Guía Asterisk:
Hacia la nueva telefonía

Alejandro Rios Peña
con Juan Manuel Coronado Z.

http://www.alerios.org/guia_asterisk

ISBN: 978-958-46-1283-0

Dedicatoria

Este libro se lo dedico a mi hijo Martín, quien estuvo en mi pensamiento en cada golpe de tecla, de cada página.

También se lo dedicamos a nuestro gran amigo Diego Andrés Asenjo González, quien se nos adelantó prematuramente al final de esta vida.

Contenido

Figuras

Tablas

Agradecimientos

Quiero agradecer especialmente a Juan Manuel por su paciencia, tesón y lealtad en todos los largos años que trabajamos juntos en nuestra empresa Avatar, así como a todos los grandes socios, colaboradores y clientes que creyeron en ella, y de los que aprendimos tanto.

Agradecemos también profundamente a todos nuestros profesores de la FIET, especialmente al Ing. Jaime Bados, cuya pasión en el aula de clase fue fue sin duda una inspiración mayúscula para nosotros, y al Ing. Álvaro Rendón, a quién citamos y parafraseamos en casi un capítulo completo.

Fueron extrañas las circunstancias que me llevaron a escribir este libro, a pesar de que la idea siempre estuvo rondando la cabeza, así que siento que debo agradecer al café de la Casa Museo del Maestro Fernando González – Otraparte, por servirme de refugio y de marco para llevar a cabo este proyecto.

Finalmente, quiero agradecer a mi familia por su apoyo y comprensión, especialmente a mi tía la Ing. Luz Elena Peña Herrera, quién siempre ha sido y será un ejemplo a seguir para mi.

Introducción

"Quien olvida su historia está condenado a repetirla"

Jorge Ruiz de Santayana- (poeta y filósofo Español)

Acerca de este libro y cómo se encuentra estructurado.

Este libro pretende ser, como su nombre lo indica, una guía de referencia para los lectores tanto del mundo de las telecomunicaciones que deseen conocer lo que ofrecen las aplicaciones de código abierto (software libre) como Asterisk [1] para la implementación de servicios de telefonía y Voz sobre IP soportados en plataformas computacionales, como para los que vienen del mundo de la informática y quieren profundizar sus conocimientos en el área de sistemas de telecomunicaciones, comprendiendo los fundamentos de la telefonía IP y su relación con la telefonía tradicional.

Está diseñado para ser una guía de consulta rápida sobre temas puntuales del diseño, instalación, configuración y administración de sistemas basados en Asterisk, a través de explicaciones sencillas pero con el suficiente bagaje teórico como para alcanzar un entendimiento global sobre la materia y sobre dónde continuar con la búsqueda de este conocimiento.

Así mismo, los talleres propuestos pretenden ayudar a implementar y profundizar los conocimientos adquiridos con la lectura del libro, todo esto abordado desde la experiencia de los autores, adquirida durante varios años de trabajo de campo alrededor de Asterisk y de otros proyectos de código abierto relacionados con él.

1 Asterisk es tanto el nombre del software, como del proyecto de código abierto que lo produce. Asterisk® y Digium ® son marcas registradas de la empresa Digium Inc. <http://www.asterisk.org>

Los ejemplos de código y otra información complementaria al libro se pueden encontrar en la página:

http://www.alerios.org/guia_asterisk

Algo de historia: la telefonía hasta nuestros días.

Desde su invención a finales del siglo XIX, la telefonía ha sido una de las tecnologías de telecomunicaciones que mayor impacto ha causado en la humanidad gracias a su capacidad de transmitir en vivo y atravesando grandes distancias la voz humana y, con ella, los sentimientos de las personas.

Pero no siempre todo giró alrededor del teléfono; antes de su invención en 1857 por Antonio Meucci, y de ser patentado y publicado por Alexander Bell en 1876, se dieron muchos otros hechos que dieron las bases para su desarrollo: desde los estudios teórico-matemáticos sobre la electricidad y el electromagnetismo del siglo XVIII y principios del XIX, que dieron origen a los conceptos de voltaje, corriente y campo electromagnético; pasando por los experimentos prácticos de diversos Físicos que inventaron el telégrafo y la red eléctrica, hasta los avances en materia de ingeniería que pusieron todas estas tecnologías al alcance del gran público a finales del siglo XIX; el genio de la humanidad tuvo que ponerse en evidencia para entender de tal manera la composición del mundo y aprovecharla para su mayor comodidad.

Es curioso cómo hoy en día a la mayoría de las personas les parece esencial el uso del teléfono, pero en sus primeros días no era así. Fue solo con el tiempo que la gente sintió que el telégrafo no era suficiente para llevar sus mensajes de un lado a otro, y se superó la dificultad de tener un cable directamente desde el teléfono origen hasta el de destino. Para no llenar las ciudades de toda una maraña de cables, donde cada persona debía tener una línea directa con cada teléfono con el que se quisiera comunicar, inicialmente surgió la idea de crear una red telefónica centralizada, a la que

cada usuario se conectara con una línea única, introduciendo el concepto de operadora telefónica, es decir una persona que cumplía la función de comunicar dos líneas telefónicas entre sí, conectando o "conmutando" los respectivos cables en un sitio central.

Posteriormente, ya entrado el siglo XX, estas operadoras fueron reemplazadas en su mayoría por sistemas automáticos de conmutación que recibían los números marcados por los usuarios, y que fueron poco a poco interconectándose entre sí, primero dentro de cada país, luego a nivel internacional, gracias a la estandarización y acuerdos orquestados por la UIT (Unión Internacional de Telecomunicaciones), y finalmente a nivel mundial, abarcando todo el planeta, gracias a la aparición del cable submarino y las comunicaciones inalámbricas. De aquí surge la Red Telefónica Pública Conmutada (RTPC[2]).

Siempre con el afán de mejorar la experiencia de los usuarios, optimizar el uso de los recursos de transmisión y hacer más eficiente la comunicación, la telefonía como industria fue avanzando poco a poco a la par de los desarrollos tecnológicos desde un mundo analógico hacia uno digital, desde los sistemas fijos hacia los inalámbricos, y fue así cómo el reinado del teléfono que alguna vez había relegado (rápidamente) al telégrafo, sólo se puso en discusión hasta la aparición del computador y de Internet, casi un siglo después.

Internet y la Voz sobre IP

Del mismo modo en que evolucionó la telefonía, rápido y como producto de una reacción en cadena, fue el desarrollo de la informática y la transmisión de datos, que dieron origen a dos de los más grandes desarrollos de la humanidad y del siglo XX: el Computador Personal e Internet, la red de redes.

La masificación de Internet y su creciente importancia como motor de una nueva sociedad con acceso a grandes cantidades de información, ha revolucionado la forma en que se comunican las personas, las ideas y el

2 En inglés: *Public Switching Telephone Network*, abreviada como PSTN

conocimiento. Sin embargo, la telefonía tradicional sigue siendo vigente y se ha intentado por todos los medios aprovechar su infraestructura para adaptarla a los nuevos retos que ha traído la revolución del Internet.

Ya desde los albores de la red de redes, en la década de los 70's, se hablaba de la posibilidad de transmitir la voz a través de la red de datos e integrar las redes de teléfono con el Internet[3], y la promesa de la convergencia entre éstas dos redes estaba por no cumplirse hasta que apareció la «Voz sobre IP» (VoIP). Esta tecnología, o mejor, familia de tecnologías, maduró y se estableció con el impulso del sector privado a la definición de una serie de estándares que permitieron, inicialmente, su uso tímido en los negocios y luego, su incursión creciente en el mercado internacional de las telecomunicaciones.

Pero... ¿Por qué se habla de dos términos diferentes: «Voz sobre IP» y «Telefonía IP»?

Por un lado, se conoce como Voz sobre IP o VoIP a la familia de protocolos estandarizados de comunicación de datos que permiten la transmisión de la voz humana a través del Protocolo de Internet (IP por sus siglas en inglés). Estos protocolos estandarizados definen la manera como se inician, mantienen y se finalizan las llamadas, especificando mecanismos para la adecuada captura, transmisión y recepción de la voz. De otro lado, la Telefonía IP es el uso práctico de la VoIP para complementar o reemplazar los servicios de la telefonía tradicional, y los nuevos servicios de voz que poco a poco van surgiendo a medida que se realiza la transición a las nuevas tecnologías y se da la correspondiente transformación socio-cultural que permite aprovecharlas.

El Software Libre

Sin embargo, el desarrollo de la Telefonía y Voz IP no se ha dado tan rápido como se esperaba. Durante mucho tiempo se criticó a la telefonía tradicional porque su tecnología de conmutación de circuitos no experimentó avances significativos durante casi medio siglo, fundamentalmente porque no

3 En 1973 se propuso el RFC 741, "Red experimental de Protocolo de Voz", inventado por ARPANET. <http://tools.ietf.org/html/rfc741>

existieron impulsores a la innovación interesados en cambiar el modelo de «voz» tradicional, ignorando por completo la potencialidad de las redes de datos y el crecimiento de Internet.

Mientras los grandes fabricantes de hardware de telefonía convencional han levantado muros a la innovación, la facilidad de uso y la reducción de costos, los servicios a través de Internet como el correo electrónico y las aplicaciones Web, han revolucionado el entorno empresarial debido en gran medida a la innovación constante en aplicaciones y servicios impulsada por los propios usuarios, basada en estándares y programas de código abierto que han contribuido a su elevada difusión y a unos costos altamente competitivos, que han permitido su rápida implantación en empresas y organizaciones de todo tipo[4].

Es así como la respuesta a la falta de innovación en la telefonía no provino de la industria de las telecomunicaciones, sino de la del Software, liderada principalmente por proyectos como Asterisk, que implementan un modelo completamente diferente, basado en el Software Libre, más al estilo de la revolución de Internet que al de la industria de las telecomunicaciones.

Asterisk: El futuro de la telefonía es abierto.

Técnicamente, Asterisk es una implementación en software de un sistema de telefonía avanzado, escrito en 1999 por Mark Spencer [5] y software libre distribuido bajo la licencia GNU GPL [6]. Es atractivo por que interactúa tanto con sistemas tradicionales de telefonía como con sistemas VoIP modernos y con sus muchas y avanzadas funcionalidades ha revolucionado el mercado de este tipo de soluciones.

Sin embrago, Asterisk no solo ha tenido éxito por el solo hecho de ser software libre. Uno de sus principales logros y razones que dieron origen a

4 Los programas de código abierto o software libre son un tipo de productos cuyo licenciamiento está orientado a la colaboración, otorgando a los usuarios libertad en el uso, copia, distribución y modificación de los programas, al contrario del software propietario o «cerrado», que restringe estas libertades para sus usuarios.

5 De la empresa Estadounidense Digium, Inc. <http://www.digium.com>

6 Pos sus siglas en inglés: Licencia Pública General del proyecto GNU, fundación abanderada del movimiento de Software Libre a nivel mundial. <http://www.gnu.org>

su popularidad, es que permite correr aplicaciones de telefonía sobre hardware relativamente económico: un computador personal (PC). Es decir, que permitió prescindir de hardware costoso como las interfaces de telefonía con DSPs[7] que eran la única alternativa en su momento, ya que el procesamiento de la voz se empezó a realizar por software en el servidor, aprovechando los grandes avances en materia de capacidad de CPU y RAM de los computadores modernos.

Hoy en día, trasegando ya el siglo XXI, Asterisk está consolidado como un proyecto maduro, con miles de usuarios y colaboradores alrededor del mundo, y cuyo aporte a la industria de las telecomunicaciones es indiscutible. Su rápida adopción lo convierten en una herramienta clave a dominar cuando se pretende tomar el camino no sólo hacia la nueva telefonía, sino hacia las nuevas comunicaciones unificadas o convergentes, que involucran además de la voz, el video, y la presencia, cualquier otra clase información que los usuarios quieran intercambiar, desde y hacia otra red o dispositivo.

Convenciones y Versiones Utilizadas

Se asume que el lector tiene conocimientos básicos de GNU/Linux, tanto de la instalación, como sistema de archivos, configuración de red y línea de comandos, así como nociones básicas de programación estructurada.

Todos los talleres incluidos en el libro están diseñados y probados sobre la distribución Ubuntu Server versión 10.4, basada en Debian GNU/Linux, pero pueden adaptarse y realizarse fácilmente en cualquier distribución moderna de GNU/Linux.

Cuando se presenta una instrucción de línea de comandos, se usa el siguiente formato de recuadro:

```
echo “esto es un comando”
```

7 Un Procesador de Señales Digitales (DSP por sus siglas en Inglés - *Digital Signal Processor*) es un tipo de micropocesador especializado en conversión y procesamiento de señales análogicas a digitales y viceversa.

Cuando se muestra el contenido de un archivo, se utiliza el siguiente formato de recuadro:

```
#!/bin/bash
#Script de ejemplo
echo “Esto un script o cualquier otro archivo”
```

Cuando se mencionan notas interesantes o puntos clave a tener en cuenta, se utiliza el siguiente tipo de recuadro para resaltarlas:

Nota importante.

Capítulo 1. Asterisk

> "No hay que empezar siempre por la noción primera de las cosas que se estudian, sino por aquello que puede facilitar el aprendizaje"
>
> Aristóteles (filósofo Griego)

¿Qué es Asterisk?

Asterisk fue inicialmente promocionado como la PBX[8] de código abierto, pero hoy en día sus creadores le dan la calificación no solo de PBX, sino de proyecto de telefonía, servidor de comunicaciones, kit de herramientas y hasta plataforma de desarrollo. Todas ellas son ciertas, puesto que Asterisk hace honor al origen de su nombre: el símbolo asterisco en el ámbito informático es un comodín que puede tomar múltiples valores.

