variable *FOPPASSWORD=passw0rd.*

Como podemos apreciar, a la izquierda del icono del teléfono de la extensión 2030, nos aparece un "sobre". Esto indica que esa extensión tiene mensajes pendientes de leer.

Si se sitúa encima del sobre y pulsa dos veces en él, la extensión correspondiente comenzará a sonar, y en el momento que descuelgue, entrará en el proceso para reproducir los mensajes de voz pendientes en su buzón.

También puede, por ejemplo, establecer la comunicación entre dos extensiones. Para ello, pulse y arrastre el icono del teléfono de una extensión hasta el icono del teléfono de otra extensión. En ese momento, empezará a sonar la primera extensión, y una vez descuelgue ésta, comenzará a sonar la segunda extensión, estableciéndose así la comunicación.

La configuración de parámetros como la centralita o centralitas, a monitorizar, usuario con permisos, puertos, tiempos de refresco, etc., son configurables en el fichero */var/www/html/panel/op_server.conf*.

Toda la documentación de instalación y configuración de esta aplicación la puede encontrar en *http://www.asternic.org/documentation.html*.

5 Varios

5.1 AUTENTIFICACIÓN SERVIDOR WEB

Independientemente de los usuarios que puede crear para la administración de *FreePBX* en el módulo *Administrators*, puede forzar que sólo ciertos usuarios se puedan conectar al servidor web, creando así una doble barrera de seguridad.

Para realizar esto, hay que crear el fichero de usuarios y los usuarios que tendrán permiso a acceder al servidor Web. Los pasos que debe realizar son los siguientes:

- Creación del fichero de usuarios:

```
# htpasswd -c /var/www/usuarios usuario1
  (nos pedirá que introduzcamos la password para el
"usuario1")
```

De esta forma, nos crea el fichero usuarios en */var/www*, que contendrá los usuarios permitidos, y hemos añadido nuestro primer usuario *usuario1*.

- Para crear los demás usuarios debe ejecutar el siguiente comando:

```
# htpasswd  /var/www/usuarios usuario2
(nos pedirá que introduzcamos la password para el "usuario2")
```

De esta forma añade el *usuario2* al fichero */var/www/usuarios*. Repita esta tarea para todos los usuarios que desee añadir.

- A continuación configure Apache para darle a los usuarios permisos sobre los directorios en concreto a los que podrán acceder.

Edite el fichero de configuración */etc/apache2/httpd.conf*, en *Debian*, o */etc/httpd/conf/httpd.conf*, en el caso de estar usando *CentOS*, y añada las siguientes líneas:

```
<Directory /var/www/html>
    AuthType Basic
    AuthName "Area Restringida"
```

```
    AuthUserFile /var/www/usuarios
    Require valid-user
</Directory>
```

Esto implica que al directorio */var/www/html* sólo podrán acceder los usuarios definidos en el fichero de usuarios */var/www/usuarios*.

Si desea que sólo algunos de los usuarios definidos en ese fichero tengan acceso a este directorio, tiene que cambiar la línea *Require valid-user* por *Require user usuario1 usuario2* ..... De esta forma, sólo los usuarios listados en esta línea podrán acceder al directorio, y por ende, a la aplicación.

- Por último, hay que reiniciar el servidor Apache para que surtan efecto los cambios:

```
En Debian:
# /etc/init.d/apache2 restart

En CentOS:
# service httpd restart
```

Es recomendable que si utiliza los mismos usuarios que los creados en el módulo *Administrators* de *FreePBX*, los cree con las mismas *passwords* y así evitar confusiones.

Si crea el mismo *usuario* y *password* que el definido en este módulo, sólo se le pedirá una vez, y validará tanto el *Area Restringida* como el acceso a la *Administración* de *FreePBX*. En caso contrario, le pedirá validación para ambas áreas.

5.2 AJUSTES EN ASTERISK MANAGER

Los programas de terceros pueden tener acceso a la operativa de Asterisk como obtener información de eventos, iniciar llamadas, cambiar la operativa del dialplan, etc. Esto se hace estableciendo la comunicación a través del *Manager de Asterisk*, tal y como hace *FOP*.

Para definir el *puerto de comunicación*, el *usuario*, la *password*, los *hosts* que tienen permisos, etc., hay que modificar el fichero de configuración */etc/asterisk/manager.conf*.

```
;
; Asterisk Call Management support
;
```

```
[general]
enabled = yes
port = 5038
bindaddr = 0.0.0.0

[admin]
secret = password
deny=0.0.0.0/0.0.0.0
permit=127.0.0.1/255.255.255.0
read = system,call,log,verbose,command,agent,user
write = system,call,log,verbose,command,agent,user

#include manager_additional.conf
#include manager_custom.conf
```

Como puede ver, el fichero tiene parte de configuración general y después la definición de cada usuario y sus permisos.

***Nota:** es muy recomendable cambiar la password por defecto que se establece al instalar FreePBX.*

5.3 AJUSTES EN FOP

Este paquete utiliza la tecnología Flash de Adobe para trabajar. Si tiene instalada la última versión de Flash Player puede encontrarse con que FOP no se refresca correctamente, sino que nos muestra los *Botones de Estado* conmutando entre verde y rojo.

Con *FreePBX* se instala *FOP* versión 0.28. Para solucionar este problema, debe actualizar algunos ficheros a la versión 0.29 del *FOP* y realizar algún cambio en la configuración.

Las tareas a realizar son las siguientes:

- Descargue FOP v.0.29 desde *http://www.asternic.org* y lo descomprime ejecutando:
  ```
  # cd /usr/src
  # wget http://www.asternic.org/files/op_panel-0.29.tar.gz
  # tar xfpvz op_panel-0.29.tar.gz
  ```

- Reemplace los ficheros *op_server.pl* y *operator_panel.swf* de la instalación por los de la versión 0.29:
  ```
  # cd op_panel-0.29
  ```

```
# cp op server.pl /var/www/html/panel
# cp flash/operator_panel.swf /var/www/html/panel
```

- Edite el fichero */var/www/html/panel/op_server.conf* de la instalación actual y añada, justamente al principio de la sección *[general]*, la siguiente línea:

```
use_amportal_conf=1
```

- Reinicie *Amportal* para que se tomen todos los cambios:

```
# amportal restart
```

Ahora si se conecta al *FOP* se deben de haber solucionado estos problemas.

APÉNDICE I

HERRAMIENTAS Y URLS REFERENCIADAS

Consolación Gil Montoya y María Dolores Gil Montoya

A continuación se muestran las herramientas y las páginas web referenciadas a lo largo del libro.

Tabla I-1. Herramientas referenciadas

Nombre	Descripción	URL
Asterisk	Centralita para comunicaciones de código libre	http://www.asterisk.org
A2Billing	Plataforma de tarificación para Asterisk realizada en php	http://www.asterisk2billing.org
AskoziaPBX	Distribución precompilada de Asterisk basada en FreeBSD y derivada de monowall	http://www.askozia.com

Nombre	Descripción	URL
AstLinux	Distribución precompilada con Asterisk, específica para dispositivos de recursos limitados	http://www.astlinux.org/
AvantFAX	Paquete de software que permite gestionar mediante interfaz web los faxes que envía y recibe un sistema Asterisk	http://www.avantfax.com
CentOS	Distribución GNU/Linux basada en RedHAT	http://www.centos.org
Debian	Distribución GNU/Linux	http://www.debian.org
Elaxtix	Distribución precompilada de Asterisk basada en CentOS y con un interfaz web basado en FreePBX	http://www.elastix.org
Fedora	Distribución GNU/Linux	http://www.fedoraproject.org
Flash Operador Panel	Interfaz flash para el manejo de extensiones vía web	http://www.asternic.org
FreePBX	Interfaz web para la gestión de las tareas más habituales de un sistema Asterisk (Código abierto)	http://www.freepbx.org
Hylafax	Gestor de faxes mediante software	http://www.hylafax.com
IAXModem	Software que emula un módem telefónico	http://sourceforge.net/projects/iaxmodem
Kamalio	Enrutador para el protocolo SIP	http://www.kamalio.org
MySQL	Herramienta paga gestión de bases de datos de código abierto	http://www.mysql.com/
PBXinaFlash	Distribución precompilada de Asterisk basada en CentOS y con un interfaz web basado en FreePBX	http://pbxinaflash.net http://pbxinaflash.com
OpenSIPS	Enrutador para el protocolo SIP	http://www.opensips.org
OSLEC	Cancelador de eco GPL	http://www.rowetel.com
SER	Enrutador para el protocolo SIP	http://www.iptel.org/ser
SJphone	Teléfono software con soporte SIP	*http://www.sjlabs.com*

Nombre	Descripción	URL
SugarCRM	Plataforma CRM para la gestión de recursos y relación con los clientes de una empresa, que se integra con Asterisk	http://www.sugarcrm.com
Trixbox	Distribución precompilada de Asterisk basada en CentOS y con un interfaz web basado en FreePBX	http://www.trixbox.org
Virtualbox	Plataforma para virtualización de sistemas operativos	http://www.virtualbox.org
VoiceOne	Distribución que incluye Asterisk y un gestor web del mismo	http://www.voiceone.it
Voip-info	Wiki sobre Asterisk	http://www.voip-info.org
VTigerCRM	CRM de código abierto	http://www.vtiger.com/
X-lite	Teléfono software con soporte SIP	http://www.counterpath.com
Zoiper	Teléfono software con soporte SIP e IAX	http://www.zoiper.com

Tabla I-2. URL referenciadas

URL	Descripción
http://www.ietf.org	IETF. Asociación de ingenieros, diseñadores, operadores, vendedores e investigadores, orientada al correcto diseño de la arquitectura de Internet así como su operación
http://ualtech.wordpress.com	Blog sobre VozIP y Asterisk
http://www.adminso.es	Página web dedicada a la Administración de Sistemas Operativos y en la que puede encontrar material relacionado con la obra
http://www.ecualug.org	Comunidad de usuario de Linux en Ecuador
http://www.gnu.org/	GNU. Sistema operativo basado en UNIX
http://www.howtoforge.com	Web que contiene numerosos tutoriales para configuración de diferente software (en inglés)
http://www.saghul.net	Blog sobre VozIP y Asterisk
http://www.sureteq.com	Web especializada en soporte técnico en redes y administración
http://www.voipnovatos.es	Web donde se alojan sonidos en español para Asterisk
http://www.wikipedia.es	Biblioteca on-line

APÉNDICE II

ADMINISTRACIÓN BÁSICA DE LINUX

Consolación Gil Montoya y María Dolores Gil Montoya

1 Introducción

Linux fue concebido por el finlandés Linus Torvalds, estudiante de la Universidad de Helsinki, quien comenzó trabajando sobre el código fuente de Minix (un pequeño UNIX desarrollado por Andy Tanembaum) para lograr un Unix mínimo, capaz de ejecutar al menos un shell y un compilador. Primero fue la versión 0.02 ya que la 0.01 nunca llegó a ser compilada con éxito. Luego Linus anunció en Internet su proyecto de la siguiente manera:

> *"Si suspiras al recordar aquellos días cuando lo hombres eran hombres y escribían sus propios manejadores (drivers). Si te sientes sin ningún proyecto interesante y te gustaría tener un verdadero sistema operativo que pudieras modificar a placer. Si te resulta frustrante tener sólo Minix. Entonces este artículo es para ti"*

De esa forma Linux fue liberado en Internet y la respuesta de los programadores y usuarios de UNIX fue contundente. Pronto todos querían aportar sus conocimientos para que Linux se convirtiera en un sistema operativo estable, robusto y potente. Finalmente llegó la primera versión estable del Kernel, la

versión 1.0. De allí en adelante Linux fue evolucionando a un ritmo vertiginoso hasta convertirse en un fuerte rival de los actuales sistemas operativos.

La estructura básica del S.O. Unix podemos verla en la figura II-1. El kernel, capa más interna en la figura, es el núcleo del sistema operativo Unix. Se encarga de secuenciar los procesos, reservar espacio de memoria y de disco, supervisar la transmisión de datos entre memoria principal y periféricos, y satisfacer las peticiones de servicios de los procesos existentes.

El kernel nunca trabaja directamente para los usuarios. Todos los servicios los proporciona el shell o los programas de utilidades (ambos actúan de interfaz entre los usuarios y el Kernel).

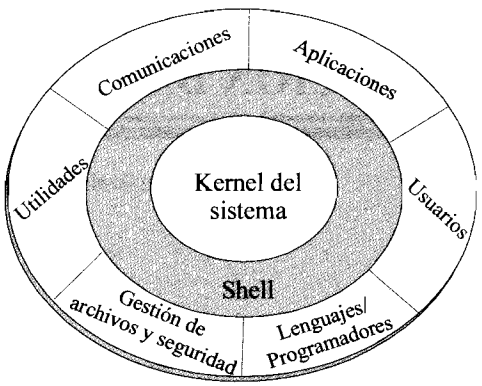


Figura II-1. Estructura del Unix

Para requerir un servicio del sistema operativo se utiliza lo que se denomina *llamadas al sistema* (*system calls*). Las llamadas al sistema lo que hacen es conmutar la máquina de *modo usuario al modo kernel* (también llamado *modo núcleo o supervisor*) y transfieren el control al sistema operativo. Se puede ver la existencia de estos modos como un esquema de protección ya que en modo núcleo se pueden ejecutar todas las instrucciones, mientras que en modo usuario las instrucciones de E/S, por ejemplo, y otras no están permitidas. Sólo se permiten algunas instrucciones como las relacionadas con las operaciones aritméticas, de acceso a memoria principal, etc. Por ello, los programas de usuario se ejecutan en modo usuario y el sistema operativo se ejecuta en modo núcleo.

El shell es el intérprete de órdenes y se encuentra por encima del kernel. Se encarga de interpretar las órdenes y convertirlas en peticiones al kernel. Normalmente se incluyen con cualquier sistema operativo Unix distintos tipos de shell, lo que permite poder elegir el que más se adapte a nuestras necesidades.

La capa más externa está formada por utilidades para la manipulación de ficheros, lenguajes de programación, utilidades de depuración de código, aplicaciones de usuarios, etc.

Son muchas las características que podemos encontrar en el sistema operativo UNIX, pero entre ellas destacamos las siguientes:

- ***Interactivo***. Se escribe órdenes y el sistema operativo obedece presentando las respuestas apropiadas. Además incluye la posibilidad de ejecutar órdenes o programas que se ejecuten de forma no interactiva.

- ***Multitarea***. Puede realizar varias tareas (llamados procesos) al mismo tiempo.

- ***Multiusuario***. Más de una persona puede usar el sistema al mismo tiempo e incluso con el mismo nombre de usuario.

- ***Independencia respecto de los dispositivos***. El acceso a la información es el mismo sin importar donde se encuentre el fichero o dispositivo. Esto permite dar un tratamiento uniforme a todos los dispositivos.

- ***Portabilidad***. Al estar escrito en C permite transportarlo a otros sistemas.

- ***Facilidad de trabajo en red***. Al disponer de un sistema de ficheros flexible permite trabajar con sistemas de ficheros en red como NFS y RFS.

2 Sistema de ficheros

Linux, al igual que UNIX, organiza la información del sistema en una estructura de árbol jerárquico de directorios compuesta de ficheros. Esta estructura se forma mediante un sistema de ficheros raíz (file system root) y un conjunto de sistemas de ficheros montables.

Un Sistema de ficheros, o File System, es una estructura de directorios completa. Para poder utilizar un Sistema de ficheros hay que montarlo; o sea, enlazarlo a la estructura de directorios ya existente. Los Sistemas de ficheros se montan automáticamente cada vez que se inicia el sistema operativo. Cuando un usuario se conecta al sistema, se encuentra un único árbol de directorios formado por los distintos Sistemas de ficheros que se encuentran montados en ese instante. La estructura que aparece al usuario será similar a la que se muestra, de forma abreviada, en la figura II-2.

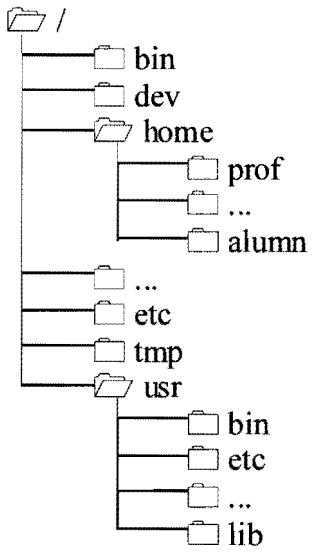


Figura II-2. Estructura de directorios del Unix

Analicemos los directorios más importantes que tiene un sistema operativo Linux:

- El directorio */etc* está reservado para ficheros de configuración de la máquina local.

- El directorio */lib* contiene normalmente las bibliotecas necesarias para ejecutar los programas residentes en */bin* y */sbin*.

- El directorio */sbin* contiene los programas de uso exclusivo por el administrador del sistema (root), y sólo contiene los necesarios para arrancar y montar el directorio */usr* y ejecutar operaciones de restablecimiento del sistema. Algunos de los programas que aparecen son: *clock*, *getty*, *init*, *update*, *mkswap*, *swapon*, *swapoff*, *halt*, *reboot*, *shutdown*, *fdisk*, *lilo*, *arp*, *ifconfig*, *route*.

- El directorio /usr contiene los ficheros que pueden ser utilizados por cualquier usuario del sistema. Normalmente este directorio se encuentra en su propia partición por motivos de seguridad. Algunos de los subdirectorios que contiene son los siguientes. El directorio *X11R6* se utiliza para el sistema X-Windows (XFree86 en Linux), *bin* es para ejecutables, *doc* para documentación diversa (no incluye páginas de manual), *etc.* para los ficheros de configuración del sistema, *include* para ficheros cabecera de C, *info* para ficheros info de GNU, *lib* para

bibliotecas, man para las páginas del manual, sbin para binarios de administración del sistema (los que no pertenecen a */sbin*), y src es para código fuente. El directorio */usr/local* tiene una estructura similar a la del directorio */usr*. Contiene los siguientes subdirectorios, con propósito similar a los de */usr*.

- En el directorio */var* se guardan los ficheros de datos variables. Esto incluye los directorios y ficheros de spool y datos de administración y registro, así como ficheros transitorios y temporales. Los ficheros de registro del sistema como *wtmp* y *lastlog* habitan en */var/log*. El directorio */var/lib* contiene también las bases de datos del sistema RPM. Las páginas del manual formateadas se encuentran en */var/catman*, y los ficheros de bloqueo en */var/lock*. El directorio */var/spool* contiene subdirectorios para los diversos sistemas que necesitan almacenar ficheros de datos.

- Cada usuario dispone de un directorio de trabajo. Normalmente este directorio se encuentra dentro del directorio */home*.

2.1 REGLAS PARA NOMBRAR FICHEROS

Los nombres de los ficheros pueden constar de hasta 255 caracteres y, al menos en teoría, pueden contener cualquier carácter. En la práctica, sin embargo, muchos de los caracteres ASCII pueden significar algo especial para el intérprete de órdenes, por lo que deberían evitarse los caracteres especiales al nombrar ficheros y directorios.

Al elegir los nombres de ficheros, como regla general es mejor limitarse a utilizar nombres que contengan sólo letras, números, el carácter de subrayado y el punto (si está al principio, se considera el fichero como oculto). Recuerde que en Linux las letras mayúsculas y minúsculas son significativas en los nombres de ficheros, es decir, no es lo mismo decir 'Dato' que 'dato', serían nombres distintos.

2. 2 NOMBRES DE CAMINOS ABSOLUTO Y RELATIVO

Linux permite que los directorios y ficheros puedan especificarse de dos formas distintas. Una es mediante un camino o direccionamiento absoluto y la otra mediante un direccionamiento relativo.

En un direccionamiento absoluto se toma como origen el directorio raíz seguido del resto del camino que hay que seguir para llegar al nombre del directorio o fichero; como por ejemplo:

```
/home/alumnos/moises/.cshrc
```

En un direccionamiento relativo, se toma como origen el directorio actual seguido del resto del camino que hay que seguir para llegar al directorio o fichero especificado. Veamos dos ejemplos:

```
home/alumnos/moises/.cshrc
../.cshrc
```

2.3 ÓRDENES DE MANIPULACIÓN DE DIRECTORIOS

Para cambiar el directorio donde se encuentra actualmente se utiliza la orden *cd (Change Directory)*. Su sintaxis es la siguiente:

```
cd [<directorio>]
```

Por ejemplo, si queremos irnos al directorio */home/alumnos/moises*, podemos escribir lo siguiente:

```
cd /home/alumnos/moises
```

El directorio también se puede escribir de forma relativa partiendo del directorio actual, como se ha indicado anteriormente. Existen algunos casos especiales cuando se usa esta orden:

```
cd ..
cd $HOME  o  cd $home
cd
```

El primer caso nos permite situarnos en el directorio padre del directorio actual. Los otros dos casos permiten situarnos automáticamente en el directorio de trabajo (*home directory*).

Si lo que desea es mostrar el directorio actual, la orden que se utiliza para esta operación es *pwd (Path Work Directory)*. El nombre del directorio aparece expresado de forma absoluta. Veamos un ejemplo:

```
$ pwd
/home/alumnos/moises
```

Para crear un directorio se utiliza la orden *mkdir (MaKe DIRectory)*. Su sintaxis es la siguiente:

```
mkdir <directorio>
```

donde *<directorio>* identifica el directorio que queremos crear. Si estando en el directorio */home/alumnos/moises* queremos crear un subdirectorio llamado ejemplos, podemos hacerlo así:

```
mkdir /home/alumnos/moises/ejemplos
```

o indicando un direccionamiento relativo, el ejemplo anterior quedará de la forma:

```
mkdir ejemplos
```

Es importante recordar que siempre podemos usar cualquiera de los dos modos de direccionamiento. El empleo de uno u otro dependerá del caso.

Si lo que queremos es borrar un directorio, se utiliza la orden *rmdir (ReMove DIRectory)*. Su sintaxis es:

```
rmdir <directorio>
```

donde *<directorio>* identifica el directorio que deseamos borrar. Por ejemplo, para borrar el directorio */home/alumnos/moises/ejemplos*, debemos estar fuera de ese directorio y ejecutar la orden siguiente:

```
rmdir /home/alumnos/moises/ejemplos
```

Si deseamos renombrar un directorio, la orden que nos permite realizar esta operación es *mv (MoVe)*. Su sintaxis es la siguiente:

```
mv <dir_antiguo> <dir_nuevo>
```

<dir_antiguo> identifica el nombre del directorio que queremos renombrar y *<dir_nuevo>* identifica el nuevo nombre. Esta orden no cambia el contenido del directorio; sólo su nombre.

También se puede usar mv para mover el directorio de un lugar a otro; o sea, cambiar el directorio padre del que depende ese directorio.

Por ejemplo, si queremos cambiar el nombre del directorio */home/moises/ejem* (suponiendo que éste exista), por el nombre */home/moises/ejer*, se puede realizar utilizando la siguiente orden:

```
mv /home/moises/ejem /home/moises/ejer
```

La copia de directorios se realiza mediante *cp (CoPy)*. La forma de utilizarla es la siguiente:

```
cp -r <dir_origen> <dir_destino>
```

<dir_origen> identifica el directorio que queremos copiar y *<dir_destino>* el directorio donde será copiado toda la información. Si el directorio a copiar incluye subdirectorios, se debe usar la opción -r o -R (indistintamente) para poder realizar una copia recursiva.

Para mostrar el contenido de un directorio se puede utilizar la orden *ls (LiSt)*. Su sintaxis es:

```
ls [<directorio>]
```

donde *<directorio>* identifica el directorio que queremos visualizar. Si no se especifica ninguno, se mostrará el contenido del directorio actual. La opción -l *(Large)* de ls es la que muestra información más detallada. Veamos un ejemplo:

```
$ ls -l /
.....
drwxr-xr-x  5  root  512  Mar 15 18:45 home
drwxr-xr-x  27  root  1024  Nov 11 18:05 usr
.....
```

El primer campo indica el modo del fichero (-) o si es un directorio (d). El resto del campo nos indica los permisos del fichero para el propietario (en nuestro ejemplo es rwx para ambos directorios), grupo al que pertenece el propietario y para el resto de los usuarios (en el ejemplo, en ambos estos permisos son r-x).

El siguiente campo es el número de enlaces. Para ficheros es normalmente 1. Si es mayor de 1 indica que existen entradas en otros directorios apuntando a ese fichero. Para directorios, indica el número de entradas de subdirectorios que contiene.

Los campos que van a continuación son el nombre del propietario del fichero; tamaño del fichero, en bytes; fecha y hora de la última modificación; y por último, el nombre del fichero.

2.4 ÓRDENES DE MANIPULACIÓN DE FICHEROS

Las órdenes de manipulación de ficheros permiten realizar determinadas operaciones sobre ficheros ordinarios. Algunas de estas operaciones permitirán mostrar el tipo de un fichero, ver su contenido, crearlos, copiarlos, etc.

Linux incluye una orden que nos permite crear un fichero de 0 bytes si éste no existía. En cuyo caso, lo que hace es actualizar las fechas del fichero. La orden es *touch* y su sintaxis es:

```
touch <fichero>
```

donde *<fichero>* identifica el fichero que queremos crear o actualizar.

Son diversas las órdenes que existen para visualizar el contenido de un fichero. Algunas de estas órdenes son *more*, *cat*, *tail*, *head*. Estudiemos cada una de estas órdenes.

La orden *more* visualiza un fichero de texto pantalla a pantalla. Su sintaxis es la siguiente:

```
more [<fichero>]
```

donde <*fichero*> identifica el fichero que queremos visualizar.

Por ejemplo, si deseamos ver todas las cuentas que hay en el sistema, podemos hacer lo siguiente:

```
$ more /etc/passwd
```

La orden *cat* permite visualizar el contenido de uno o más ficheros, sin paradas, a través del dispositivo de salida por defecto, la pantalla. En algunos sistemas se puede detener la imagen pulsando <CTRL> <S>, y continuar pulsando <CTRL> <Q>. Veamos la sintaxis de esta orden:

```
cat [<fichero>...]
```

donde <*fichero*> identifica el fichero a visualizar. Si deseamos ver el contenido de varios ficheros, éstos deberán separarse por espacios en blanco. Veamos un ejemplo:

```
$ cat .profile .exrc
```

La orden pg permite paginar la salida. Si el fichero que vamos a visualizar tiene más de 24 líneas, pg visualizará las 23 primeras y presentará en la línea 24 un carácter: para solicitar una orden. Dos posibles órdenes son:

- pulsar <retorno> para ver la siguiente página.
- abandonar la ejecución de pg, pulsando "q".

La orden *tail* permite examinar el final de un fichero. Su sintaxis es:

```
tail [+|-<número>] <fichero>
```

Por defecto visualizará las diez últimas líneas del fichero <*fichero*>, pero es posible modificar dicho número usando la opción <*número*>. Si le antecede un +, la cuenta comenzará por la primera línea mientras que si le antecede un -, la cuenta comenzará por la última línea. Veamos unos ejemplos:

```
$ tail -3 /etc/passwd
$ tail +10 /etc/group
```

En el primer ejemplo se visualizan las tres últimas líneas del fichero passwd; mientras que el segundo ejemplo visualiza el fichero group a partir de la línea 10. Si se utiliza la opción +<*número*>, y el fichero no tiene tantas líneas, no se visualizará nada.

La orden head nos permite visualizar las diez primeras líneas de un fichero, aunque como en el caso de la orden *tail*, podemos modificar ese valor. Su sintaxis es la siguiente:

```
head [-<número>] <fichero>
```

La opción -<*número*> indica la cantidad de líneas a visualizar del fichero <*fichero*>. Como ejemplo, veamos cómo se pueden visualizar las primeras 20 líneas de un fichero:

```
$ head -20 /etc/tempcap
```

Si lo que queremos es mover o renombrar un fichero, la orden que nos permite realizar esta operación es *mv (MoVe)*. Dependiendo de lo que queramos hacer, su sintaxis será:

```
mv <fichero1> <fichero2>
```

o

```
mv <fichero>... <directorio>
```

El primer caso se utilizará para renombrar el fichero <fichero1> por el nombre de fichero especificado en <*fichero2*>. En el segundo caso, *mv* se utiliza para mover el fichero <*fichero*> al directorio <*directorio*>. Se puede mover más de un fichero a la vez. Veamos unos ejemplos:

```
$ mv mensaje_01 mesj.01
$ mv moises/mbox erik
```

La primera orden nos permite cambiar el nombre del fichero mensaje_01; y en el caso de que existiese mesj.01 éste cambiaría su contenido por el de mensaje_01. Con la segunda orden, movemos el fichero mbox del directorio moises al directorio erik.

El sistema operativo incluye la orden *cp (CoPy)* para copiar ficheros. La copia se puede realizar en el mismo directorio, en cuyo caso no puede tener el mismo nombre, o copiarlo en un directorio distinto. La sintaxis para esta orden es la siguiente:

```
cp <fichero1> <fichero2>
```

o

```
cp <fichero>... <directorio>
```

El primer caso se utiliza para copiar el fichero <*fichero1*> en el mismo directorio donde se encuentra éste. Unix no permite tener dos ficheros con el mismo nombre en el mismo directorio, por lo que se necesita indicarle un nombre distinto. Por tanto necesitamos utilizar <*fichero2*> para indicar otro nombre al fichero. Si la copia se realiza a otro directorio, se utiliza la segunda forma. El argumento <*directorio*> indicará el directorio destino de la copia. cp permite copiar varios ficheros a otro directorio. Por ejemplo, si deseamos copiar en el directorio actual el fichero motd y renombra el fichero llamándole mensaje, escribiríamos lo siguiente.

```
$ cp /etc/motd mensaje
```

Estamos suponiendo que en el directorio actual no existía un subdirectorio llamado mensaje, en cuyo caso, esta orden hubiera copiado el fichero motd en dicho subdirectorio. De cualquier forma, después de realizar la copia, el modo del fichero es el mismo que tenía el fichero original.

Para borrar uno o más ficheros de un directorio se utiliza la orden *rm (ReMove)*. En el caso de que uno de los ficheros a suprimir estuviese protegido contra escritura, rm le informaría del modo real del fichero, y esperaría una respuesta: si pulsa y procede a borrarlo y si pulsa cualquier otra tecla no lo borrará. Su sintaxis es:

```
rm [ifr] <fichero>....
```

La opción -i (Interactiva) lo utiliza rm para pedir confirmación de cada uno de ficheros que va a borrar, independientemente de que esté o no protegido. La opción -f se puede utilizar si se quiere forzar la supresión de ficheros aunque estén protegidos. La opción -r permite borrar el contenido del directorio actual, y de los posibles subdirectorios que existan a partir de él (de forma recursiva).

Unix permite que un fichero o directorio pueda ser referenciado desde el mismo o desde distinto directorio más de una vez. Esto permite no tener duplicados ficheros o directorios y se puede emplear, por ejemplo, para compartir información entre grupos de usuarios. La forma de conseguirlo es mediante la orden ln. Su sintaxis es:

```
ln [-s] <fichero>
```

Hay dos tipos de enlaces a un determinado fichero especificado con *<fichero>*, los enlaces fijos y los simbólicos. Los enlaces fijos sólo se hacen a ficheros existentes y dentro del Sistema de ficheros (file system) al que pertenezca el fichero. Para eliminar un fichero, se tienen que eliminar todos los enlaces fijos (incluyendo el primer nombre que se le dio al fichero).

Los enlaces simbólicos se realizan con la opción -s. Es una entrada de directorio especial que apunta a otro fichero existente. El fichero puede estar en otro sistema de ficheros. Si se elimina el fichero al que apunta, no afecta ni altera al propio enlace simbólico.

2.5. ACCESO A LOS FICHEROS

Desde el punto de vista del acceso a un fichero, existen tres tipos de usuarios a los que se les pueden dar o denegar permisos sobre un fichero:

- (u) user propietario del fichero

- (g) group usuarios pertenecientes a su grupo
- (o) other resto de usuarios que no pertenecen a su grupo

La capacidad de un usuario para trabajar con ficheros depende del tipo de acceso que tenga a dicho fichero. Los accesos disponibles son:

Para un fichero:

- Permiso de lectura (r): permite ver el contenido del fichero.
- Permiso de escritura (w): permite cambiar el contenido del fichero.
- Permiso de ejecución (x): permite ejecutar un fichero (como cualquier orden del Unix.

Para un directorio:

- Permiso de lectura (r): permite ver los nombres de los ficheros de un directorio. Si se quiere información detallada sobre dichos ficheros (mediante la opción -l de ls), el directorio tiene que tener el permiso de ejecución para dicho usuario.
- Permiso de escritura (w): permite cambiar el contenido de dicho directorio; crear nuevos ficheros y suprimir los existentes (este último caso depende de los permisos de escritura de los propios ficheros).
- Permiso de búsqueda (x): se debe hablar más bien de permiso de búsqueda ya que permite situarse en dicho directorio y según el resto de los permisos, permitirá crear, borrar, modificar o copiar ficheros.

Para definir los permisos de creación de un fichero o directorio se emplea el comando *chmod.* Su sintaxis es:

```
chmod <modo> fichero
```

donde *<modo>* indica los permisos que le queremos dar al fichero. Por ejemplo, si queremos darle los permisos de rw- para el propietario y r-- para el resto, el comando que se debe utilizar es:

```
$chmod 644 fichero
```

2.6 MODIFICACIÓN DE PERMISOS Y PROPIETARIOS

El propietario de un fichero es aquel usuario que creó dicho fichero. Unix permite cambiar al propietario de cualquier fichero o directorio. Opcionalmente se

puede cambiar también al grupo al que pertenece dicho fichero o directorio. Para ello se utiliza la orden chown. Veamos su sintaxis:

```
chown <NombreUsuario> [.<NombreGrupo>] <fichero>...
```

<NombreUsuario> identifica el nuevo propietario de fichero o directorio. *<NombreGrupo>* el nuevo grupo y *<fichero>* identifica el fichero o directorio sobre el que se va a actuar.

La limitación que tiene esta orden es que sólo el súper usuario puede utilizarlo.

Por otro lado, para cambiar el grupo al que pertenece un directorio se utiliza chgrp. Su sintaxis es:

```
chgrp <NombreGrupo> <fichero>...
```

<NombreGrupo> identifica el nuevo nombre de grupo que se le va a asignar al fichero o directorio *<fichero>*. Se puede actuar sobre varios ficheros a la vez.

3 Comandos más importantes

A modo de resumen en la tabla II-1 puede ver los comandos más importantes de Linux.