Es por esto que hay que verlo de esa manera: Asterisk es un conjunto de piezas que pueden combinarse de varias formas para construir diferentes tipos de sistemas de telefonía. A lo largo de este libro se podrán encontrar las diferentes piezas y algunos ejemplos de su combinación para obtener los más comunes sistemas de telefonía de hoy en día. En el Capítulo 13 se podrá encontrar una guía para la realización de proyectos y diseños en escenarios reales utilizando Asterisk.

8 Acrónimo de Planta Telefónica o Centralita empresarial, por sus siglas en inglés (*Private Branch Exchange*).

¿Qué se puede construir con Asterisk?

Una PBX IP

La principal aplicación de Asterisk es una solución completa de PBX IP con avanzadas funcionalidades de comunicaciones y con la posibilidad de integrarse a escenarios de Comunicación Unificada[9]. Entre las principales características que presenta Asterisk para el montaje de una PBX están:

- Manejo básico de llamadas: extensiones, llamada en espera, transferencia, identificador de llamante, no molestar y redirección de llamadas.

- Manejo avanzado de llamadas: estacionamiento (parqueo) de llamadas, grupos de captura, colas de espera, directorio y conferencia.

- Plan de Marcado flexible y potente: se puede controlar exactamente cómo se manejan las llamadas entrantes y salientes de la PBX. Restricción por horarios, duración, origen, destino, entre otros muchos parámetros y según la lógica deseada.

- Manipulación avanzada de voz y video: Música en espera, transcodificación, grabación de llamadas y videollamadas.

- Aplicaciones de mensajería: Correo de voz con notificación al correo electrónico y correo de voz con notificación al correo electrónico, mensajes de texto cortos SMS.

- Aplicaciones administrativas: Consola de línea de comandos, herramientas de administración web, registros de llamadas.

9 Se refiere a los nuevos servicios de integración de telefonía, correo de voz, correo electrónico, mensajería instantánea, conferencia, video y fax (ver Capítulo 12).

IVR

Asterisk permite establecer completos IVR o sistema de Respuesta de Voz Interactiva (*Interactive Voice Response*, en inglés), donde al usuario llamante se le presenta un árbol de opciones pregrabadas con el que debe interactuar introduciendo tonos telefónicos (*Dual Tone Multi-Frequency* - DTMF) conforme se le van presentando las opciones.

Esto permite a las organizaciones extender los horarios de atención de un negocio determinado y disminuir las tareas de consultas telefónicas repetitivas a seres humanos, pudiendo incluso brindar información específica a partir de consultas realizadas a bases de datos. En los centros de llamadas son de mucha utilidad para segmentar a los usuarios que llaman y conducirlos al servicio o área específica que pueda darles el servicio preciso requerido, constituyéndose así en soluciones clave en los esquemas de atención al cliente

Entre las principales características que presenta Asterisk para el montaje de sistemas de IVR están:

- AGI (*Asterisk Gateway Interface*): API[10] independiente del lenguaje de programación que permite la manipulación de las llamadas desde un software externo, extendiendo el plan de marcado de la PBX al límite de la imaginación de los programadores.

- Posibilidad de integración con motores sintetizadores de habla (en inglés *Text-To-Speech*) para realizar funciones avanzadas como obtener información conectándose a Bases de Datos o sistemas de información y leerla al usuario con una voz artificial.

Centro de llamadas

Asterisk permite contar con una completa plataforma de comunicaciones y gestión para un Centro de Llamadas (*Call Center* en inglés), para lo cual incluye entre otras las siguientes características:

10 Biblioteca de funciones u objetos de programación disponible para ser utilizada por programas externos (*Application Programming Interface*).

- Encolamiento de llamadas, con música en espera y distribución entre los agentes de acuerdo a diferentes políticas o estrategias.

- Gestión de agentes: ingreso, salida, pausas y prioridades dentro de cada cola de atención.

- Supervisión en vivo de llamadas en curso: el supervisor puede escuchar solamente, o también dar indicaciones al agente sobre cómo debe entablar la conversación.

- AMI (*Asterisk Management Interface*): Interfaz de gestión remota para la integración telefonía-computador (CTI: *Computer Telephony Integration*), muy utilizada para obtener información el contacto con el que está hablando el agente y de esta manera, personalizar la atención prestada.

El Proyecto y su Ecosistema

Asterisk se ha mantenido como un exitoso proyecto de código abierto, gracias a una activa comunidad de desarrolladores que contribuyen tanto con el programa principal, como con todas las herramientas y aplicaciones relacionadas, que constituyen en su conjunto, todo un ecosistema de proyectos, empresas y contribuyentes independientes.

Tradicionalmente, el punto de encuentro por excelencia de la comunidad de Voz IP de código abierto ha sido el wiki de VoIP-Info.org[11], pero con el ánimo de tener más organización y control sobre los proyectos relacionados con Asterisk, Digium, la empresa que lidera su desarrollo, creó los siguientes instrumentos:

- AstriCon[12]: Congreso y feria de usuarios realizada anualmente y en donde se reúne la comunidad para mostrar las nuevas tendencias tecnológicas, realizar anuncios sobre nuevos productos, y para discutir el futuro del proyecto presencialmente.

11 <http://www.voip-info.org>
12 <http://www.astricon.net>

- AsteriskForge[13]: Sitio web para gestionar proyectos de software *open source* relacionados con Asterisk, a la manera de SourceForge.net, es decir, con herramientas como sistemas de control de versiones, gestor de incidencias (*bug tracker*), wiki, estadísticas de descargas, etc.

- AsteriskExchange[14]: Directorio web de las empresas proveedoras de soluciones relacionadas con Asterisk.

Desarrollo del proyecto y manejo de versiones

El desarrollo de Asterisk se gestiona mediante un Sistema de Control de Versiones, llamado Subversion o SVN, que permite llevar un registro y control sobre quién, dónde y cuándo se realizan cambios en el código fuente del programa.

Subversion utiliza un modelo de organización, con un tronco principal de desarrollo (*Trunk*), y varias ramas para las diferentes versiones (*Branches*). Por ejemplo, la versión 1.4 tiene una rama, de la cuál se desprenden ramas secundarias para cada publicación de la serie 1.4.X.

Diariamente, la versión de la rama de desarrollo (*Trunk*) está en constante cambio, ya que es allí donde se agregan las nuevas funcionalidades y donde se realizan los cambios fuertes en el código. Cada cierto tiempo, cuando los cambios se han estabilizado y se hace una publicación oficial, se toma una copia instantánea de la rama de desarrollo para crear una nueva rama asociada a la versión publicada.

Una vez que una nueva versión es publicada, ésta es soportada oficialmente por cierto tiempo, inicialmente con actualizaciones para todo tipo de fallos, y posteriormente sólo con actualizaciones de seguridad. Eventualmente, la versión llega al estado EOL (*End of life*) donde se descontinúa y ya no es soportada ni actualizada[15].

13 <http://forge.asterisk.org>
14 <http://www.asteriskexchange.com>
15 Asterisk Project Wiki. Asterisk Versions. [en línea].
<https://wiki.asterisk.org/wiki/display/AST/Asterisk+Versions> [citado en 22 de agosto de 2012].

Las versiones publicadas se clasifican de acuerdo al tiempo en el que estarán soportadas. Una versión de tipo LTS (*Long Term Support*) estará soportada oficialmente por cuatro años, con un año adicional de actualizaciones de seguridad. Las versiones «*Standard*» estarán soportadas durante un periodo variable de hasta un año, más un año adicional de actualizaciones de seguridad.

Serie	Tipo	Fecha de Publicación	Sólo Actualizaciones de Seguridad	Descontinuación
1.2.X		2005-11-21	2007-08-07	2010-11-21
1.4.X	LTS	2006-12-23	2011-04-21	2012-04-21
1.6.0.X	Standard	2008-10-01	2010-05-01	2010-10-01
1.6.1.X	Standard	2009-04-27	2010-05-01	2011-04-27
1.6.2.X	Standard	2009-12-18	2011-04-21	2012-04-21
1.8.X	LTS	2010-10-21	2014-10-21	2015-10-21
10.X	Standard	2011-12-15	2012-12-15	2013-12-15

Tabla 1.1: Programa de versiones en Asterisk

En versiones posteriores a la 1.8.X, se elimina el número uno precedente a la versión, es decir, que la siguiente versión LTS no será la ***1.11*** sino la ***11***, etc.

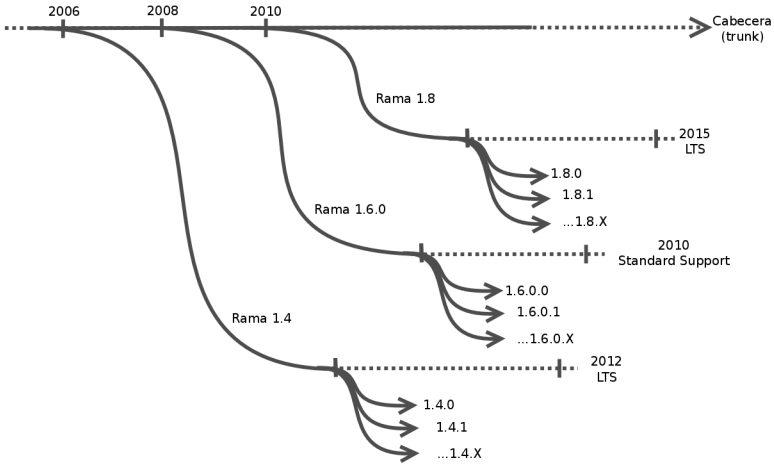


Figura 1.1: Manejo de versiones en Asterisk

Se recomienda utilizar siempre la versión más actualizada, si se quieren probar las últimas funcionalidades, o la última versión de tipo LTS, que estará soportada por más tiempo, y que es lo recomendado para sistemas en producción.

Otros proyectos afines y complementarios

Se han desarrollado muchas aplicaciones por parte de terceros que o bien facilitan la instalación y configuración de Asterisk para cualquier escenario posible, o que se especializan en un propósito específico, por ejemplo PBX, tarificación o *Call Center.*

La ventaja de estas aplicaciones es que empaquetan una configuración estándar, lista para usar, evitando la complejidad de tener que aprender la configuración de bajo nivel de Asterisk. Sin embargo, su principal desventaja es que se pierde la flexibilidad de adaptar Asterisk a escenarios muy específicos, así como la capacidad de conocer a fondo las entrañas del sistema para analizar y resolver los posibles problemas que se presenten.

En el Capítulo 10 se encontrará un listado de las herramientas y proyectos

más conocidos.

Documentación en línea

Asterisk es ya un proyecto muy maduro, con miles de usuarios y millones de descargas, de modo que es muy fácil encontrar documentación sobre todo tipo de configuraciones y de posibles problemas. A continuación se listan algunos de los repositorios de información más recomendados:

- Repositorio de Conocimiento de Digium (Digium's Knowledge Base): http://kb.digium.com/
- Centro de soporte de Digium:

  http://www.digium.com/en/supportcenter/
- Wiki de Voip-info: http://www.Voip-info.org
- Wiki oficial de Asterisk: http://wiki.asterisk.org
- Tutoriales de Asterisk Gurú:

  http://www.asteriskguru.com/tutorials/
- Videos de Asterikast: http://www.asterikast.com/
- Proyecto Asterisk Docs: http://www.asteriskdocs.org
- The Asterisk Book: http://www.the-asterisk-book.com

Soporte Comunitario

Listas de Correo

Las listas de correo son una de las herramientas de comunicación mas importantes en Internet, y su idea principal es la de poder ayudar y obtener ayuda en los temas de interés de la lista. En ocasiones las listas de correo hacen referencia a un sitio web para explicar el servicio que prestan. El sitio web de las listas relacionadas con Asterisk es: http://asterisk.org/support/mailing-lists

Otra forma de entender el funcionamiento de las listas del correo es la siguiente: Cuando una persona escribe pidiendo un cierto tipo de ayuda u opinión, su mensaje llega a todas las personas suscritas a la lista, quienes podrán responder por el mismo medio. Una lista de correo podría ser interpretada como una mesa redonda donde se discuten ciertos temas propuestos, pero en este caso, a través de Internet.

Las principales listas de correo de Asterisk para usuarios son:

- **asterisk-users:** Ayuda en general a usuarios.
- **asterisk-announce:** Anuncios de nuevos lanzamientos y eventos importantes.
- **asterisk-biz:** Discusiones sobre productos y servicios comerciales alrededor de Asterisk.

> **!** Si se sabe preguntar, los miembros de la lista responderán con gusto, si no, evitarán hacerlo, por eso es importante aprender las reglas de etiqueta de la respectiva comunidad. Para Asterisk, las reglas están en: http://asterisk.org/community/rules

Foros

Dos de los foros de ayuda más importantes relacionados con Asterisk son:

- http://forums.digium.com
- y http://forum.voxilla.com/asterisk-support-forum

Chat IRC

El IRC (*Internet Relay Chat*) es uno de los medios de comunicación en tiempo real más populares para interactuar con personas interesadas en algún tema de software libre.

El canal de ayuda a usuarios de Asterisk es #asterisk, puerto 6667, en la red de IRC FreeNode (irc.freenode.net). Para conectarse es necesario utilizar un cliente de IRC, como gaim o xchat, disponibles tanto para Linux como para Windows.

Es muy importante buscar primero en los archivos de las listas y en los foros las respuestas a los problemas encontrados, puesto que muchas personas pueden haberlos ya consultado y respondido antes, y las personas en el canal de IRC no contestarán preguntas que parezcan trilladas.

Arquitectura y Componentes

Desde el punto de vista de su arquitectura, Asterisk se define como un conector universal que enlaza protocolos de telefonía con servicios de telefonía. Es decir, que es posible conectar cualquier teléfono, línea, o paquete de voz a otra interfaz o servicio de destino con el fin de proveer la funcionalidad requerida.

La flexibilidad de Asterisk está dada por su diseño, ya que se compone esencialmente de un núcleo y varios tipos de módulos que pueden cargarse y

descargarse dinámicamente. Son los módulos los que hacen posible que Asterisk tenga gran cantidad de características, ya que cada día crecen en número y en funcionalidades de acuerdo al progreso y el desarrollo del proyecto [16].

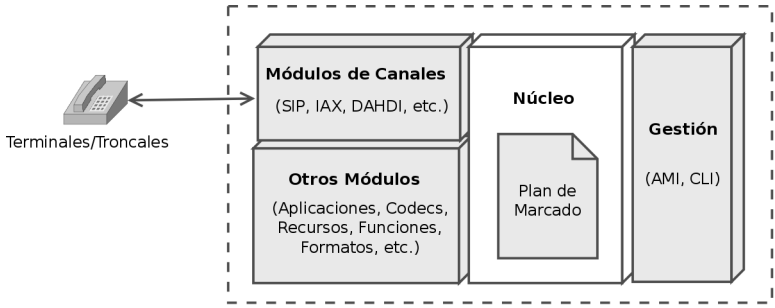


Figura 1.2: Arquitectura de Asterisk

Núcleo

Se encarga de la actividad principal de Asterisk: conectar las llamadas entre los diferentes usuarios y funciones de manera transparente, de acuerdo al plan de marcado definido por el administrador.

Módulos de Canales

Manejan el tipo de conexión, protocolo o tecnología a través de la cuál llegan o salen las llamadas. Un canal puede conectar a Asterisk con un terminal o con una troncal. Asterisk incluye módulos para manejar canales SIP, IAX2, Zaptel/DAHDI, MGCP, H323, entre otros.

16 Asterisk Project. Architecture.[en línea]. <http://www.asterisk.org/architecture> [citado en 14 de septiembre de 2011]

Módulos de Aplicaciones y Funciones

Permiten la ejecución de los diversos servicios (conferencia, correo de voz, decir la hora, etc.) y funciones (matemáticas, utilidades, etc.).

Módulos de traducción de Códecs

Cargan los diferentes códecs a utilizar. Asterisk soporta códecs como: GSM, Ley-µ, Ley-A, H.264, etc.

Módulos de formatos de audio

Manejan las tareas de lectura y escritura de diversos formatos en los que puede almacenarse el audio en el sistema, como WAV, GSM, etc.

Otros Módulos

Asterisk también provee módulos de Recursos, que definen el tipo de ubicación que se tendrá para la configuración del sistema y otras interfaces externas; módulos de CDR (Call Detail Records) para almacenar los registros de llamadas en diferentes ubicaciones, y otros módulos de configuración general de la PBX.

Terminales

Asterisk soporta diferentes tipos de terminales para que los usuarios se conecten y puedan hacer y recibir llamadas. A continuación se mencionan los terminales más frecuentemente utilizados.

Softphones SIP

Como su nombre lo indica, son programas cliente que se basan en la especificación del protocolo SIP (RFC3261)[17]. Se instalan en un computador o dispositivo móvil, y requieren que el usuario tenga una diadema (headset)

17 El protocolo SIP es cubierto con detalle en el Capítulo 6.

con auricular y micrófono.

Algunos ejemplos de los Softphones SIP más populares para usar con Asterisk son:

Fabricante	Softphone	Plataformas Soportadas
Counterpath	X-Lite, Bria, EyeBeam	Windows, Mac OS, Linux, Android, IPhone
Zoiper	Zoiper Classic, Zoiper Communicator	Windows, Mac OS, Linux, Solaris, Windows Mobile
Michel de Boer, distribuido bajo con licencia GPL	Twinkle	Linux
Ekiga.org, distribuido bajo licencia GPL	Ekiga	Windows(beta), Linux
SIP-Communicator.org, distribuido bajo licencia GPL	SIP-Communicator	Windows, Mac OS, Linux

Tabla 1.2: Softphones compatibles con Asterisk

Los Softphones son la opción más adecuada para usuarios que permanecen todo el tiempo frente al computador y con necesidades de comunicación altas, como agentes de servicio, de telemercadeo o de mesa de ayuda.

En general, no es una buena práctica asignarle a todos los usuarios Softphones solo porque pueda ser la opción más económica, ya que hay muchos usuarios para los cuales resulta más cómodo usar teléfonos IP (*hardware*).

Teléfonos IP SIP

Son clientes SIP implementados en Hardware en forma de teléfono convencional. En su mayoría se pueden configurar tanto por medio del teclado numérico y panel LCD, así como por medio de una interfaz web, lo

cuál facilita la gestión remota por parte del encargado del sistema de telefonía.

Algunos ejemplos de los teléfonos SIP más populares para usar con Asterisk son:

Fabricante	Modelo
Grandstream	Series BT y GXP.
Linksys/Cisco	Serie SPA.
Polycom	Serie SoundPoint.
Snom	Serie 3XX.

Tabla 1.3: Teléfonos IP compatibles con Asterisk

Los puntos clave a tener en cuenta a la hora de escoger un Teléfono IP son:

- *Número de Líneas:* algunos usuarios como la recepcionista requieren contestar varias llamadas al tiempo.

- *Número de puertos Ethernet:* cada teléfono es un equipo de red activo, que requiere su propio puerto de red. Muchos modelos vienen con doble puerto de red para hacer una derivación del puerto de red de un computador cercano, y de esta manera no tener que invertir en más switches y cableado estructurado.

- *Aprovisionamiento:* la mayoría de los fabricantes han diseñado mecanismos de aprovisionamiento para realizar la configuración automática de los teléfonos, pensando en escenarios donde hay decenas de equipos que deben configurarse y actualizarse periódicamente.

Gateways SIP con puertos FXS

Una pasarela o puerta de enlace (*Gateway* en Inglés), es un dispositivo que permite conectar redes distintas. Una *Gateway* SIP, permite conectar teléfonos análogos convencionales a redes SIP.

Al igual que con los Teléfonos IP, se deben tener en cuenta aspectos como el número de puertos de red y características como gestión Web y aprovisionamiento automatizado. Sin embargo, el parámetro principal a tener en cuenta es el número de puertos FXS (ver detalles en el Capítulo 5), que son los que permiten conectar los teléfonos análogos convencionales a través de conectores RJ11.

Son populares las presentaciones de uno (1) y dos (2) puertos FXS, las cuales son conocidas como ATA (Adaptador Telefónico Análogo, por sus siglas en inglés). Algunos ejemplos de Gateways SIP más populares para usar con Asterisk son:

Fabricante	Modelos	Número de puertos FXS
Cisco/Linksys	ATA series PAP2 y SPA2XXX	1 y 2
Cisco/Linksys	Serie 8XXX	4 y 8
Grandstream	Serie Handy-Tone	1 y 2
Grandstream	Serie GXW4XXX	4, 8 y 24
Audiocodes	Serie Media-Pack	2 a 24

Tabla 1.4: Gateways FXS compatibles con Asterisk

Generalmente se emplean Gateways cuando existe una inversión previa en un gran número de teléfonos analógicos convencionales, o cuando se requieren conectar máquinas de FAX o puertos de troncal (FXO) de plantas telefónicas PBX.

Softphones, ATAs y Teléfonos IP IAX

Al igual que para el protocolo SIP, existen en el mercado muchos programas y dispositivos para conectarse con Asterisk como un cliente a través del protocolo IAX[18]. Sin embargo, no existe mucha variedad, dado que el protocolo SIP es mucho más popular en la industria de la telefonía IP.

Tarjetas de telefonía con puertos FXS

Cuando nació Asterisk, el mercado de las Gateways y Teléfonos SIP no estaba muy desarrollado y eran equipos muy costosos. Esta fue una de las causas por las que las tarjetas de telefonía, que se podían conectar a cualquier computador con puertos PCI, disminuyendo el costo de implementación de sistemas con conexión a telefonía convencional, fueran una alternativa muy popular con Asterisk.

Hoy en día, puede resultar mucho más económico utilizar Gateways para proveer puertos FXS, pero hay escenarios en donde puede resultar más conveniente usar tarjetas de telefonía, bien sea para tener un mayor control sobre los puertos, o para disminuir los puntos de falla del sistema, como conexiones de red y fallas de configuración en las Gateways.

Para utilizar teléfonos convencionales conectados a puertos FXS en tarjetas de telefonía insertadas en el mismo equipo donde se corre Asterisk, se utiliza DAHDI (*Digium Asterisk Hardware Device Interface*), que es un conjunto de controladores (drivers) y herramientas para configurar y utilizar tarjetas de telefonía. DAHDI será cubierto con detalle en el Capítulo 5.

18 El protocolo IAX es cubierto con detalle en el Capítulo 6.

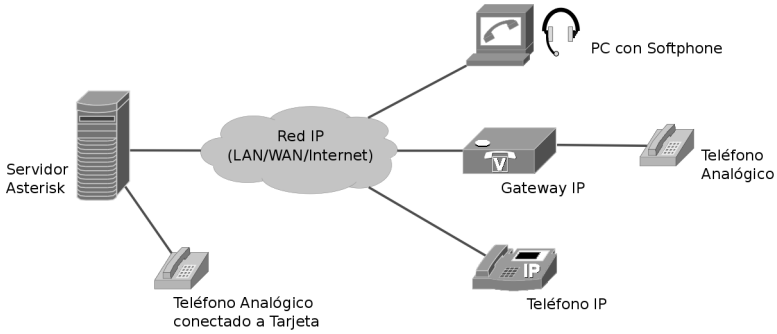


Figura 1.3: Tipos de terminales

Troncales

Una troncal es la conexión que tiene Asterisk para hacer y recibir llamadas hacia y desde el exterior. Puede ser a la red telefónica pública conmutada (RTPC), o a otra red o sistema de telefonía privada. A continuación se mencionan los tipos de troncal que se encuentran más frecuentemente.

Tarjetas de telefonía con puertos FXO

Al igual que con las tarjetas de puertos FXS para terminales, el uso de tarjetas para troncales fue el principal punto de entrada de Asterisk al mercado, pero poco a poco ha sido desplazado por troncales o gateways SIP.

Los puertos FXO permiten conectar Asterisk con líneas de telefonía analógica a través de conectores RJ11, por ejemplo, las líneas telefónicas residenciales. Los puertos digitales son generalmente de tipo E1/T1 (Capítulo 5), y se usan en conexiones de telefonía para empresas que requieren un mayor número de canales de comunicación.

Así como con las tarjetas de puertos FXS, o que combinan diferentes tipos de puertos, se requiere configurar DAHDI para su uso con Asterisk, el cuál

será cubierto con detalle en el Capítulo 5.

Algunos ejemplos de tarjetas de telefonía con puertos FXS, FXO y Digitales para usar con Asterisk son:

Fabricante	Modelos	Número de puertos
Digium	Series TDM (PCI) y AEX (PCI express)	Análogos: 4, 8, y 24, con módulos intercambiables FXS o FXO.
Digium	Series B (BRI) y TE (T1/E1/J1)	Digitales: 1, 2 y 4
Sangoma	Series B600, A200 y A400	Análogos: 2 a 24, modular FXS y FXO
Sangoma	Series A10X (T1/E1/J1) y A500 (BRI)	Digitales: 1, 2, 4 y 8
Rhino	R4FXO, R8FXX, R24FXX	FXO: 4 fijo, 8 y 24 modulares
Rhino	RXT1 (T1/E1/J1)	Digitales: 1, 2 y 4

Tabla 1.5: Tarjetas para conectar troncales tradicionales

De igual manera se han hecho populares soluciones como la de la serie Astribank de Xorcom, que en lugar de usar una tarjeta interna PCI, utiliza una conexión USB, pero que también requiere de DAHDI para su uso con Asterisk.

En general, a la hora de elegir una tarjeta de telefonía debe tenerse en cuenta la cantidad de puertos requeridos, y el tipo y capacidad de interfaces del servidor (ranuras PCI-X/PCI-Ex disponibles). Adicionalmente, para obtener una mejor calidad de sonido y menor procesamiento de la CPU, es preferible adquirir tarjetas que incluyan cancelación de eco por hardware.

Gateways SIP con puertos FXO, Digitales o Móviles

Al igual que en el caso de los terminales, una Gateway SIP con puertos FXO o Digitales permite conectar líneas telefónicas convencionales a redes SIP.

Algunos ejemplos de Gateways SIP con puertos FXO o digitales más populares para usar con Asterisk son:

Fabricante	Modelos	Número de puertos FXS
Grandstream	GXW410X	4-8
Audiocodes	MP11X	4-8
Quintum	AFTX00 - AXTXX00	2-24

Tabla 1.6: Gateways FXO compatibles con Asterisk

Tarjetas y *Gateways* para SIM cards móviles

Como era de esperarse, tan pronto fue posible establecer un puente entre las líneas de telefonía tradicional con las redes SIP, el siguiente paso fue buscar la interoperabilidad con las redes de telefonía móvil celular (principalmente GSM/UMTS).

En el mercado existen fabricantes que ofrecen tanto interfaces PCI-X para su instalación en un servidor Asterisk, como *Gateways* con compatibilidad para el protocolo SIP y de forma simultánea con los distintas bandas bajo las que operan las redes GSM/UMTS[19].

Dentro de algunos de los más representativos, se encuentran:

19 VoIP Wiki. GSM Gateways. [en linea] <http://www.voip-info.org/wiki/view/VOIP+GSM+Gateways> y <http://www.voip-info.org/wiki/view/GSM#PCIadapters> [citado en 2 de octubre de 2012]

Fabricante	Modelos	Canales GSM/VoIP
Junghanns (Interfaces PCI-X)	Uno/duo/quad-GSM	1-4
Topex (Gateways GSM/VoIP)	Mobilink - VoiBridge	1-4
2N Telecommunications (Gateways GSM/VoIP)	VoiceBlue - StartGate	4-256
Hypermedia (Gateways GSM/VoIP)	HG-4000	4-72

Tabla 1.7: Interfaces PCI-X y Gateways GSM/VoIP compatibles con Asterisk

Troncales SIP

Una de las formas más económicas y flexibles de conectar Asterisk con el exterior es a través de troncales SIP, ya que no se requiere de un hardware especializado que realice la conversión a Voz sobre IP.