Tabla II-1. Estructura del cableado

Categoría	Comando	Descripción
Sistema de ficheros		
	cp <origen> <destino>	Copia ficheros
	mv <actual> <nuevo>	Mueve o cambia el nombre de un fichero o directorio
	rm <fichero>	Borra un fichero o directorio
	cd <directorio>	Cambia de directorio
	pwd	Muestra el directorio actual de trabajo
	mkdir <directorio>	Crea un directorio
	less <fichero> **cat <fichero>**	Muestra el contenido de un fichero
	ls	Muestra el contenido de un directorio

Particionamiento	
fdisk	Permite administrar las particiones del sistema
fsck	Permite comprobar el estado de un sistema de ficheros
mkfs	Permite formatear un sistema de ficheros
df	Indica el espacio libre de un sistema de ficheros
du	Indica el espacio utilizado por un usuario en el sistema de ficheros
mount	Permite montar sistemas de ficheros
umount	Permite desmontar sistemas de ficheros

Comandos generales	
startx	Inicia el modo gráfico
halt	Apaga el equipo
Reboot	Reinicia el equipo
Date	Muestra y permite cambiar la fecha del sistema
clear	Borra la pantalla
man	Permite obtener ayuda del sistema

Procesos	
ps	Muestra los procesos activos del sistema
top	Muestra los procesos del sistema y su rendimiento
kill	Permite matar un proceso

Cuotas de usuario	
edquota <usuario> **edquota –g <grupo>**	Permite modificar la cuota de un usuario o grupo
quota <usuario> **quota –g <grupo>**	Muestra la cuota de un usuario o grupo
repquota <directorio>	Muestra las cuotas de usuario de un sistema de un directorio
checkquota –cug <directorio>	Crea las cuotas de usuario de un directorio
quotaoff –aug	Desactiva las cuotas de usuario
quotaon –aug	Activa las cuotas de usuario

Permisos	
chmod <permisos> <fichero/directorio>	Establece los permisos de un fichero o directorio
chown <usuario> <fichero/directorio>	Cambia el usuario propietario de un fichero o directorio
chgrp <grupo> <fichero/directorio>	Cambia el grupo propietario de un fichero o directorio

Redes	
ifconfig	Permite obtener información y configurar los adaptadores de red
iwconfig	Permite obtener información y configurar los adaptadores de red inalámbrica
ping <host>	Permite realizar un ping para comprobar la comunicación con un equipo
route	Muestra y configura la tabla de enrutado del sistema
iptables	Muestra y configura el cortafuegos del sistema
service	Permite administrar los servicios del sistema

Usuarios	
adduser <usuario>	Da de alta un usuario
userdel <usuario>	Borra un usuario
usermod	Permite modificar las propiedades de un usuario
passwd	Cambia la contraseña de un usuario
addgroup	Permite dar de alta un usuario dentro de un grupo
su	Permite cambiar de usuario
id	Muestra el usuario que se está utilizando

Grupos	
groups	Muestra los grupos a los que pertenece el usuario
groupadd	Permite dar de alta a un grupo
groupdel	Permite borrar un grupo de usuarios

APÉNDICE III

ASPECTOS BÁSICOS DE REDES

Consolación Gil Montoya y María Dolores Gil Montoya

1 Introducción

Podría decirse que Internet se ha convertido en la entidad virtual más variada que ha desarrollado el hombre. El número de usuarios crece periódicamente en cientos de miles por todo el mundo, sin que parezca que vaya a dejar de aumentar. Internet es un lugar virtual donde todo el mundo es bienvenido para hacer negocios, comunicarse, buscar información o, simplemente, divertirse navegando por la red.

En este capítulo vamos a ver todos los pasos que son necesarios para la puesta en marcha de una red:

- **Creación de la red a nivel físico:** se crea la infraestructura necesaria para poner la red en funcionamiento. Para ello se instala el cableado de la red y luego se ponen en marcha los dispositivos de interconexión (hub, switch, routers...).

- **Creación de la red a nivel lógico:** se crean las diferentes redes lógicas y se asignan las direcciones IP a los diferentes equipos de la red.

- **Configuración de los routers:** se configuran los routers para permitir aceptar o denegar la comunicación que se realizan a través de él.

2 Tipos de cable

Los diferentes tipos de cables ofrecen distintas características de funcionamiento. La variedad de velocidad de transmisión que un sistema de cableado puede soportar se conoce como el ancho de banda utilizable. La capacidad del ancho de banda está condicionada por las características físicas que tienen los componentes del sistema de cableado.

El funcionamiento del sistema de cableado deberá ser considerado no sólo cuando se está cubriendo las necesidades actuales sino también con las necesidades del mañana. Conseguir esto permitirá la migración a aplicaciones de redes más rápidas sin necesidad de incurrir en costosas actualizaciones del sistema de cableado.

A continuación veremos los medios de comunicación más utilizados en la actualidad.

- **Coaxial.** Este tipo de cable está compuesto de un hilo conductor central de cobre rodeado por una malla de hilos de cobre. El espacio entre el hilo y la malla lo ocupa un conducto de plástico que separa los dos conductores y mantiene las propiedades eléctricas. Todo el cable está cubierto por un aislamiento de protección para reducir las emisiones eléctricas. El ejemplo más común de este tipo de cables es el cable coaxial de televisión.

  Originalmente fue el cable más utilizado en las redes locales debido a su alta capacidad y resistencia a las interferencias, pero en la actualidad su uso está en declive. Presenta dos grandes limitaciones, como es la seguridad y la velocidad de transmisión.

  En la tabla III-1 y III-2 se puede ver la estructura del cable así como sus datos técnicos.

- **Par trenzado.** Es el tipo de cable más común y se originó como solución para conectar teléfonos, terminales y ordenadores sobre el mismo cableado.

  Cada cable de este tipo está compuesto por una serie de pares de cables trenzados. Los pares se trenzan para reducir la interferencia entre pares adyacentes. Normalmente una serie de pares se agrupan en una única funda

de color codificado para reducir el número de cables físicos que se introducen en un conducto.

Este cable es el más utilizado en la actualidad y permite velocidades de hasta 1Gb/s. En la tabla III-1 y III-2 puede ver la estructura del cable, conectores, así como sus características físicas.

- **Fibra óptica.** Este cable está constituido por uno o más hilos de fibra de vidrio. Tal y como muestra la tabla III-1, cada fibra de vidrio consta de: un núcleo central de fibra con un alto índice de refracción; una cubierta que rodea al núcleo, de material similar, con un índice de refracción ligeramente menor; y una envoltura que aísla las fibras y evita que se produzcan interferencias entre fibras adyacentes, a la vez que proporciona protección al núcleo. Cada una de ellas está rodeada por un revestimiento y reforzada para proteger a la fibra.

  La fibra óptica es un medio excelente para la transmisión de información debido a sus excelentes características: gran ancho de banda, baja atenuación de la señal, integridad, inmunidad a interferencias electromagnéticas, alta seguridad y larga duración. Su mayor desventaja es su coste de producción superior al resto de los tipos de cable, debido a las necesidades de empleo de vidrio de alta calidad y la fragilidad de su manejo en producción.

  Dependiendo del número de haces de luz que se transmiten a la vez, se distinguen dos tipos de fibra óptica:

  - **Monomodo.** En las fibras monomodo tan sólo se envía un único haz de luz a través del cable. Por lo tanto, su velocidad es menor pero la distancia máxima del segmento es mucho mayor.

  - **Multimodo.** Se transmiten varios haces de luz a la vez por lo que su velocidad es mayor pero la distancia de segmento es menor ya que a mayor distancia, es posible que un haz "adelante" a otro haz produciéndose un error en la transmisión.

  En la tabla III-1 y III-2 puede ver la estructura de la fibra óptica y sus características.

Tabla III-1. Estructura del cableado

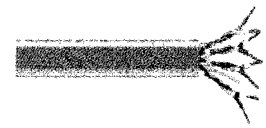
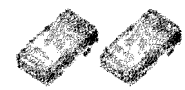
	Coaxial	Par trenzado	Fibra óptica
Estructura interna del Cable			
Conectores			

Tabla III-2. Datos técnicos del cableado

	Coaxial	Par trenzado	Fibra óptica
Velocidad	10 Mb/s	10 Mb/s (10 Base- TX) 100 Mb/s (100 Base-TX) 1000 Mb/s (1000 Base-TX)	Hasta 10 Tb/s
Distancia máxima de segmento	185 metros	100 metros	Monomodo: 100 Km Multimodo: 2,4 Km

3 Dispositivos de interconexión

Los dispositivos de interconexión permiten conectar segmentos de una misma red, o redes diferentes. Los dispositivos que se utilizan en una red son:

- **Repetidores.** Tal y como hemos visto en el apartado anterior, cualquier medio físico tiene una longitud máxima de segmento. Por ejemplo, la longitud máxima de un cable UTP Cat 5 es de 100 metros. Esto quiere decir que si utilizamos un cable con una longitud mayor, en la señal eléctrica existe demasiada atenuación o interferencias que hacen que la comunicación tenga muchos errores o que incluso sea impracticable.

  Tal y como puede ver en la figura III-1, el repetidor es un dispositivo que regenera la señal transmitida evitando su atenuación; de esta forma se puede ampliar la longitud del cable que soporta la red. Por ejemplo, si

queremos conectar dos equipos que se encuentran a una distancia de 150 metros, necesitaremos un repetidor que divida el cable en dos partes; de forma que ninguna exceda la longitud máxima del segmento del cable (100m).

Figura III-1. Esquema de un repetidor

- **Hub.** Un Hub es un dispositivo de interconexión que permite conectar varios host o varios segmentos de una misma red. El tamaño de un hub viene determinado por el número de entradas que tiene (puertos). Existen Hub desde 4 puertos a 128 puertos. En la figura III.2 podemos ver un Hub de la familia 3Com.

  El funcionamiento interno del Hub es como el de un "enchufe ladrón". El Hub recibe una señal por un puerto y lo que hace es enviar la señal recibida por todos los demás puertos. Por lo tanto, una restricción que tiene un Hub es evitar que se produzcan colisiones cuando recibe una señal por varios puertos.

  Un hub tiene dos grandes desventajas: es un dispositivos lento e inseguro ya que toda la información de un puerto se envía a los demás puertos.

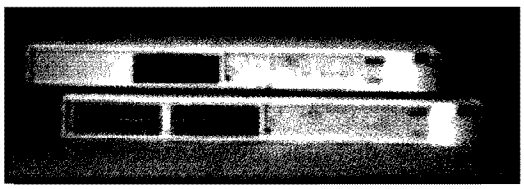


Figura III-2. HUB

- **Switch.** Al igual que un Hub, un Switch es un dispositivo de interconexión que permite conectar varios host o varios segmentos de una misma red. La diferencia entre un Hub y un Switch es que un Switch tiene una pequeña memoria asociativa en la que guarda la dirección física (MAC) del equipo que está conectado a cada uno de sus puertos. De esta forma, al recibir un mensaje el switch mira la dirección de destino y lo envía sólo a su destinatario.

El switch resuelve los problemas de rendimiento y de seguridad de la red que tienen los hubs. El switch puede agregar mayor ancho de banda, acelerar la salida de paquetes, reducir el tiempo de espera y bajar el coste por puerto.

En la figura III-3, se puede ver un ejemplo de un switch Catalyst de Cisco System.

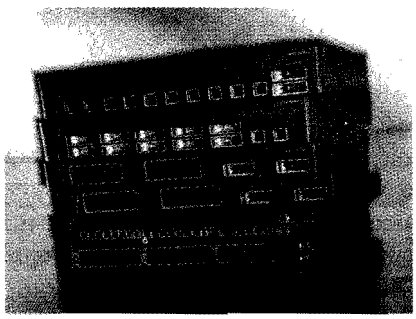


Figura III-3. Switch Cisco Catalyst 2950

- **Bridges.** Los Bridges o puentes son dispositivos que ayudan a resolver el problema de limitación de distancias, junto con el problema de limitación del número de nodos de una red. Trabajan al nivel de enlace del modelo OSI, por lo que pueden interconectar redes que cumplan las normas del modelo 802 (3, 4 y 5). Si los protocolos por encima de estos niveles son diferentes en ambas redes, el bridge no es consciente, y por tanto no puede resolver los problemas que puedan presentarse.

  Los Bridges se utilizan para ampliar la extensión de la red, para reducir la carga de una red o para unir redes de diferente topología.

- **Gateways.** Se trata de un ordenador u otro dispositivo que interconecta redes radicalmente distintas. Son capaces de traducir información de una red a otra, como por ejemplo las pasarelas de correo electrónico. Un gateway trabaja en la capa de aplicación ya que necesita conocer el tipo de información que tiene que traducir de una red a otra.

- **Routers.** Un router es un dispositivo de propósito general diseñado para segmentar la red, con la idea de limitar el tráfico de broadcast y proporcionar seguridad, control y redundancia entre dominios broadcast. También puede dar servicio de firewall.

  En la figura III-4, podemos encontrar un ejemplo de utilización de un router para conectar una red interna con Internet.

Un router opera en la capa de red del modelo TCP/IP y tiene más prestaciones que un switch. El router distingue entre los diferentes protocolos de red tales como IP, IPX, AppleTalk, etc. Para poder trabajar un router en la capa de red es necesario que tenga una dirección IP por cada interfaz del router.

El router tiene dos funciones básicas:

- **Enrutamiento.** El router es responsable de crear y mantener las tablas de enrutamiento para cada capa de protocolo de red. Estas tablas son creadas estáticamente o dinámicamente. De esta manera, el router extrae del paquete IP la dirección de destino y selecciona el mejor camino basándose en diversos factores. Estos factores pueden incluir el número de saltos hacia el destino, la velocidad de línea, el coste de la transmisión, las condiciones del tráfico, etc.

- **Filtrado de paquetes.** El router al comunicar varias redes es el encargado ideal para decidir qué información tiene que pasar o qué información tiene que ser bloqueada. A partir de las tablas de enrutado (o tablas de filtrado de paquetes) el router toma la decisión de qué acción tiene que realizar con cada paquete. Por lo tanto un router es un dispositivo que nos puede ayudar a mantener la seguridad de una red.

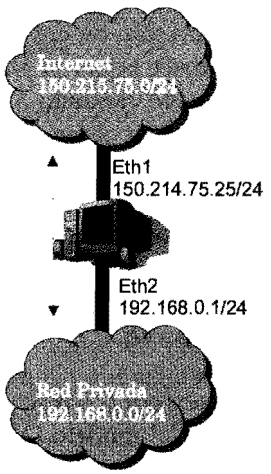


Figura III-4. Ejemplo de utilización de un router

En la figura III-5, podemos ver un ejemplo de un router hardware.

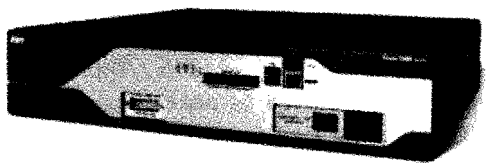


Figura III-5. Router Cisco 2821

4 El protocolo TCP/IP

En 1969 la agencia ARPA (*Advanced Research Projects Agency*) del Departamento de Defensa de los Estados Unidos inició un proyecto de interconexión de ordenadores mediante redes telefónicas. Al ser un proyecto desarrollado por militares en plena Guerra Fría, un principio básico de diseño era que la red debía poder resistir la destrucción de parte de su infraestructura (por ejemplo a causa de un ataque nuclear), de forma que dos nodos cualesquiera pudieran seguir comunicados siempre que hubiera alguna ruta que los uniera. Esto se consiguió en 1972 creando una red de conmutación de paquetes denominada ARPAnet, la primera de este tipo que operó en el mundo. La conmutación de paquetes unida al uso de topologías malladas mediante múltiples líneas punto a punto dio como resultado una red altamente fiable y robusta.

ARPAnet fue creciendo paulatinamente, y pronto se hicieron experimentos utilizando otros medios de transmisión de datos, en particular enlaces por radio y vía satélite; los protocolos existentes tuvieron problemas para interoperar con estas redes, por lo que se diseñó un nuevo conjunto o pila de protocolos, y con ellos una arquitectura. Este nuevo conjunto se denominó TCP/IP (*Transmission Control Protocol/Internet Protocol*), nombre que provenía de los dos protocolos más importantes que componían la pila; la nueva arquitectura se llamó sencillamente *modelo TCP/IP*, los nuevos protocolos fueron especificados por vez primera por Cerf y Kahn en un artículo publicado en 1974. A la nueva red, que se creó como consecuencia de la fusión de ARPAnet con las redes basadas en otras tecnologías de transmisión, se la denominó Internet.

La aproximación adoptada por los diseñadores del TCP/IP fue mucho más pragmática que la de los autores del modelo OSI. Mientras que en el caso de OSI se emplearon varios años en definir con sumo cuidado una arquitectura de capas donde la función y servicios de cada una estaban perfectamente definidas, y sólo después se planteó desarrollar los protocolos para cada una de ellas, en el caso de TCP/IP la operación fue a la inversa; primero se especificaron los protocolos, y luego se definió el modelo como una simple descripción de los protocolos ya existentes. Por este motivo el modelo TCP/IP es mucho más simple que el OSI.

También por este motivo el modelo OSI se utiliza a menudo para describir otras arquitecturas, como por ejemplo TCP/IP, mientras que el modelo TCP/IP nunca suele emplearse para describir otras arquitecturas que no sean la suya propia.

Tal y como se muestra en la figura III.6, el modelo TCP/IP tiene sólo cuatro capas:

1. **La capa host-red.** Esta capa permite enviar la información a través del medio físico y nos permite el direccionamiento físico. Para enviar la información a través del medio físico es necesario conocer las características físicas del medio. Por ejemplo, es totalmente diferente el envío de información a través de un cable coaxial que a través de una fibra óptica.

   El direccionamiento físico permite que los equipos puedan comunicarse. Cuando se encuentra en un medio broadcast, por ejemplo un cable ethernet, el origen y destino de un paquete se obtiene a través de la dirección MAC del adaptador de red. La dirección MAC es un número serie único que tiene cada adaptador de red.

2. **La capa de red.** El objetivo de esta capa es permitir que los nodos introduzcan paquetes en cualquier red y lo hagan viajar de forma independiente a su destino (que podría estar en una red diferente). Los paquetes pueden llegar incluso en un orden diferente a aquel en que se enviaron, en cuyo caso corresponde a las capas superiores reordenándolos, si se desea la entrega ordenada.

   La capa de red define un formato de paquete y protocolo oficial llamado IP (Internet Protocol). El objetivo de la capa de red es entregar paquetes IP donde se supone que deben ir. Aquí la consideración más importante es decidir el camino que tienen que seguir los paquetes (encaminamiento), y también evitar la congestión.

3. **La capa de transporte.** La capa de transporte se diseñó para permitir que las entidades pares en los nodos de origen y destino lleven a cabo una conversación. Aquí se definieron dos protocolos punto a punto. El primero, TCP (Transmission Control Protocol), es un protocolo orientado a la conexión que permite que un flujo de bytes originado en una máquina se entregue sin errores en cualquier máquina destino. Este protocolo fragmenta el flujo entrante de bytes en mensajes y pasa cada uno a la capa de interred. En el diseño, el proceso TCP receptor reensambla los mensajes recibidos para formar el flujo de salida. TCP también se encarga del control de flujo para asegurar que un emisor rápido no pueda saturar a un receptor lento con más mensajes de los que pueda gestionar.

El segundo protocolo de esta capa, UDP (User Datagram Protocol), es un protocolo sin conexión, para aplicaciones que no necesitan la asignación de secuencia ni el control de flujo TCP y que desean utilizar los suyos propios.

En esta capa además se realiza el direccionamiento por puertos. Gracias a la capa anterior, la de interred, los paquetes viajan de un ordenador origen a un ordenador destino. La capa de transporte se encarga de que, dentro del ordenador, la información se envíe a la aplicación adecuada (mediante un determinado puerto).

4. **La capa de aplicación.** Por encima de la capa de transporte está la capa de aplicación, que contiene todos los protocolos de alto nivel. El protocolo de terminal virtual (TELNET), el de transferencia de ficheros (FTP), el de correo electrónico (SMTP), etc. Con los años se han añadido muchos otros protocolos, como el servicio de nombres de dominio (DNS) para relacionar los nombres de los nodos con sus direcciones de red; NNTP, el protocolo que se utiliza para transferir noticias; HTTP, el protocolo que se usa para ver páginas Web, y muchos más.

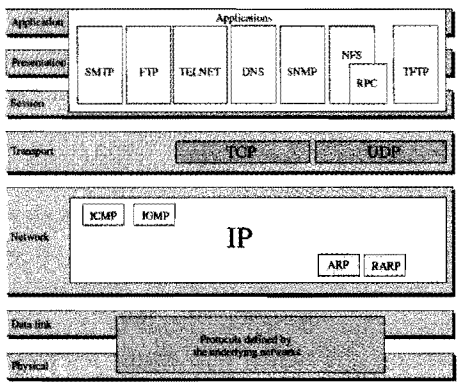


Figura III-6. Modelo TCP/IP

5 Direccionamiento IP

Cada interfaz de red de cada nodo (host o router) en una red IP se identifica mediante una dirección única de 32 bits. Las direcciones IP se suelen representar por cuatro números decimales separados por puntos, que equivalen al valor de cada uno de los cuatro bytes que componen la dirección. Por ejemplo una dirección IP válida sería 147.156.23.208.

Si un nodo dispone de varias interfaces físicas (cosa habitual en los routers) cada una de ellas deberá tener necesariamente una dirección IP distinta. Es posible además, y en algunas situaciones resulta útil, definir varias direcciones IP asociadas a una misma interfaz física.

5.1 CLASES DE DIRECCIONES

Las direcciones IP tienen una estructura jerárquica. Tal y como muestra la figura III-7, una parte de la dirección corresponde a la red (netid), y la otra al host dentro de la red (hostid). Cuando un router recibe un datagrama (mensaje) por una de sus interfaces, compara la parte de red de la dirección con las entradas contenidas en sus tablas (que normalmente sólo contienen direcciones de red, no de host) y envía el datagrama por la interfaz correspondiente.

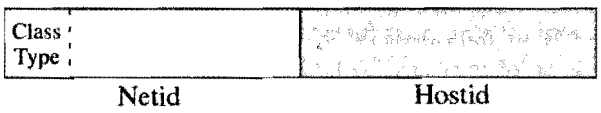


Figura III-7. Partes de una dirección IP

En el diseño inicial de Internet se reservaron los ocho primeros bits para la red, dejando los 24 restantes para el host; se creía que con 254 redes habría suficiente para una red experimental que era fruto de un proyecto de investigación del Departamento de Defensa americano. Ya en 1980 se vio que esto resultaba insuficiente, por lo que se reorganizó el espacio de direcciones reservando una parte para poder definir redes más pequeñas. Para dar mayor flexibilidad y permitir diferentes tamaños se optó por dividir el rango de direcciones en tres partes adecuadas para redes grandes, medianas y pequeñas, conocidas como redes de clase A, B y C, respectivamente:

- Una red de clase A (que corresponde a las redes originalmente diseñadas) se caracteriza por tener a 0 el primer bit de dirección; el campo red ocupa los 7 bits siguientes y el campo host los últimos 24 bits. Puede haber hasta 126 redes de clase A con 16 millones de hosts cada una.

- Una red de clase B tiene el primer bit a 1 y el segundo a 0; el campo red ocupa los 14 bits siguientes, y el campo host los 16 últimos bits. Puede haber 16382 redes clase B con 65534 hosts cada una.

- Una red clase C tiene los primeros tres bits a 110; el campo red ocupa los siguientes 21 bits, y el campo host los 8 últimos. Puede haber hasta dos millones de redes de clase C con 254 hosts cada una.

Para indicar qué parte de la dirección corresponde a la red y qué parte al host, se suele utilizar una notación denominada "*máscara de red*", consistente en poner a 1 los bits que corresponden a la parte de red y a 0 los que corresponden a la parte host. Así, por ejemplo, diremos que una red clase A tiene una máscara 255.0.0.0, lo cual equivale a decir que los ocho primeros bits especifican la red y los 24 restantes el host. Análogamente decimos que una red clase B tiene una máscara 255.255.0.0 y una clase C una máscara 255.255.255.0. Otra notación utilizada en muchos sistemas es expresar de forma conjunta con la dirección IP el número de bits de la máscara de red. Así por ejemplo, para expresar una dirección de clase A sería 12.15.19.1/8, 12.15.19.1/16 de clase B y 12.15.19.1/24 de clase C.

Además existen direcciones de clase D (no redes) cuyos primeros cuatro bits valen 1110, que se utilizan para definir grupos multicast (el grupo viene definido por los 28 bits siguientes).

Por último, la clase E, que corresponde al valor 11110 en los primeros cinco bits, está reservada para usos futuros.

A partir de los valores de los primeros bits de cada una de las clases mencionadas anteriormente, se puede deducir el rango de direcciones que corresponde a cada una de ellas. Así pues, en la práctica es inmediato saber a qué clase pertenece una dirección determinada sin más que leer el primer byte de su dirección. La siguiente tabla resume toda la información esencial sobre los tipos de direcciones de Internet.

A modo de resumen, en la figura III-8 puede ver un esquema de las diferentes clases de direcciones y en la tabla III-9 puede ver las características principales de las clases de direcciones.

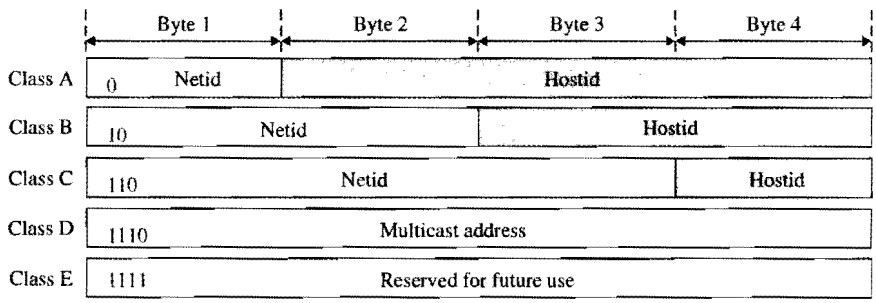


Figura III-8. Clases de direcciones

Tabla III-3. Características principales de las clases de direcciones

Clase	Bits Reservados	Bits red/host	Número de redes	Número de ordenadores	Rango
A	0---	7/24	126	16777214	1.0.0.0 127.255.255.255
B	10--	14/16	16384	65334	128.0.0.0 191.255.255.255
C	110-	21/8	2097152		192.0.0.0 223.255.255.255
D	1110				224.0.0.0 239.255.255.255
E	1111				240.0.0.0 255.255.255.255

La asignación de direcciones válidas de Internet la realizan los NICs (NIC = *Network Information Center*). Al principio había un NIC para toda Internet pero luego se crearon NICs regionales (por continentes). Actualmente muchos países tienen un NIC propio, así ocurre en España donde el NIC es administrado por RedIRIS.

5.2 DIRECCIONES ESPECÍFICAS

Existen unas reglas y convenios en cuanto a determinadas direcciones IP que es importante conocer:

1. La dirección **255.255.255.255** se utiliza para indicar broadcast en la propia red, cualquiera que sea (y sea del tipo que sea).

2. La dirección **0.0.0.0** identifica al host actual.

3. La dirección con el **campo host todo a ceros** se utiliza para indicar la red misma, y por tanto no se utiliza para ningún host. Por ejemplo la dirección 193.147.7.0 identifica la red clase B que pertenece a la Universidad de Valencia.

4. La dirección con el **campo host todo a unos** se utiliza como la dirección broadcast de la red indicada, y por tanto no se utiliza para ningún host. Por ejemplo para enviar un mensaje broadcast en la red anterior, utilizaríamos la dirección 193.147.7.255.

5. La dirección con el **campo red todo a ceros** identifica a un host en la propia red, cualquiera que sea; por ejemplo si queremos enviar un

datagrama al primer host (1) de una red clase B podemos utilizar la dirección 0.0.0.1. Esto permite enviar datagramas sin saber en qué red nos encontramos, aunque es preciso conocer si es clase A, B o C para saber qué parte de la dirección es red y qué parte es host.

6. La dirección **127.0.0.1** se utiliza para pruebas loopback; todas las implementaciones de IP devuelven a la dirección de origen los datagramas enviados a esta dirección sin intentar enviarlos a ninguna parte.

Como consecuencia de las reglas 3 y 4 siempre hay dos direcciones no asignables a hosts en una red. Por ejemplo, si tenemos la red 200.200.200.0 (clase C) tendremos que reservar la dirección 200.200.200.0 para denotar la red misma, y la dirección 200.200.200.255 para envíos broadcast a toda la red; dispondremos pues de 254 direcciones para hosts, no de 256.

A modo de resumen, en la tabla III-4 puede ver un ejemplo de las diferentes direcciones específicas.

Tabla III-4. Direcciones específicas

Dirección especial	Netid	Hostid	Ejemplo (193.147.7.32/24)
Dirección de red	Específica	Todo a 0	193.147.7.0
Dirección directa de broadcast	Específica	Todo a 1	193.147.7.255
Dirección broadcast limitada	Todo a 1	Todo a 1	255.255.255.255
Este host en esta red	Todo a 0	Todo a 0	0.0.0.0
Host específico en esta red	Todo a 0	Específica	0.0.0.32
Dirección loopback	127.	Cualquiera	127.0.0.1

5.3 DIRECCIONES PRIVADAS

La tabla III-5 nos muestra que las direcciones de red **10.0.0.0, 172.16.0.0 a 172.31.0.0, y 192.168.0.0 a 192.168.255.0** están reservadas para redes privadas (intranets) por el RFC 1918. Estos números no se asignan a ninguna dirección válida en Internet y por tanto pueden utilizarse para construir redes privadas. Por ejemplo detrás de un cortafuegos, sin riesgo de entrar en conflicto de acceso a redes válidas de Internet.

Tabla III-5. Direcciones privadas

Clase	Rango	Número de redes
A	10.x.x.x	1
B	De 172.16.x.x A 172.31.x.x	16
C	De 192.168.x.x a 192.168.255.x	256

6 Configuración de routers

Como hemos visto anteriormente, un router es un dispositivo de interconexión que permite regular el tráfico que pasa entre varias redes. Un router es muy útil a la hora de defendernos de posibles intrusiones o ataques externos. Pero la desventaja es que un router no se configura por sí solo. Mientras que un router bien configurado puede ser muy útil, un router mal configurado no nos proporciona ningún tipo de protección o, simplemente, no llega a comunicar dos redes.

6.1 TABLAS DE ENRUTADO

Para configurar un router debemos crear lo que se denomina "tabla de enrutado". En ella se guardan las acciones que hay que realizar sobre los mensajes que recibe el router para redirigirlos a su destino. Existen dos tipos de encaminamiento: "encaminamiento clásico" y "encaminamiento regulado".

6.1.1 Encaminamiento clásico

Con el "encaminamiento clásico", las reglas utilizadas para encaminar los paquetes se basan, exclusivamente, en la dirección destino que aparece en la cabecera del paquete. Así se distinguen las siguientes reglas:

- Permitir un equipo de nuestra red.
- Permitir cualquier equipo de nuestra red.
- Permitir un equipo de otra red.
- Permitir cualquier equipo de otra red.

La última regla (por defecto) se aplica en el caso de que no se cumpla ninguna de las anteriores y se suele utilizar para poder enviar los mensajes a la puerta de enlace de la red.

6.1.2 Encaminamiento regulado

Sin embargo, en la actualidad, con la explosión del uso de Internet y la llegada del concepto de calidad de servicio (QoS) y la seguridad, los routers utilizan el llamado "encaminamiento regulado", con el que, a la hora de escribir la tabla de enrutado, se pueden utilizar los siguientes elementos:

A la hora de escribir una tabla de enrutado se pueden utilizar los siguientes elementos:

- **Interfaz:** interfaz de red por donde se recibe la información.
- **Origen / Destino:** origen y destino del mensaje. Normalmente el origen y el destino de un mensaje es una dirección IP, pero algunos routers permiten utilizar como dirección origen y destino usuarios o grupos de usuarios.
- **Protocolo:** permitir o denegar el acceso a los puertos es importante porque las aplicaciones servidoras (que aceptan conexiones originadas en otro ordenador) deben 'escuchar' en un puerto para que un cliente (que inicia la conexión) pueda conectarse. Por ejemplo un servidor web trabaja en el puerto 80, un servidor de FTP en el puerto 21, etc.
- **Seguimiento:** indica si el router debe realizar un seguimiento de los lugares por los que pasa un mensaje.
- **Tiempo:** espacio temporal en el que es válida la regla.
- **Autentificación de usuarios:** indica si el usuario debe estar autentificado para utilizar la regla.
- **Acción:** especifica la acción que debe realizar el router. Un router puede realizar las siguientes acciones:
    - **Aceptar:** dejar pasar la información.
    - **Denegar:** no deja pasar la información.
    - **Reenviar:** envía el paquete a una determinada dirección IP.

Existen diferentes tipos de routers por lo que en un principio podemos caer en la tentación de pensar que el proceso de configuración para cada router es totalmente diferente a los demás. Los router más utilizados son:

- *FireWall 1* de CheckPoint
- *Private Internet Exchange (PIX)* de Cisco System
- *IOS Firewall Feature Set* de Cisco System
- *Firewall del núcleo* de Linux, Iptables
- *Enterprise Firewall* de Symantec
- *Internet Security and Acelerador (ISA Server)* de Microsoft

Si comparamos los elementos que utilizan los diferentes routers (ver tabla III-6) podemos ver cómo los más utilizados a la hora de realizar una tabla de enrutado son la **interfaz**, la **dirección origen y destino**, el **puerto** y la **acción** que debe realizar el router.

Tabla III-6. Comparativa sobre los elementos de las tablas de enrutado

Modelo	Interfaz	Origen /Destino	Protocolo	Seguimiento	Tiempo	Autentificación de usuarios	Acción
FireWall 1	√	√**	√	√	√		√
PIX	√	√	√*				√
IOS Firewall	√	√	√				√
Firewall Linux	√	√	√				√
Enterprise Firewall	√	√**	√		√	√	√
ISA Server	√	√**	√*		√	√	√

* Distingue entre puerto de origen y destino

** Permiten especificar como origen o destino direcciones IPs o usuarios.

A la hora de indicar la **dirección de origen** o la **dirección de destino** es importante utilizar la máscara de red para indicar un mayor o menor número de ordenadores. Así por ejemplo, si en la dirección destino utiliza la dirección de clase B 142.165.2.0/16 se hace referencia a todas las direcciones IP del tipo 142.165.x.x. Si utiliza la dirección de clase C 192.165.2.0/24, hace referencia a las direcciones del tipo 192.165.2.x. Por lo tanto, si aumentamos la máscara de red, estamos disminuyendo el número de direcciones IP a las que se hace referencia y si

disminuimos la máscara de red, entonces se hace referencia a un mayor número de direcciones IP. En la tabla III-7, puede ver algunas de las posibilidades más habituales.

Tabla III-7. Ejemplos de utilización de la máscara de red en la configuración de routers

Ejemplo	Comentario
192.165.2.23/32	Representa a un único ordenador (por ejemplo, servidor web)
192.165.2.0./24	Representa a todas las direcciones IP del tipo 192.165.2.X
192.165.0.0/16	Representa a todas las direcciones IP del tipo 192.165.X.X
192.0.0.0/8	Representa a todas las direcciones IP del tipo 192.X.X.X
0.0.0.0/0	Representa a todas las direcciones IP del tipo X.X.X.X

Durante el filtrado de paquetes se aplica la regla de "coincidencia total". Todos los criterios de la regla tienen que coincidir con el paquete entrante; en caso contrario, no se aplica la regla. Esto no significa que se rechace el paquete o que se elimine, sino que la regla no entra en vigor. Normalmente, las reglas se aplican en orden secuencial, de arriba hacia abajo. Aunque hay varias estrategias para implementar filtros de paquetes, las dos que se describen a continuación son las más utilizadas por los especialistas de seguridad:

- **Construir reglas desde la más específica a la más general.** Esto se hace así para que una regla general no "omita" a otra más específica, pero conflictiva, que entra dentro del ámbito de la regla general.

- **Las reglas deberían ordenarse de tal forma que las que más se utilizan estén en la parte superior de la lista.** Esto se hace por cuestiones de rendimiento. Normalmente un router detiene el procesamiento de una lista cuando encuentra una coincidencia total.

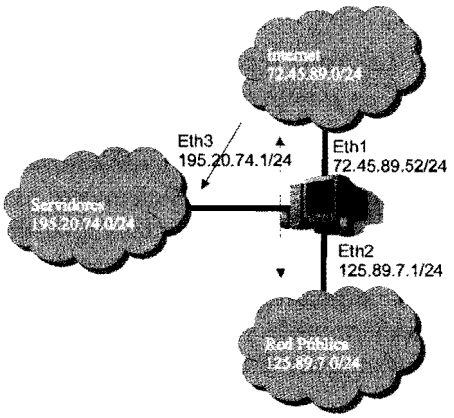
6.2 EJEMPLO DE CREACIÓN DE UNA TABLA DE ENRUTADO

La figura III-9 muestra un router conectado a tres redes diferentes. Debemos crear el conjunto de reglas para permitir que: la red pública se conecte a Internet y que los servidores sean accesibles desde Internet; el servidor web se encuentra en la dirección 195.20.74.5 y el servidor de correo se encuentra en la dirección 195.20.74.5.

La tabla de enrutado representa el conjunto de reglas que actúan como medida de seguridad para determinar si se permite que un paquete pase o no.

El conjunto de reglas está formado por seis reglas sencillas. La complejidad de las reglas tiene propósitos educativos para mostrar los conceptos del procesamiento de reglas (directiva) del filtrado de paquetes. Las notas acerca de la implementación se incluyen siguiendo la descripción de cada línea del conjunto de reglas.

Las reglas están agrupadas en tres grandes grupos: las primeras tres reglas se aplican al tráfico que tiene como origen Internet y como destino la red de servidores. Las reglas 4 y 5 permiten la comunicación entre Internet y la red pública. Y la última regla se utiliza siempre para indicar que el tráfico que no cumpla las reglas anteriores debe ser denegado.



	Interfaz	Dir. origen	Dir. destino	Puerto	Acción
1	Eth1	0.0.0.0/0	195.20.74.5/32	80	Aceptar
2	Eth1	0.0.0.0/0	195.20.74.7/32	25, 110	Aceptar
3	Eth1	0.0.0.0/0	195.20.74.0/24	-	Denegar
4	Eth1	0.0.0.0/0	125.89.7.0/24	-	Aceptar
5	Eth2	125.89.7.0/24	0.0.0.0/0	-	Aceptar
6	-	-	-	-	Denegar

Figura III-9. Ejemplo de red y de tabla de enrutado

- **Regla 1.** Esta regla permite el acceso entrante en el puerto 80, que normalmente se utiliza para el tráfico http. El host que está en 195.20.74.5 es el servidor web. La organización no puede predecir quién va a tener acceso a su sitio Web, por lo que no hay restricción en las direcciones IP de origen.

- **Regla 2.** Esta regla permite el acceso entrante a los puertos 25 y 110, que normalmente se utiliza para correo electrónico (el puerto 25 es el servidor smtp o correo saliente y el puerto 110 es el servidor pop3 o correo entrante). El servidor de correo está en la dirección 195.20.74.7. Al igual que en la regla anterior, como no se puede predecir quién va a tener acceso al servidor de correo no se restringen las direcciones IP de origen.

- **Regla 3.** Esta regla elimina todos los paquetes que tienen como destino la red donde se encuentran los servidores. Como la regla 1 y 2, si se ejecuta antes, se permite el tráfico que va dirigido a los servidores web y correo electrónico. Si se pone esta regla al principio de la tabla de enrutado, no se podrá acceder a ningún servidor.

- **Reglas 4 y 5.** La cuarta regla deja pasar el tráfico que va desde Internet a la red pública. Y la quinta regla deja pasar el tráfico que va desde la red pública a la red de Internet.

- **Regla 6.** Esta regla bloquea explícitamente todos los paquetes que no han coincidido con ningún criterio de las reglas anteriores. La mayoría de los dispositivos de análisis realizan este paso de forma predeterminada, pero es útil incluir esta última regla de limpieza. Incluirla aclara la aplicación de la directiva predeterminada y, en la mayoría de los casos, permite registrar los paquetes que coinciden con ella. Esto es útil por motivos jurídicos y administrativos.

APÉNDICE IV

CLIENTES DE VOIP

Francisco Gil Montoya y Julio Gómez López

1. Introducción

Para que se pueda mantener una conversación telefónica se necesitan terminales que puedan procesar la voz humana, transformándola en impulsos eléctricos que son enviados por la red RTC. O al menos, esto era así hasta la llegada de la telefonía IP. Bajo este nuevo enfoque, las conversaciones ya no son codificadas en impulsos eléctricos analógicos, es decir, en señales cuyo nivel de tensión y/o corriente es proporcional a la presión sonora recogida por los micros o emitida por los altavoces. Ahora, la voz es codificada/decodificada mediante conversores analógico/digitales y posteriormente (ya digital) paquetizada y lista para ser enviada por redes Ethernet (u otro tipo de redes IP) mediante TCP/IP.

Esta labor se puede realizar de diversas formas hoy en día, siendo las más utilizadas, las siguientes:

- ***Softphones*** **o Teléfonos software.**
- ***Hardphones*** **o Teléfonos IP.**
- ***Webphones*** **o Teléfonos web.**

Aun a pesar de la mejora tecnológica que implica el uso de la VozIP, hay que destacar que todavía se sigue manteniendo el componente analógico a la hora de transformar la voz en señales eléctricas. Esta fase sigue la técnica tradicional del muestreo de señales tal y como se aprecia en la figura IV-1.

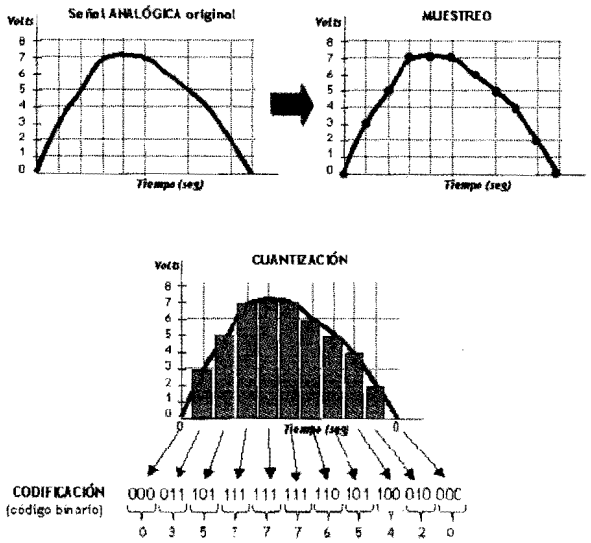


Figura IV-1. Esquema del proceso de conversión analógico-digital de una señal de voz (Fuente: http://www.iua.upf.es/~ramon/redes/digital/digital.htm)

2. Teléfono software o softphone

Los softphones son programas informáticos que simulan las funcionalidades de un teléfono convencional, pero ayudados de dos periféricos habituales: micrófono y auriculares o altavoces. Estos últimos dispositivos, conectados al ordenador, permiten que la voz pueda ser tratada por el softphone y así, ser transformada en paquetes IP.

Además de las típicas características de un teléfono, los softphones incorporan diversas funcionalidades avanzadas, muy presentes en el ámbito de la VozIP, como son videoconferencia, chat, grabación, multilínea, etc.

Existe una larga lista de ellos, ya sea de pago o de uso libre, con un sinfín de características interesantes. Quizá entre los más usados se encuentren dos muy conocidos: X-lite y SJphone.

En este apéndice se darán unas instrucciones acerca de cómo se deben configurar para que puedan funcionar con un proxy SIP cualquiera, inclusive con Asterisk.

2.1. X-LITE

El siguiente desarrollo se detalla para un sistema Windows, aunque es muy similar para cualquier otro sistema operativo.

Para instalar X-Lite debe realizar los siguientes pasos:

- Descargue el ejecutable de la página web del fabricante *Counterpath* (http://www.counterpath.com).
- Ejecute el proceso de instalación (véase la figura IV-2).
- En la pantalla que aparece a continuación, acepte la licencia y pulse *Next.*
- A continuación especifique el directorio de instalación y pulse *Next.*
- En la pantalla que aparece en la figura IV-3 indique los accesos directos que desea crear y si quiere que *x-lite* se ejecute automáticamente al iniciar el programa. Pulse *Next* para continuar.
- Se inicia el proceso de instalación y una vez finalizado, pulse *Finish.*
- Por último, reinicie el ordenador para completar la instalación.

Figura IV-2. X-Lite. Inicio del proceso de instalación

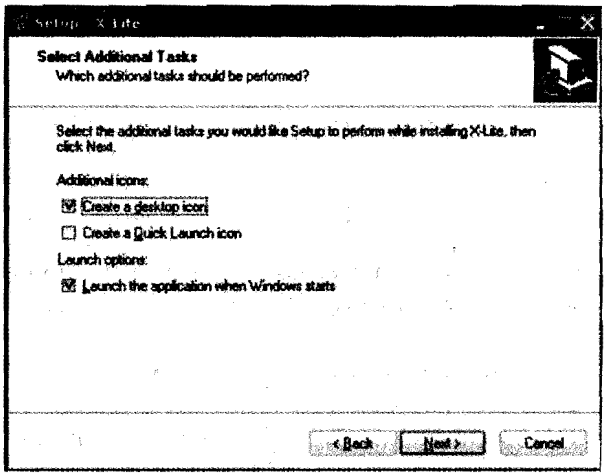


Figura III-3. X-Lite. – Tareas adicionales

***Nota:** si disponemos de Cortafuegos (Panda, McAfee, Norton, etc.), deberemos permitir las comunicaciones del software...*

Seguidamente, se abrirá automáticamente el programa para realizar la configuración de la cuenta de acceso (si esto no es así, pulse con el botón derecho y seleccione *SIP Account Settings*) y aparecerá la pantalla que aparece en la figura IV-6. Pulse *Add.*

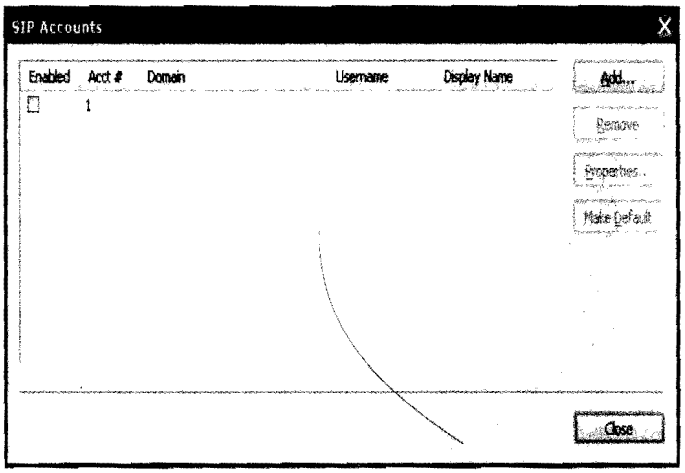


Figura IV-4. X-Lite - SIP Accounts

Introduzca los datos de configuración de la cuenta SIP tal y como muestra la figura IV-7.

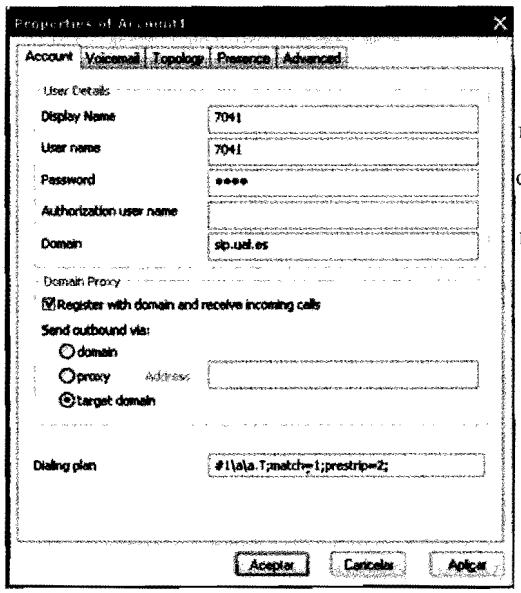


Figura IV-5. X-Lite - Configuración de una cuenta SIP

Introduzca los datos de la cuenta SIP, pulse *Aceptar* y seguidamente *Close.*

Una vez configurada la cuenta SIP, X-Lite ya está listo para realizar llamadas.

Figura IV-6. X-Lite

2.2. SJPHONE

Al igual que el detalle de X-lite, la configuración para el SJphone se presenta en entorno Windows, aunque como se dijo, es muy similar para otro sistema operativo.

Para realizar el proceso de instalación debe realizar los siguientes pasos:

- Lo primero es descargar el archivo ejecutable desde la web del fabricante *SJLabs* (http://www.sjlabs.com) y ejecútelo.
- En la ventana que aparece en la figura IV-7, pulse *Next* para iniciar el asistente de instalación.
- A continuación acepte la licencia del software, pulse *Next* y automáticamente se inicia el proceso de instalación.
- Una vez finalizado el proceso pulse *Finish* (véase la figura IV-8).

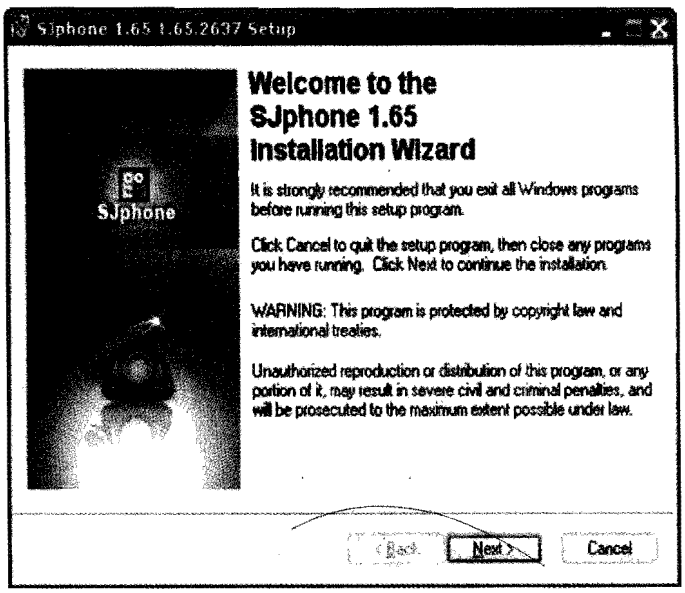


Figura IV-7. SJphone – Inicio del proceso de instalación

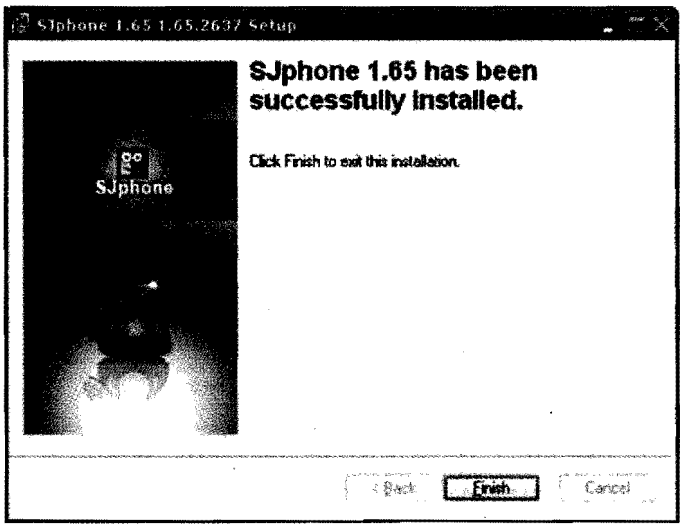


Figura IV-8. SJphone – Proceso de instalación finalizado

Si por casualidad tiene instalado algún tipo de Firewall (Panda, Norton, etc.), cuando ejecute por primera vez SJphone debe permite las comunicaciones del softphone (véase la figura IV-9).

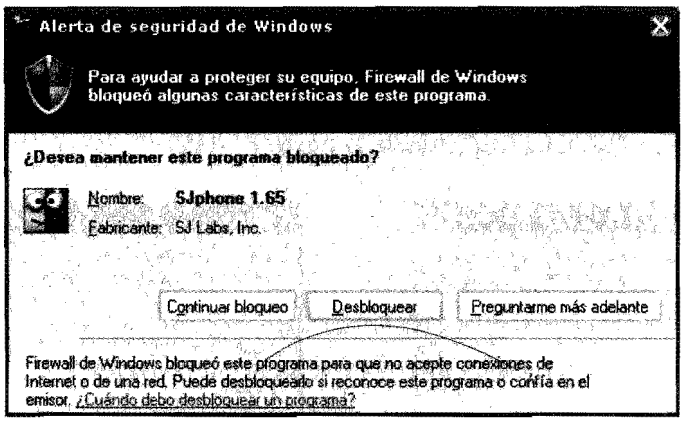


Figura IV-9. SJphone – Permitir tráfico

Seguidamente, se abrirá automáticamente el asistente para realizar una configuración automática del audio. Pulse *Next* y realice los ajustes necesarios para comprobar que el micro y el altavoz (o auriculares) funcionan correctamente.

Una vez instalado el sistema, debe configurarlo para ser utilizado con el servicio de VoIP escogido.

La vista inicial del software se muestra una pantalla como la de la figura IV-10. Inmediatamente utilice el interfaz en modo avanzado. Para ello ejecute la opción *To Advanced Mode* que se encuentra dentro de *Menu.*

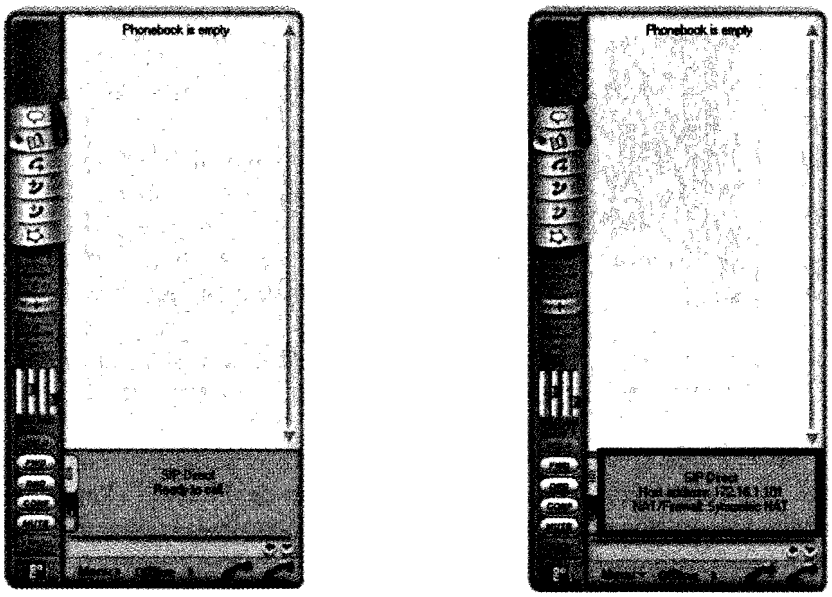


Figura IV-10. Xlite en modo inicial (izquierda) y modo avanzado (derecha)

Seguidamente configure la cuenta de su proveedor. Para ello pulse en *Options* dentro de *Menú* y seleccione *Profiles.* Debe añadir un nuevo perfil pulsando en *New* (pantalla izquierda de la figura IV-11).

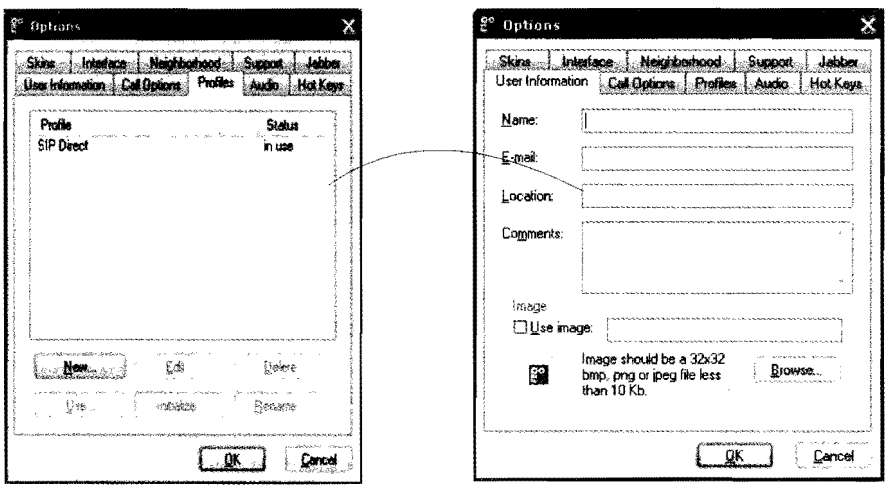


Figura IV-11. Xlite en modo inicial (izquierda) y modo avanzado (derecha)

Tal y como muestra la figura IV-12, introduzca el nombre del nuevo perfil y pulse OK.

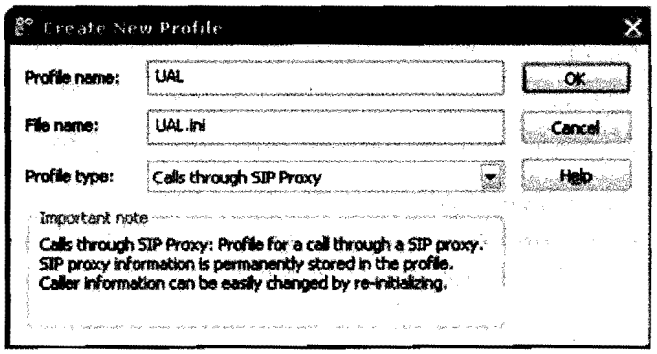


Figura IV-12. Xlite en modo inicial (izquierda)

Seguidamente configure los parámetros de conexión al servidor SIP.

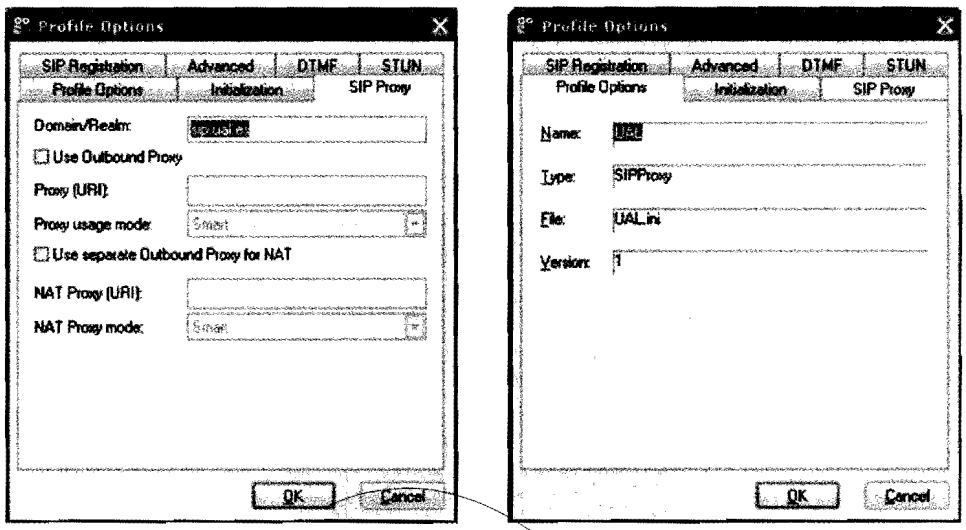


Figura IV-13. Xlite en modo inicial (izquierda)

Una vez introducidos los datos de la figura IV-13, pulse *OK* y el sistema nos pedirá el nombre de usuario y la contraseña de acceso (véase la figura IV-14).

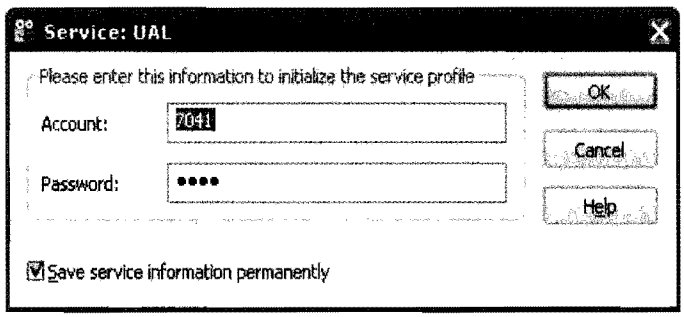


Figura IV-14. Xlite en modo inicial (izquierda) y modo avanzado (derecha)

Introducimos nuestros datos de usuario, pulsamos *OK* y ya estamos listos para hacer y recibir llamadas.

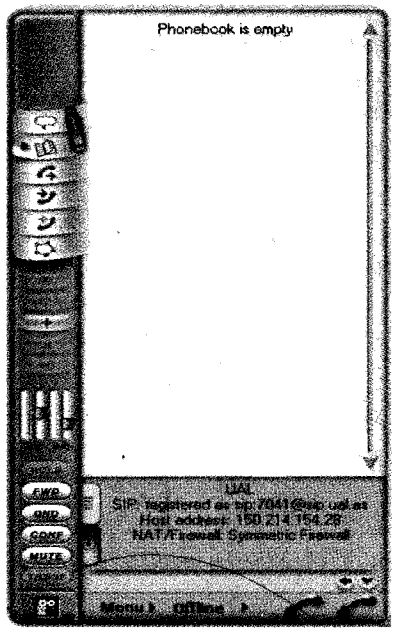


Figura IV-15. Jphone

3. Teléfono web o webphone

Un teléfono web no es más que un cliente de VozIP pero que está desarrollado completamente para funcionar en una página web. Su gran ventaja es que no requiere, generalmente, de la descarga de ningún tipo de software. Esta característica puede ser muy importante en ciertos entornos donde las políticas de

seguridad que aplican los administradores en los equipos impiden la instalación de programas no autorizados.

Otra punto fuerte es la escasez de problemas relacionados con los cortafuegos, NAT y, en general, toda aquella técnica que tenga que ver con el trasiego de paquetes desde algún software instalado en el ordenador. Esto se debe a que el navegador web utiliza el puerto 80 para comunicarse con el exterior y habitualmente este puerto suele estar libre y abierto.

El uso de teléfonos web no está tan extendido como el de los softphones, aunque cada vez se pueden encontrar con más frecuencia.

La tecnología empleada puede ser muy variada, aunque los que gozan de mayor popularidad son los basados en la tecnología Java de Sun Microsystems. Últimamente, están apareciendo otros sistemas que usan la tecnología Flash de Adobe Systems.

La configuración de estas utilidades es muy similar a la de los softphones, por lo que no se ahondará en los mismos, dejando al lector que pueda investigar por su cuenta.

A modo de ejemplo se pueden apreciar algunos modelos de *webphones* en las figuras IV-16 y IV-17.

Figura IV-16. Teléfono web con tecnología Flash (www.flashphone.ru)

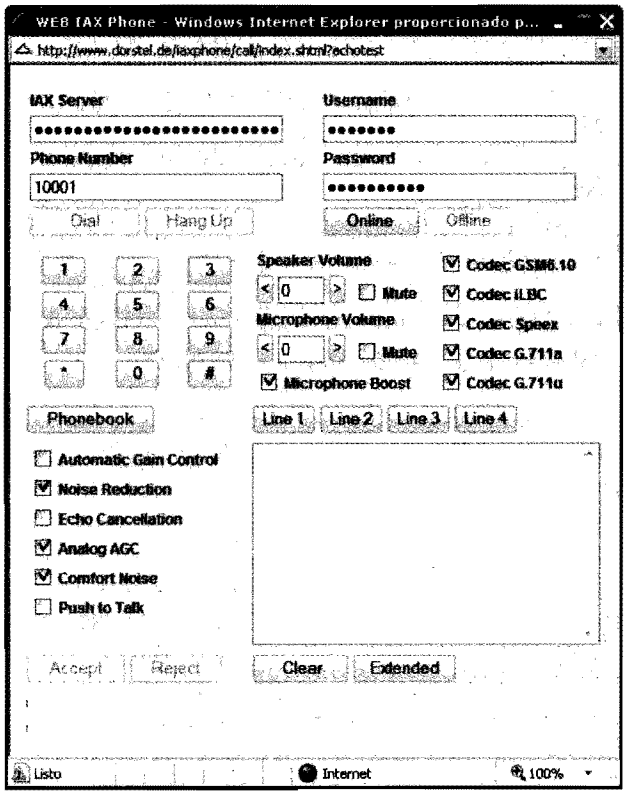


Figura IV-17. Teléfono web con tecnología JAVA

4. Teléfono IP o hardphone

El teléfono IP es el dispositivo por excelencia en las redes de VozIP. Está compuesto por la unión de un ATA, un micro con auricular, pantalla LCD y botones. Por tanto, podemos considerar que un teléfono IP es el resultado en la evolución de los teléfonos tradicionales, y al cual se le ha añadido una pantalla LCD más o menos avanzada junto al adaptador analógico-IP. Es más, hoy en día incluso es posible que dentro de un teléfono IP encontremos una placa base con un microprocesador (más o menos potente) corriendo como sistema operativo, Linux. Por tanto, y teniendo en cuenta la potencia de dicho sistema, un teléfono IP puede llegar a hacer prácticamente cualquier cosa que el fabricante desee incorporar. Valga a modo de ejemplo, que ciertos teléfonos IP contienen navegadores web, pueden trabajar con código XML e incluso permiten configurarse ellos solos para recibir instrucciones específicas de algún servidor local o externo.

Debido al gran auge de la VozIP, hoy en día existen numerosos fabricantes que ponen al alcance del usuario una gama extensa donde elegir. En el capítulo 2,

ya se habló de las diversas categorías que se han creado en función del precio final y de las características que ofrecen

Figura IV-18. Teléfono IP modelo Linksys SPA962, con funciones avanzadas y pantalla a color

Figura IV-19. Teléfono IP marca CISCO de altas prestaciones y precio

La configuración de los teléfonos IP es exactamente igual que la de los adaptadores analógicos (muchos de ellos suelen ser un ATA de la marca más pantalla y teclado, como se ha dicho anteriormente), siguiendo un patrón parecido al de la figura IV-20.

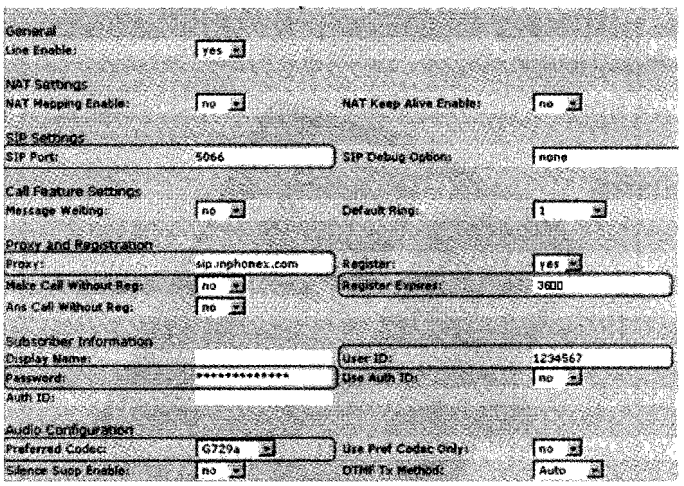


Figura IV-20. Página de configuración de un teléfono IP de Linksys

APÉNDICE V

DISTRIBUCIONES PRECOMPILADAS DE ASTERISK

Francisco Gil Montoya y Julio Gómez López

1 Introducción

Gran parte del auge que ha tenido Asterisk, y la VozIP en general, se ha basado en hacer accesible su instalación y configuración. Hay que recordar que la gran mayoría de usuarios que se acercan a Asterisk por primera vez suelen estar familiarizados, habitualmente, sólo con entornos Windows, por lo que el hecho de trabajar con Linux (o cualquier otro sistema operativo que no sea Windows, como FreeBSD, Solaris, etc.) supone una barrera inicial muy fuerte. Además de no conocer el sistema operativo sobre el cual se asienta Asterisk, muchos de los usuarios novatos tampoco presentan destrezas en el uso de sistemas informáticos o en el manejo de dispositivos de red. Por si fuera poco, es difícil conseguir, en un tiempo razonable, las nociones técnicas mínimas que serían deseables para dicho usuario inexperto, con lo que esto supone un factor en contra, por ejemplo, a nivel empresarial. Si hay algo que no sobra en una empresa es tiempo.

Realmente, existe una necesidad de poder trabajar con Asterisk de una forma más accesible e inmediata, de manera que sea posible poder "echar a andar"

un sistema, más o menos completo, de forma automatizada. Desde una perspectiva económica, una empresa que no dispone de servicio informático propio o, si lo posee, es escaso, no se puede permitir el lujo de emplear el valioso tiempo de sus técnicos en entender la idiosincrasia de cientos de sistemas novedosos que afloran en el mercado y que prometen "revolucionar" la manera de trabajar, relacionarse, etc. Pero a lo mejor sí pueden dedicar una pequeña fracción de tiempo en probar un sistema que está listo en 30 minutos, y que, efectivamente, ayuda a ahorrar costes y a optimizar muchas de las rutinas de la propia empresa. Si ese sistema precompilado existe, sería cuanto menos interesante verlo en funcionamiento. De la experiencia obtenida al "juguetear" con el sistema, se podrían inferir conclusiones que desembocasen (o no) en una mayor profundización en sus características y posibilidades.

Pues bien, esta funcionalidad la aportan los sistemas precompilados para Asterisk, de entre lo cuales se puede citar al archiconocido *Trixbox*, o los más novedosos como *Elastix*, *PiaF* o *Voicebuntu*.

Mucho se ha escrito acerca de la conveniencia en el uso de distribuciones precompiladas frente a la utilización de Asterisk *"a pelo"* (en cualquier Linux), de cara al aprendizaje del usuario no iniciado. Existen muchos detractores del uso de sistemas automatizados, los cuales argumentan, principalmente, que desnaturaliza y enmascara el uso y aprendizaje de la filosofía de Asterisk, así como que se acaba desconociendo la potencia real del sistema en su conjunto. Por otro lado, los entusiastas de estas distribuciones defienden la facilidad de instalación y manejo, la sencillez de configuración y el elevado grado de automatización de tareas que conlleva. Además, ciertas distribuciones (como Trixbox o Elastix) suelen incorporar, de serie, otras herramientas adicionales al propio servidor Asterisk (como Apache, MySQL, etc.) y que permiten añadir nuevas funcionalidades en el manejo diario del sistema.

Es el usuario final el que debe entender las bondades del uso de estos sistemas precompilados, y decidir dónde le es más conveniente su utilización (si es que realmente le interesa). Así mismo, el usuario debe ser consciente de que no puede descuidar la investigación y el aprendizaje sobre Asterisk, por más que pueda ayudarse de un *front-end*[1] o *GUI*[2] para determinadas labores concretas. No conocer la filosofía de trabajo de Asterisk es un grave error en el que no debe incurrir aquel usuario que pretenda mantener o instalar servidores de telefonía basados en Asterisk.

---

[1] El front-end es la parte de un software que interactúa con el usuario.

[2] GUI: *Graphical User Interface* o Interfaz Gráfica de Usuario.

2 Sistemas para servidor

Seguidamente se pasará a describir las distribuciones que incluyen Asterisk y que tienen más aceptación en el mercado. Cabe destacar que casi todas ellas son de código abierto (*OpenSource*) y, por tanto, el usuario tiene cierta libertad para usar, modificar y/o distribuir el código con las personalizaciones que estime oportunas.

Para dar una idea de la potencia y facilidad de uso de las mismas, se detallará de manera particular y más específica el uso de Elastix, el cual ha irrumpido fuertemente en el mercado gracias al apoyo inicial de un grupo de usuarios latinoamericanos. Puesto que este libro está dirigido especialmente al mercado hispanohablante, se ha considerado oportuno hacer esta mención por su relevancia.

2.1 ELASTIX

Tal y como lo describen sus propios autores, Elastix es:

> *"... un software que integra las mejores herramientas disponibles para PBX's[3] basadas en Asterisk pero con una interfaz simple y fácil de usar. Además añade su propio conjunto de utilidades y permite la creación de módulos de terceros para hacer de este el mejor paquete de software disponible para la telefonía de código abierto.*
>
> *La meta de Elastix son la confiabilidad, modularidad y fácil uso. Estas características, añadidas a la robustez para reportar, hacen de él la mejor opción para implementar un PBX basada en Asterisk".*

La idea que subyace tras Elastix (igual que en las otras distribuciones) es poder tener una centralita telefónica avanzada en menos de 30 minutos. Para ello, se hace uso de una imagen *.iso* que incluye todo lo necesario para ser instalado en un servidor (o PC convencional) de forma que tras el reinicio del mismo, podamos estar en condiciones de efectuar llamadas y usar servicios.