Ya en muchos países hay proveedores de este tipo de líneas troncales y son conocidos como ITSP (*Internet Telephony Service Provider*). También los operadores tradicionales de telefonía han venido actualizando su infraestructura para ofrecer este tipo de troncales a la par de las troncales analógicas y digitales.

Las troncales SIP suelen usarse principalmente para realizar llamadas de salida a bajo costo o para recibir llamadas a través de Internet desde la red telefónica pública. Para este último caso, además de la troncal SIP se requiere un número para recibir las llamadas, el cuál es conocido como DID (*Direct Inward Dialing*).

El concepto de DID proviene de la telefonía digital, en donde a través de una misma troncal, por ejemplo un E1 de 30 canales, pueden recibirse llamadas dirigidas a diferentes números asignándole a cada uno, un código (comúnmente de 3 o 4 dígitos) que sirva para darle un tratamiento diferente al número de principal, al que suele asociarse un mensaje de bienvenida o un menú de audiorespuesta (IVR – *Interactive Voice Response*).

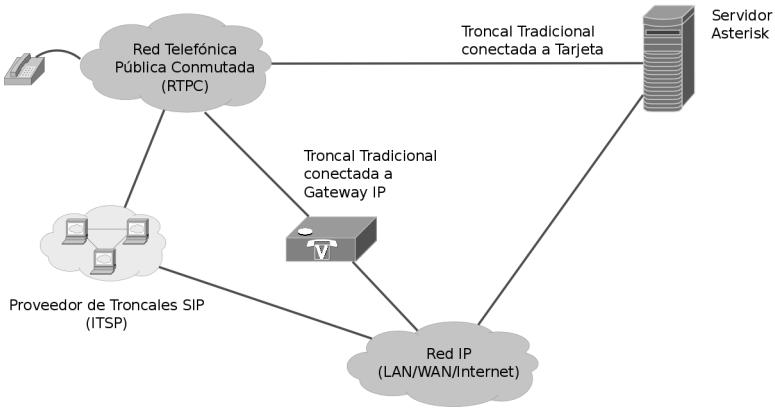


Figura 1.4: Tipos de troncales

Capítulo 2. Instalación y Primeros Pasos

> “El genio es uno por ciento de inspiración y un noventa y nueve por ciento de transpiración”
>
> Thomas Alva Edison (inventor Estadounidense)

¿Cómo y dónde obtener Asterisk?

Asterisk puede obtenerse desde diferentes fuentes:

- A partir de una distribución especializada en Asterisk como Elastix, Trixbox o AsteriskNow. Éstas incluyen pre-empaquetadas todas las dependencias requeridas por Asterisk, así como gran cantidad de herramientas de administración y programas relacionados.

- Usando los paquetes pre-compilados de la distribución GNU/Linux que se vaya a utilizar. Éstos hacen fácil de gestionar el manejo de dependencias y posteriores actualizaciones.

- Compilando el programa desde el código fuente. Es la opción más compleja, pero también la que permite controlar exactamente qué módulos se incluirán.

En este capítulo se realizará la instalación manual desde el código fuente, con el fin de explorar y entender cada uno de los componentes.

Asterisk está desarrollado principalmente en GNU/Linux para arquitecturas computacionales x/86, pero corre también en OpenBSD, FreeBSD y Mac

OS X. Como ya se mencionó en el capítulo de Introducción, en este libro se describe el proceso de instalación y configuración de Asterisk sobre la distribución Ubuntu Server versión 10-4, basada en Debian GNU/Linux.

Asterisk

El código fuente de Asterisk puede descargarse del sitio web oficial del proyecto[20], donde además se pueden encontrar otros componentes que suelen instalarse en compañía de Asterisk para contar con funcionalidades extra.

Se recomienda descargar los archivos de extensión TAR.GZ o TGZ (Tarballs comprimidos con GZip) de Asterisk, DAHDI y LibPRI, y ubicarlos en la carpeta "/usr/src" para mantener organizado el entorno de trabajo.

> **!** En las versiones 1.2 a 1.4, se descargaban también los paquetes asterisk-addons (módulos adicionales) y asterisk-sounds (sonidos pregrabados), pero a partir de la versión 1.6 los sonidos se pueden seleccionar y descargar cuando se hace la compilación, y a partir de la versión 1.8, los módulos adicionales vienen incluidos en el programa principal como opcionales.

DAHDI

DAHDI (*Digium Asterisk Hardware Device Interface* en Inglés), es un conjunto de módulos del kernel (*drivers*) y herramientas asociadas para controlar interfaces digitales y analógicas de telefonía.

DAHDI es independiente de Asterisk y puede ser utilizado por otros programas de telefonía. Anteriormente, era conocido con el nombre de Zaptel debido a su origen en el proyecto de telefonía Zapata.

Al descargar DAHDI, está la opción de un sólo paquete llamado "dahdi-linux-complete", o de bajar por separado el paquete de módulos específicos del kernel de Linux (dahdi-linux) y el paquete con las herramientas

20 <http://www.asterisk.org/downloads>

asociadas (dahdi-tools).

LibPRI

Es una librería que implementa la señalización requerida para interfaces de telefonía digital RDSI PRI (T1/E1/J1), la cuál será cubierta en el Capítulo 5. Su instalación es opcional, pero se recomienda llevarla a cabo con el fin de tener un sistema completo desde el principio.

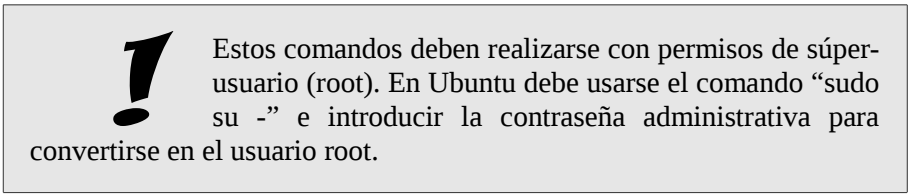
Requerimientos para la instalación

Asterisk requiere las siguientes librerías y sus cabeceras asociadas:

- ncurses, para la consola de comandos interactiva.
- openssl, la librería de sockets seguros.
- zlib, librería para compresión.
- newt, para las utilidades de DAHDI.
- curl, para interactuar con sitios Web.
- xml2, para la documentación incorporada.

En Ubuntu, las cabeceras de las librerías vienen en un paquete con el mismo nombre y el sufijo “-dev”, mientras en otras distribuciones suelen venir con el sufijo “-devel”. Por ejemplo, en Ubuntu 10.4 la instalación se hace con el gestor de paquetes APT, de la siguiente manera:

```
apt-get install libncurses5-dev zlib1g-dev libreadline-dev
libssl-dev libnewt-dev libcurl4-openssl-dev libxml2-dev
```


Estos comandos deben realizarse con permisos de súper-usuario (root). En Ubuntu debe usarse el comando “sudo su -” e introducir la contraseña administrativa para convertirse en el usuario root.

También se debe instalar el compilador para procesar el código fuente. Asterisk está escrito en el lenguaje de programación C y algunos módulos en C++, así que se recomienda utilizar el compilador GCC, que es el utilizado por los desarrolladores de Asterisk.

En Ubuntu, la manera más sencilla de instalar GCC y otras herramientas de compilación es instalando el paquete virtual “build-essential”:

```
apt-get install build-essential
```

Por último, se requieren las cabeceras del Kernel de Linux para compilar los controladores (drivers) de las tarjetas de telefonía (DAHDI):

```
apt-get install linux-headers-`uname -r`
```

Compilación e Instalación

En este libro se utilizan las siguientes versiones, donde la letra “X” puede reemplazarse con la versión exacta más actualizada:

- libpri-1.4.X.tar.gz
- dahdi-linux-complete-2.X.tar.gz
- asterisk-1.8.X.tar.gz

Asumiendo que estos archivos se han descargado y ubicado en la carpeta "/usr/src", se procede a descomprimirlos usando el comando "tar":

```
tar zxf libpri-1.4.X.tar.gz
tar zxf dahdi-linux-complete-2.X.tar.gz
tar zxf asterisk-1.8.X.tar.gz
```

Las opciones utilizadas con el comando "tar" indican lo siguiente:

- x: Descomprimir.
- z: El archivo está comprimido con gZIP.
- f: El nombre del archivo

La compilación debe hacerse en el siguiente orden: primero LibPRI, luego DAHDI y por último Asterisk.

Para compilar LibPRI, se ingresa al directorio que se creó al descomprimir el archivo de las fuentes:

```
cd libpri-1.4.X
```

Se ejecuta el comando de compilación "make", primero con la opción "clean" para limpiar posibles intentos anteriores de compilación, y luego con la opción "install" para construir e instalar el programa:

```
make clean
make install
cd ..
```

DAHDI se compila de la manera similar, pero agregando el comando “make” con la opción “all” para que se revisen las dependencias necesarias antes de compilar:

```
cd dahdi-linux-complete-2.X
make clean
make all
make install
```

Adicionalmente, DAHDI incluye una opción para instalar archivos de configuración y de arranque de ejemplo:

```
make config
cd ..
```

Para Asterisk, el proceso cambia un poco, ya que antes de compilar se ejecuta el script o guión de configuración, que revisará que todas las dependencias requeridas estén presentes en el sistema:

```
cd asterisk-1.8.X
./configure
```

Si se quiere tener control o revisar cuáles son los módulos que se van a compilar e instalar, se ejecuta el siguiente comando, que mostrará un menú navegable con el teclado y organizado por categorías:

```
make menuselect
```

En este menú se puede por ejemplo navegar a la opción “Core Sound Packages” y seleccionar “CORE-SOUNDS-ES-GSM”, para agregar los

sonidos pre-grabados en Español.

Si alguno de los módulos que interesan no se puede seleccionar y aparece marcado con XXX, es porque requiere una dependencia específica adicional, la cuál se muestra en la parte de abajo del menú. Simplemente se debe salir, instalar la librería correspondiente, y repetir los pasos desde el comando "./configure".

Al igual que con DAHDI, se puede instalar una configuración de ejemplo y scripts de arranque luego de compilar:

```
make
make install
make samples
make config
cd ..
```

Arranque de la aplicación

Antes de arrancar o inicializar Asterisk, se debe verificar que los módulos de DAHDI estén cargados. Para esto se puede utilizar el comando "lsmod", que muestra los módulos del kernel activos y filtrar la lista con el comando "grep":

```
lsmod | grep dahdi
```

La salida debe mostrar los módulos de DAHDI cargados. Por ejemplo:

```
dahdi_transcode          6765  1 wctc4xxp
dahdi_voicebus           45993  2 wctdm24xxp,wcte12xp
dahdi             213764  11 xpp,dahdi_transcode,
```

```
wcb4xxp,wctdm,wcfxo,wctdm24xxp,wcte11xp,wct1xxp,wcte12xp,dah
di_voicebus,wct4xxp
crc_ccitt               1675  2 wctdm24xxp,dahdi
```

Si no se muestra nada, es porque DAHDI no está cargado, entonces se puede arrancar con el siguiente comando:

```
/etc/init.d/dahdi start
```

Una vez verificado el estado de DAHDI, se puede verificar si Asterisk está corriendo, intentando conectarse a la consola de comandos (CLI), así:

```
asterisk -r
```

Si Asterisk está corriendo, se podrá ver la consola CLI:

```
*CLI>
```

Si Asterisk no está corriendo, se puede iniciar con el script de arranque:

```
/etc/init.d/asterisk start
```

A veces se requiere arrancar Asterisk directamente, y no como un servicio en el trasfondo, con el fin de encontrar problemas asociados al arranque y configuración. Esto se hace usando el comando “asterisk -c”, el cuál inicia el programa en modo de consola de línea de comandos CLI.

La Interfaz de Línea de Comandos (CLI)

Asterisk normalmente está corriendo en el trasfondo del sistema, como un

servicio o demonio. Iniciando la Consola o Interfaz de Línea de Comandos (CLI), se puede establecer una conexión con el programa para interactuar con él en tiempo de ejecución.

Esto se hace con invocando al comando “asterisk” con la opción “r”, que indica conectarse a un proceso de Asterisk que ya se encuentre corriendo. Adicionalmente, se puede agregar la opción “v” el número de veces deseado para indicar el nivel de detalle de la información que se mostrará (se recomienda usar al menos 3):

```
asterisk -rvvv
```

Todos los comandos de la CLI tienen la siguiente sintaxis:

```
*CLI> módulo acción parámetros
```

Por ejemplo, el siguiente comando muestra la ayuda incorporada de los comandos disponibles:

```
*CLI> core show help
```

Para salir de la CLI, se usa el comando “quit”:

```
*CLI> quit
```

Para detener el servicio de Asterisk inmediatamente, colgando todas las llamadas en curso:

```
*CLI> core stop now
```

Para dejar de recibir nuevas llamadas, y detener el servicio de Asterisk cuando las llamadas en curso terminen:

```
*CLI> core stop gracefully
```

Para continuar aceptando nuevas llamadas y detener el servicio de Asterisk cuando no haya ninguna llamada en curso:

```
*CLI> core stop when convenient
```

También existe la opción de reiniciar el servicio en lugar de detenerlo, con el comando “restart” y las mismas opciones del comando “stop”:

```
*CLI> core restart now
*CLI> core restart gracefully
*CLI> core restart when convenient
```

Dentro de la CLI se puede usar la tecla [Tab] para autocompletar y ver los comandos disponibles. También se puede usar la tecla de flecha arriba para repetir un comando anterior.