---

[3] Centralita telefónica. Ver http://es.wikipedia.org/wiki/PBX

Elastix está basado en Centos 5 e incluye una larga lista de módulos incorporados que, a modo de resumen, se enumeran a continuación:

- Soporte de video para realizar videollamadas (gracias al soporte de video en Asterisk 1.4).
- Soporte para virtualización que permite múltiples máquinas virtuales Elastix sobre el mismo servidor.
- Interfaz Web para el usuario realmente amigable.
- "Fax a email" para faxes entrantes. También se pueden enviar documentos para fax a través de una impresora virtual.
- Interfaz Web con sistema de tarificación incluido.
- Configuración gráfica de parámetros de red.
- Reportes de uso de recursos.
- Opciones para reiniciar/apagar remotamente.
- Detalle de llamadas entrantes/salientes y uso de canales (*Call Detail Recording* o *CDR* en inglés).
- Módulo de correo de voz integrado (buzón de voz).
- Interfaz Web para correo de voz.
- Módulo de panel operador integrado (*Flash Operador Panel FOP*[4]).
- Módulos extras: aplicación para CRM SugarCRM[5] y aplicación para gestión de tarjetas de llamadas "*Calling Card*"[6] .
- Sección de descargas con software de uso común.

---

[4] http://www.asterinic.org

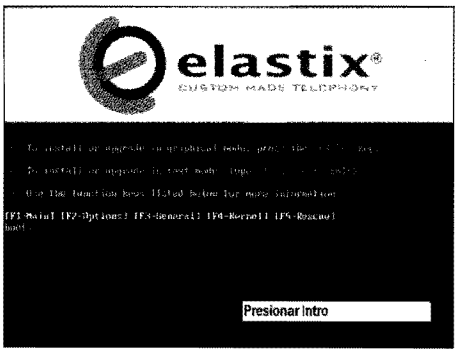
[5] http://www.sugarcrm.com/crm/

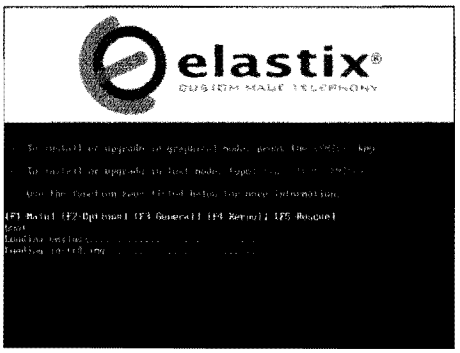
[6] http://www.asterisk2billing.org

- Interfaz de ayuda empotrado.
- Servidor de mensajería instantáneo *Openfire*[7] integrado.
- Soporte multi-lenguaje: inglés, español, ruso, coreano, griego, chino, polaco, alemán, francés, rumano, esloveno, portugués, danés, italiano, húngaro, búlgaro, serbio, croata y persa.
- Servidor de correo integrado que incluye soporte multi-dominio.
- Interfaz web para correo electrónico.
- Módulo para *Call Center* o Centro de Atención Telefónica.

2.1.1 Instalación

Lo primero que debe realizar es descargar la versión ISO disponible en http://www.elastix.org. Una vez descargada, grábela en un CD mediante algún software específico. Existen multitud de ellos en versión libre tanto para Windows (*Express Burn*, *MagicISO*, etc.) como para Linux (*K3B*, *GnomeMaker*, etc.). Finalmente, inserte el CD en el servidor donde desea instalar el sistema, e inicie el ordenador. El proceso de instalación debe ser como el que seguidamente se detalla en la figura V-1, teniendo en cuenta las instrucciones que se proporcionan en las esquinas inferiores derechas.





[7] http://www.igniterealtime.org/projects/openfire/index.jsp

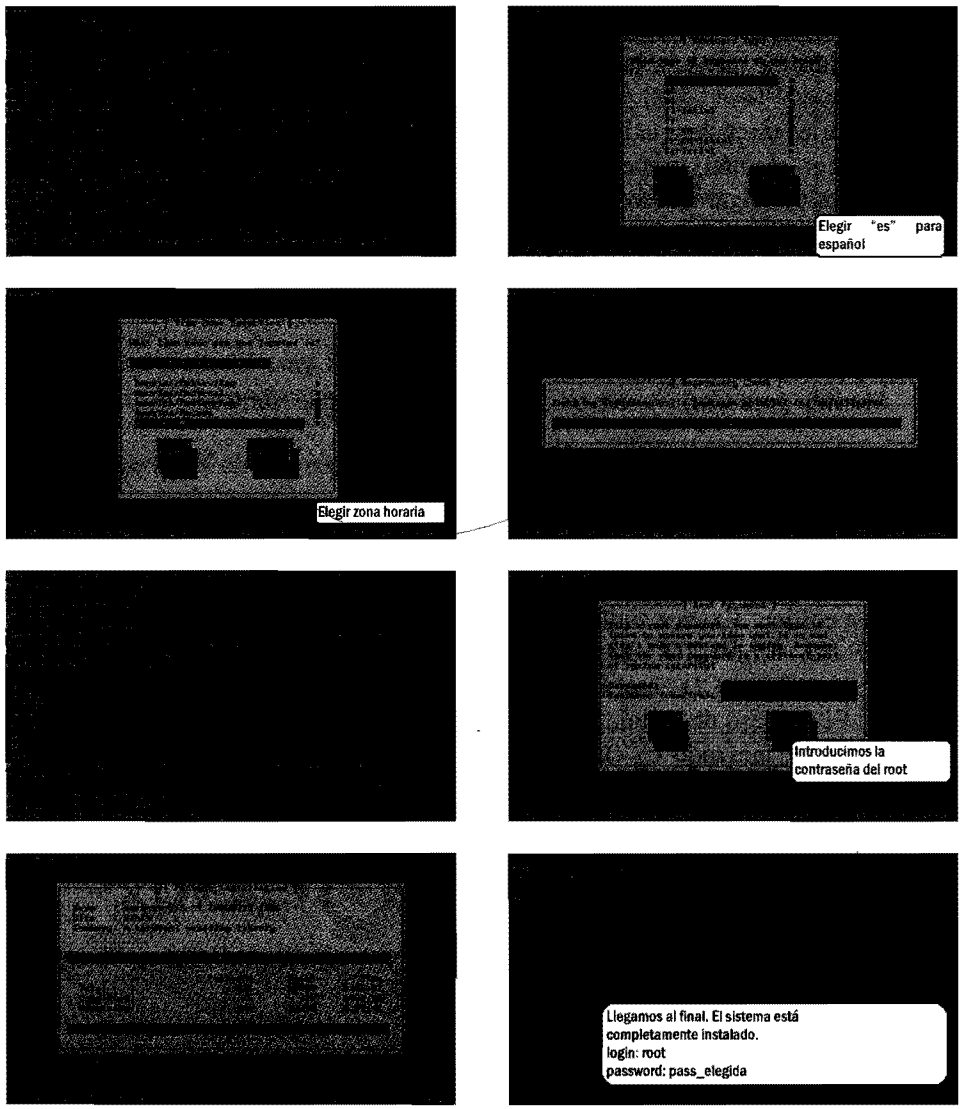


Figura V-1. Proceso de instalación de Elastix

Como se ha podido apreciar, el proceso de instalación no tiene mayor misterio que presionar la tecla *Intro* o *Enter* alguna que otra vez, e introducir contraseñas. Si ocurriese algún error, se deberá empezar desde el principio, ya que lo más probable es que haya algún conflicto con el hardware del equipo, por lo que deberá examinarse el mismo para verificarlo.

Una vez que tiene el sistema completamente instalado, ya puede acceder a la interfaz web de administración.

Es importante resaltar, para los no iniciados, que el servidor Elastix está montado sin la funcionalidad de escritorio, por lo que sólo dispone de la consola de comandos. Para acceder a la interfaz web necesita otro PC que esté en la misma red del servidor o bien, si desea acceder desde fuera de nuestra red, hay de configurar el router para que redirija las peticiones del puerto 80 (http) hacia la IP del servidor.

Habiendo realizado lo anterior, inicie el navegador (una buena elección puede ser Mozilla Firefox), escriba la dirección IP asignada al servidor Elastix y aparecerán las pantallas que puede ver en la figura V-2.

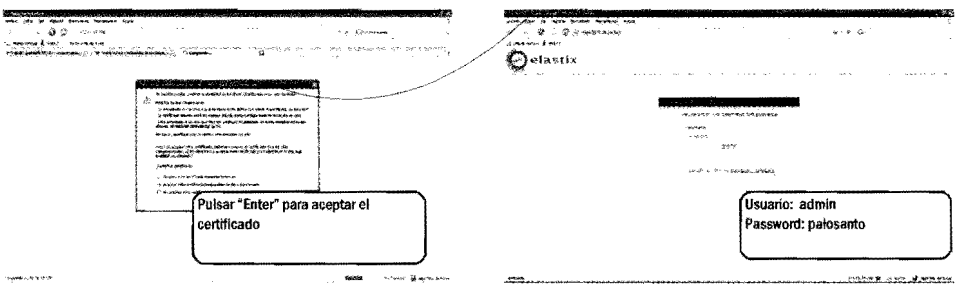


Figura V-2. Elastix - Inicio

Ya tiene acceso a la interfaz web, por lo que puede configurar el sistema.

2.1.2 Configuración

A continuación se va a describir cada una de las funcionalidades que incorpora Elastix, deteniéndonos a explicar en detalle qué se puede conseguir con las mismas.

Sistema - Información de sistema

Se presenta información sobre el estado de los recursos del sistema. Permite monitorizar la carga de la CPU, el uso de memoria, el tiempo que lleva el sistema encendido y el estado de uso de los discos duros. Además, también muestra información gráfica que relaciona el número de llamadas simultáneas con el uso de CPU y memoria.

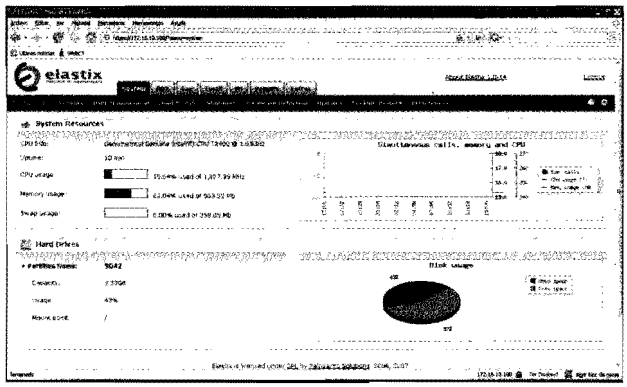


Figura V-3. Elastix - Información del sistema

Sistema - Red

Muestra información sobre las interfaces de red y los parámetros configurados durante la instalación. Por defecto, el sistema se configura para obtener una IP automáticamente (DHCP), pero se puede modificar a conveniencia pulsando en *Editar Parámetros de Red*. Si no se dispone de servidor DHCP en la red, Elastix incorpora uno propio que puede configurar pulsando en *Servidor DHCP*.

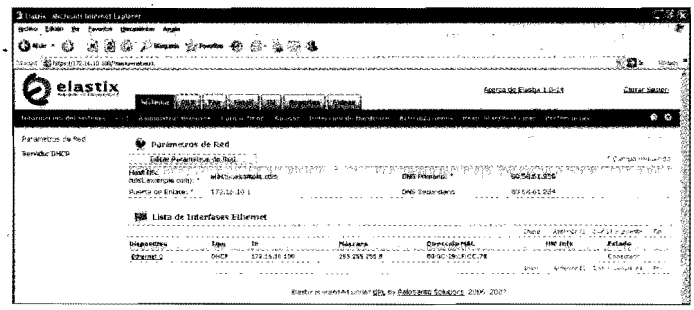


Figura V-4. Elastix - Configuración de la red

Sistema - Administrar Usuarios

En esta pestaña se configuran los usuarios del sistema. Se pueden definir grupos de usuarios que a su vez tendrán acceso a ciertas partes del sistema, previamente definidas en *Permisos de Grupo*. Desde esta pestaña puede controlar qué puede y qué no puede hacer cada usuario en particular (ya sea una extensión telefónica o un usuario externo).

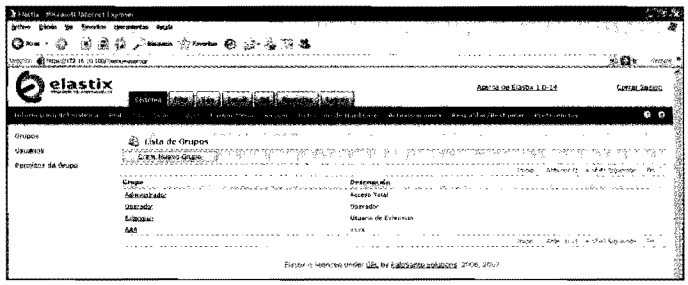


Figura V-5. Elastix - Administración de usuarios

Sistema - Cargar Módulos

Elastix incorpora la posibilidad de añadir módulos personalizados que haya a disposición de la comunidad. También puede desarrollar un módulo personalizado (que cumpla con los estándares definidos por ellos) para añadir más funcionalidades al sistema.

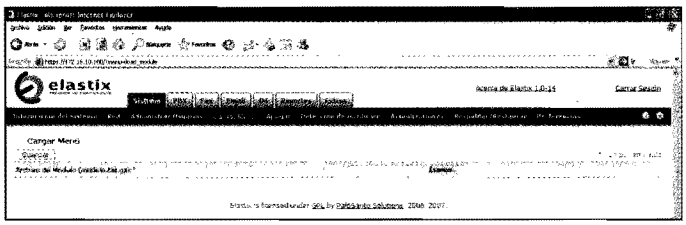


Figura V-6. Elastix - Módulos del sistema

Sistema - Gestión de apagado

Desde esta pestaña se puede apagar el sistema o reiniciarlo.

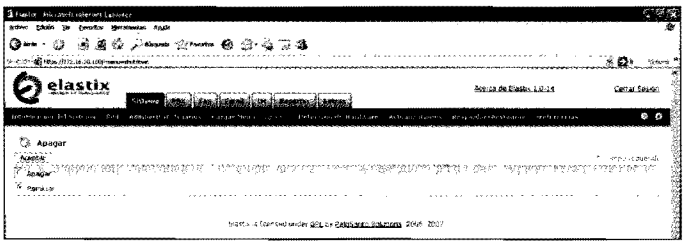


Figura V-7. Elastix - Gestión de apagado del sistema

Sistema - Detección automática de hardware

Una funcionalidad muy importante, sobre todo para los no iniciados, es la posibilidad de configurar hardware zaptel de manera automática. Uno de los principales quebraderos de cabeza para los usuarios que comienzan con Asterisk está en configurar, de manera adecuada, las interfaces de acceso a la red pública (tipo Digium TDM400, TE121, etc.). Elastix puede ayudarnos en esta tarea, de manera que puede configurar el sistema en cuestión de segundos (se recomienda analizar el resultado de esta configuración, editando los ficheros */etc/zaptel.conf* y */etc/asterisk/zapata.conf*).

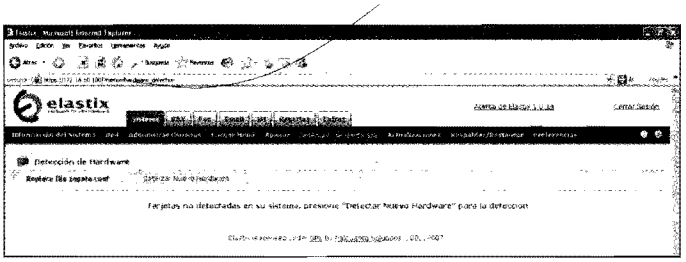


Figura V-8. Elastix - Detección automática de Hardware

Sistema - Actualizaciones

Al igual que otros sistemas de escritorio, Elastix incorpora la posibilidad de actualizar los diferentes módulos que componen el sistema base Linux, así como los propios módulos del sistema Elastix. Algo muy útil para solventar fallos o mejorar las funcionalidades del sistema. Además, también podemos instalar aquellos paquetes que nos puedan interesar (todo ello de manera visual), o incluso puede añadir o quitar repositorios según sus intereses.

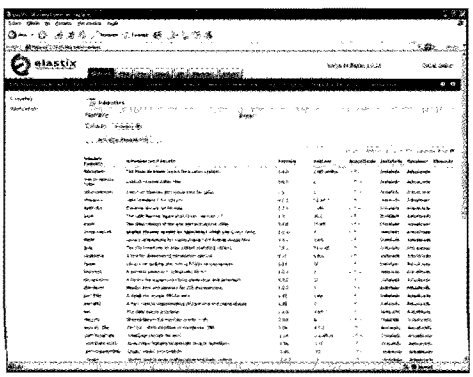


Figura V-9. Elastix - Actualizaciones

Sistema - Copias de seguridad

Se incorpora la posibilidad de realizar copias de seguridad del sistema. Es posible realizar copias personalizadas que incluyen: base de datos, sonidos, archivos de configuración de Asterisk, fax, buzón de voz, cuentas de correo y monitorizaciones. Las copias se almacenan en el disco local, por lo que deben moverse manualmente si desea guardarlas en un lugar más seguro.

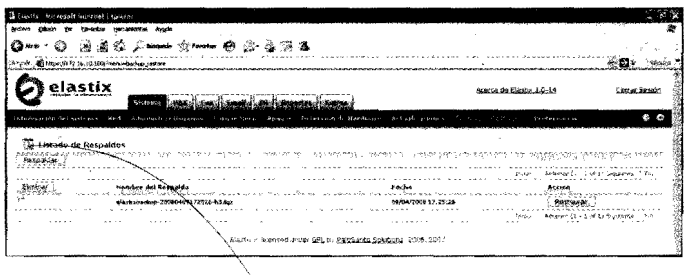


Figura V-10. Elastix - Gestión de copias de seguridad

Sistema - Preferencias

Es posible configurar el idioma de la interfaz, la fecha y la hora, así como el aspecto visual.

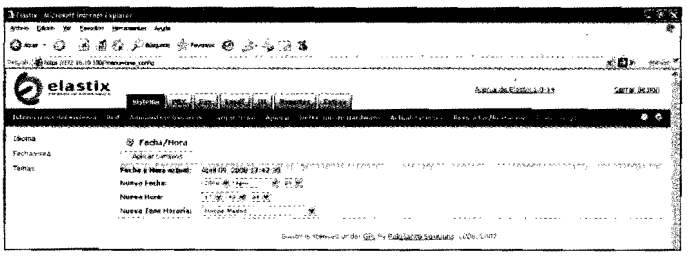


Figura V-11. Elastix - Preferencias

PBX – Configuración de PBX

En la pestaña correspondiente a PBX, se puede encontrar la interfaz de gestión de Asterisk, FreePBX, pero embebida en la hoja de estilo del propio Elastix. Sólo aparecen las funcionalidades típicas o de más uso, aunque si se pulsa en *unembedded FreePBX*, es posible abrir una nueva ventana con la versión completa de dicha interfaz, donde se tienen todas las funcionalidades disponibles.

Puesto que en el *Capítulo 5. Gestión de Asterisk mediante interface Web,* se hace una revisión a fondo de FreePBX, se remite al lector a su análisis para un mejor entendimiento.

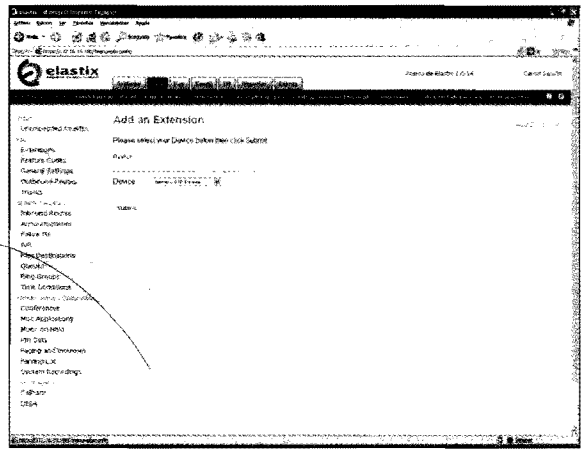


Figura V-12. Elastix - Configuración de PBX

PBX – Flash Operator Panel (FOP)

Elastisk incorpora el módulo Flash Operation Panel (FOP) para poder ver en tiempo real la actividad del sistema.

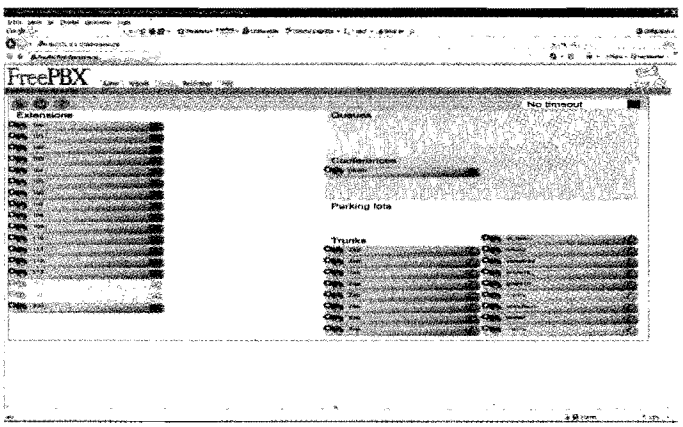


Figura V-13. Elastix - Flash Operator Panel

PBX – Buzón de voz

Elastix dispone de una interfaz para el manejo de los buzones de voz de los usuarios. Cada vez que entra un mensaje en el buzón de algún usuario, se puede consultar inmediatamente en esta ventana. Para poder acceder a esta funcionalidad, se debe haber dado de alta al usuario en cuestión en la interfaz de administración.

Es posible escuchar el mensaje directamente o descargarlo al PC.

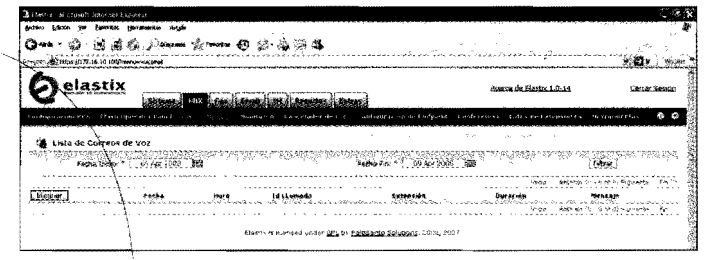


Figura V-14. Elastix - Buzón de voz

PBX – Monitorización

En la ventana de monitorización se pueden listar todas las grabaciones que se han realizado a las extensiones a las cuales se les tiene habilitado tal funcionalidad. Aparece detallada toda la información de cuándo se realizó la llamada, a quién se hizo, la duración, etc. y además puede escuchar "online" o bajársela al PC.

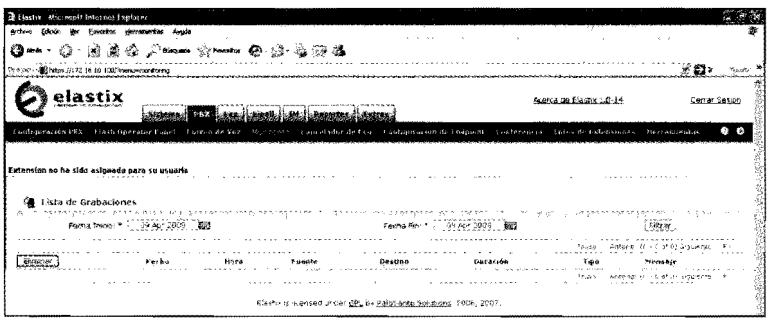


Figura V-15. Elastix - Monitorización

PBX – Cancelador de eco

Elastix incorpora de serie una cancelador de eco para su uso en los canales tipo *Zap*. En esta ventana se habilita o deshabilita su uso.

El cancelador usado es OSLEC[8], el cual ha sido desarrollado por David Rowe, y ha sido liberado bajo licencia GPL. OSLEC se acerca (sin llegar a cumplirlo del todo) al estándar G168, aunque proporciona unos resultados excelentes en aquellos entornos en los que hay presencia de eco.

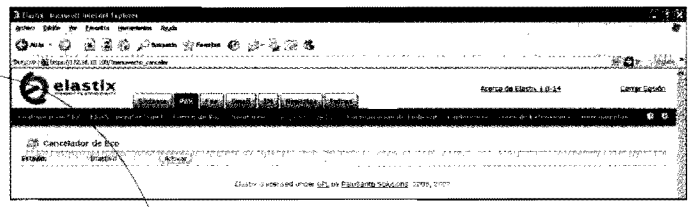


Figura V-16. Elastix - Cancelador de eco

PBX – Configuración de Endpoints

A través de esta pestaña puede autoprovisionar dispositivos presentes en nuestra red. Elastix soporta actualmente algunos modelos de Linksys, Grandstream, etc.

Si el dispositivo tiene habilitado la funcionalidad de autoprovisionamiento, Elastix lo detectará y podrá asignarle un perfil de extensión, previamente creado en FreePBX.

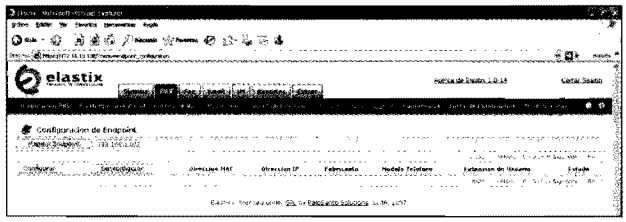


Figura V-17. Elastix - Configuración de Endpoints

[8] http://www.rowetel.com/ucasterisk/oslec.html

PBX – Conferencias

Elastix incorpora un gestor propio de conferencias que ayuda al administrador en el manejo de las mismas. Una vez que se ha creado la conferencia en FreePBX, se pulsa en *Nueva conferencia* y se introducen los datos requeridos, y de esta forma tendrá un gestor que permite realizar diversas acciones sobre los usuarios de dicha conferencia: quitar la voz, expulsar de la conferencia, etc.

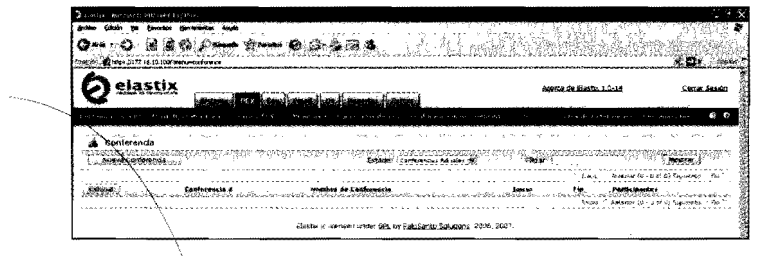


Figura V-18. Elastix - Conferencias

PBX – Importar Extensiones

Se pueden importar lotes de extensiones hacia el sistema. Muy útil cuando hay que migrar de forma masiva a usuarios de un sistema a otro o hay que introducirlos por primera vez. También es posible descargar los datos de los usuarios existentes.

El formato de trabajo es un archivo de texto separado por comas (.csv) con los campos adecuados.

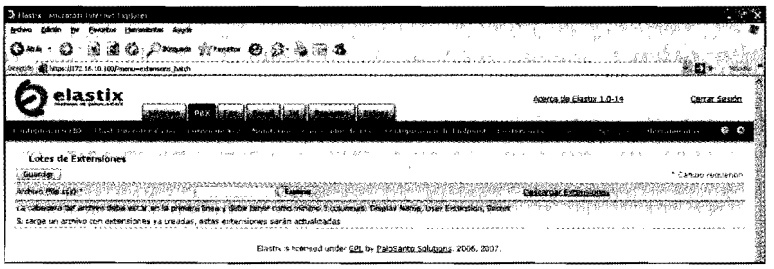


Figura V-19. Elastix- Lotes de extensiones

PBX – Herramientas

A través de esta ventana es posible realizar dos tareas: ejecutar comandos en Asterisk desde el *CLI* (*Command Line Interface*) y editar los archivos de configuración (.conf).

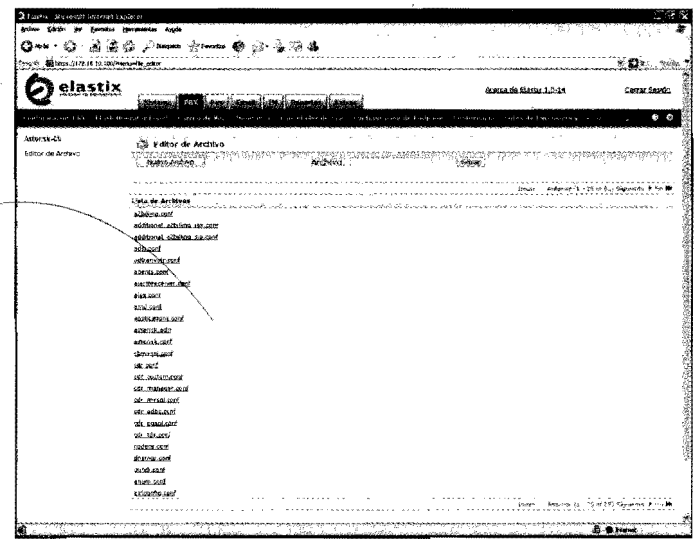


Figura V-20. Elastix - Herramientas del sistema

Fax – Listado de faxes

Elastix incorpora una funcionalidad muy destacada en cualquier centralita telefónica comercial: la posibilidad de manejar faxes. Pero no sólo los maneja, es que también permite actuar como un fax, pudiendo recibir y enviar faxes a cualquier destino. Para poder desplegar esta característica se apoya en dos módulos: IAXmodem e Hylafax. El primero es un software que emula un módem de datos y que se comunica con Asterisk mediante el protocolo IAX, mientras que el segundo es un software para gestión de faxes.

En la pestaña de *Listado de Faxes* puede ver las unidades de fax que hay definidas en el sistema, así como su estado y otros datos relevantes.

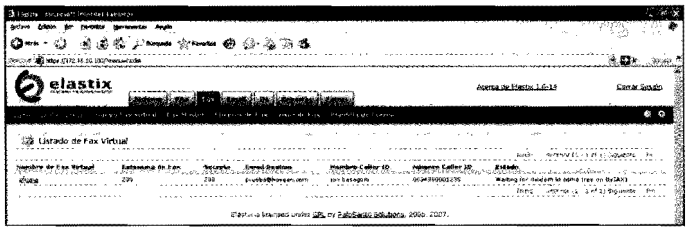


Figura V-21. Elastix - Listado de Faxes

Fax – Añadir fax virtual

En este apartado se introducen los datos necesarios para configurar una unidad de fax. Para gestionar cada línea de fax, habrá que definir una extensión tipo IAX que, a su vez, se comunicará con IAXmodem. Se puede configurar un correo electrónico de destino para recibir notificaciones de envío y recepción.

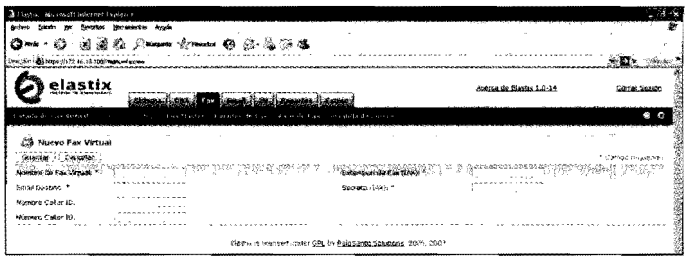


Figura V-22. Elastix - Fax virtual

Fax – Fax master

Pestaña para introducir el correo electrónico del administrador del servidor de fax.

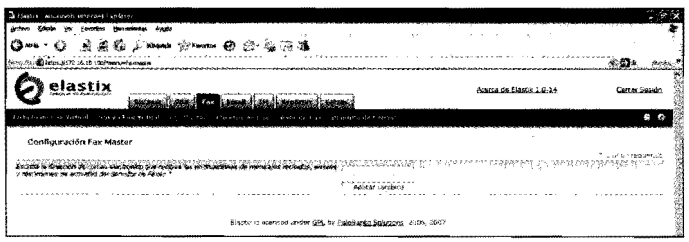


Figura V-23. Elastix - Fax Master

Fax – Clientes de fax

Permite definir qué máquinas pueden enviar faxes usando software al efecto (por ejemplo, *Winprint Hylafax*). Debe introducir la IP de aquellas máquinas que pueden enviar faxes al sistema para ser procesados.

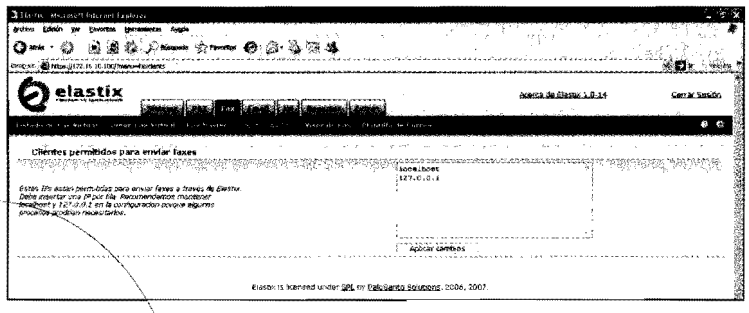


Figura V-24. Elastix - Clientes de Fax

Fax – Visor de faxes

Elastix incorpora un visor donde pueden verse todos los faxes enviados o recibidos en formato PDF. Sólo es necesario hacer clic en el enlace adecuado y se nos abrirá el documento de fax correspondiente.

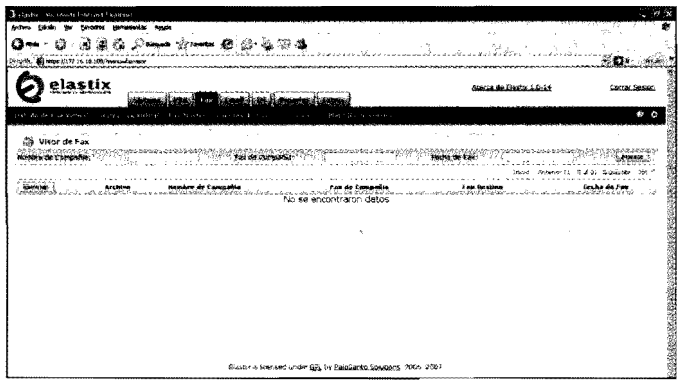


Figura V-25. Elastix - Visor de faxes

Fax – Plantilla de correo electrónico

En esta pestaña se configuran los parámetros relativos al envío por correo electrónico de los faxes recibidos. Normalmente, cuando un fax entre en el sistema, será enviado a la cuenta de correo electrónico correspondiente, usando los datos de envío de la plantilla.

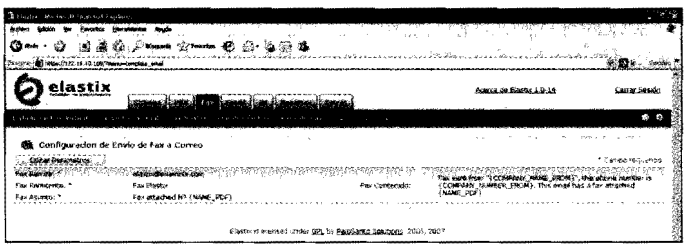


Figura V-26. Elastix - Plantilla de correo electrónico

Correo Electrónico – Dominios

Además de la centralita telefónica y del servidor de faxes, Elastix incorpora un servidor de correo electrónico.

En la pestaña se crean los dominios disponibles y que desee gestionar con Elastix para el correo electrónico.

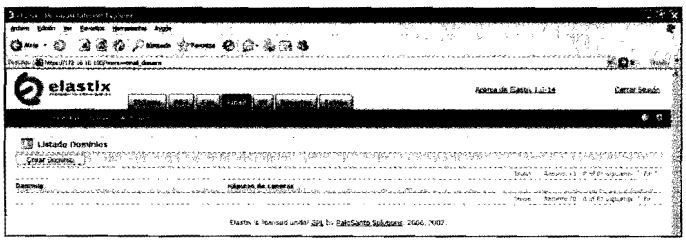


Figura V-27. Elastix - Dominios del correo electrónico

Correo Electrónico – Cuentas

Iremos creando cuentas de correo electrónico asociadas a dominios. Introduzca el nombre de usuario, la contraseña y la cuota de espacio en disco asignada.

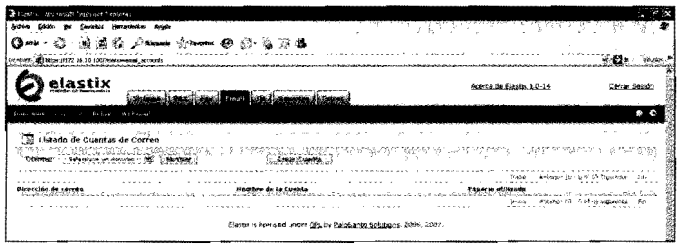


Figura V-28. Elastix – Cuentas de correo electrónico

Correo Electrónico – Relay

Sólo se pueden enviar correos electrónicos desde redes habilitadas. Por defecto, Elastix no permite enviar correos a cualquiera (para evitar ataques con correos basura), por lo que deben darse de alta aquellas redes que estén autorizadas.

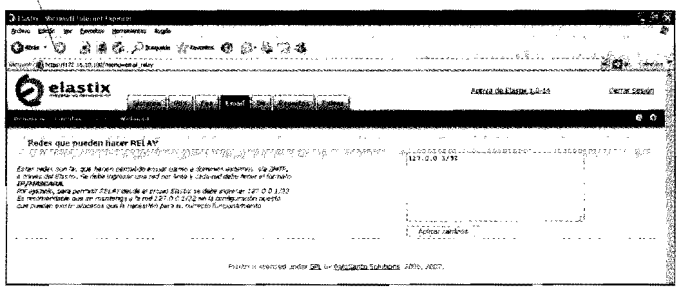


Figura V-29. Elastix - Correo electrónico - Relay

Correo Electrónico – Correo web

Como servicio para el usuario, se dispone de una interfaz para el correo electrónico vía web. Se usa la plataforma *RoundCube*.

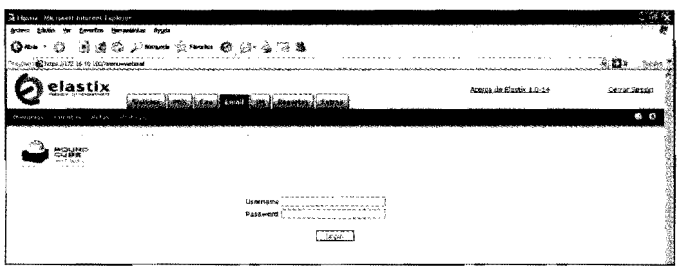


Figura V-30. Elastix - Correo web

Mensajería Instantánea

Otra de las funcionalidades estrella de Elastix es la integración de la mensajería instantánea con Asterisk. Se utiliza para ello el servidor de mensajería Openfire, el cual nos va a permitir tener tanto mensajería instantánea entre clientes como indicación de presencia, usando el protocolo *Jabber/XMPP*.

Mediante el módulo adicional *Asterisk IM* y el uso de un cliente de mensajería (como *Spark*) se puede interactuar entre los usuarios de nuestro Asterisk consiguiendo:

- Mensajería instantánea entre usuarios de la red corporativa e incluso, usuarios de fuera de la red corporativa.
- Mensajería instantánea con usuarios de otras redes de mensajería como *ICQ, MSN, Yahoo, AIM*, etc.
- Indicación de presencia. Mediante el uso de módulos adicionales, se puede publicar el estado de un determinado usuario (está al teléfono, está desconectado...), no sólo entre clientes de mensajería, sino en páginas web.

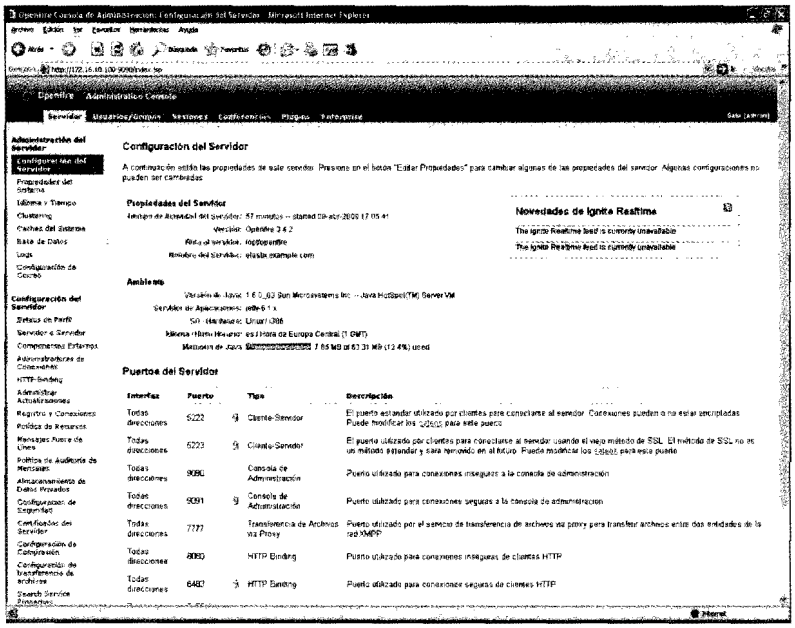


Figura V-31. Elastix - Mensajería instantánea

Reportes – CDR

Se puede mostrar un detalle de llamadas bastante completo, con indicación de diferentes parámetros de la llamada como fecha, número destino, duración de llamada, etc.

La interfaz permite una búsqueda de cualquier llamada efectuada en base a diferentes campos de la misma, proporcionando una gran flexibilidad a la hora de obtener información relevante.

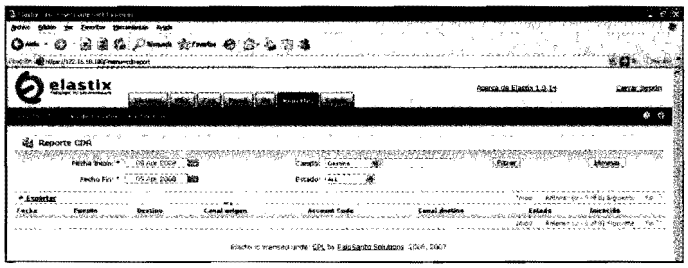


Figura V-32. Elastix - CDR

Reportes – Uso de canales

En el reporte de uso de canales se muestra de manera gráfica el uso de los diferentes canales a lo largo del tiempo. Es útil para observar de un vistazo la carga en llamadas que tiene nuestro sistema.

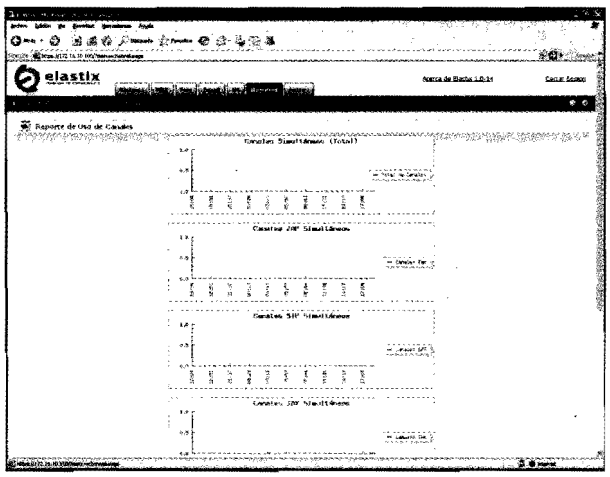


Figura V-33. Elastix - Estadísticas de uso de canales

Reportes – Facturación

Otra funcionalidad interesante que aporta Elastix es la posibilidad de tarificar las llamadas de una manera simple y rápida. Se definen las tarifas por cada destino mediante el nombre, el precio por minuto y el establecimiento de llamada.

Además, se tiene un reporte tanto gráfico como en texto de las llamadas facturadas de nuestro sistema. Por último, puede indicarle al sistema qué canales de salida serán tarificados en el apartado *Configurar facturación*.

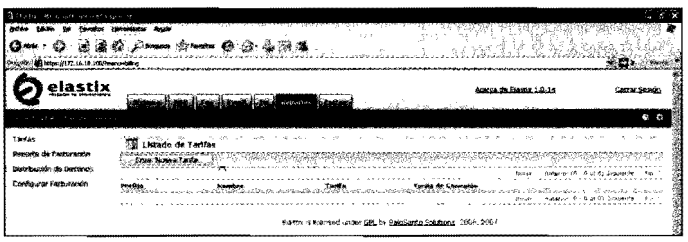


Figura V-34. Elastix - Facturación

Extras – vTigerCRM

En la pestaña Extras, se incluyen diferentes aplicaciones realizadas por terceros y que aportan diferentes funcionalidades.

El caso de *vTigerCRM* es una solución CRM de código abierto y gratuita que se integra con Asterisk. Para más información sobre el uso de esta aplicación se puede consultar http://www.vtiger.com/

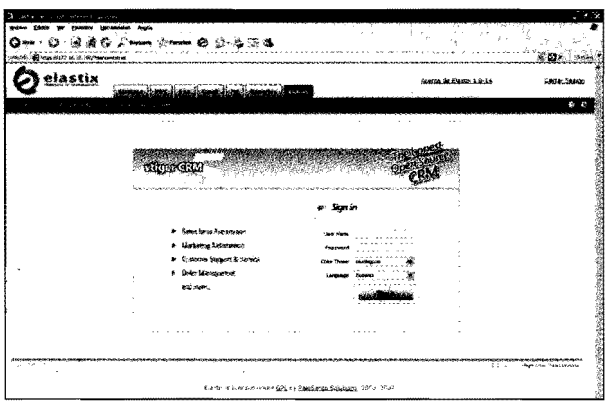


Figura V-35. Elastix - vTigerCRM

Extras – A2Billing

Otra aplicación de terceros muy utilizada e interesante es *A2Billing*. Mediante esta interfaz, tenemos un completo sistema de manejo de tarjetas telefónicas y de usuarios SIP e IAX, a los que podemos tarificar y ofrecer una gran cantidad de servicios personalizados.

Esta aplicación será objeto de un detalle más pormenorizado en el *Anexo VI – Software de terceros para Asterisk*.

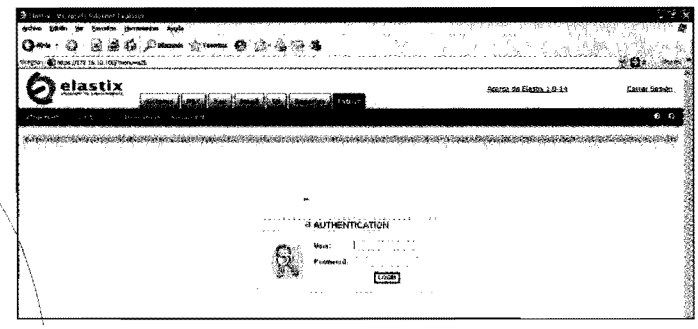


Figura V-36. Elastix - A2Billing

Extras – Descargas

Elastix tiene información y enlaces sobre utilidades varias que serán necesarias en el uso de la centralita. Entre estas utilidades están clientes software de telefonía, clientes de fax y clientes de mensajería instantánea.

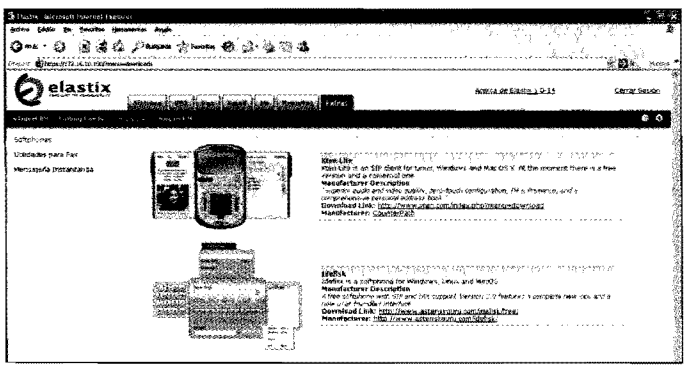


Figura V-37. Elastix - Descargas de softphones

Extras – SugarCRM

SugarCRM es otro módulo de terceros parecido a *vTigerCRM*, es decir, se usa como aplicación CRM y se integra con Asterisk para ofrecer a los agentes información de los clientes que llaman al sistema mediante el uso del identificador de llamada.

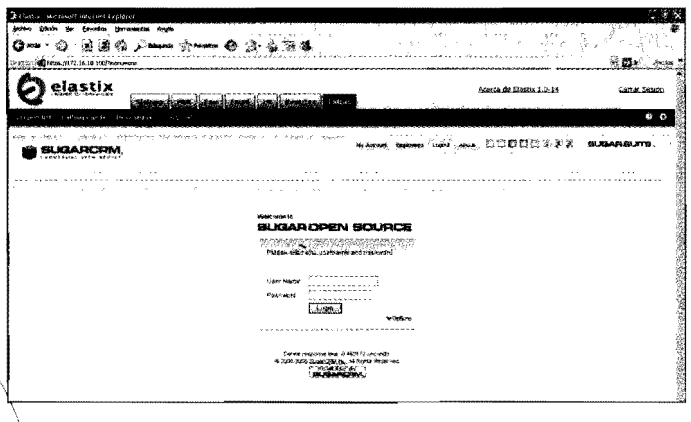


Figura V-38. Elastix - SugarCRM

2.1.3 Conclusión e impresiones

Como se ha podido apreciar, Elastix es una buena solución para aquellos usuarios que desean realizar un despliegue rápido y seguro de una centralita telefónica. No sólo eso, Elastix permite tener mucho más que un sistema telefónico, permite tener un sistema de mensajería, un sistema completo de fax, un sistema de tarificación para telefonía, un sistema de gestión CRM, y todo lo anterior sin que el usuario necesite emplear excesivo tiempo en descargas, configuraciones e instalaciones.

Esta solución también permite que usuarios inexpertos en el mundo de las comunicaciones, Linux y redes informáticas, puedan hacer sus primeros pinitos, sin acabar desmoralizados o irritados con las complejas configuraciones en un entorno "hostil" para el no iniciado.

La potencia del sistema instalado, su interfaz sencillo e intuitivo, y el apoyo de una gran comunidad de usuarios y empresas del sector, hacen de Elastix un gran candidato para convertirse en el sistema telefónico ideal para las pequeñas y medianas empresas que buscan la sencillez y la facilidad de uso.

2.2 PBX IN A FLASH

2.2.1 Instalación y configuración

Si lo que se persigue es una instalación más sencilla y limpia, sin depender excesivamente de personalizaciones externas, entonces el candidato perfecto es *PBX in a Flash* (PIAF).

Esta distribución está basada fundamentalmente en la web de gestión *FreePBX*. A diferencia de Elastix, PIAF intenta introducir los mínimos cambios posibles en el sistema original Asterisk. Para ello hace uso de *scripts*[9] que permiten descargar el código fuente de Asterisk desde su ubicación en Internet, para su posterior instalación sin que el usuario intervenga en absoluto en dicho proceso.

El resultado final es un sistema dedicado Asterisk instalado en la distribución Centos 5 y que permite ser configurado a través de FreePBX. Adicionalmente, los creadores de PIAF han puesto a disposición de la comunidad de usuarios diferentes scripts mediante los cuales es posible habilitar nuevas funcionalidades a conveniencia. Todo lo anterior, siguiendo una metodología automatizada, pensada para el usuario con conocimientos limitados o básicos. Entre los scripts más importantes están:

- Instalación de Gtalk y speex
- Instalación de codecs (en versión educacional) g.729 y g.723
- Copias de seguridad de disco completas
- Aplicación para conocer el tiempo en determinados lugares o aeropuertos
- Recordatorios telefónicos

El conjunto final acaba siendo un sistema Asterisk al que el usuario puede ir añadiendo a conveniencia aquellas funcionalidades que le parecen de interés, aun sin tener unos conocimientos avanzados de Linux o VozIP.

Una característica importante de PIAF es que no parchea, ni modifica, ni retoca el código fuente original de Asterisk, Zaptel o Libpri, por lo que, de alguna forma, no se enmascaran la potencia y funcionalidad original de Asterisk.

---

[9] http://www.alegsa.com.ar/Dic/script.php

A modo de ejemplo, se adjuntan algunas capturas de la interfaz web que gestiona el sistema.

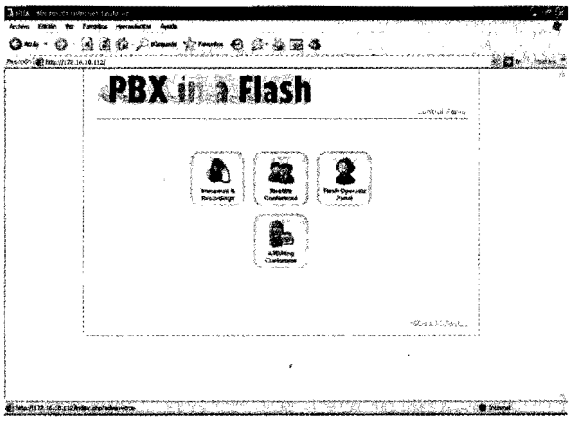


Figura V-39. Interfaz de usuario PIAF

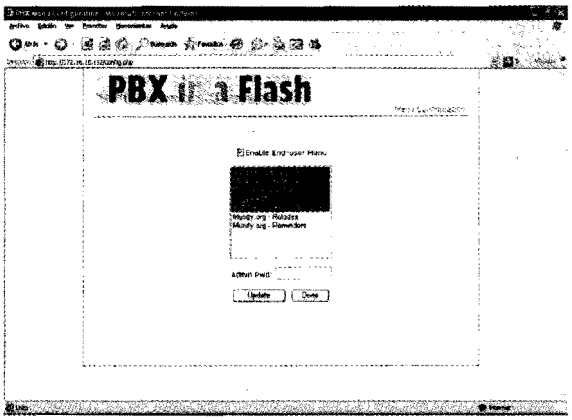


Figura V-40. Menu de configuración para el usuario final

Como se puede apreciar en las figuras V-39 y V-41, el sistema tiene habilitados unos accesos directos a funcionalidades típicas de la central como pueden ser buzones de voz, grabaciones, panel de operador e incluso, interfaz de usuario para software de tarificación (A2Billing). En la interfaz de administrador se puede acceder a otros sitios como Webmin[10] (véase la figura V-42) o FreePBX.

---

[10] http://es.wikipedia.org/wiki/Webmin

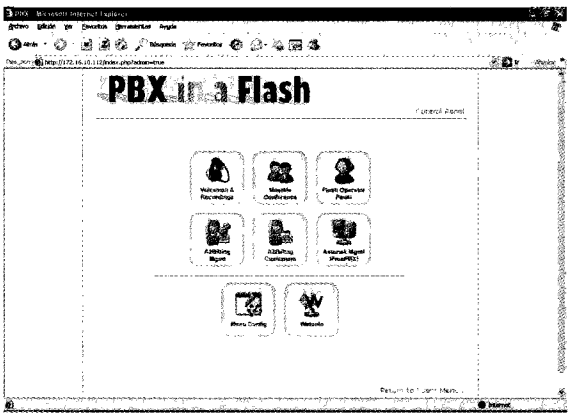


Figura V-41. Interfaz de administración de P1AF

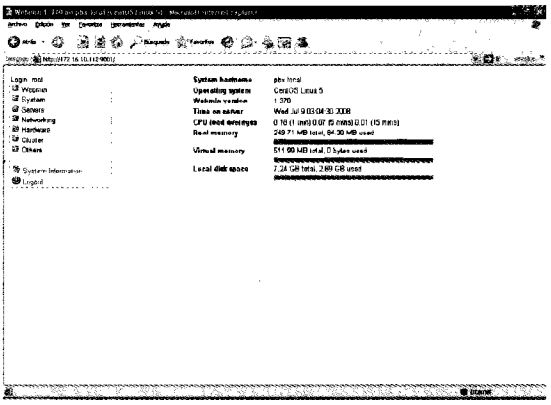


Figura V-42. Interfaz Webmin

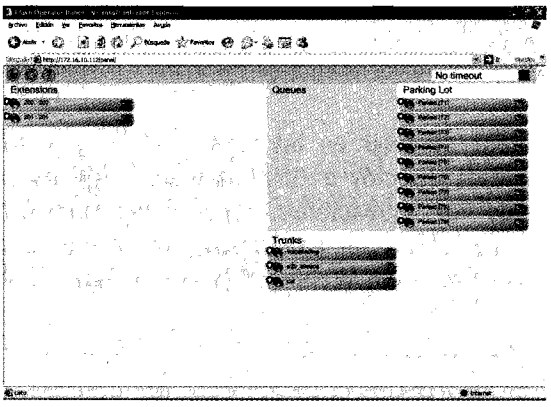


Figura V-43. Flash Operator Panel

2.2.2 Conclusión e impresiones

Al igual que Elastix, PIAF permite tener operativo un sistema completo telefónico en pocos minutos, y además, sin conocimientos avanzados de Linux o telefonía. Su posterior configuración recae en la correcta utilización del entorno FreePBX para poder poner en marcha todo lo necesario, como son extensiones, troncales de salida, proveedores, etc.

PIAF no introduce mucha variedad en cuanto a aplicaciones de terceros, a diferencia de Elastix, limitándose exclusivamente a Asterisk y algunas pequeñas utilidades adicionales. Esto redunda en un sistema más limpio y mejor controlado por el administrador.

Desde este punto de vista, PIAF parece más recomendable para aquellos usuarios iniciados que sólo desean disponer del sistema Asterisk, pero a través de un manejo más o menos automatizado, y añadiendo la posibilidad de actualización mediante el uso de scripts confeccionados al efecto por terceros.

2.3 ASTERISKNOW

Otra distribución que está cosechando gran popularidad debido a su sencillez de instalación y configuración es la desarrollada por Digium bajo el nombre de *AsteriskNow*. A pesar de haber sido el primer desarrollo de este tipo, Digium adquirió en 2007 la plataforma SwitchBox, por lo que pasó a centrarse comercialmente en la misma. El proyecto original no ha sido abandonado y actualmente, siguen ofreciéndose versiones y mejoras, aunque todo apunta a que AsteriskNow no alcanzará el nivel profesional de otras soluciones de *Código Abierto* o Comerciales.

Digium la califica como "Asterisk en pocos minutos". En realidad es una distribución de Linux (Rpath) que incluye de manera nativa Asterisk 1.4 más el interfaz web de gestión *AsteriskGUI*. Además posee todo el software necesario para el pleno funcionamiento de Asterisk.

Su interfaz web usa la tecnología AJAM (*Aynchronous Javascript Asterisk Manager*), la cual permite a los navegadores web u otros aplicaciones que usen *http* acceder directamente al *Asterisk Manager Interface (AMI)*.[11] De esta forma se

[11] http://www.voip-info.org/wiki-Asterisk+manager+API

pueden leer eventos y ejecutar comandos en el sistema de manera apropiada y vía web.

A continuación se muestran algunas capturas que dan idea del proceso de configuración de una central telefónica usando el interfaz *AsteriskGUI*.

***Nota:** el sistema no se encuentra actualmente disponible en español.*

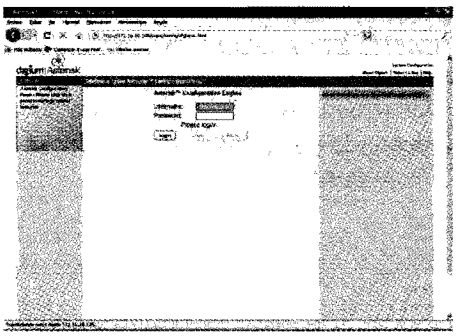


Figura V-44. Autentificación del sistema

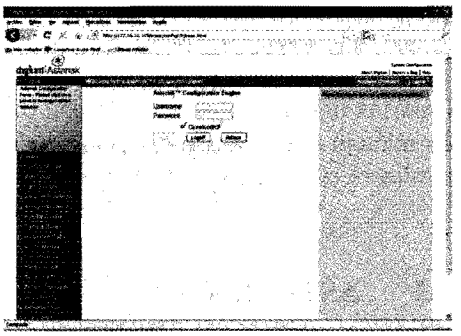


Figura V-45. Pantalla de inicio

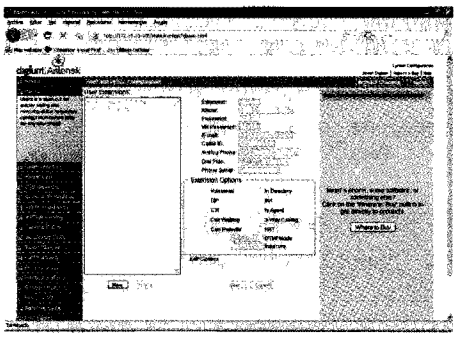


Figura V-46. Creación de usuarios

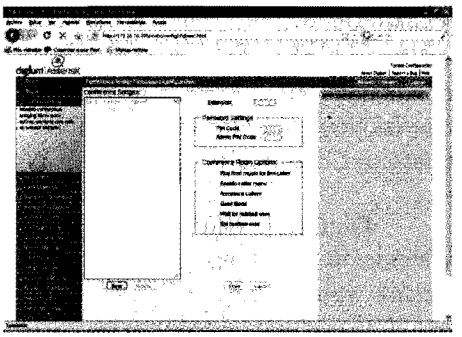


Figura V-47. Manejo de salas de conferencias

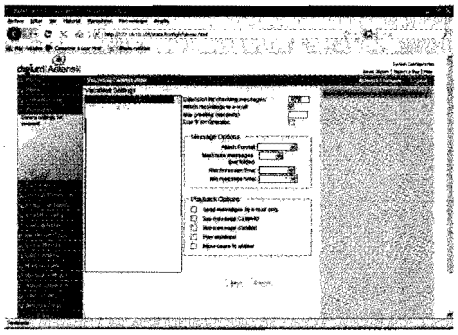


Figura V-48. Buzón de voz

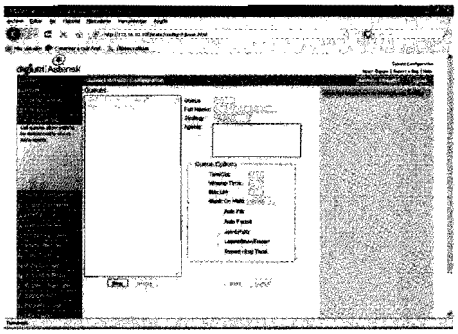


Figura V-49. Creación y configuración de colas

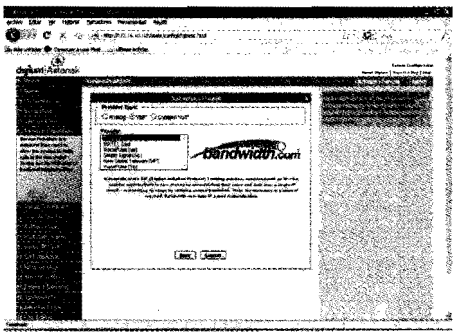


Figura V-50. Creación de proveedores telefónicos

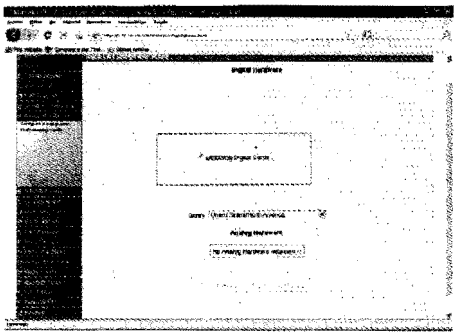


Figura V-51. Detección de hardware telefónico

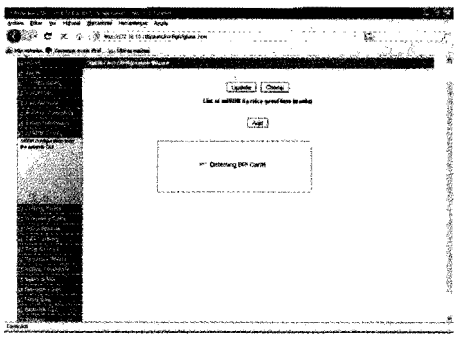


Figura V-52. Detección de sistemas RDSI

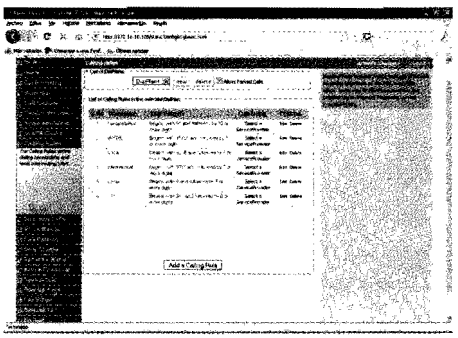


Figura V-53. Creación de rutas de salida

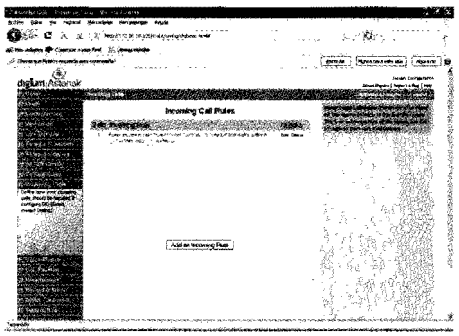


Figura V-54. Creación de rutas de entrada

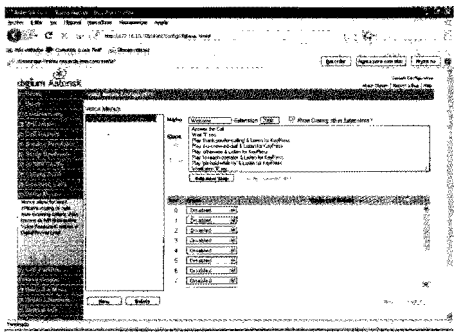


Figura V-55. Configuración de menús vocales IVR

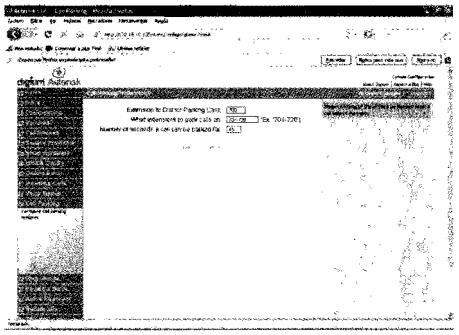


Figura V-56. Parking de llamadas

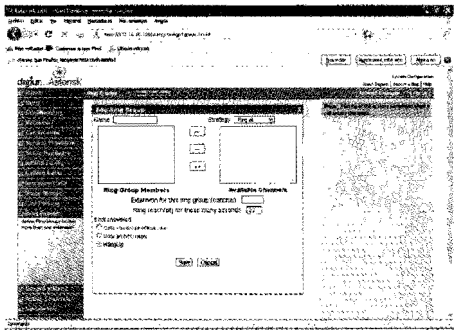


Figura V-57. Creación de grupos de salto

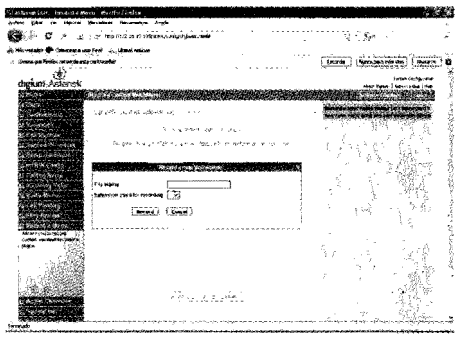


Figura V-58. Grabación de menús vocales

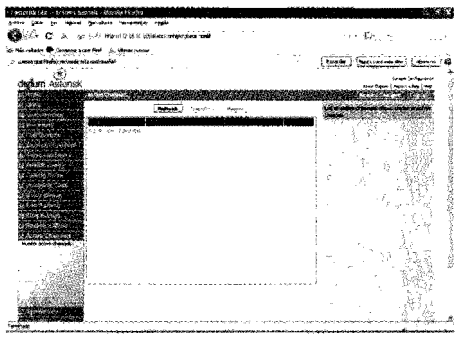


Figura V-59. Pantalla de llamadas activas

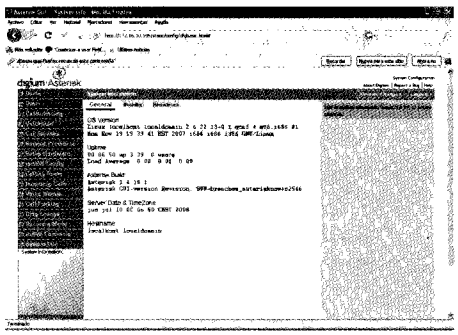


Figura V-60. Información del sistema

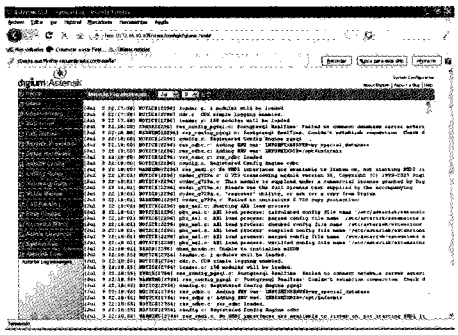


Figura V-61. Log del sistema

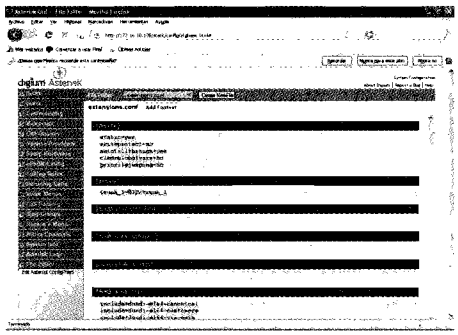


Figura V-62. Editor de ficheros de configuración

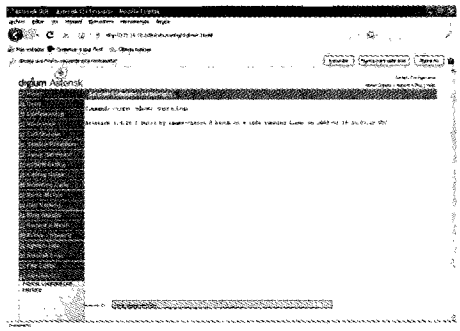


Figura V-63. Consola de Asterisk

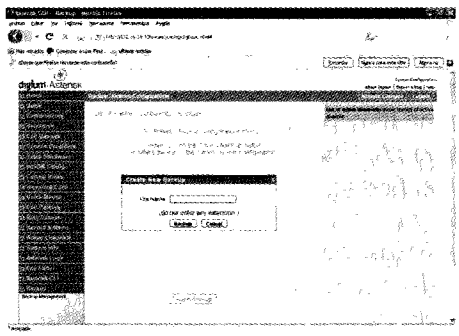


Figura V-64. Realización de copias de seguridad

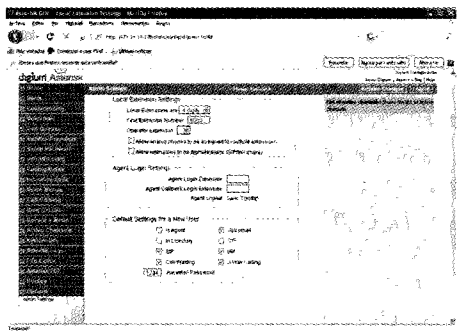


Figura V-65. Opciones varias

3 Sistemas integrados o de capacidad limitada

Los sistemas de capacidad limitada (SCL), también llamados sistemas integrados (a veces traducido del inglés como *empotrado* o *embebido*), se usan fundamentalmente en tareas muy concretas y que requieren la dedicación de pocos recursos en comparación con sistemas tipo servidor o PC de sobremesa.

Tradicionalmente han sido sistemas con unas características limitadas en cuanto a capacidad de procesamiento, almacenamiento, etc. forzados por sus reducidas dimensiones y su escasa potencia eléctrica.

Tal y como se explica en *wikibooks.org*,[12] los sistemas SCL pueden ser descritos más bien por lo que no son, que por lo que sí son. Desde este punto de vista, un PC sirve para realizar múltiples tareas como leer correos, navegar por Internet, escuchar música, etc., sin embargo, un SCL realizará, usualmente, una única tarea o un número reducido de ellas preprogramadas de fábrica. Como ejemplo de SCL's podemos citar reproductores de música MP3, sistemas de navegación GPS, alarmas, enrutadores, etc.