¿Qué se obtuvo?

Con la instalación de Asterisk y DAHDI, se crearon diferente archivos y directorios, de los cuáles se puede hacer un barrido para tener una idea general de qué se obtuvo.

Directorios

- **/etc/asterisk/**: Configuración de Asterisk.
- **/usr/bin/**,**/usr/sbin/**: Ejecutables y scripts de Asterisk.
- **/usr/lib/asterisk/**: Objetos binarios.
- **/usr/lib/asterisk/modules/**: Módulos para aplicaciones, canales, controladores, etc.

- **/usr/include/asterisk/**: Archivos de cabecera.
- **/var/lib/asterisk/**: Datos usados por Asterisk en tiempo de ejecución.
- **/var/lib/asterisk/keys/**: Llaves para autenticación RSA.
- **/var/lib/asterisk/mohmp3/**: Archivos música en espera MoH (Music on Hold en Inglés).
- **/var/lib/asterisk/sounds/**: Directorio de sonidos.
- **/var/run/asterisk/**: Identificador de los procesos del servicio de Asterisk (PID).
- **/var/spool/asterisk/outgoing/**: directorio para generar llamadas salientes con archivos ".call".
- **/var/spool/asterisk/voicemail/**: Buzones de voz de los usuarios.
- **/var/spool/asterisk/monitor/**: Archivos de llamadas grabadas.
- **/var/log/asterisk/**: Registro de Asterisk (eventos, detalle de llamadas - CDR, etc.).
- **/usr/share/asterisk/agi-bin**: donde deben ir las aplicaciones basadas en AGI (Asterisk Gateway Interface).

Archivos de Configuración

- **/etc/dahdi/system.conf:** Configuración de canales de DAHDI.
- **/etc/asterisk/asterisk.conf**: Configuración básica de los directorios de Asterisk y sus opciones de arranque.

- **/etc/asterisk/modules.conf**: Configuración de los módulos dinámicos de Asterisk a cargar en el arranque.

- **/etc/asterisk/extensions.conf**: Especifica la lógica del plan de marcado. Define qué hacer con los diferentes tipo de canal y las llamadas entrantes y salientes. Es el corazón de la PBX.

- **/etc/asterisk/*.conf**: Otros múltiples archivos de configuración, dependiendo de los módulos a utilizar.

El formato estándar de todos los archivos de configuración es el siguiente:

```
[sección_1]
parámetro_a=valor
parámetro_b=valor

; Comentario
[sección_n]
parámetro_a=valor   ; otro comentario
parámetro_c=valor

;-- Comentario en bloque
de varias líneas. --;
```

Los archivos están divididos en varias secciones, cuyo nombre no puede contener espacios, y debe estar encerrado entre corchetes.

Dentro de cada sección, se pueden asignar valores a diferentes parámetros de configuración. En general, esta asignación es independiente de la que se haga en otras secciones, pudiendo repetirse un parámetro dentro de otras secciones del mismo archivo.

Los comentarios se inician con punto y coma, y van hasta el final de la línea. Los comentarios en bloque inician con “;--” y terminan con “--;”.

Capítulo 3. Entendiendo y configurando Asterisk por primera vez

> "Dime y lo olvido, enséñame y lo recuerdo, involúcrame y lo aprendo"
>
> Benjamin Franklin (1706-1790) Estadista y científico estadounidense.

Flujo de una llamada en Asterisk

Antes de empezar a configurar Asterisk y hacer llamadas, es importante tener una idea de los pasos que componen el flujo de una llamada.

En el siguiente ejercicio, se asume que existen dos usuarios hipotéticos, cada uno con un tipo de teléfono diferente habilitado en Asterisk: Alicia, que tiene un teléfono SIP, y Carlos, que tiene un teléfono IAX.

El flujo de una llamada de Alicia hacia Carlos sería de la siguiente manera[21]:

1. Alicia levanta el teléfono y marca el número de extensión de Carlos, por ejemplo la 2002.

2. El teléfono de Alicia inicia una llamada usando el protocolo de comunicaciones SIP, la cuál es recibida por el módulo correspondiente en Asterisk.

3. Asterisk verifica que la cuenta SIP de Alicia esté autenticada para realizar llamadas, de lo contrario colgará la llamada.

21 Asterisk Project Wiki. Call Flow and Bridging Model. [en línea]. <https://wiki.asterisk.org/wiki/display/AST/Call+Flow+and+Bridging+Model> [citado en 22 de agosto de 2012].

4. En este punto, se establece un primer canal de comunicación (bajo el protocolo SIP) entre Alicia y Asterisk.

5. Ahora Asterisk buscará en el plan de marcado del sistema las instrucciones a realizar para la extensión 2002 solicitada por Alicia, encontrando que la instrucción es llamar al teléfono IAX de Carlos.

6. Asterisk genera un segundo canal, a través de su módulo IAX, para llamar al teléfono de Carlos.

7. Carlos contesta el teléfono.

8. En este punto, existen dos canales independientes: un canal entre Alicia y Asterisk, y otro entre Asterisk y Carlos. Asterisk realiza un "puente" (bridge) interno entre los dos canales, con el fin de establecer la llamada completamente.

9. La llamada continua establecida hasta que alguno de los dos usuarios cuelga, luego de lo cuál Asterisk cuelga el canal que quede abierto.

De esta manera es como Asterisk puede conectar llamadas desde cualquier protocolo y dispositivo hacia otro, siempre y cuando los dos estén soportados.

Configuración de Teléfonos SIP

Tal como se vio en el Capítulo 1, hay muchos tipos de terminales que se pueden utilizar como clientes del servidor Asterisk, pero los más populares son teléfonos o softphones que usen el protocolo SIP.

La configuración de todos los canales de tipo SIP, tanto de teléfonos como de troncales, se realiza en el archivo "**sip.conf**", dentro de la carpeta de configuración "**/etc/asterisk**". Al dar una mirada por el archivo de ejemplo que se copió al momento de la instalación (ver Capítulo 2.), se puede

apreciar que es bastante largo y está lleno de secciones y parámetros deshabilitados con comentarios, pero da una idea de las muchas posibles configuraciones que se pueden realizar.

Por el momento, se pueden dejar estas configuraciones de ejemplo sin modificaciones, y agregar al final del archivo la configuración de las cuentas SIP para los usuarios de ejemplo Alicia y Benito:

```
[alicia]                    ;El nombre de la cuenta.
type=friend                 ;El tipo de canal[22].
host=dynamic                ;Se permitirá el registro
                            ;desde IPs dinámicas.
secret=clave_secreta        ;Se debe cambiar por una
                            ;contraseña segura.
context=usuarios            ;La sección del archivo
                            ;"extensions.conf" donde se
                            ;procesarán las llamadas iniciadas
                            ;por este teléfono.
language=es                 ;para escuchar los sonidos
                            ;pre-grabados en Español

[benito]
type=friend
host=dynamic
secret=clave_secreta
context=usuarios
language=es
```

Luego de guardar los cambios en el archivo, se debe indicar a Asterisk que recargue la configuración del módulo SIP. Esto se puede hacer desde la consola CLI, con el siguiente comando:

22 Los tipos de canal se detallan en el Capítulo 6.

```
*CLI> sip reload
 Reloading SIP
```

Para verificar la correcta configuración de las cuentas SIP:

```
*CLI> sip show users
Username    Secret       Accountcode  Def.Context  ACL
     ForcerPort
benito      clave_secreta             usuarios     No     No
alicia      clave_secreta             usuarios     No     No
```

Para que un teléfono SIP se comunique a través de Asterisk, debe configurarse con los datos de la cuenta creada, luego de lo cuál intentará "registrarse" en Asterisk, autenticando su usuario y clave para ser habilitados para hacer y recibir llamadas.

Para verificar el estado de registro de las cuentas SIP, se usa el siguiente comando, donde se puede ver si el estado es "online" (conectado) u "offline" (desconectado):

```
*CLI> sip show peers
Name/username     Host                Dyn    Forcerport
     ACL    Port   Status
alicia            (Unspecified)D      0
     Unmonitored
benito            (Unspecified)D
     Unmonitored
2 sip peers [Monitored: 0 online, 0 offline Unmonitored: 0
online, 2 offline]
```

El lector es libre de realizar los siguientes pasos con los teléfonos SIP de su

preferencia. La configuración de cada tipo de teléfono está por fuera del alcance de este libro, pero en general todos los clientes SIP solicitan al menos los siguientes datos:

- *Servidor de Registro o Proxy SIP (SIP Registrar):* la dirección IP del servidor Asterisk.

- *Usuario o Autenticación SIP:* el nombre de usuario de la cuenta SIP, tal como se especificó en la sección correspondiente del archivo "sip.conf" (alicia y benito).

- *Clave o Contraseña:* la clave especificada para el usuario en la cuenta SIP.

Una vez realizada la configuración de los teléfonos, se puede verificar si se encuentran registrados, y desde qué equipo:

```
*CLI> sip show peers
Name/username     Host                Dyn    Forcerport
     ACL    Port   Status
alicia            192.168.0.5         D      0
     5060   Unmonitored
benito            192.168.0.6         D
     5060   Unmonitored
2 sip peers [Monitored: 0 online, 0 offline Unmonitored: 2
online, 0 offline]
```

De esta manera ya se tienen los teléfonos configurados y registrados, pero se requiere de un plan de marcado para que Asterisk sepa qué hacer cuando los teléfonos inician llamadas, y para enviarles llamadas cuando alguien quiera comunicarse con ellos.

Plan de Marcado

El plan de marcado (dialplan) es el centro en torno del cual se concibe cualquier solución de telefonía basada en Asterisk y consiste en un conjunto de instrucciones que seguirá Asterisk paso a paso para saber qué hacer con cada llamada que se genera hacia o desde el sistema y que se encarga de enlazar aplicaciones, líneas, extensiones, y servicios entre sí.

El plan de marcado se define en el archivo "**extensions.conf**", que se encuentra ubicado en la carpeta de configuración "**/etc/asterisk/**". Al igual que con el archivo "sip.conf", por el momento no se harán modificaciones a las configuraciones de ejemplo que se copiaron al momento de la instalación (ver Capítulo 2.).

Continuando con el ejemplo de Alicia y Benito, se debe agregar lo siguiente al final del archivo "extensions.conf":

```
[usuarios]   ; Contexto donde se procesan
             ;las llamadas de los usuarios
exten = 2000,1,Dial(SIP/alicia,25) ; Extensión para llamar a Alicia
exten = 2001,1,Dial(SIP/benito,25) ; Extensión para llamar a Benito
```

Estas instrucciones definen los pasos que se deben realizar cuando alguien marque la extensión 2000, para llamar a Alicia, y la extensión 2001, para llamar a Benito. Esta sintaxis se explicará en la siguiente sección.

Luego de guardar los cambios, se debe indicar a Asterisk que recarge la configuración del módulo "pbx_config", que es el que procesa el archivo "extensions.conf". Esto se puede hacer desde la consola CLI, con el siguiente comando:

```
*CLI> dialplan reload
```

Luego, se puede usar el siguiente comando para verificar que la configuración del contexto "usuarios" haya sido procesada correctamente:

```
*CLI> dialplan show usuarios
[ Context 'usuarios' created by 'pbx_config' ]
  '2000' => 1. Dial(SIP/alicia,25)    [pbx_config]
  '2001' => 1. Dial(SIP/benito,25)    [pbx_config]
-= 2 extensions (2 priorities) in 1 context. =-
```

En este punto ya se pueden realizar llamadas de prueba, por ejemplo, desde el teléfono de Alicia marcar la extensión 2001 para comunicarse con Benito. Es importante ir revisando todo el proceso que ocurre desde la CLI, para entender mejor la ejecución del plan de marcado:

```
*CLI>
     -- Executing [2001@usuarios:1] Dial("SIP/alicia-
00000006", "SIP/benito,25") in new stack
     -- Called benito
     -- SIP/benito-00000007 is ringing
     -- SIP/benito-00000007 answered SIP/alicia-00000006
     -- Remotely bridging SIP/alicia-00000006 and
SIP/benito-00000007
  == Spawn extension (usuarios, 2001, 1) exited non-zero on
'SIP/alicia-00000006'
```

Nótese como Asterisk asigna un identificador único para cada canal establecido en esta llamada. Para el canal entre Alicia y Asterisk, el canal es "*SIP/alicia-00000006*", y para la llamada entre Asterisk y Benito, el canal es "*SIP/benito-00000007*".

El Plan de Marcado consta de cuatro elementos principales: contextos, extensiones, prioridades y aplicaciones.

Contextos

Los contextos son las secciones en las que se divide el plan de marcado y se encuentran delimitados por palabras dentro de corchetes ([ ]). Estas palabras pueden ser una combinación de caracteres alfanuméricos (AaZz0-9), guiones (-) y guiones al piso (_). No se admiten espacios ni caracteres espaciales.

Un contexto termina donde empieza el siguiente. De este modo, se consideran extensiones integrantes de un contexto todas las que se encuentren debajo de éste en el archivo “extensions.conf”.

Los contextos también pueden considerarse agrupaciones de extensiones, y delimitan qué extensiones pueden interactuar con qué otras. En otra palabras una extensión dentro de un contexto determinado se encuentra aislada de extensiones definidas en otros contextos.

Así por ejemplo, dados los siguientes contextos:

```
[contexto_A]
extension_A1,...
extension_A2,...
extension_A3,...

[contexto_B]
extension_B1,...
extension_B2,...
extension_B3,...
```

Un usuario que inicie una llamada para procesarse en el contexto_A, sólo podrá marcar a las extensiones A1, A2 o A3, pero no a las extensiones B1, B2 y B3. Este aislamiento de extensiones es muy útil para definir permisos de marcación, como se verá más adelante.