Muchos sistemas SCL suelen trabajar, además, en entornos donde la computación en tiempo real es crítica. Cada resultado u operación es efectiva si se produce en el lapso de tiempo correcto. Un piloto automático de un avión es un claro ejemplo de sistemas integrados en tiempo real. Si el piloto automático detecta una variación de la ruta, entonces debe seguir los pasos necesarios en cuestión de milisegundos para corregir dicha desviación, o de lo contrario, el desastre está asegurado.

Cada día que pasa, los fabricantes están redefiniendo el concepto que tenemos sobre sistemas integrados, consiguiendo reducir el tamaño de los dispositivos hardware más y más a la vez que aumentan sus prestaciones. De esta forma vemos que en la actualidad están surgiendo infinidad de ordenadores que tienen un tamaño muy reducido pero a la vez son muy potentes y versátiles. Este es el caso de los nuevos equipos que emplean placas base con formato Mini-ITX, Nano-ITX o Pico-ITX del fabricante Via Technologies. En la figura V-66 se muestra la placa base Pico-ITX.

---

[12] http://en.wikibooks.org/wiki/Embedded_Systems/Embedded_Systems_Introduction

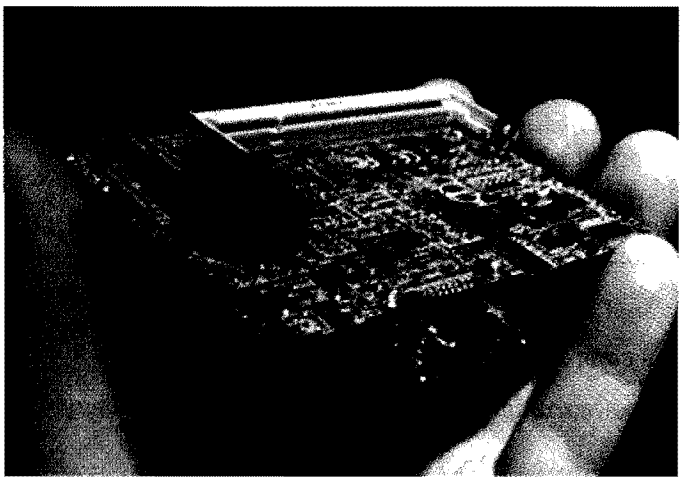


Figura V-66. Placa base Via Pico-ITX

Estos ordenadores están llevando la arquitectura x86 a extremos inimaginables hace tan sólo unos años, reduciendo al máximo el consumo de energía así como el calor producido por estos dispositivos. Esto los hace ideales para diseñar soluciones reducidas que realicen funciones específicas, como pueden ser las propias de un servidor web, un reproductor multimedia o un enrutador.

Juntando la potencia de esta nueva generación de miniordenadores con cualquiera de la amplia gama de sistemas operativos GNU/Linux podremos obtener un rendimiento excelente para la ejecución de todo este tipo de tareas. Esto se debe a que los sistemas GNU/Linux pueden aprovechar al máximo el hardware sobre el que son empleados, algo que constituye una de sus principales ventajas. A diferencia de otros sistemas operativos propietarios, en GNU/Linux podemos configurar nuestro sistema para que se adapte al máximo a nuestras necesidades, obteniendo así el mejor rendimiento sobre cualquier tipo de hardware.

Desde este punto de vista, y gracias a la perfecta adaptación de GNU/Linux a los SCL, es posible hoy en día realizar la instalación de Asterisk en una gran variedad de dispositivos integrados. El resultado es una mini-central telefónica que puede gestionar una cantidad aceptable de llamadas simultáneas (del orden de una decena o menos, habitualmente) de manera exitosa, así como diversos servicios asociados como buzón de voz, colas, salas de conferencias, etc. Para entornos de pequeña oficina, residencial, etc. ¡más que suficiente!

Puesto que existen numerosas distribuciones GNU/Linux con Asterisk, y que se adaptan a casi cualquier máquina, se van a detallar las distribuciones más extendidas para este tipo de dispositivos.

3.1 ASKOZIA PBX

El proyecto AskoziaPBX nació en junio de 2007, concretamente en el *Institut für Kommunikations systeme und Technologies* (IKT), de Wolfenbüttel (Alemania). Su principal mantenedor es Michael Iedema.

Askozia es una distribución basada en FreeBSD y está pensada para ser empleada en sistemas pequeños y de poca potencia. Sus principales características son:

- Sistema completo en menos de 15 MB.
- Diseñado específicamente para sistemas integrados de bajo consumo eléctrico.
- Basado en FreeBSD 6.2, incluyendo la rama 1.4 de Asterisk.
- Configuración de sistema archivada en un único fichero XML.

Debido a estas características, Askozia puede ser instalado prácticamente sobre cualquier tipo de soporte (USB, MMC, SD, HDD...). Un detalle importante es la posibilidad de guardar toda la configuración del sistema en un solo archivo XML, lo cual permite una gran flexibilidad a la hora de poder trabajar con diferentes configuraciones, realizar copias de seguridad o incluso a la hora de recuperar sistemas en producción que han tenido algún tipo de problema: se cambia la máquina, se graba la imagen y se carga la configuración... todo muy sencillo.

Askozia puede ser empleado en diferentes plataformas hardware, entre ellas:

- Sistemas de PC basados en tecnología x86 (min. 64MB RAM, 200MHz CPU)
- PC Engines: Wrap, Alix1x, Alix2x y Alix3x
- Soekris: Net48xx, Net55xx
- Herologic: Hl4xx
- VMware

Además de lo anterior, Askozia presenta unas funcionalidades bastante interesantes:

- Configuración vía web muy sencilla para teléfonos y proveedores, ya sean SIP, IAX, RDSI o Analógicos.

- Gestión de dispositivos inalámbricos (Wi-Fi).
- Uso de conferencias, parking de llamadas, transferencias, grupos de salto, etc.
- Buzón de Voz (incluido el reenvío como fichero adjunto vía correo electrónico).
- Inclusión de audios pregrabados en numerosos idiomas, entre ellos el español (versión de voces del español Alberto Sagredo[13]).
- Notificación de correo electrónico en español y varios idiomas más.

En las figuras que siguen se pueden ver varias capturas del sistema y su configuración.

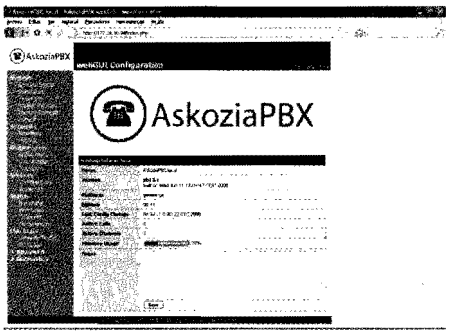


Figura V-67. Página de inicio

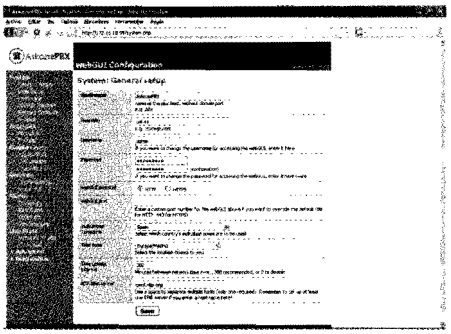


Figura V-68. Configuración general

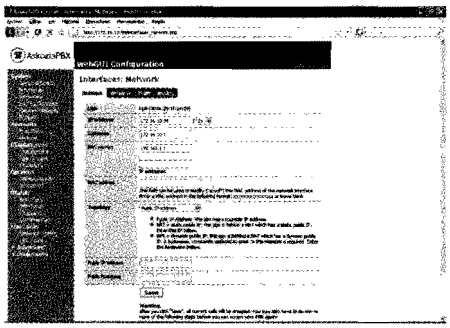


Figura V-69. Configuración de red

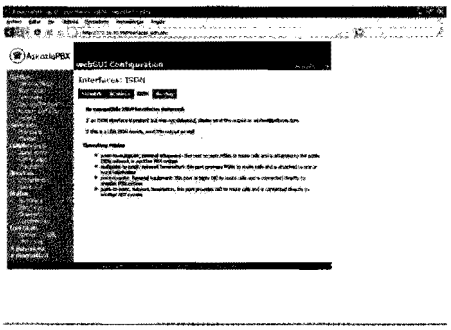


Figura V-70. Configuración de RDSI

[13] http://www.voipnovatos.es/voces

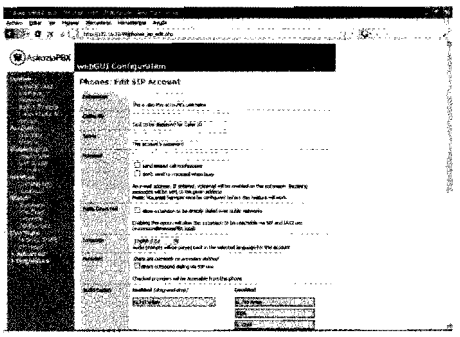


Figura V-71. Configuración de teléfono SIP

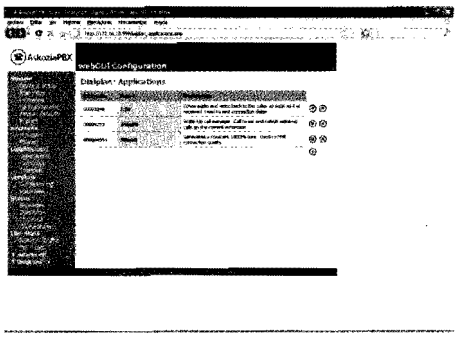


Figura V-72. Aplicaciones para Dialplan

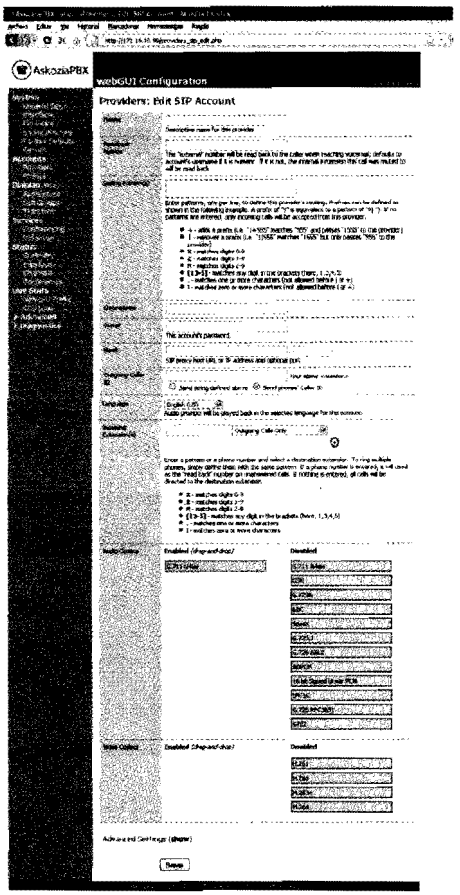


Figura V-73. Configuración de proveedor SIP

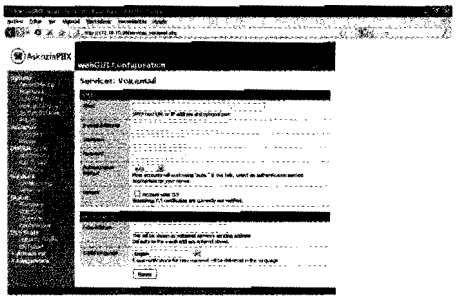


Figura V-74. Configuración del buzón de voz

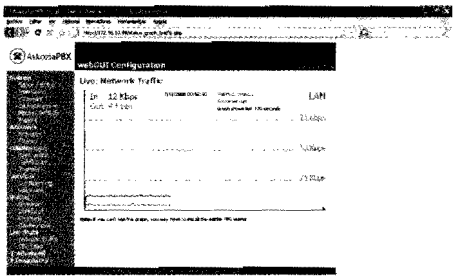


Figura V-75. Gráficos de ancho de banda

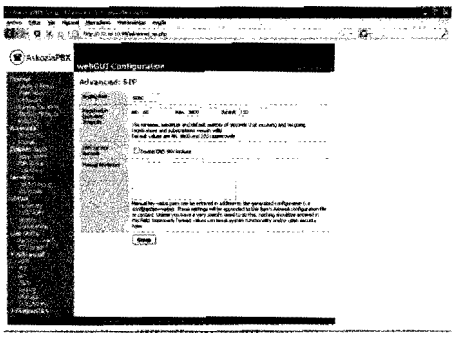


Figura V-76. Opciones avanzadas

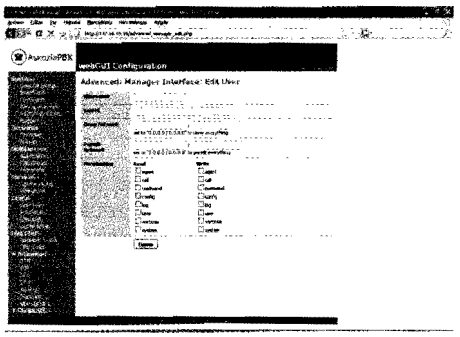


Figura V-77. Interfaz del AMI

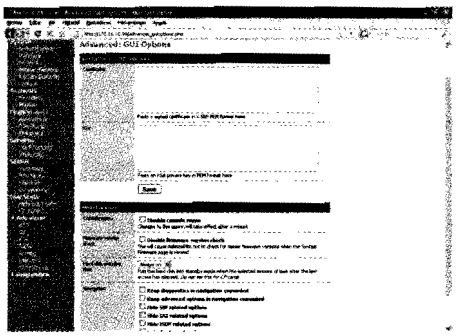


Figura V-78. Opciones avanzadas del interfaz web

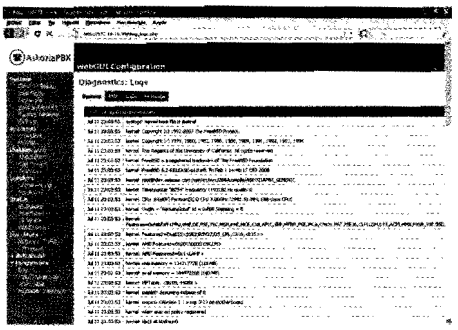


Figura V-79. Logs del sistema

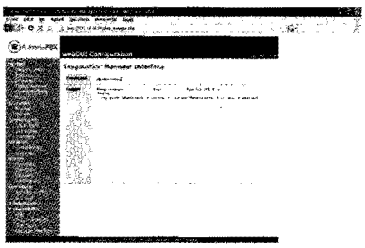


Figura V-80. Consola de comandos

Figura V-81. Terminal de AskoziaPBX

3.2 ASTLINUX

Otro proyecto pensado para ser usado en sistemas integrados de potencia limitada es AstLinux. Creado originalmente por Kristian Kielhofner en 2004 y la colaboración de otros autores, está basado en *Gentoo* y ha sido durante mucho tiempo (y hasta la llegada de AskoziaPBX) la mejor distribución con Asterisk preinstalado en el mercado de los dispositivos SCL.

Su filosofía de trabajo es un tanto peculiar y difiere de la que comúnmente suele hacerse en sistemas más grandes, tipo servidor. Precisamente, esas diferencias fueron las que impulsaron a Kristian Kielhofner a diseñar un sistema que no usase discos duros tradicionales (IDE, SATA, etc.) sino memorias flash (Compact Flash, USB, etc.) con la consiguiente mejora en fiabilidad al no usar partes móviles. Para evitar uno de los principales limitantes de las memorias flash, como es la cantidad de ciclos de lectura y escritura, introdujo el concepto de *keydisk*. Inicialmente, todo el sistema se monta en una memoria flash, en la cual no se puede escribir nada. De esta forma, la vida útil de la tarjeta se alarga enormemente y se evitan ciertos fallos. Como quiera que el sistema debe tener la posibilidad de guardar ciertos datos importantes (buzones de voz, logs, copias de seguridad, etc.), se hace uso de una segunda memoria flash (*keydisk*).

Así, uno puede tener varias *keydisk* con diferentes configuraciones listas para ser cargadas en el sistema, mientras que el sistema base está alojado en otra memoria (incluso en un CD o DVD) exclusiva para el arranque (no pudiendo escribir en ella).

Además, Kristian concibió el sistema para funcionar con los elementos mínimos, a diferencia de las distribuciones de Linux tradicionales. De esta forma, consiguió un sistema en menos de 100 MB.

Las principales funcionalidades y características son:

- Kernel Linux 2.6 (Gentoo)

- Asterisk 1.2 (1.4 en versiones 0.6 y siguientes) y zaptel
- mini_httpd
- openssh
- vsftpd
- busybox
- php
- PTXDist
- m0n0wall

Existen versiones para Soekris net48xx y net55xx, VIA EPIA, PCEngines WRAP y ALIX, Gumstix y arquitectura x86. También se soporta el hardware de los fabricantes Digium, Sangoma, Rhino y Pika (incluyendo soporte para mISDN).

En las figuras siguientes se pueden ver algunos detalles de la configuración de Asterisk mediante el interfaz web de AstLinux.

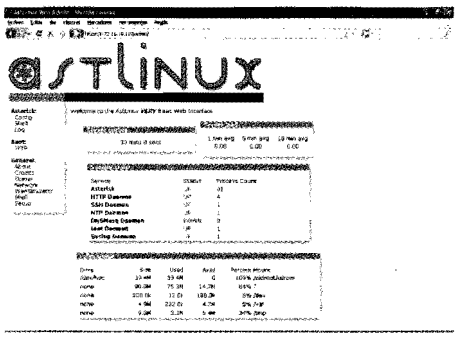


Figura V-82. Estado del sistema AstLinux

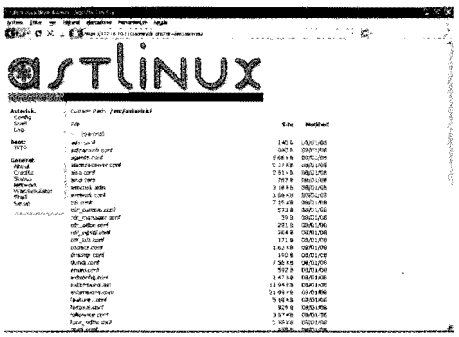


Figura V-83. Configuración del sistema

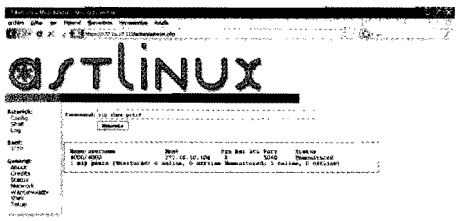


Figura V-84. Consola de comandos vía web

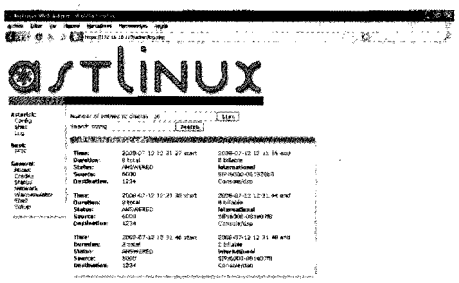


Figura V-85. Log de llamadas

Figura V-86. TFTP

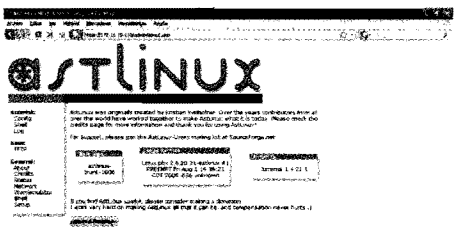


Figura V-87. Estado del sistema

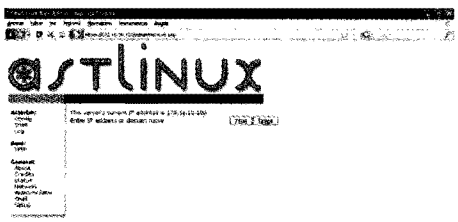


Figura V-88. Utilidades de red

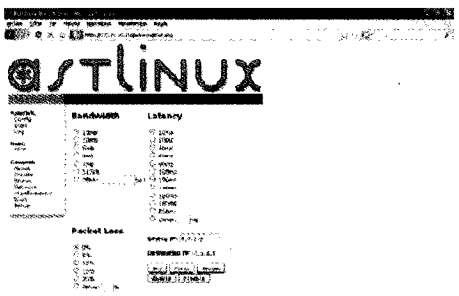


Figura V-89. Emulación de WAN

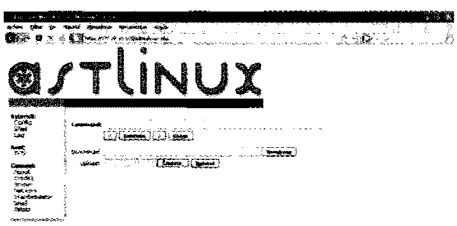


Figura V-90. Shell para comandos GNU/Linux

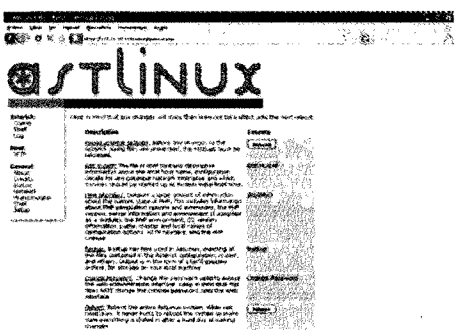


Figura V-91. Utilidades polivalentes

APÉNDICE VI

SOFTWARE DE TERCEROS PARA ASTERISK

Francisco Gil Montoya y Julio Gómez López

1 Introducción

El rápido despegue de Asterisk y de la voz IP en general ha provocado una gran revolución en el mercado de la telefonía y las nuevas tecnologías. Como suele suceder cuando algún producto novedoso consigue conectar con tanta gente a lo largo y ancho del mundo, se crea una gran comunidad de trabajo que aporta mucha creatividad y, sobre todo, nuevas aplicaciones que interactúan o basan su capacidad en dicho producto estrella.

Este ha sido el caso de Asterisk. Numerosos desarrolladores han visto el gran potencial que tiene la telefonía libre y de código abierto gracias a la aplicación creada por Mark Spencer, y han apostado por la integración y desarrollo de nuevo software, permitiendo alcanzar, en su conjunto, niveles de calidad, fiabilidad, potencia y facilidad de uso, jamás pensados en sus inicios.

Piénsese que las posibilidades que ofrece el uso de software de código libre o abierto sólo quedan limitadas por la imaginación o destreza del desarrollador.

Este factor inigualable hace que sea sumamente sencillo y abordable crear múltiples líneas de desarrollo tan polifacéticas como las siguientes:

- Plataformas de tarificación (*Billing and Accounting*)
- Envío y recepción de faxes
- Soporte para bases de datos (MySQL, PostgreSQL)
- Centros de atención al cliente (*Call Center* o *Contact Center*)
- Integración con CRM[1] (por ejemplo, SugarCRM)
- Gestión hotelera y locutorios

Figura VI-1. Puesto de trabajo de un centro de atención de llamadas

En definitiva, es posible desarrollar cualquier aplicación que integre, de manera efectiva, bases de datos con gestión telefónica, y todo en tiempo real.

Este anexo está dedicado a algunas de las más importantes aplicaciones que se han desarrollado para interactuar con Asterisk y, por supuesto, todas están disponibles bajo licencia GPL[2]. Concretamente, se mostrará un entorno web, realizado mayoritariamente en PHP, para labores de tarificación de usuarios y

---

[1] http://es.wikipedia.org/wiki/CRM

[2] http://es.wikipedia.org/wiki/Licencia_pública_general_de_GNU

control de tarjetas telefónicas denominado *A2Billing*. Además, también se presentará otra solución muy ligada a la manera tradicional de comunicación en la empresa privada y pública, como es el envío y recepción de faxes, denominado *Avantfax*.

2 Tarificación mediante A2billing

El paquete *A2Billing* (*A2B*) es una interfaz web usada para tarificación de sistemas basados en Asterisk. Fue diseñado e implementado por Belaid Areski y está considerado como el mejor paquete *OpenSource* para tarificación en Asterisk.

Sus funcionalidades son muy amplias y profesionales, por lo que muchos usuarios en todo el mundo se están decantando para su uso a nivel de pequeños operadores de VozIP, locutorios y ciber-cafés.

Entre las principales funcionalidades del sistema están:

- **Servicio de tarjetas telefónicas prepago.** A2B puede hacer de proveedor de las típicas tarjetas telefónicas prepago, en las que el usuario marca un número de la Red Pública Conmutada (RTC o *PSTN* en inglés), el sistema responde y da un nuevo tono de llamada tras un proceso de autentificación (PIN o autentificación por identificador del llamante).

- **Servicio de Retorno de Llamada** (*Callback*[3] en inglés). A2B permite ser usado como servidor de callback. Se puede utilizar bien vía web o de la forma tradicional, es decir, reconociendo el identificador de llamante.

- **Proveedor de telefonía IP residencial.** A2B puede gestionar y facturar a usuarios asociados a un servicio telefónico mediante VozIP, y que usan teléfonos IP o *softphones*.

- **Terminación mayorista para VozIP.** A2B y Asterisk pueden usarse a modo de *softswitch*[4] para gestionar la terminación de minutos en diferentes redes de comunicaciones, provenientes de diversas fuentes y revendedores o mayoristas.

---

[3] http://es.wikipedia.org/wiki/Callback

[4] http://es.wikipedia.org/wiki/Softswitch

- **Terminación para otros equipos Asterisk.** Debido al crecimiento de las centrales IP para telefonía, así como los sistemas basados en Asterisk, A2B puede proporcionar servicios de tarificación a otros integradores o revendedores de sistemas Asterisk.

- **Terminación y redirección de DDI-DID[5].** A2B puede redireccionar hacia cuentas SIP, IAX o Digital/Analógica cualquier DID entrante, así como facturar cargos mensuales o de cualquier otro tipo.

En este anexo se detallarán los aspectos más importantes de su configuración para poder tener operativo un sistema de tarificación profesional usando A2B. No se incluye ninguna guía para su instalación, puesto que existen numerosas referencias en Internet.

- En español:
  - http://www.ecualug.org/?q=2006/07/19/comos/instalar_un_sistema_de_facturacion_para_asterisk
  - http://www.ecualug.org/?q=2006/12/12/comos/configurar_a2billing_en_menos_de_10_minutos
- En inglés:
  - http://wiki.asterisk2billing.org/index.php/Installation_guide
  - http://www.sureteq.com/asterisk/a2bv1.2.3install.htm

2.1 CONFIGURACIÓN

A2B tiene dos modos de operación: administrador y usuario. En el modo administrador se realizan la mayoría de las acciones de configuración del sistema, como creación de usuarios, tarifas, facturas, etc., aunque hay una parte de la configuración que reside en el archivo */etc/asterisk/a2billing.conf*. En el modo

---

[5] DID: *Direct Inward DIaling* y DDI: *Direct Dial-In* son términos usados en literatura inglesa para referirse a la numeración asignada por el operador de telefonía a una central telefónica de empresa. Se asignan rangos de numeración (DID's) que la central gestiona cuando una llamada entra en la misma, según los patrones propios de gestión.

usuario se pueden realizar consultas sobre detalle de llamadas, ver tarifas, realizar web callback, etc.

A continuación se hará un recorrido por las principales pantallas de configuración en modo administrador.

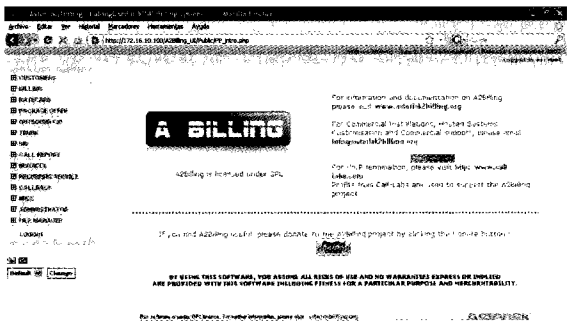


Figura VI-2. A2billing – Pantalla de inicio

Pantalla de inicio de sesión. Se muestran las opciones principales:

- **Customers**. Gestión de usuarios.
- **Billing**. Métodos de pago, cargos específicos, transacciones, pagos realizados, etc.
- **Ratecard**. Creación de tarifas y planes de precios.
- **Package offer**. Paquetes con ofertas de minutos.
- **Outbound CID**. Creación de identificadores de llamadas personalizados para los clientes.
- **Trunk**. Creación de proveedores y canales de salida.
- **DID**. Manejo de numeración entrante personalizada. Facturación de numeración asignada al cliente.
- **Call Report**. Detalles de llamadas, gráficas de consumo y carga del sistema.

- **Invoices**. Manejo de facturación de los usuarios.
- **Recurring Services**. Manejo de alarmas y cargos a usuarios por suscripción.
- **Callback**. Manejo y estadísticas del servicio de callback.
- **Misc**. Manejo de prefijos telefónicos y creación de plantillas para correo electrónico.
- **Administrator**. Manejo de administradores, sub-administradores. Creación de copias de seguridad y visualización de archivos de registro.
- **File Manager**. Gestión de archivos para música en espera y archivos de audio.

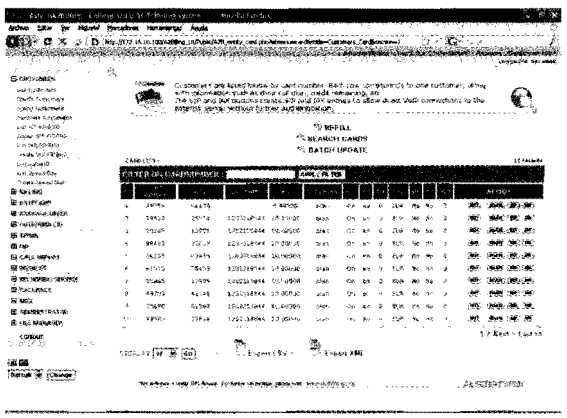


Figura VI-3. A2billing – Listado de clientes

Customers

Se pueden crear y administrar usuarios. Para crear basta con pulsar *Create Customers* y rellenar los campos necesarios. Conforme se crean usuarios, aparecerán en la pestaña *List Customers* y podremos ver sus detalles. Cada usuario del sistema puede tener asociada, o no, una cuenta SIP e IAX.

Sobre cada usuario pueden realizarse diversas labores de actualización o mantenimiento. Para ello se hará uso de la búsqueda de usuarios *Search Cards* y luego se actualizará mediante *Batch Update*.

Creación de usuarios

Se deben completar una serie de campos para identificar al usuario correctamente y que pueda operar en el sistema. Los más importantes son:

- ***Card Number.*** Número de usuario, se asigna automáticamente, aunque puede ser personalizado en */etc/asterisk/a2billing.conf.*
- ***Card Alias.*** Identificador para acceder a la web de usuario.
- ***WebUI Password.*** Contraseña para acceder a la web de usuario.
- ***Balance.*** Crédito inicial asignado.
- ***Language.*** Idioma del interfaz de usuario.
- ***Call Plan.*** Plan de precios asignado al usuario. Se debe crear en la sección *Ratecard.*
- ***Currency.*** Moneda de uso por el usuario. Útil a efectos de tarificación.
- ***Card Type.*** Define si el usuario será de tipo prepago o postpago.
- ***Sip account.*** Define si se crea un usuario SIP en Asterisk.
- ***IAX account.*** Define si se crea un usuario IAX en Asterisk.
- ***CallerID.*** Permite asociar identificadores de llamadas al usuario al objeto de usarlos en otros servicios como el callback o la autentificación de usuarios SIP o IAX.

Billing

A través de la opción de *Billing*, se puede administrar todo lo que tiene que ver con movimientos monetarios, medios de pago, etc.

A2B se integra con varios sistemas on-line de pago como Paypal, Moneybookers y Authorize. Para una correcta configuración es necesario configurar también, *a2billing.conf.*

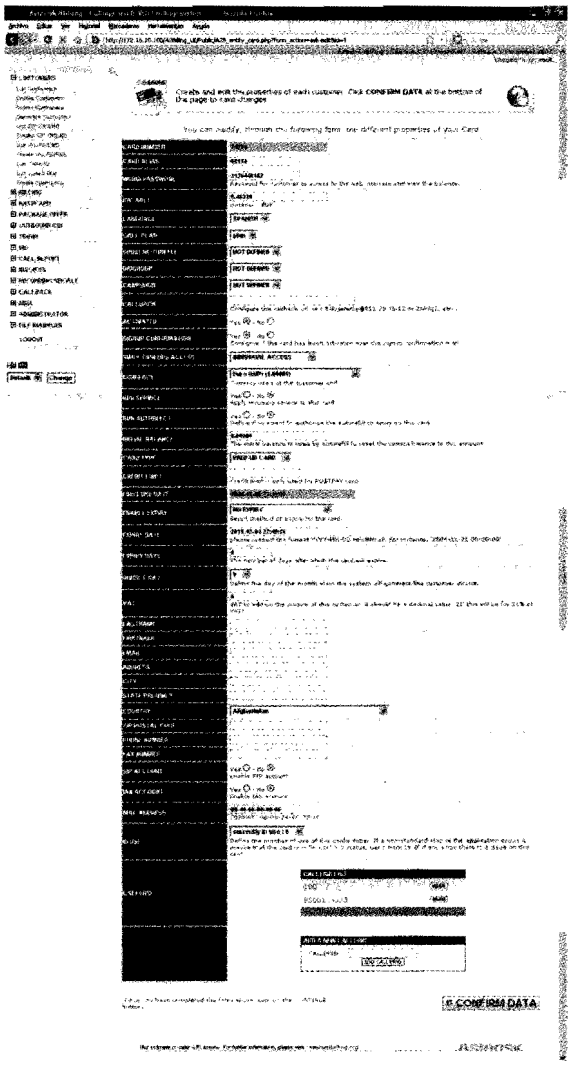


Figura VI-4. A2billing – Alta de clientes

En *View Money Situation* se pueden listar todos los pagos que han realizado los usuarios (*Payments)* ya sea por los métodos on-line, como de forma manual. También se listan las actualizaciones de saldo (*Refill*) que A2B maneja por separado, es decir, al hacer un pago (que no sea on-line) se debe usar la opción *Refill* disponible en *List Customers* para actualizar el saldo.

Además de obtener listados acerca de los pagos y la situación financiera, también se pueden emitir cupones preactivados (*vouchers*). Sólo es necesario acceder a *Creat Voucher* e introducir la cantidad a recargar. El usuario final sólo

tiene que acceder a su página personal e introducir el código, y será automáticamente recargado en la cantidad correcta.

Por último, en el apartado de *Currency List*, se tiene un listado de paridad monetaria entre divisas (que se actualiza a través del portal financiero de Yahoo), muy útil si se están usando diferentes monedas en el sistema.