Adicionalmente, un contexto puede enlazar a otros, con el fin de construir planes de marcado más complejos y organizados. Para esto se utiliza el parámetro “include”, así:

```
[contexto_A]
extension_A1,...
extension_A2,...
extension_A3,...
include => contexto_B

[contexto_B]
extension_B1,...
extension_B2,...
extension_B3,...
```

De esta manera, un usuario que inicie una llamada para procesarse en el contexto_A, ahora podrá marcar no solo a las extensiones A1, A2 y A3, sino también a las extensiones B1, B2 y B3.

Extensiones

Como se vio anteriormente, una extensión se define dentro de un contexto y consiste en un conjunto de instrucciones generadas tras el evento de una llamada entrante o unos dígitos pulsados dentro de un canal de voz, que Asterisk seguirá para darle un determinado tratamiento a esa llamada.

De este modo, las extensiones trazan el camino que seguirá una llamada dentro del sistema a través del plan de marcado.

Una extensión se define por la palabra reservada “exten” seguida del signo igual (=) o los signos igual y mayor qué (=>), y se compone de tres (3) elementos separados por comas:

```
exten => nombre,prioridad,aplicación(parámetros)
```

- Un nombre, número o patrón de marcación.
- Una prioridad que define múltiples pasos a seguir dentro de una misma extensión y en qué orden se deben seguir éstos.
- Una aplicación que ejecuta una acción específica sobre la llamada activa en la extensión, y que puede tener sus propios parámetros (separados por comas).

Ejemplo:

```
; Reproducir el archivo “hello-world”
; cuando se marque la extensión 123:
exten => 123,1,Playback(hello-world)
```

En este orden de ideas, las extensiones pueden indicar timbrar a un teléfono, consultar un correo de voz, ingresar a una sala de conferencia o realizar una llamada a través de una línea troncal.

Prioridades

Cada extensión tiene múltiples pasos. El orden en el que se ejecutan estos pasos es su prioridad. Cada prioridad se encuentra enumerada secuencialmente comenzando por 1.

El siguiente ejemplo muestra un esquema de prioridades secuencial donde al marcar «123», se ejecutan diferentes acciones sobre la llamada:

```
exten => 123,1,Acción A
exten => 123,2,Acción B
exten => 123,3,Acción C
```

La prioridad especial “n” o «siguiente» (en inglés *next*), permite obviar las secuencias y evita tener que enumerar manualmente las prioridades cada vez

que se introduce un cambio en éstas. Por ejemplo:

```
exten => 123,1,Acción A
exten => 123,n,Acción B
exten => 123,n,Acción C
```

Adicionalmente, la prioridad n, permite manejar etiquetas, facilitando con esto la lectura del plan de marcado y los saltos entre prioridades cuando se requiera:

```
exten => 123,1,Acción A
exten => 123,n(etiqueta1),Acción B
exten => 123,n,Acción C
```

Aplicaciones

Las aplicaciones son los caballitos de batalla del plan de marcado. Como su nombre lo indica, se encargan de ejecutar distintas acciones disponibles en Asterisk para manipular las llamadas: reproducir un sonido, contestar un canal, colgar un canal, capturar unos tonos, etc.

La mayoría de las aplicaciones aceptan parámetros para configurar su comportamiento, algunos opcionales y otros obligatorios. Estos parámetros deben ir separados por comas, por ejemplo:

```
exten => 123,1,Aplicación(parámetro1,parámetro2,parámetro3)
```

Se puede obtener el listado completo de aplicaciones cargadas en el sistema (con una breve descripción en inglés) introduciendo el siguiente comando en la CLI:

```
*CLI> core show applications
```

Poco a poco en este libro se irán presentando algunas de las aplicaciones más frecuentemente utilizadas.

Primeras Aplicaciones

Answer

Esta aplicación permite contestar una llamada entrante por un canal, si no se ha contestado antes. Si el canal ya se ha contestado, no tiene ningún efecto sobre la llamada.

Para obtener la ayuda detallada de cualquier aplicación, se puede utilizar el siguiente comando en la CLI, con el nombre de la aplicación correspondiente:

```
*CLI> core show application Answer
```

La aplicación Answer se utiliza en situaciones donde se requiere reproducir un mensaje al usuario, antes de contestar, con el fin de que no se cobre la llamada. Por ejemplo, decirle que la persona llamada no está disponible y que puede dejarle un mensaje de voz en el buzón o colgar. Esto se conoce como “flujo multimedia anticipado” (Early-Media en Inglés), característica permitida dependiendo del marco legal del país respectivo, y solo disponible en troncales VoIP o digitales.

En general, es un buen hábito hacer un llamado a la aplicación Answer antes de reproducir cualquier mensaje al usuario llamante, especialmente si se espera que introduzca dígitos.

Playback

Esta aplicación reproduce archivos de sonido disponibles en formatos compatibles con Asterisk. La sintaxis es la siguiente:

```
Playback(archivo|opciones)
```

Se pueden concatenar varios archivos usando el signo “&”. Por ejemplo:

```
Playback(archivo1&archivo2&archivo3|opciones)
```

El archivo a reproducir debe estar ubicado dentro del directorio "/var/lib/asterisk/sounds/", o en la sub-carpeta correspondiente al idioma en el que se está ejecutando la llamada[23]. También se puede dar la ruta absoluta (completa) al archivo, si se encuentra en otra ubicación del sistema de archivos.

No es necesario especificar la extensión con el formato del archivo, ya que Asterisk utilizará el formato disponible que sea más eficiente de acuerdo al CODEC que utilice la llamada.

Wait

Espera el número de segundos que se pasa como argumento. Los segundos pueden especificarse con fracciones, por ejemplo "1.5" esperará un segundo y medio.

Sintaxis:

```
Wait(segundos)
```

Hangup

Esta aplicación cuelga explícitamente un canal que se encuentre activo. Asterisk cuelga automáticamente las llamadas cuando se termina la última prioridad definida en el plan de marcado, lo cuál se conoce como "auto-falltrough", pero se recomienda hacer uso explícito de la aplicación Hangup como la última prioridad en todas las extensiones.

Asterisk Habla

Combinando todo lo visto en este capítulo, se puede crear la extensión 123 para contestar una llamada, escuchar un mensaje de bienvenida y luego

23 Ver Capítulo 4 para más información sobre lo referente a sonidos en Asterisk.

colgar. Se puede ubicar en un nuevo contexto “servicios”, luego del contexto “usuarios” creado previamente:

```
[usuarios]
exten = 2000,1,Dial(SIP/alicia,25)
exten = 2001,1,Dial(SIP/benito,25)
include => servicios

[servicios]
; Extensión 123 – Escuchar un mensaje y colgar
exten => 123,1,Answer()
exten => 123,n,Playback(hello-world)
exten => 123,n,Wait(2)
exten => 123,n,Hangup()
```

Nótese que se agregó una línea “*include => servicios*” en el contexto “*usuarios*”, para permitir que tanto Alicia como Benito puedan marcar la extensión 123 del contexto “*servicios*”, tal como se mencionó en el apartado referente a los contextos.

Luego de ejecutar el comando “dialplan reload” en la CLI para recargar la configuración, y realizar la llamada, se puede verificar la ejecución correcta del plan de marcado:

```
*CLI>
 -- Executing [123@usuarios:1] Answer(“SIP/alicia-00000010”, “”) in new stack
   -- Executing [123@usuarios:2] Playback(“SIP/alicia-00000010”, “hello-world”) in new stack
```

```
   --   <SIP/alicia-00000010>   Playing   'hello-world.gsm'
(language 'es')
  -- Executing [123@usuarios:3] Wait(“SIP/alicia-00000010”,
“2”) in new stack
 -- Executing [123@usuarios:4] Hangup(“SIP/alicia-00000010”,
“”) in new stack
== Spawn extension (usuarios, 123, 4) exited non-zero on
'SIP/alicia-00000010'
```

Capítulo 4. Añadiendo funcionalidad a la PBX

> “Si añades un poco a lo poco y lo haces así con frecuencia, pronto llegará a ser mucho”
>
> Hesíodo (S. VIII AC-?) Poeta griego.

Marcando al interior y al exterior: La aplicación Dial

Dial es probablemente la aplicación más importante del plan de marcado. Intenta establecer una llamada saliente con un canal de salida y lo enlaza al canal origen a través de un puente interno (bridge).

Sintaxis:

```
Dial(tecnología/recurso..., tiempo_limite, opciones, URL)
```

- *Tecnología/recurso*: Se refiere al tipo de protocolo o tecnología utilizada para la marcación, por ejemplo SIP, IAX, DAHDI, etc., y el recurso o canal sobre el cuál se va a hacer la marcación. Se pueden especificar varios, concatenados con el signo “&” para hacer marcación en paralelo.

- *Tiempo límite*: Especifica el número de segundos durante los que se debe intentar la marcación antes de pasar a la siguiente prioridad del plan de marcado. Si no se especifica, por omisión toma un valor de 136 años.

- *Opciones*: La aplicación Dial tiene una gran cantidad de opciones, que pueden consultarse haciendo uso del comando "*core show application Dial*" en la CLI.

- *URL*: Envía una URL al teléfono llamado. No todos los teléfonos soportan esta característica.

Ejemplo: Marcar a un canal SIP:

```
exten => 123,n,Dial(SIP/alicia)
```

Ejemplo: Marcar a un canal DAHDI:

```
exten => 123,n,Dial(DAHDI/1)
```

Ejemplo: marcar simultáneamente en los canales DAHDI 1 y SIP alicia:

```
exten => 123,n,Dial(DAHDI/1&SIP/alicia)
```

Ejemplo: marcar al número 1234567 por una troncal SIP:

```
exten => 123,n,Dial(SIP/troncal/1234567)
```

Dial salta a la prioridad siguiente si el canal de destino no pudo ser llamado o si el tiempo límite expira.

El correo de Voz

Asterisk soporta, entre otras características de correo de voz (*VoiceMail* en Inglés), las siguientes:

- Buzones protegidos por contraseña
- Saludos personalizados
- Reenvío de mensajes
- Notificación del correo de voz por correo electrónico.

La definición de los buzones de correo de voz se realiza en el archivo "**voicemail.conf**", dentro de la carpeta de configuración "**/etc/asterisk**".

Sintaxis de la configuración:

```
buzón=> clave,nombre_completo[,email[,opciones]]
```

- *Buzón*: Es el número del buzón para correos de voz.
- *Clave*: una contraseña numérica para consultar el buzón.
- *Nombre Completo*: Nombre del propietario del buzón, para ponerlo como destinatario en las notificaciones enviadas por correo electrónico
- *E-Mail*: Para notificación del correo de voz por correo electrónico.

El archivo se divide en contextos por cada grupo de buzones, los cuales

permiten poder separar y organizar los buzones para una mayor flexibilidad. Siguiendo con el ejemplo de Alicia y Benito, así sería la configuración de sus buzones de voz:

```
[default] ; Contexto por omisión
2000=>1234,Alicia,alicia@ejemplo.com
2001=>5678,Benito,benito@ejemplo.com
```

Luego de guardar los cambios en el archivo, se debe indicar a Asterisk que recarge la configuración del módulo de VoiceMail. Esto se puede hacer desde la consola CLI, con el siguiente comando:

```
*CLI> voicemail reload
Reloading voicemail configuration...
```

Algunos teléfonos soportan una función para alertar al usuario cuando se tienen correos de voz pendientes de leer en el buzón. Para que el teléfono pueda suscribirse al buzón que se creó en el paso anterior, se debe agregar el parámetro "*mailbox*" a la cuenta correspondiente en el archivo "**sip.conf**", de la siguiente manera:

```
[alicia]
type=friend
host=dynamic
secret=clave_secreta
context=usuarios
mailbox=2000@default        ; Número_de_buzón@contexto_de_buzón

[benito]
type=friend
host=dynamic
```

```
secret=clave_secreta
context=usuarios
mailbox=2001@default
```

Luego de guardar los cambios en el archivo, se debe indicar a Asterisk que recargue la configuración del módulo SIP con el comando “sip reload” en la consola CLI.

Algunos cambios realizados en las cuentas SIP sólo tomarán efecto la próxima vez que el teléfono se registre.

Las aplicaciones que permiten configurar un servicio de correo de voz completo en Asterisk son **VoiceMail**, para dejar mensajes, y **VoiceMailMain**, para consultarlos.

Sintaxis de la utilización de VoiceMail:

```
VoiceMail(buzon[@contexto][&buzon[@contexto]][...][|opciones])
```

Las principales opciones son:

- b - reproducir el mensaje de ocupado del buzón.
- u - reproducir el mensaje de no-disponible del buzón.
- s - saltar instrucciones.

Por ejemplo, para enviar a un buzón cuando no contesten Alicia y Benito, el archivo “extensions.conf” se debe modificar de la siguiente manera:

```
[usuarios]
exten = 2000,1,Dial(SIP/alicia,25)
exten = 2000,n,VoiceMail(2000@default)

exten = 2001,1,Dial(SIP/benito,25)
exten = 2001,n,VoiceMail(2001@default)
```

Para consultar el buzón se utiliza la aplicación VoiceMailMain que, si no se especifican parámetros, preguntará el número del buzón y la contraseña:

```
[servicios]
exten => 90,1,VoiceMailMain()
```

Luego de guardar los cambios en el archivo, se debe indicar a Asterisk que recarge la configuración del plan de marcado con el comando “dialplan reload” en la consola CLI.

En este punto se puede marcar a cualquiera de los teléfonos y no contestar la llamada para dejar un mensaje de voz. Luego se puede consultar marcando la extensión 90 e indicando el número de buzón y la clave cuando el sistema lo solicita.