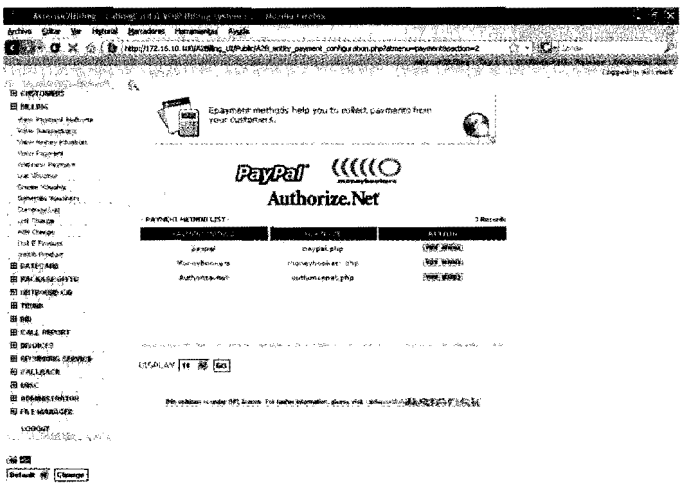


Figura VI-5. A2billing – Paypal

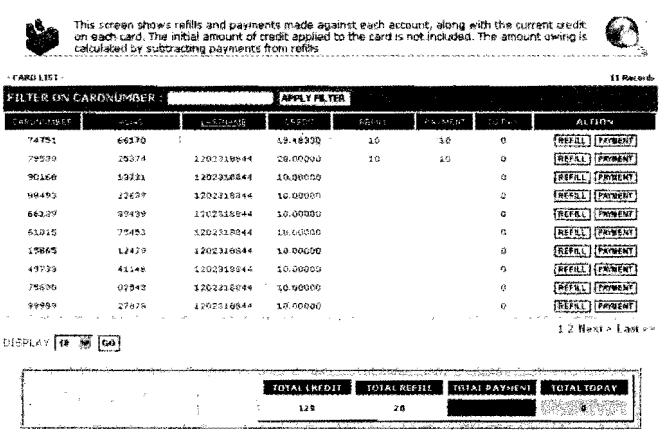


Figura VI-6. A2billing – Listado de tarjetas

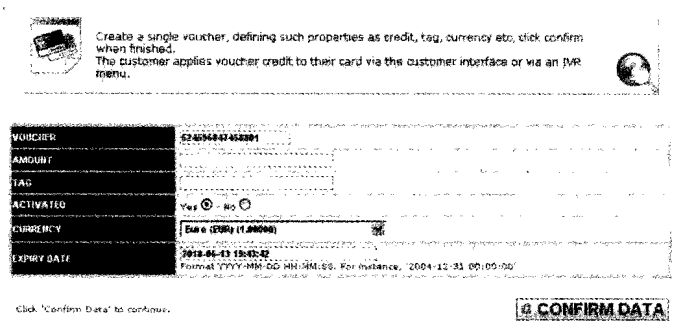


Figura VI-7. A2billing – Cupón de recarga

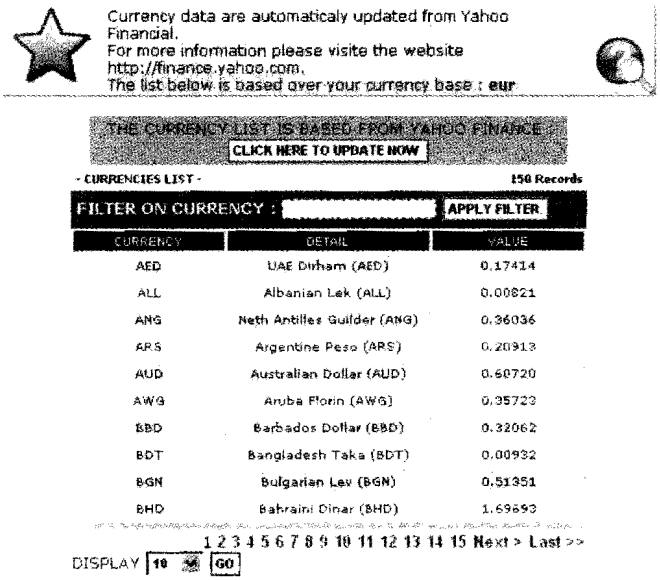


Figura VI-8. A2billing – Tarifas

Ratecard

En el apartado *Ratecard* se define todo lo que tiene que ver con prefijos, rutas y precios de las llamadas.

En *List Call Plan* se encuentran los planes de precios sobre los que se tarifica a los usuarios.

Un plan de precios puede comprender varias tarifas (*Ratecard*), que son elegidas según una lógica preestablecida de enrutado, es decir, se puede hacer que

un plan de precios elija una tarifa u otra del plan, según se defina una ruta de mínimo coste (*LCR* en inglés) para el operador o mínimo coste para el usuario.

Dentro del plan de precios se puede elegir qué tarifas son añadidas de entre las disponibles ya creadas.

Una vez creado el Plan de Precios, es necesario crear las distintas tarifas que compondrán dicho plan u otros planes. Para ello es necesario situarse sobre *Create New Ratecard* y podremos introducir los datos necesarios. En este caso, se da nombre a la tarifa, se asigna un periodo de validez y se elige el proveedor a utilizar por defecto.

Por último, sólo queda crear los destinos pertenecientes a cada tarifa, por lo que se usará *Add Rate* y se irán rellenando los campos necesarios.

Ratecard. Tarifa a la que pertenece el destino.

Dial Prefix. Se introducirá el patrón de marcado para reconocer al destino. Puede introducirse a mano o seleccionarlo de entre la base de datos que incorpora A2B.

Destination. Nombre del destino en la tarifa.

Después de introducir los anteriores datos, aparecen una serie de campos relativos a los precios de compra y venta del minuto telefónico, periodo de tarificación (por segundos, por minutos, etc.), así como costes típicos de algunos operadores como establecimiento de llamada, desconexión, etc.

Se puede establecer un periodo de validez para la tarifa, así como personalizar el proveedor de salida para el destino en concreto, independientemente del que tenga asignado la Tarifa en general.

El resultado final será una tabla donde aparecerán las distintas Tarifas (*Ratecard*) y sus correspondientes destinos.

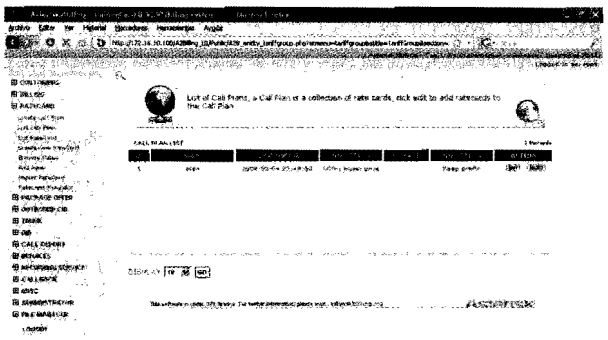


Figura VI-9. A2billing – Listado de Callplan

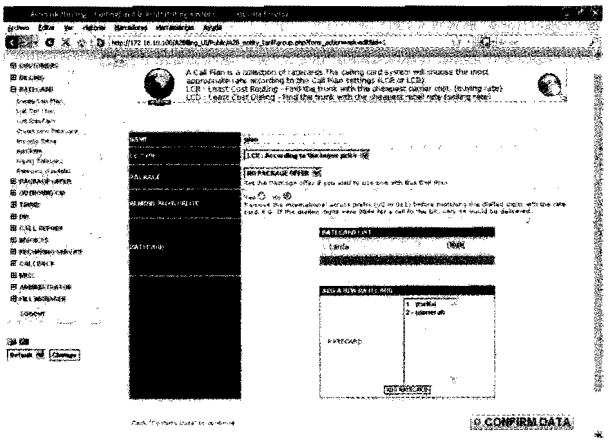


Figura VI-10. A2billing – Callplan

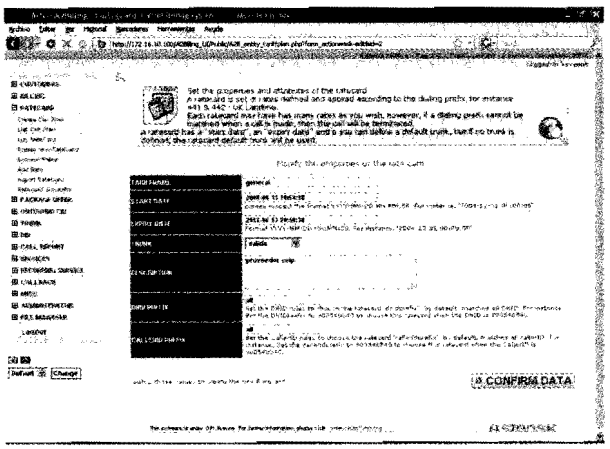


Figura VI-11. A2billing – Propiedades de una tarjeta

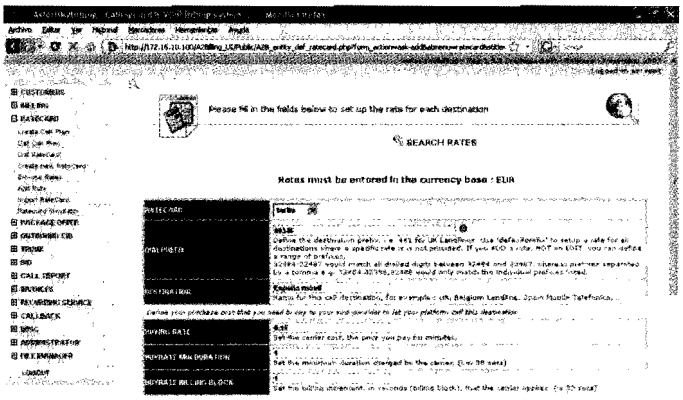


Figura VI-12. A2billing – Recarga de tarjetas

Trunk

En este apartado se definen los canales de salida o proveedores. Existen dos opciones: *trunk* o *provider*. *Provider* se usa para propósitos estadísticos y no es imprescindible para el funcionamiento del sistema.

Add Trunk permite añadir un proveedor de cualquier tipo de tecnología (SIP, IAX, Zaptel, etc.) por el que sacar llamadas. Se puede añadir o quitar prefijos a conveniencia, así como definir otros proveedores ya creados como ruta de fallo *(failover)*.

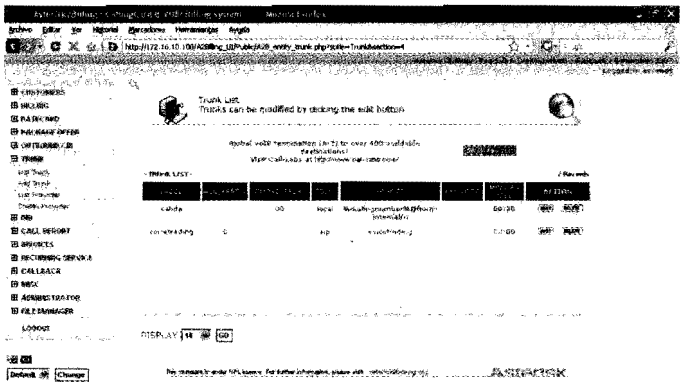


Figura VI-13. A2billing – Listado de proveedores

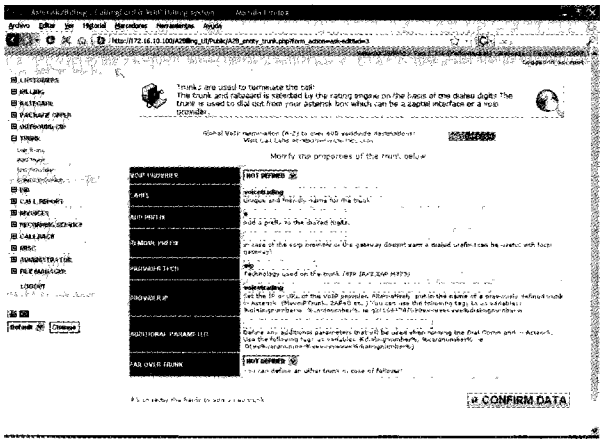


Figura VI-14. A2billing – Propiedades de un proveedor

DID

A2B puede manejar rangos de numeración asignados por un operador, pudiendo facturar su uso a cada usuario de diversas formas.

Para su manejo se deben crear grupos de DID's a los cuales asignar DID individuales.

Se introduce el DID en cuestión y se asigna un modo de tarificación así como un grupo de DID.

Se puede redireccionar cualquier DID a cualquier cuenta o tecnología usando *Add Destination*.

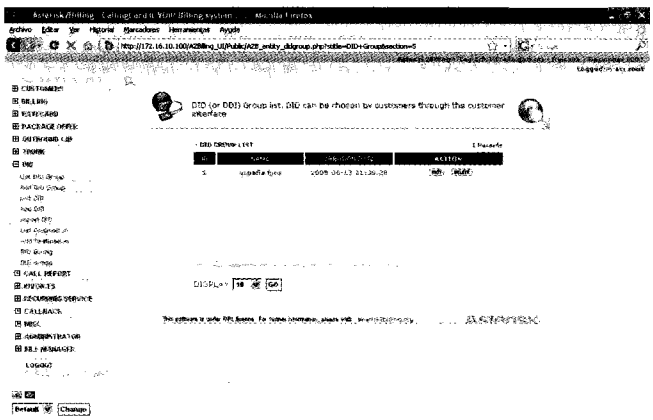


Figura VI-15. A2billing – Listado de DID

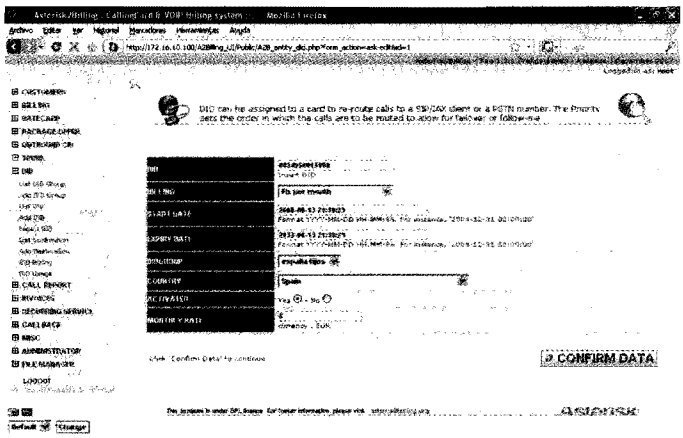


Figura VI-16. A2billing – Propiedades de un DID

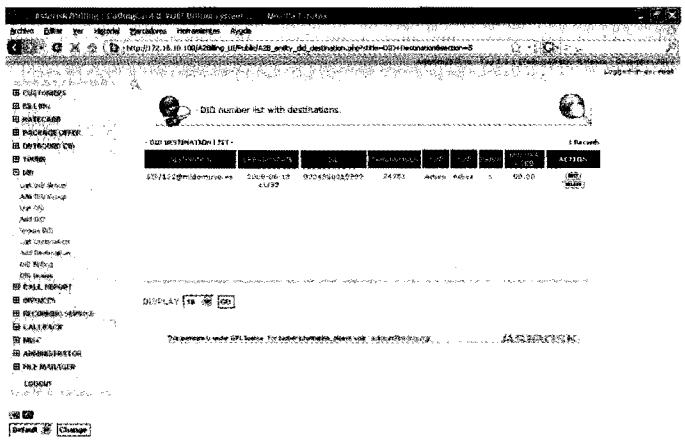


Figura VI-17. A2billing – Listados de DIDs con sus destinos

Call Report

A2B dispone de un potente sistema de detalle de llamadas y estadísticas que permiten obtener precisos reportes de todo el tráfico telefónico efectuado en el sistema Asterisk.

Es posible obtener parámetros típicos referentes a calidad y fiabilidad de las llamadas como son:

- *Average Length of Call (ALOC).* Duración media de llamada.

- *Answer Seizure Ratio (ASR).* Relación entre las llamadas correctamente contestadas y las efectuadas.

- *Maximum number Failed Calls Successively (MFCS).* Número máximo de llamadas fallidas sucesivamente.

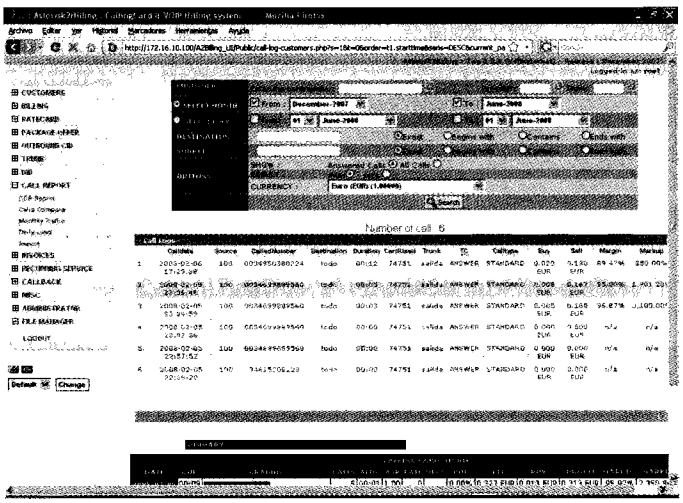


Figura VI-18. A2billing – Reporte de llamadas

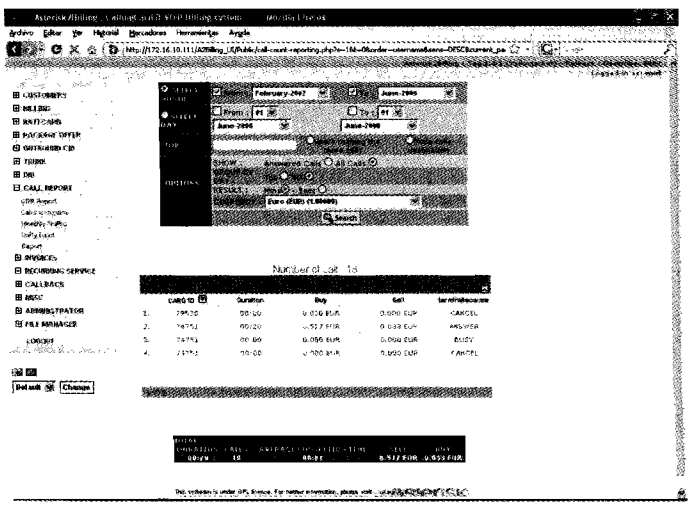


Figura VI-19. A2billing – Filtrado de llamadas

Invoices

A2B también permite llevar una contabilidad y facturación de cada uno de los clientes del sistema. Es posible asignar pagos, facturar mensualmente, emitir detalle de facturas, etc.

También es posible enviar toda la información de facturación mediante correo electrónico a los clientes en formato PDF o HTML.

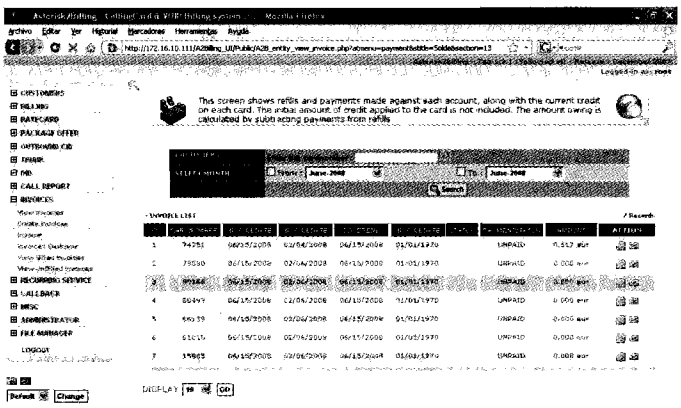


Figura VI-20. A2billing – Facturas

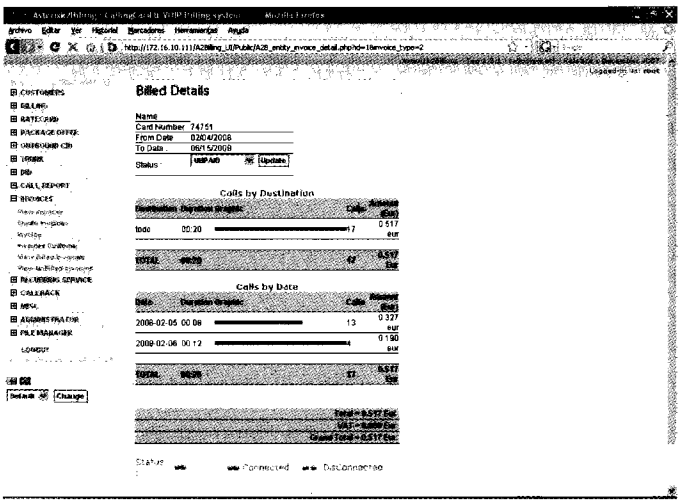


Figura VI-21. A2billing – Detalles de consumo de una factura

2.2 CONCLUSIÓN

Como se ha podido apreciar *A2Billing* es un potente paquete que rivaliza con caros sistemas de facturación propietarios, y que se integra con Asterisk perfectamente.

Sus funcionalidades permiten tener al alcance de la mano la posibilidad de montar un sistema completo de gestión y facturación para un operador de telefonía IP por una mínima fracción de dinero, y lo más importante, podremos adaptarlo completamente a nuestros requisitos en caso de que no nos satisfaga completamente.

Su uso es sencillo y a la vez potente, permitiendo gestionar una cantidad aceptable de usuarios[6] en entornos reales.

3 Manejo de Faxes mediante Avantfax

Otra gran solución que trabaja con Asterisk es *Avantfax*, disponible en http://www.avantfax.com. En este caso, la idea es poder manejar faxes desde un entorno web, en formato PDF y con la posibilidad de usar el correo electrónico para reenviar dichos faxes.

AvantFAX, como tal, es una aplicación web que hace uso de otro paquete, denominado *Hylafax*[7]. Este paquete es realmente el servidor de fax implementado mediante software. A su vez, y para completar la integración IP, se usa un tercer paquete denominado *IAXModem*[8] que sirve como módem telefónico y se comunica con Asterisk mediante el protocolo IAX. De esta forma, esta combinación de software sustituye completamente a las tradicionales máquinas de fax, sin perder una funcionalidad que, aunque arcaica, sigue arraigada todavía en la estructura empresarial.

---

[6] Aunque no existen datos contrastados sobre carga de trabajo y pruebas de rendimiento, se estima que un sistema de múltiples núcleos y algunos gigas de RAM podría gestionar algunos centenares de llamadas simultáneas.

[7] http://www.hylafax.org

[8] http://iaxmodem.sourceforge.net/

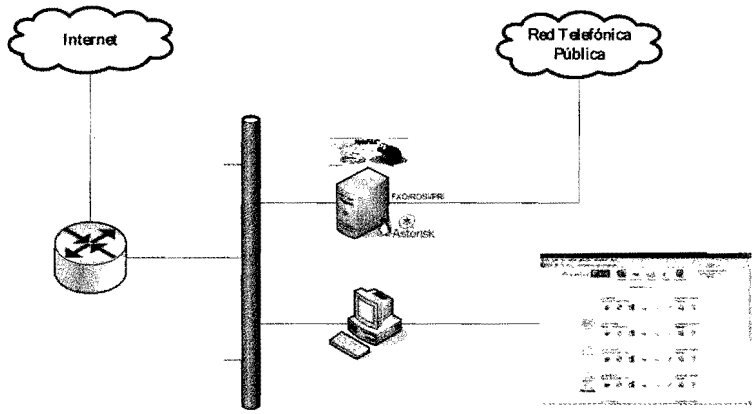


Figura VI-22. Esquema de red para servidor de fax

Los usuarios de Avantfax pueden ver y gestionar los faxes sin la necesidad de instalar o tener ningún software especial. Además, permite a los administradores manejar usuarios, permisos, líneas de fax, categorías de fax, etc.

El interfaz web de Avantfax es accesible no sólo en la red local sino también desde el exterior de la red.

Las características más destacadas de Avantfax son las siguientes:

- Los usuarios pueden ver los faxes en tiempo real mediante un navegador web estándar (IE 6/7, Firefox, Safari y Opera).
- Se pueden descargar los faxes en formato PDF.
- Reenvío automático de faxes mediante correo electrónico y por PDF, para número de fax preestablecidos.
- Se pueden reenviar faxes en PDF sin necesidad de ningún cliente de correo instalado.
- Número de usuarios "casi" ilimitado (dependiente de la capacidad de hardware del servidor).
- Número ilimitado de líneas de fax.
- Soporte para envío de faxes mediante correo electrónico.
- Soporte para reconocimiento automático de caracteres (*OCR* en inglés).
- Libreta de direcciones.
- Soporte para varios idiomas.

- Capacidad de archivar faxes para una posterior búsqueda por diferentes campos.

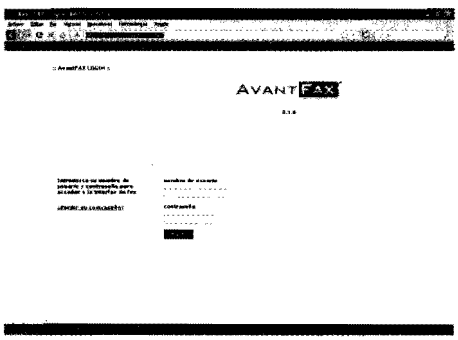
3.1 INSTALACIÓN

Al igual que A2Billing, la instalación de Avantfax está bien documentada en la red. Entre otros, se pueden encontrar los siguientes enlaces:

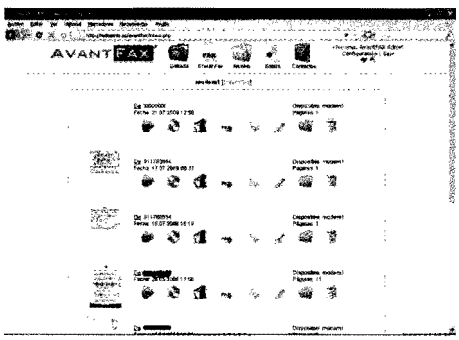
- En español:
  - http://www.ecualug.org/?q=2007/02/26/comos/instalar_avantfax
  - http://ualtech.wordpress.com/2008/05/04/jautu-sobre-avantfax-con-asterisk-y-freepbx/
- En inglés:
  - http://www.avantfax.com/install.php
  - http://www.howtoforge.com/build-a-hylafax-server-with-avantfax-on-debian-etch

3.2 CONFIGURACIÓN

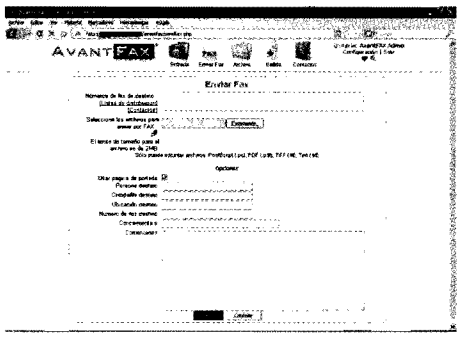
A continuación en las siguientes figuras puede ver los elementos de configuración más importante del sistema.



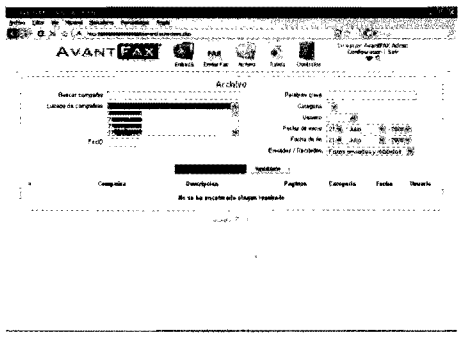
Pantalla de inicio



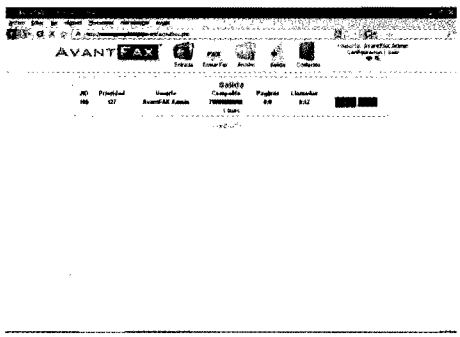
Bandeja de entrada del sistema



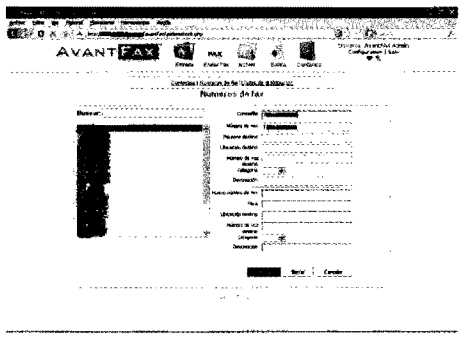
Envío de faxes vía web



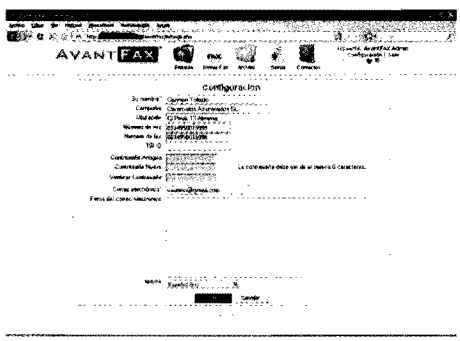
Archivo de faxes entrantes y salientes



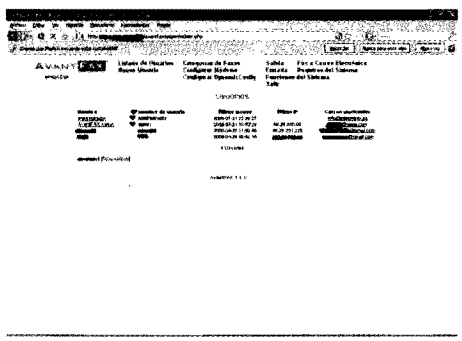
Bandeja de salida



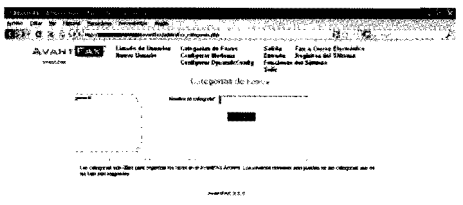
Libreta de contactos



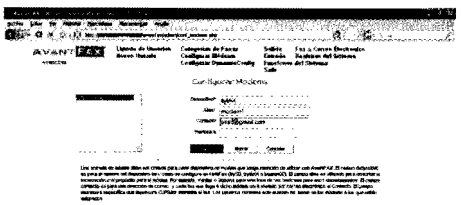
Configuración de usuario



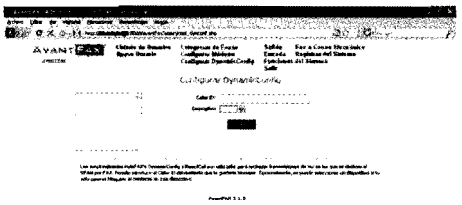
Configuración de administrador



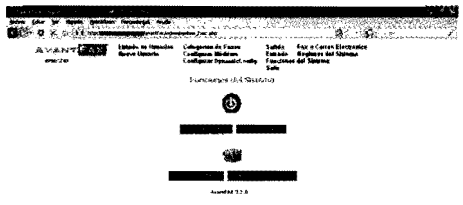
Creación de categorías de faxes



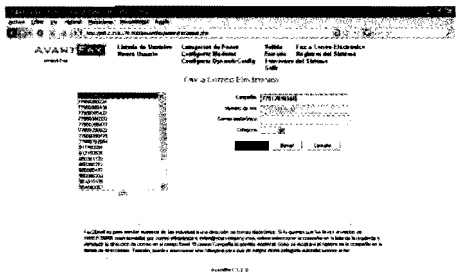
Configuración de módems (IAXmodem)



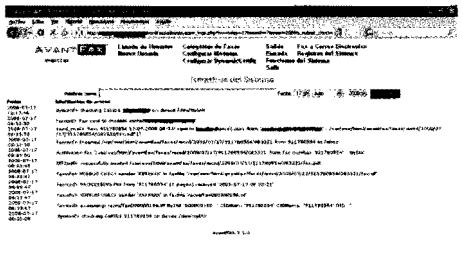
Funcionalidades avanzadas



Funcionalidades de Sistema



Sistema fax a correo electrónico



Registro de actividad del sistema

Figura VI-23. Pantallas de configuración AvantFax

4 Ejemplos prácticos

4.1 TARIFICACIÓN EN LOCUTORIOS

Un caso típico donde se tiene la necesidad de tarificar llamadas telefónicas es el de los locutorios telefónicos (también conocidos como *callshop* o *teleboutique*). Un locutorio es un establecimiento frecuentado mayoritariamente por personas con necesidad de comunicarse con destinos internacionales a un precio razonable. Tradicionalmente, las llamadas desde la red fija y móvil hacia destinos internacionales han sido extremadamente caras por oscuros motivos, pero comúnmente aceptados debido a la relación conceptual entre distancia y precio del servicio:

> *"... es lógico pensar que si uno llama a un destino lejano, el precio debe ser más caro que si llamo a un teléfono de mi misma ciudad..."*

Craso error. La gran mayoría de operadores telefónicos tienen acuerdos de interconexión con *carriers*[9] internacionales que les permiten terminar llamadas a precios muy asequibles, pero que nunca son trasladados al cliente final. Sólo si se tiene un cierto volumen de llamadas (como sucede en los locutorios) es posible realizar una negociación directa en la que se pueda conseguir un precio mucho más razonable y que, a su vez, el locutorio pueda trasladar el cliente final.

Lo anterior ha sido la tónica de trabajo durante muchos años, aunque con la llegada de la liberalización de las comunicaciones móviles, la VoIP y los Operadores Móviles Virtuales, cualquier persona tiene acceso hoy en día a tarifas internacionales (e incluso nacionales) del mismo orden o incluso inferior que ciertos locutorios.

Aun así, en muchos países, no es posible acceder a estos servicios por no estar desarrolladas las infraestructuras mínimas, y sigue siendo necesario hacer uso de los locutorios.

---

[9] En telefonía, el término *carrier* hace referencia a grandes compañías de telecomunicaciones especializadas en interconexiones internacionales, manejando gran cantidad de datos y tráfico telefónico.

4.2 ESQUEMA DE TRABAJO DE UN LOCUTORIO

Los locutorios suelen tener diversos tipos de funcionamiento, aunque el más extendido consiste en:

- El cliente entra al local y solicita una cabina para llamar.
- El operador asigna un número y cabina al cliente.
- El cliente entra en la cabina asignada y el operador desbloquea la misma para su uso.
- El cliente puede, a partir de ese momento, realizar llamadas a cualquier destino permitido (suele ser típico inutilizar número gratuitos como los 900 en España, teléfonos satelitales, números Premium, etc.).
- Se efectúan una o varias comunicaciones por parte del cliente.
- El cliente termina de hablar y se dirige hacia el operador.
- El operador usa software específico para tarificación de locutorios y factura la llamada.
- El cliente obtiene un ticket por el importe total y paga el montante.
- La cabina está lista para recibir a nuevos clientes.

Como se puede apreciar en la descripción anterior, realizar una llamada telefónica en un locutorio tradicional implica disponer de:

- Teléfonos analógicos
- Hardware para control de líneas telefónicas
- Software para tarificación en tiempo real
- Tantas líneas telefónicas como cabinas instaladas

Además, la contratación de las líneas telefónicas supone incurrir en un coste mensual con la compañía telefónica, y sobre todo, tener que "tragar" con sus tarifas de trabajo. Esto último supone no tener margen de maniobra puesto que muchas veces no es posible contratar con otros proveedores telefónicos al tener la compañía propietaria de las líneas una exclusividad total sobre las mismas. En ciertos países (como España), es posible contratar con revendedores de minutos

que proporcionan precios más asequibles que la compañía dominante, aunque aun así, sigue siendo un escenario no óptimo. Hoy en día existen numerosas compañías en Internet que son capaces de ofrecer unas tarifas más atractivas si se usa tecnología de voz sobre IP.

A2Billing, Asterisk y cierto hardware para VoIP pueden ayudarnos a lograr que el locutorio pueda gestionar de forma completamente autónoma todo lo necesario para seguir ofreciendo el servicio de llamadas telefónicas pero con una mayor rentabilidad y fiabilidad.