En el CLI, se puede utilizar el siguiente comando para monitorear el estado de los buzones de los usuarios:

```
*CLI> voicemail show users
Context    Mbox  User                      Zone       NewMsg
default    2000  Alicia                                   0
default    2001  Benito                                   1
```

```
2 voicemail users configured.
```

También llamando a la extensión de VoiceMailMain se pueden realizar diferentes operaciones sobre el buzón, como grabar el saludo personal, los mensajes de “ocupado” y “no disponible”, mover los correos de una carpeta a otra, etc.

Operadora automática

Una operadora o recepcionista automática es el tipo de IVR[24] más sencillo. Normalmente se implementa su funcionamiento de la siguiente manera:

- Las llamadas que provienen del exterior de la PBX son dirigidas al contexto del plan de marcado correspondiente a la operadora automática, por ejemplo “menu_bienvenida”

- Se reproduce el archivo de sonido con las opciones del menú y se da la opción al usuario llamante para que marque dígitos o tonos (DTMF).

- Los dígitos introducidos por el usuario son procesados dentro del contexto “menu_bienvenida”.

- Generalmente se incluye otro contexto que es el que finalmente tiene las extensiones que se ejecutarán cuando el usuario presione una secuencia de dígitos válida, por ejemplo: “usuarios”.

A continuación se irán introduciendo otras aplicaciones que se requieren para implementar esta operadora automática.

Goto

La aplicación Goto permite saltar al contexto, extensión y prioridad indicados.

24 Ver la definición de IVR en el Capítulo 1.

Sintaxis:

```
Goto([[contexto,]extensión],prioridad)
```

Si se omite el contexto, se asume que el salto debe realizarse en el contexto actual. Si también se omite la extensión, se asume que se debe saltar a la prioridad indicada de la extensión actual.

También se pueden realizar saltos a las etiquetas especificadas en prioridades de tipo “n”. Retomando el ejemplo de la aplicación Playback del Capítulo 1, se puede usar Goto de la siguiente manera:

```
[servicios]
exten => 123,1,Answer()
exten => 123,n(hola),Playback(hello-world)
exten => 123,n,Wait(2)
exten => 123,n,Goto(hola)   ; Esto crea un bucle infinito hasta
                            ; que el usuario cuelgue la llamada.
exten => 123,n,Hangup()
```

La instrucción “Goto(hola)” del ejemplo, podría reemplazarse con cualquiera de las siguientes:

- “Goto(servicios,123,2)”
- “Goto(123,2)”
- “Goto(2)”

Background y WaitExten

La aplicación Background, de manera similar a Playback, reproduce archivos de sonido, pero acepta tonos marcados durante la reproducción. Es decir, que mientras el sonido se reproduce, el usuario puede marcar una extensión para saltar dentro del plan de marcado.

Sintaxis:

```
Background(archivo[&archivo2...]|opciones)
```

Generalmente se suele usar en conjunto con WaitExten, que permite esperar dígitos durante una determinada cantidad de segundos, con lo cuál se puede aumentar el tiempo que tiene el usuario para marcar la extensión.

Sintaxis:

```
WaitExten(segundos)
```

A continuación un ejemplo de la aplicación Background donde el usuario puede marcar el dígito 1 o el dígito 2 cuando escucha el mensaje de saludo:

```
[menu_bienvenida]
exten => s,1,Answer()       ; Se comienza en una extensión “s”.
exten => s,n,Background(hello-world); Reproducir y esperar tonos.
exten => s,n,WaitExten(2)          ; Esperar tonos 2 segundos más.

exten => 1,1,Playback(digits/1)    ; Extensión 1, reproducir sonido
exten => 1,n,Goto(s,1)      ; Saltar de nuevo a la extensión “s”.

exten => 2,1,Playback(digits/2)    ; Extensión 2, reproducir sonido
exten => 2,n,Goto(s,1)      ; Saltar de nuevo a la extensión “s”.
```

Para probar este IVR desde la PBX de ejemplo que se ha ido construyendo, se debe permitir la marcación a los teléfonos de Alicia y Benito, agregando la extensión 3000 al contexto “usuarios”:

```
exten => 3000,1,Goto(menu_bienvenida,s,1)
```

El usuario puede marcar cualquiera de las extensiones que se han definido en el contexto de la operadora automática, pero también puede marcar una

extensión equivocada, que no existe, o simplemente no marcar nada. Asterisk cuenta con unas extensiones especiales para manejar estos casos.

Extensiones especiales

Las extensiones estándar más usadas son:

- **s** (Start): Se ejecuta cuando se recibe una llamada dentro de un contexto sin un destino específico, generalmente desde troncales de entrada.
- **t** (Timeout): Se ejecuta cuando se termina el tiempo máximo para que el usuario digite una extensión después de un mensaje.
- **T** (Absolute Timeout): Se ejecuta cuando se termina el tiempo máximo de duración total permitido para una llamada.
- **h** (Hangup): Se ejecuta al colgarse una llamada, permitiendo realizar algunas tareas cuando el canal ya no está disponible.
- **i** (Invalid): Se ejecuta cuando se digita una opción que no se encuentra definida en el contexto de ejecución.

De esta manera, la operadora automática de ejemplo se vuelve más robusta al tener en cuenta los casos especiales:

```
[menu_bienvenida]
exten => s,1,Answer()
exten => s,n,Background(hello-world)
exten => s,n,WaitExten(2)

exten => 1,1,Playback(digits/1)
exten => 1,n,Goto(s,1)
```

```
exten => 2,1,Playback(digits/2)
exten => 2,n,Goto(s,1)

exten => t,1,Playback(digits/0)    ; Si no se presiona nada
exten => t,n,Playback(vm-pls-try-again) ; se reproduce un mensaje
exten => t,n,Goto(s,1)      ; y se devuelve a la extensión "s".

exten => i,1,Playback(invalid)     ; Si se marca una extensión
exten => i,n,Goto(s,1)             ; inválida, se reproduce un
                                   ; mensaje y se devuelve a "s".
```

Sonidos Personalizados

Hasta ahora se han realizado todos los ejemplo utilizando los sonidos incorporados en Asterisk, que se seleccionaron en el momento de la instalación (Capítulo 2), pero en la práctica se requieren reproducir mensajes personalizados de acuerdo al propósito de la PBX.

El directorio de sonidos de Asterisk "/var/lib/asterisk/sounds/" a su vez contiene subdirectorios por cada idioma bajo el cuál se quieran reproducir tales sonidos (Ej: "es" para Español, "en" para Inglés, etc.) y otros subdirectorios como:

- "*silence*": archivos de silencio de N segundos.
- "*digits*": números, horas, y días del mes.
- "*letters*": letras del abecedario y algunos caracteres.
- "*phonetic*": Se usa para decir un determinado caracter o deletrear una palabra con elementos fonéticos (alfa, charlie, bravo, etc.).

Para reproducir mensajes personalizados en Asterisk se tienen dos

alternativas: copiar a la carpeta de sonidos el archivo grabado previamente en un software de edición de audio especializado o usar la aplicación Record de Asterisk para grabar el mensaje usando un teléfono.

Cuando se utiliza un programa de edición de audio, se deben guardar o convertir a un formato adecuado para su reproducción en Asterisk. En general, para aplicaciones telefónicas se requiere que los sonidos sean monofónicos y que se encuentren muestreados a 8khz.

La aplicación Record permite grabar sonidos en los formatos soportados por Asterisk (generalmente WAV o GSM) para ser usados luego en alguna porción del plan de marcado.

Sintaxis:

```
Record(archivo.formato|silencio|max|opciones)
```

Para conocer los formatos de sonido que soporta Asterisk, se puede ejecutar el siguiente comando en la CLI:

```
*CLI> core show file formats
```

Por ejemplo, para grabar un mensaje llamado “bienvenida” en formato GSM:

```
[servicios]
exten => 91,1,Answer()
exten => 91,n,Wait(1)
exten => 91,n,Record(bienvenida.gsm)
exten => 91,n,Wait(1)
exten => 91,n,Playback(bienvenida)
exten => 91,n,Hangup()
```

Al marcar a la extensión 91, sonará un pitido (beep) y se podrá empezar a grabar el mensaje. Para terminar la grabación, se puede presionar la tecla “#”

o simplemente colgar.

Combinando todo lo visto, el IVR de ejemplo de operadora automática quedaría de la siguiente manera:

```
[menu_bienvenida]
exten => s,1,Answer()
exten => s,n,Background(bienvenida)
exten => s,n,WaitExten(2)

exten => 1,1,Playback(digits/1)
exten => 1,n,Goto(s,1)

exten => 2,1,Playback(digits/2)
exten => 2,n,Goto(s,1)

exten => t,1,Playback(digits/0)
exten => t,n,Playback(vm-pls-try-again)
exten => t,n,Goto(s,1)

exten => i,1,Playback(invalid)
exten => i,n,Goto(s,1)
```

Capítulo 5. Interacción con Redes de Telefonía tradicional

> "Aprende a resolver todos los problemas que ya hayan sido resueltos"
>
> Richard P. Feynman (1918 - 1988), físico estadounidense

La Red Telefónica Tradicional

RTPC (PSTN)

RTPC (Red Telefónica Pública Conmutada) o PSTN, por sus siglas en inglés, es como se conoce a la totalidad de elementos de telecomunicación que permiten la interconexión de los millones de teléfonos dispersos por todo el mundo, desde la central telefónica de un operador fijo o móvil de su ciudad, pasando por las centrales de tránsito nacionales, hasta los cables interoceánicos y otras conexiones internacionales.

Figura 5.1: Concepto de la RTPC

Para establecer una comunicación entre dos abonados o teléfonos, la RTPC establece temporalmente un circuito que va de extremo a extremo de la red. Este procedimiento, conocido como "Conmutación", es lo que permite establecer los caminos de comunicación dinámicamente, sin necesidad de

tener conexiones directas entre todos los teléfonos del mundo, lo cuál sería imposible.

En los albores de la telefonía, en algunas ciudades se llegaron a tender redes de cables para comunicar teléfonos directamente entre sí, creando una maraña, que unida a la red de cables de electricidad, hacía un uso muy ineficiente de recursos.

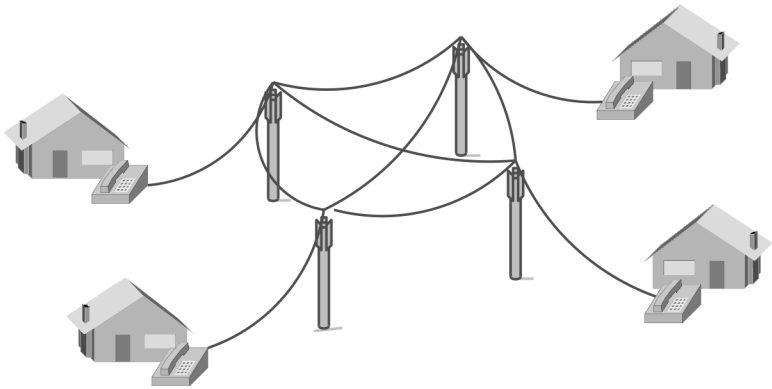


Figura 5.2: Necesidad de la conmutación telefónica

Rápidamente, se generaron desarrollos para optimizar la forma en que dos abonados telefónicos se pudieran conectar entre sí, dando paso los conceptos de operadora y central telefónica.

Con la creciente demanda de aparatos telefónicos, se instalaron centrales a las cuales se conectaban todas las líneas de abonado. Las personas realizaban primero una llamada a la central telefónica, donde la operadora contestaba solicitando el número de destino, luego de lo cuál procedía a conectar las dos líneas a través de cables intercambiables sobre diferentes clavijas. De esta manera quedaba manualmente establecido el circuito telefónico, que al finalizar la llamada era liberado nuevamente por la operadora.

Posteriormente se diseñaron centrales telefónicas electromecánicas, que

automatizaban este proceso. Los teléfonos incluían ahora un disco de marcación, con el cuál indicaban el número de destino a través de pulsos eléctricos enviados a la central. La central realizaba la conmutación a través de selectores mecánicos rotatorios y relés.

Finalmente, con el desarrollo de la electrónica digital, se diseñaron las centrales telefónicas actuales, que convierten las señales analógicas del teléfono a señales digitales, las cuales a su vez pueden ser transportadas con mayor fidelidad a mayores distancias, y cuyo proceso de conmutación está completamente automatizado.

Centrales telefónicas

Las centrales telefónicas digitales o híbridas, junto a los pares de cobre que llegan finalmente a los abonados, son el corazón de la RTPC. Las central local se encarga de conectar los abonados llamante y llamado directamente, si se encuentran bajo su jurisdicción, o de solicitar la conexión a otra central si el abonado de destino se encuentra en otro lugar.

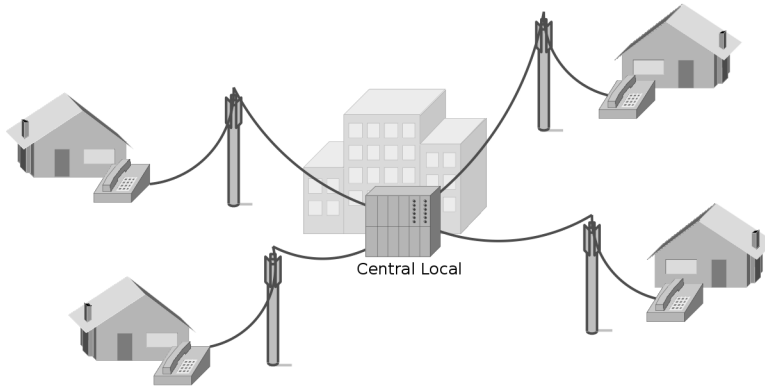


Figura 5.3: Central telefónica de área local

Con el fin de compartir entre los diferentes abonados locales la conexión que existe entre una central y otra, se utilizan técnicas para repartir el canal a través de modulación y multiplexación de las señales digitales, como se verá

más adelante. De esta manera se puede utilizar una sola línea para establecer varios circuitos de comunicación simultáneamente, optimizando los recursos.