Tabla VI-1. Analogía entre locutorio IP y tradicional

Elemento IP	Elemento tradicional	Funcionalidad
A2billing	Software propietario tarificación	Software para tarificación
ATA o Teléfono IP	Teléfono Analógico	Teléfonos para llamar
Asterisk	Hardware para gestión de líneas	Control de líneas telefónicas
Proveedor VoIP	Compañía telefónica	Terminación de llamadas

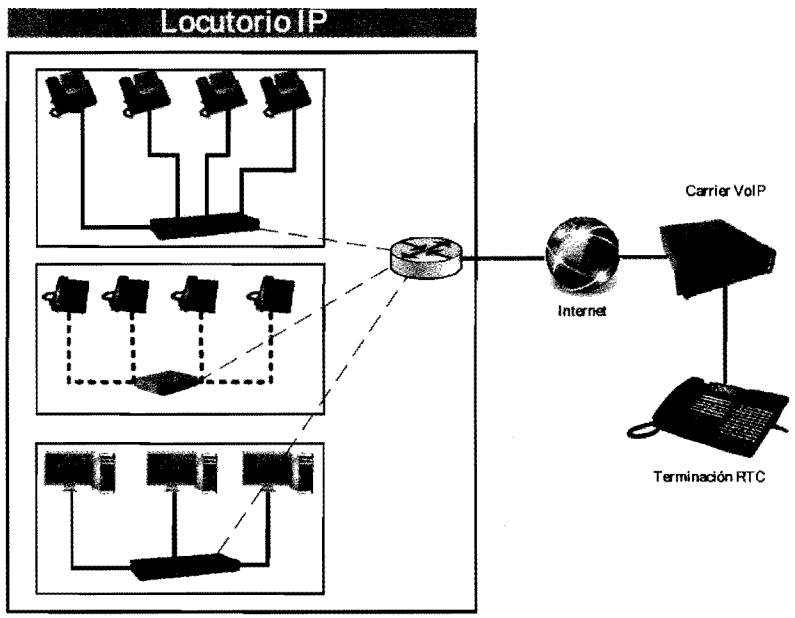


Figura VI-24. Diagrama de un locutorio IP

4.3 PUESTA EN MARCHA Y CONFIGURACIÓN DE UN LOCUTORIO IP

Para poder poner en marcha un locutorio completamente IP, será necesario tener instalado y configurado un servidor con Asterisk y A2Billing. Además será necesario usar adaptadores FXS, si dispone de teléfonos analógicos, o bien teléfonos IP en caso contrario.

Si utiliza teléfonos analógicos, debe tener en cuenta que el cliente no tendrá posibilidad de controlar el tiempo de la llamada así como el coste de la misma. En el caso de teléfonos IP, es muy probable que disponga de un visor con el tiempo transcurrido (aunque no el coste).

Si además quiere tener alguna salida hacia un proveedor analógico o GSM, deberemos instalar un adaptador FXO o tarjeta PCI-PCIe con puertos FXO.

La configuración necesaria se realizará fundamentalmente en A2billing. Para ello, acceda al sistema con un explorador Web y autentifíquese en el sistema (véase la figura VI-25).

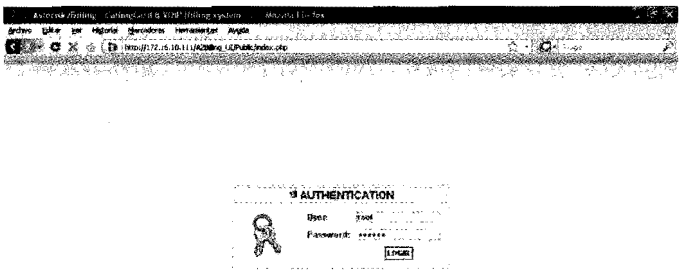


Figura VI-25. A2Billing – Página de inicio

Introduzca su nombre de usuario y contraseña. A continuación, pulse *Create Call Plan* para crear una lista de precios. Se le asigna un nombre y se pulsa en *Confirmar*.

Figura VI-26. A2Billing – Create Call Plan

Creado el plan de precios, pasamos a crear un proveedor por el que enrutar las llamadas hacia el exterior. Para ello pulse en *Create Provider*. Introduzca los datos que nos interesan y pulse *Confirm data*.

En este caso, la funcionalidad de proveedor es meramente estadística, ya que es en la opción *Trunk* donde realmente se define la ruta de salida.

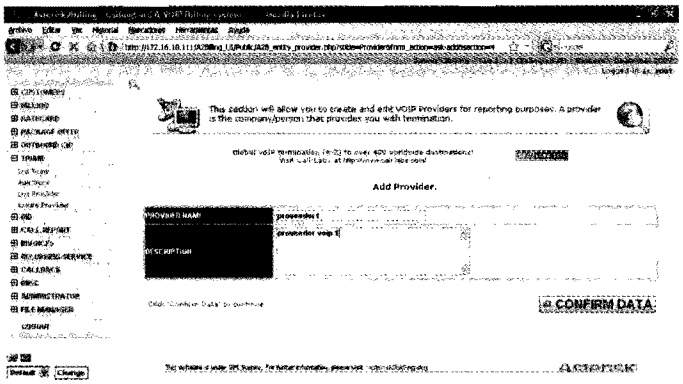


Figura VI-27. A2Billing – Configuración de un trunk

Seguidamente hay que crear el canal de salida pulsando en *Create Trunk*. Seleccione el proveedor, el nombre del trunk, la tecnología a usar y por último, la dirección del Proxy SIP y pulse *Confirm Data*.

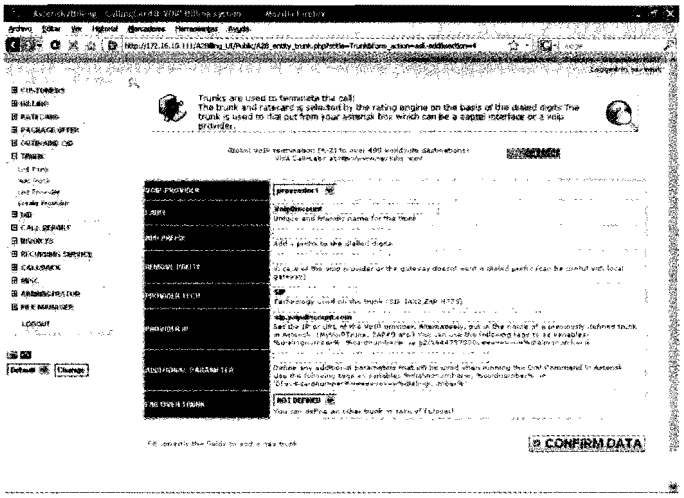


Figura VI-28. A2Billing – Creación de un trunk

Seguidamente se va a crear una tarifa de precios. Pulse en *Create Ratecard* y asigne valores: nombre de la tarifa, trunk de salida y una descripción sobre la misma.

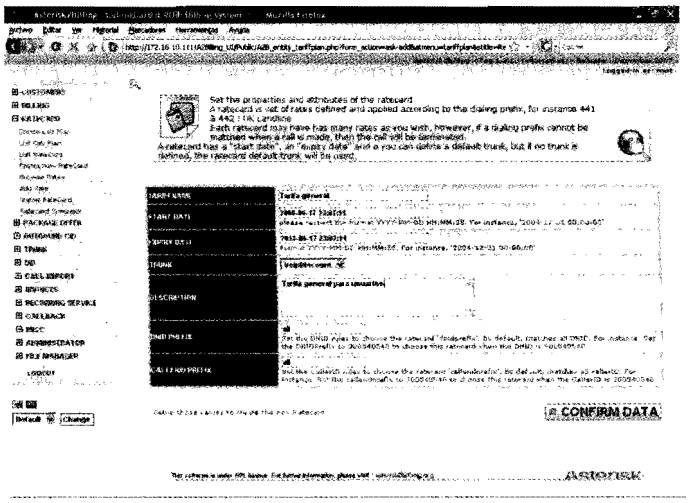


Figura VI-29. A2Billing – Create RateCard

Hay que dar de alta la tarifa y asignarle a un plan de precios.

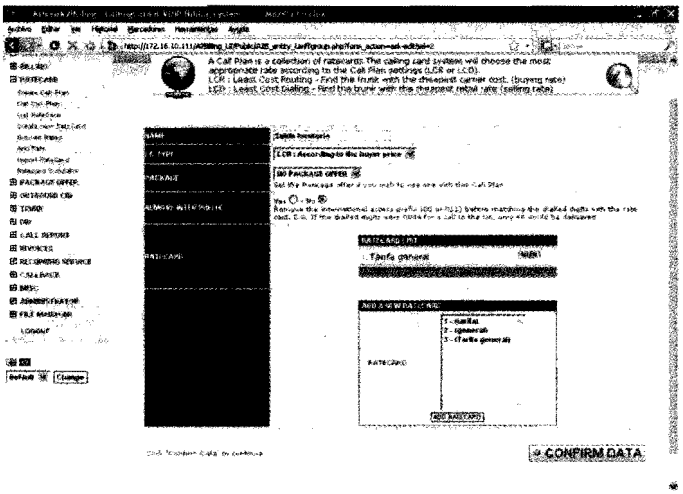


Figura VI-30. A2Billing – Asignación de la tarifa al plan de precios

La tarifa creada debe tener asignados destinos con sus respectivos prefijos, por lo que iremos añadiéndolos mediante *Add Rates*. Si es la primera vez que lo hace, y tiene muchos, lo lógico será hacer una importación a escala, a través de un archivo de texto plano (csv, txt, etc.).

Deberemos elegir la tarifa asignada, el trunk por defecto y los campos que considere oportunos.

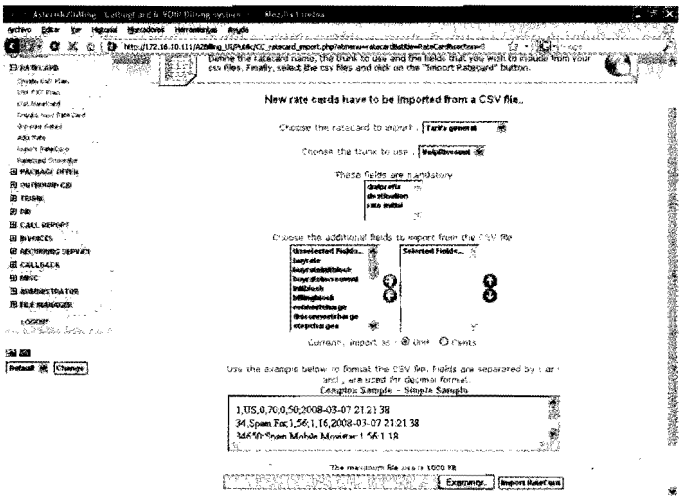


Figura VI-31. A2Billing – Importación de tarifas

Si el proceso ha tenido éxito, se habrán importado todos los destinos con sus correspondientes precios y forma de tarificar.

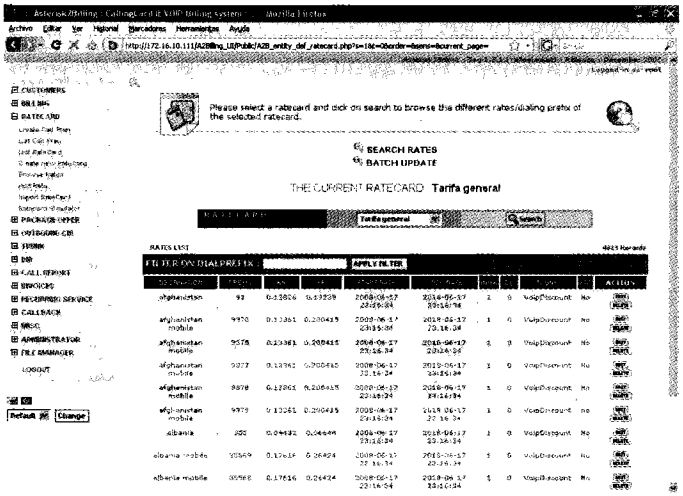


Figura VI-32. A2Billing – Tarifas importadas

Ya tiene listo toda la parte referente a las tarifas y rutas de salida. Sólo queda dar de alta los teléfonos de las cabinas para que puedan realizar llamadas en el sistema. Esta acción será realizada dando de alta a un único usuario que tendrá varios usuarios SIP haciendo cargos en su cuenta.

Hay que dar de alta dicho usuario en *Create Customers*. Deje prácticamente todos los valores por defecto, a excepción de *Card Type* que pondrá a *Postpay,* y asigne un *Credit Limit* lo suficientemente alto como para que no tenga que preocuparse de él durante un buen tiempo. Además, debe permitir acceso simultáneo con *Simultaneous Access*.

El siguiente paso es ir creando los diferentes usuarios SIP que harán cargos a la tarjeta anterior. Para ellos pulse en *Create Sip Friend* y asigne el *ID Card* a la tarjeta creada previamente. En *Name* y en *UserName* escriba "cabina 1". En *Accountcode* es importante que se ponga el número de la tarjeta para autentificar las llamadas. Si no se hace así, no será posible realizar ninguna llamada (fallo en la autentificación).

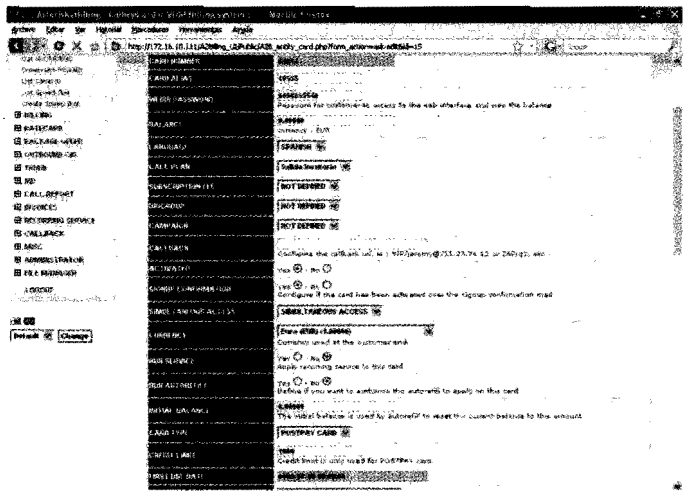


Figura VI-33. A2Billing – Create Customer

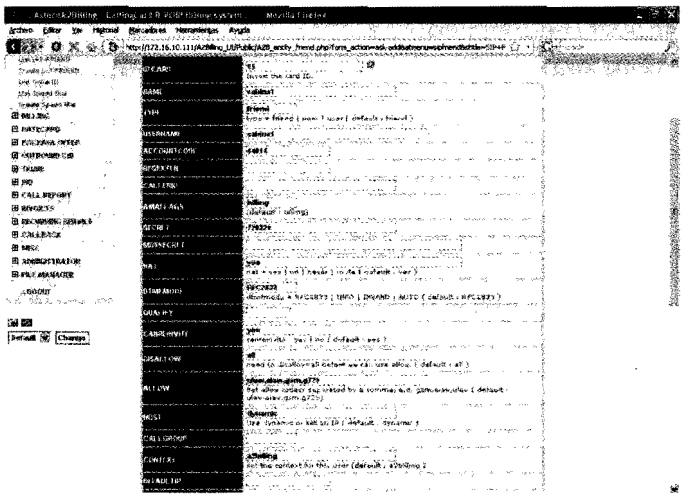


Figura VI-34. A2Billing – Create SIP friend

Además, se deberá tener en cuenta que la configuración del fichero */etc/asterisk/a2billing.conf* debe ser similar a la siguiente:

```
[agi conf1]
debug = 1
asterisk version = 1 4
answer call = no
play audio = no
say goodbye = NO
play_menulanguage = NO
```

```
force language =
intro prompt =
min credit 2call = 0
min duration 2bill = 0
notenoughcredit cardnumber = no
notenoughcredit assign newcardnumber cid = no
use dnid = yes
no auth dnid = 2400,2300
number try = 1
force callplan id =
say balance after auth = no
say balance after call = no
say rateinitial = no
say timetocall = no
auto setcallerid = YES
force callerid =
cid sanitize = NO
cid enable = no
cid askpincode ifnot callerid = no
cid auto assign card to cid = no
cid auto create card = NO
cid auto create card len = 10
cid auto create card typepaid = POSTPAY
cid auto create card credit = 0
cid auto create card credit limit = 1000
cid auto create card tariffgroup = 6
callerid authentication over cardnumber = NO
sip iax friends = No
sip iax pstn direct call prefix = 555
sip iax pstn direct call = NO
ivr voucher = NO
ivr voucher prefix = 8
jump voucher if min credit = NO
extracharge did =
extracharge fee =
international prefixes = 00
dialcommand param = "|60|iCHgL(%timeout%:61000:30000)"
dialcommand param sipiax friend =
"|60|HRgirL(3600000:61000:30000)"
switchdialcommand = NO
failover recursive limit = 2
maxtime tocall negatif free route = 5400
send reminder = NO
record call = NO
monitor formatfile = gsm
agi_force_currency =
```

```
currency association =
usd:dollars,mxn:pesos,eur:euros,all:credit
file conf enter destination = prepaid-enter-dest
file conf enter menulang = prepaid-menulang2
callback_bill_1stleg_ifcall_notconnected = YES
```

Tras crear tantos usuarios SIP como cabinas tenga el locutorio, deberá haber un listado como el de la figura VI-35.

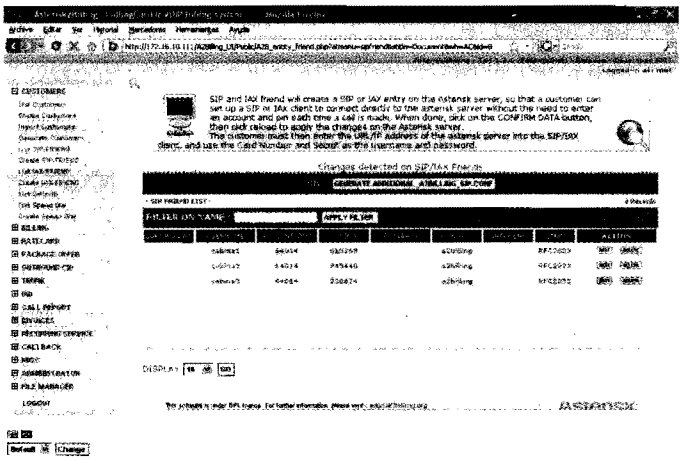


Figura VI-35. A2Billing – Listado de usuarios SIP

En este caso, existen tres cabinas que tienen el crédito asignado a la tarjeta 64014 y, por tanto, son tarificados de la misma forma (según lo definido en el plan de precios asignado al usuario 64014).

A partir de este momento, los dispositivos instalados en las cabinas, ya sean teléfonos IP, ATA's o PC's con softphones, podrán enviar llamadas al sistema para ser tarificadas, teniendo en cuenta que por cada llamada se generará un detalle para su posterior facturación.

Como paso final, sólo queda acceder a *CDR Report* y visualizar el detalle de facturación de cada cabina. Es importante resaltar que A2Billing carece de la opción de acumular las llamadas efectuadas por una persona en una cabina, así que es necesario hacer una suma aparte. Tampoco es posible realizar una impresión típica en una impresora de tickets, tal y como es costumbre en los sistemas tradicionales.

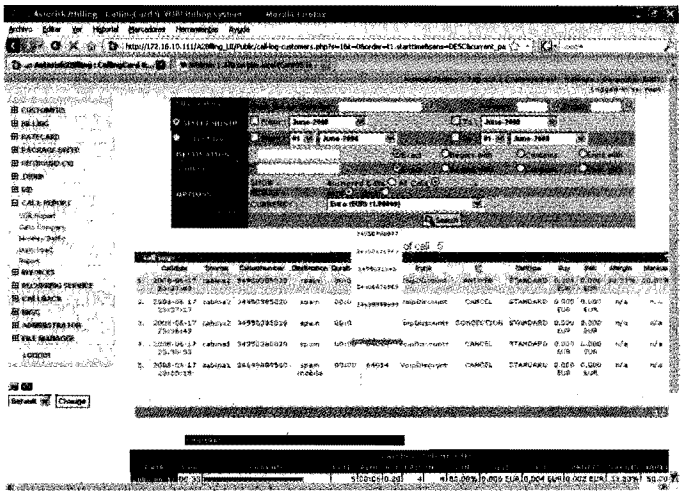


Figura VI-36. A2Billing – CDR Report

APÉNDICE VII

ASEGURANDO LA CALIDAD DE UN SISTEMA VOIP

Saúl Ibarra Corretge

1 Introducción

La seguridad es uno de esos aspectos que se suelen dejar para el final y que muchos ni siquiera miran, aunque pueda ser un factor crítico.

Nadie cuestiona que un servidor de correo electrónico tenga que ser seguro, ya que sino podríamos convertirnos en distribuidores de spam, pero con la VoIP no está pasando esto. Ni la VoIP ni Asterisk tienen tantos años como el correo electrónico, y por ahora se nota un cierto vacío en lo que a seguridad se refiere. Mucha gente no asegura correctamente sus servidores Asterisk, pero como tampoco es la regla general tener los puertos del servidor abiertos, las inseguridades quedan ocultas.

En este anexo se tratará de ofrecer una visión global sobre la seguridad en un servidor Asterisk, examinando distintos tipos de ataques que pueden realizarse,

tratando de mitigar sus efectos, para tener un sistema lo más seguro posible, ya que un sistema sólo estará 100% seguro si no está encendido.

2 Análisis inicial de la seguridad VoIP

Antes de comenzar a analizar los puntos de una instalación susceptibles de ataques es necesario hacer una valoración inicial del estado de la seguridad en instalaciones de VoIP.

En comparación con la telefonía tradicional, la VoIP puede resultar más segura ya que, por ejemplo, para escuchar una conversación en telefonía tradicional basta con irse a la habitación de al lado y descolgar el teléfono, mientras que en VoIP esto no sucede.

La VoIP convierte la voz en datos, por lo que estos datos se pueden cifrar, haciendo imposible su uso por parte de terceros. No obstante, es necesario tener en cuenta que, en muchas instalaciones de VoIP, Internet se encuentra involucrado, y en este caso Internet es un medio hostil. Todo dato que viaje en claro a través de Internet es susceptible de ser capturado y analizado por personas con fines dudosos, por lo que conviene tener especial cuidado en este tipo de conexiones.

Como se puede apreciar la VoIP es a priori un servicio que es posible asegurar, pero que dispone de puntos delicados a los que es necesario prestar atención para no abrir brechas de seguridad.

3 Elementos susceptibles de ataques

Teniendo en cuenta la accesibilidad de los elementos que componen una instalación de VoIP, podríamos dividir en 3 capas los elementos susceptibles de ser atacados:

- Terminales
- Red de VoIP
- Servidor Asterisk (PBX)

3.1. SEGURIDAD EN LOS TERMINALES

Los terminales son el elemento más vulnerable de toda una instalación de VoIP ya que están al alcance de los usuarios o de otras personas que podrían tratar de manipularlos para obtener datos delicados acerca de nuestra instalación de VoIP. Es posible atacar un terminal de varias formas diferentes:

- Fuzzing
- Flooding
- Fallos de configuración
- Servicios no deshabilitados

El *Fuzzing* consiste en el envío masivo de paquetes malformados a un dispositivo. Muchas veces, la implementación SIP de los terminales tiene fallos, y es posible inutilizar un terminal mandando paquetes erróneos a propósito, ya que podrían darse overflows y se sobrescribirían posiciones de memoria que harían que el terminal dejara de responder. Si un terminal ha quedado inutilizado por este tipo de ataque, basta con reiniciarlo para que recupere su estado habitual.

El *flooding* consiste en el envío masivo de paquetes hacia un terminal. Lo que puede suceder es que si el número de paquetes es suficientemente elevado, el terminal quede completamente inutilizado, al ser incapaz de diferenciar entre peticiones "buenas" y las procedentes de un ataque. Para inutilizar un terminal es posible realizar flooding de paquetes UDP, peticiones INVITE de SIP o paquetes RTP, obteniendo un resultado similar. Al cesar el ataque puede que el terminal se recupere aunque podría quedarse inutilizado, porque sería necesario reiniciarlo.

Otra forma de atacar un terminal consiste en aprovechar errores en la configuración de los mismos. No es tan difícil encontrarse con las contraseñas por defecto, o la administración por telnet habilitada, lo que puede facilitar la labor de hackers que desean interrumpir el servicio de ese terminal, o conseguir credenciales del usuario.

3.2. SEGURIDAD EN LA RED VOIP

En una red de VoIP hay muchos servicios involucrados como por ejemplo servidores DHCP o TFTP. Esto hace que la red disponga de más puntos de fallo por los que se puede atacar la misma, causando diversos efectos.

Se pueden realizar multitud de ataques contra una red de VoIP, pero a diferencia de los ataques contra terminales, podría quedar expuesta información más sensible e importante. Los tipos de ataques que pueden realizarse son los siguientes:

- Flooding
- Man In The Middle
- Eavesdropping
- Ataques a diversos servicios

Al igual que ocurría con los terminales, es posible utilizar ataques de *flooding* en una red de VoIP, con el objetivo de saturarla. Esto causaría una interrupción del servicio ya que sería imposible descartar todos los paquetes erróneos para quedarnos sólo con los válidos. Este tipo de ataque podría inutilizar completamente la red de VoIP, recuperándose cuando el ataque finalice.

El *Man In The Middle (MITM)* no es un ataque *per se*, es un paso previo. Consiste en infiltrarse en una red y situarse en un sitio estratégico, redirigiendo todo el tráfico. Por ejemplo un atacante podría situarse entre los terminales y la PBX, recibiendo todo el tráfico dirigido a la PBX y redirigiéndolo hacia ella. Al recibir todo el tráfico, el atacante podría analizar la señalización en busca de contraseñas, e incluso podría decodificar el flujo multimedia para escuchar las conversaciones.

Una vez el atacante se ha infiltrado en la red objetivo y ha utilizado la técnica del MITM puede llevar a cabo el que posiblemente sea el más temido de los ataques: *eavesdropping*. El *eavesdropping* consiste en el análisis del flujo multimedia de las conversaciones, para poder escuchar su contenido. Esto es especialmente peligroso ya que conversaciones confidenciales podrían ser vulneradas.

En las redes de VoIP están involucrados más servicios además del propio servidor de VoIP, por ejemplo servidores de TFTP o DHCP. Estos dos servicios también pueden ser blancos de atacantes. No es complicado consumir todas las IPs de un servidor DHCP, de manera que los terminales no puedan conseguir una IP y por lo tanto no puedan utilizarse. TFTP tampoco es un servicio seguro y con conocer el fichero que se desea obtener es posible descargarlo. Normalmente los terminales se autoprovisionan de un servidor TFTP, por lo que si un atacante consiguiera uno de estos ficheros dispondría de datos muy sensibles como el usuario y la contraseña.

3.3. SEGURIDAD EN EL SERVIDOR ASTERISK (PBX)

Como último punto a examinar en lo que a seguridad se refiere tenemos la PBX en sí. Al ser el servidor que controla todas las comunicaciones, es importante tenerlo lo más asegurado posible, ya que los ataques contra la PBX pueden causar problemas como la interrupción del servicio, suplantación de identidad, llamadas fraudulentas, etc.

A continuación se muestran los ataques más comunes y que afectan a la PBX, algunos de ellos centrados en Asterisk:

- Flooding
- Cracking de passwords
- REGISTER hijacking
- Exploits
- Errores de configuración

Al igual que los terminales y la red, la PBX también resulta vulnerable a los ataques de *flooding*, y por la misma razón: el servicio se ve desbordado de peticiones erróneas, y es incapaz de atender las peticiones auténticas, quedando por tanto el servicio interrumpido.

El mecanismo de autenticación utilizado en SIP (HTTP digest) se basa en el intercambio de un texto de desafío y su respuesta, todo cifrado con MD5. Si un atacante captura ambos mensajes (algo sencillo, ya que prácticamente todos los mensajes SIP son autenticados) dispone de todo lo necesario para averiguar la clave del usuario mediante fuerza bruta o ataques de diccionario. Si un atacante se hace con la contraseña de un usuario del sistema, podría suplantarle, hacer llamadas en su nombre, etc., algo nada deseable.

Si un atacante dispone del usuario y contraseña de alguna cuenta de la PBX éste puede llevar a cabo un ataque de *REGISTER hiajcking*, consistente en enviar un paquete REGISTER al servidor, y así suplantar al otro usuario. También podría consultar las ubicaciones (en Asterisk sólo es posible una) o des-registrar al usuario, de manera que no pueda recibir llamadas, provocándole una interrupción en el servicio sin que se diera cuenta.

Otro de los problemas más importantes para la seguridad de la PBX son los exploits. Al ser software, puede contener bugs, por lo que es posible que un paquete malformado provoque algún error en la ejecución que desemboque en la

caída del servicio. Por ejemplo, Asterisk 1.4.0 tenía un bug que causaba un core dump (caída del servicio) si le llegaba un paquete SIP con la cabecera "Content-Length" en negativo.

Por último, también es posible que haya agujeros de seguridad en nuestro Asterisk debido a una mala configuración. Para evitarlo es importante tener en cuenta lo siguiente:

- Configurar correctamente la opción allowguest en el fichero sip.conf, para evitar que usuarios no autorizados utilicen nuestro sistema Asterisk para realizar llamadas.

- Configurar los contextos de los usuarios de acuerdo a sus "privilegios". Si un usuario no debe poder llamar al extranjero, no debe tener la posibilidad de hacerlo desde el contexto en el que se halle definido.

- No utilizar la opción T en las llamadas entrantes, ya que permiten al usuario que ha llamado transferir su llamada, por lo que podría transferirse a cualquier destino sin asumir su coste.

4 Conclusiones

No hay un sistema 100% seguro y además Asterisk no dispone de todos los medios necesarios para hacer nuestras comunicaciones suficientemente seguras, por lo que se opta por poner diversas trabas al atacante (seguridad por oscuridad).

Para defendernos de los ataques arriba mencionados, se recomienda adoptar las siguientes medidas:

- Utilizar VLANs separadas para voz y datos, de manera que resulte más complicado (aunque no imposible) el acceso a la red de VoIP.

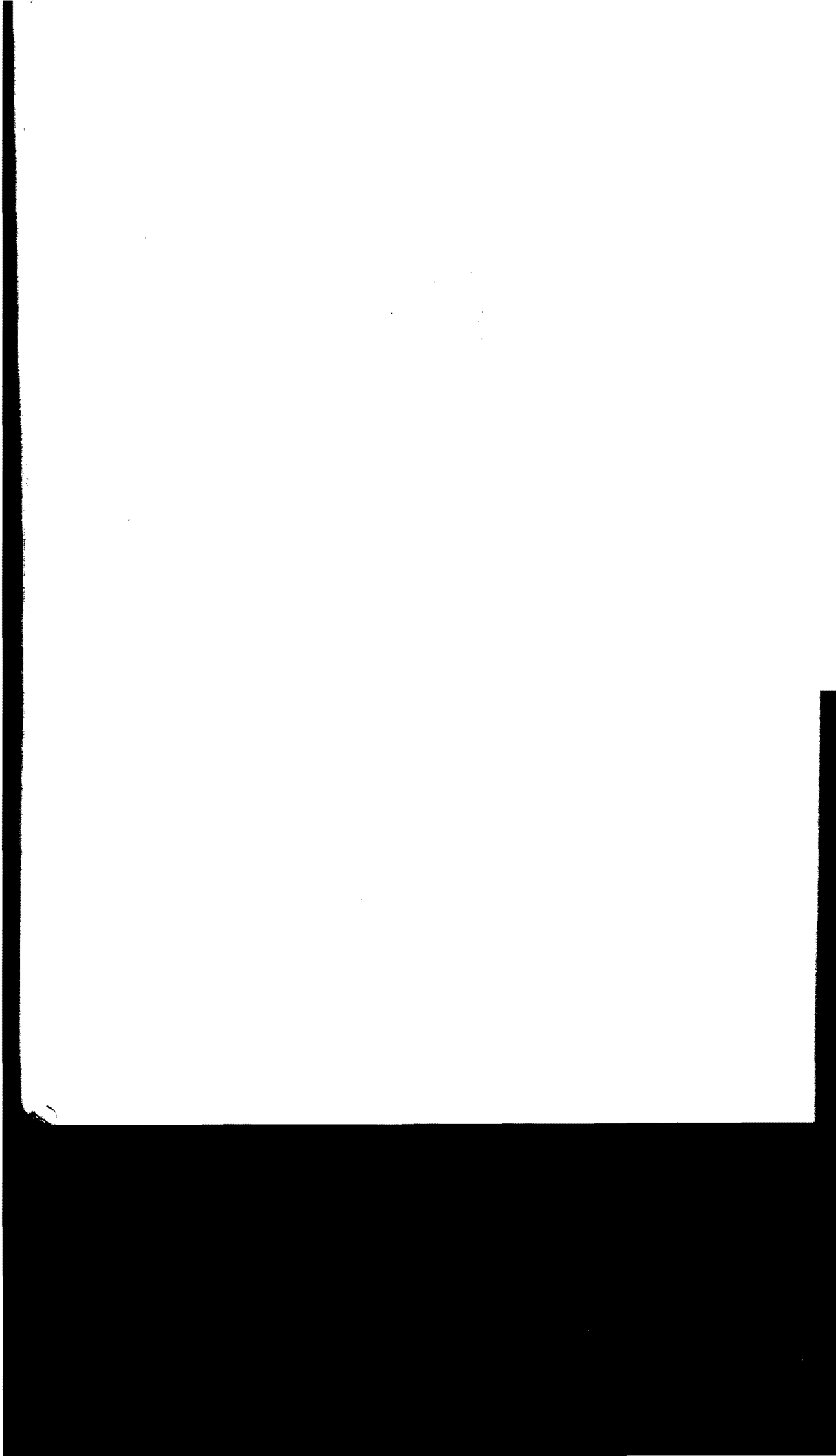
- No utilizar softphones en la medida de lo posible. Dado que los softphones son programas de escritorio, posibilitar su uso implicaría permitir el acceso de la red de datos a la red de VoIP, perdiendo así las ventajas logradas con la separación de redes.

- Mantener el firmware de los terminales actualizado. Es importante estar al día de los boletines de seguridad de los fabricantes, y en caso de anunciarse una vulnerabilidad realizar la actualización pertinente, para evitar posibles ataques.

- Encriptar el audio de las llamadas mediante SRTP o ZRTP. Esto todavía no es posible con Asterisk, pero debería estar disponible en breve.

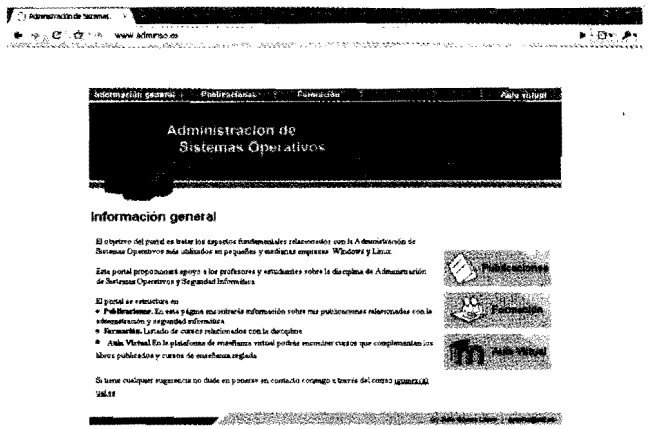
- Utilizar VPNs cifradas al realizar enlaces a través de Internet.

Además, es posible utilizar alguna herramienta de mitigación de ataques de denegación de servicio (DoS) como por ejemplo Snort.



PÁGINA WEB

Este libro dispone de su propia página web:



www.adminso.es

A través, de esta página web los usuarios que se registren, podrán acceder a un curso virtual realizado con moodle. Para poder acceder al curso, el sistema le pedirá la contraseña:

ZAPTEL

El curso, que complementa la obra, le permite ampliar sus conocimientos sobre redes inalámbricas a través de los siguientes materiales:

Software, máquinas virtuales y archivos de configuración referenciados a lo largo de la obra.

Conjunto de **presentaciones** y **herramientas de autoevaluación** para evaluar sus conocimientos.

Animaciones interactivas de las prácticas realizadas a lo largo de la obra.

ÍNDICE ALFABÉTICO

W

X

Z

Esta edición se terminó de imprimir en enero de 2009. Publicada por
por ALFAOMEGA GRUPO EDITOR, S.A. de C.V. Apartado Postal
73-267, 03311, México, D.F. La impresión y encuadernación se realizaron
en IMPRESIONES EDITORIALES FT, S.A de C.V. Calle 20 Mz. 105, Lt, 11,
Col. José López Portillo, Iztapalapa, 09920, México, D.F