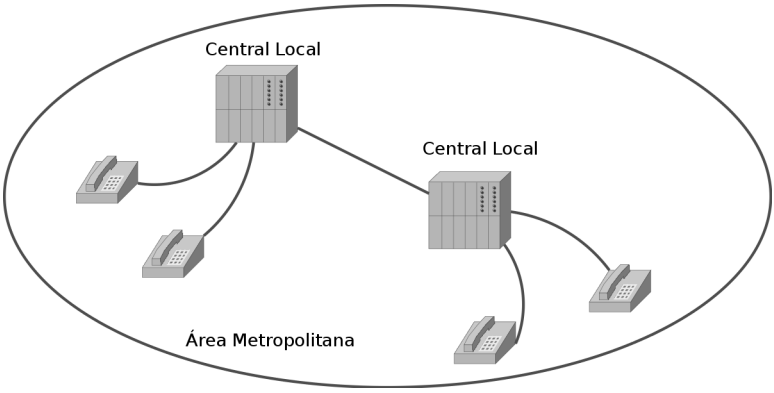


Figura 5.4: Conexión entre centrales locales

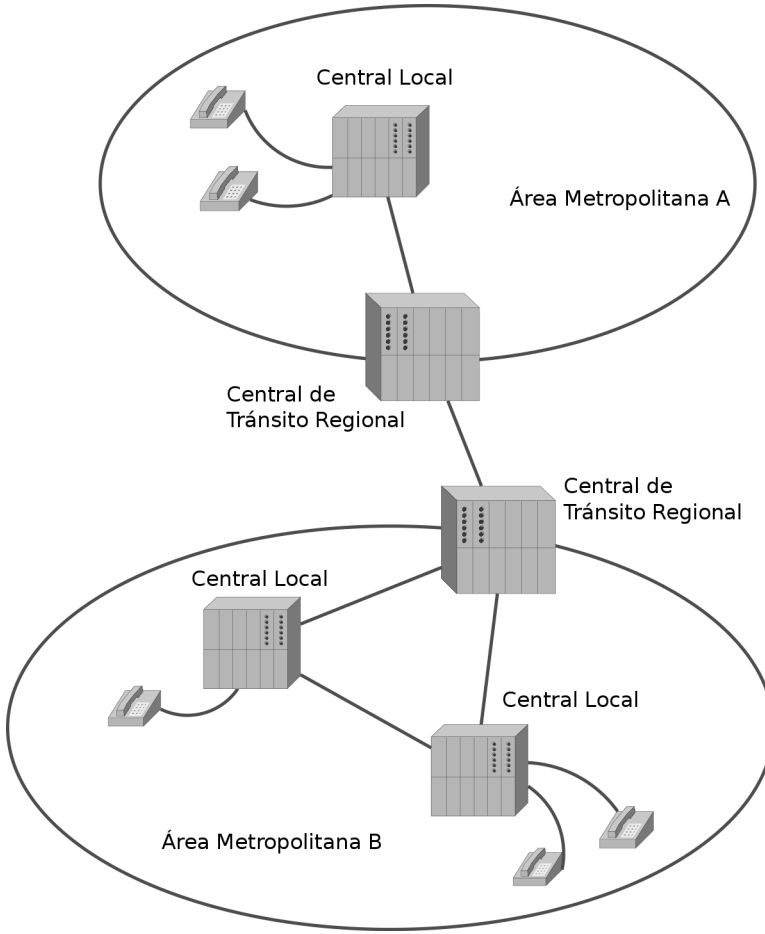


Figura 5.5: Conexión entre centrales de diferente orden

PBX

Las PBX (*Private Branch eXchange*) tradicionales son sistemas que permiten tomar las líneas que tenga contratada una empresa con su operador de telefonía y compartirlas entre un gran número de extensiones distribuidas

por todas sus instalaciones. Son una especie de pequeña central telefónica local privada.

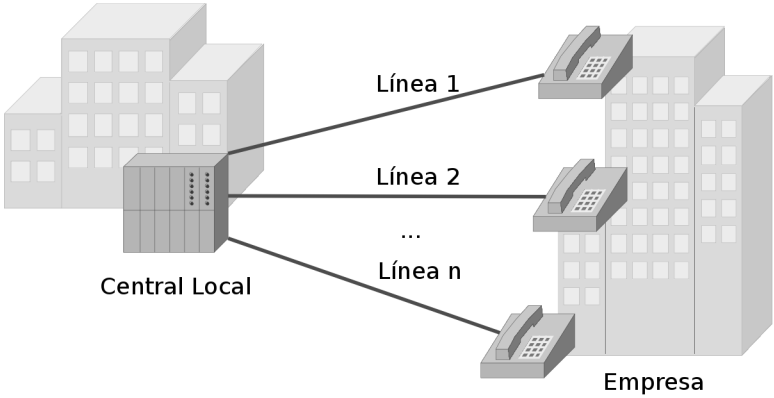


Figura 5.6: Una empresa sin PBX requiere de un mayor número de líneas telefónicas.

La PBX suele ser propiedad de la empresa, lo cuál permite adaptarla rápidamente a las necesidades específicas de su negocio.

Las PBX son conmutadores telefónicos, de tal forma que para realizar llamadas entre extensiones no es necesario interactuar con la RTPC. La PBX sólo enruta llamadas hacia la RTPC si éstas están dirigidas a destinatarios que no se encuentran dentro de la empresa.

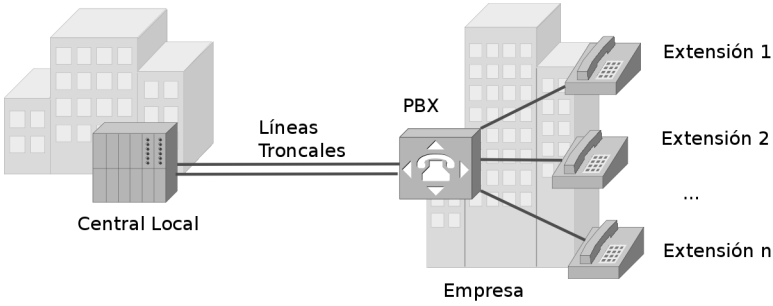


Figura 5.7: Con una PBX, las líneas telefónicas pueden compartirse entre muchas extensiones internas.

Telefonía Analógica

Por telefonía analógica se entiende al uso de redes de telecomunicaciones para el transporte a distancia de la voz, utilizando equipos y medios de transmisión que emplean señales analógicas, es decir, señales eléctricas que siguen fielmente la información de las ondas sonoras capturadas por el teléfono.

El Teléfono

Un teléfono convencional (a menudo también llamado terminal, aparato o abonado telefónico) se compone esencialmente de dos circuitos: el de conversación o voz y el de marcación. Los dos circuitos se alimentan a través de la línea telefónica empleando los 48 voltios que ésta suministra.

Circuito de conversación

El circuito de conversación se divide en tres partes: micrófono, auricular y bobina híbrida.

El micrófono (generalmente fabricado a base de gránulos de carbón comprimidos) proporciona una corriente eléctrica que varía en respuesta a las ondas de presión acústicas producidas por la voz.

Las variaciones en la corriente eléctrica resultantes son transmitidas a lo largo de la línea telefónica a la central. En el mismo orden, las señales eléctricas provenientes de la central serán enviadas a la bocina y convertidas en señales audibles al oído humano.

Tanto las señales que se transmiten como las que se reciben ocupan 2 hilos de cobre cada una, pero ambas comparten el mismo medio de transmisión, una línea de un solo par de hilos de cobre. La bobina híbrida, se encarga de balancear los dos tipos de señales a la línea, mediante un transformador con tres embobinados y una impedancia de 600Ω.

Circuito de marcación

Existen dos clases de circuitos dependiendo del método empleado para transmitir información de marcación a la central, que puede ser dećadica o por tonos:

Circuito de marcación electromecánico (marcación decádica)

Se compone de un disco mecánico giratorio provisto de diez agujeros enumerados del 0 al 9 que cuando retrocede, acciona un interruptor el número de veces que corresponda a cada dígito, generando una secuencia de pulsos. Así pues, al discar el cero se generan 10 pulsos; al discar el nueve, nueve; el ocho, ocho; y así sucesivamente hasta el uno que genera un solo pulso. A este tipo de marcación se le denomina decádica por pulsos.

En la actualidad, las centrales telefónicas modernas siguen aceptando este tipo de marcación, incluyendo Asterisk, pero estos terminales han caído en desuso para ser reemplazados por otros que emplean un tipo de marcación por tonos duales de multi-frecuencia (DTMF por sus siglas en inglés).

Circuito de marcación por tonos - DTMF

Se compone de un teclado numérico, por lo general de tres (3) columnas por cuatro (4) filas, en donde cada fila y columna tiene un valor determinado de frecuencia que se suman al oprimir un dígito, a este sistema se le denomina DTMF (*Dual-Tone Multi-Frequency* por sus siglas en inglés) o de tonos duales de multi-frecuencia.

La siguiente son las frecuencias que encontramos en un teclado DTMF estándar:

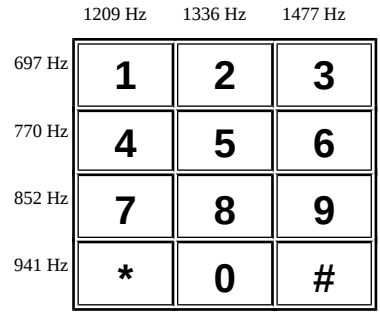


Figura 5.8: Teclado DTMF

Por ejemplo, al marcar 4, se enviará un tono con la combinación de las frecuencias de 770 Hz y 1209 Hz.

El envío de tonos compuestos de frecuencias distintas en lugar de una sola, permite disminuir los errores en la detección dígitos y aumentar la confiabilidad en la transmisión de dígitos entre el abonado y la central. Es además más veloz que la marcación decádica.

Señalización analógica

La señalización es el proceso de generación y procesamiento de la información requerida para establecer, mantener y terminar una conexión dentro de un sistema telefónico.

En señalización analógica, los dos extremos de la comunicación se definen como puertos FXS y FXO:

FXS

Un puerto FXS - Estación de Intercambio Extranjero (en inglés Foreign Exchange Station) es una interfaz de telefonía que proporciona potencia de alimentación eléctrica, envío de tonos de señalización y voltaje de timbre.

Un teléfono convencional se conecta a una interfaz FXS para recibir servicio

telefónico de un operador.

FXO

Un puerto FXO - Oficina de Intercambio Extranjero (en inglés Foreign Exchange Office), es una interfaz de telefonía que recibe telefonía pública básica conmutada. Esta interfaz se encarga de generar los indicadores de cuelgue y descuelgue empleados para señalizar la apertura y cierre del bucle en el extremo FXS de un circuito.

Los teléfonos analógicos convencionales, máquinas de FAX y MODEMs son dispositivos con puertos FXO, al igual que los puertos de una planta telefónica PBX que se conectan a las líneas provistas por el operador de telefonía pública básica conmutada.

A través del tiempo se han definido distintos tipos de señalización para conectar los puertos FXS y FXO en telefonía analógica, de los cuales los más importantes y soportados por Asterisk son el *Loop start*, y *Kewl start*:

Loop start

Loop start es un tipo de señalización provista por un teléfono o una centralita como respuesta al cierre del bucle en un circuito.

Cuando el FXS necesita alertar al usuario de la línea sobre una llamada entrante, se superpone una señal de corriente alterna AC sobre la misma, con una frecuencia especial, la cuál origina el timbre del teléfono.

Cuando el teléfono es descolgado (en inglés Off-hook), el bucle de abonado se cierra sobre la línea, indicándole al extremo FXS que se prepare para la comunicación. Para finalizar la llamada, el circuito se abre y el voltaje sobre el bucle de abonado retorna a su valor inicial, dejando la línea disponible de nuevo.

Kewl start

Es un tipo de señalización analógica derivada de *Loop start*. En esta señalización, además de la operación proporcionada por *Loop start*, la central telefónica le indica al terminal del usuario que el extremo remoto colgó, desconectando momentáneamente la corriente en la línea. La mayoría de las centrales telefónicas actuales soportan esta característica, usualmente requerida para tener confirmación de cuelgue.

Usando puertos FXS/FXO en Asterisk

Como se vio en el Capítulo 1, Asterisk puede interactuar con terminales o troncales analógicas mediante tarjetas de telefonía con puertos FXS o FXO, respectivamente.

Para configurar y utilizar tarjetas analógicas compatibles con Asterisk, se utiliza DAHDI, cuya instalación se cubre en el Capítulo 2.

Configuración de DAHDI

DAHDI es un módulo del núcleo de Linux que presenta una abstracción entre los controladores de hardware y el módulo chan_dahdi de Asterisk.

El archivo /etc/dahdi/system.conf es donde se configuran los parámetros específicos requeridos por una interfaz DAHDI determinada. Los controladores de los dispositivos permiten la comunicación directa con el hardware y el paso de información entre DAHDI y el hardware de telefonía.

Los parámetros de configuración que se especifican en el archivo /etc/dahdi/system.conf son:

- Un número de identificación único para cada tarjeta de telefonía que será usada en el plan de marcado para referenciar un canal o grupo de canales determinado.

- El tipo de señalización que usará cada tarjeta de telefonía.

- El idioma de los tonos a usar asociados con determinada tarjeta de telefonía (español, inglés, francés, etc). Estos tonos deben ser consistentes con el estándar nacional en el que se vaya a implementar la solución de telefonía.

Un archivo /etc/dahdi/system.conf tiene la siguiente estructura típica para canales analógicos:

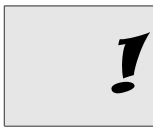
```
loadzone=es          # Tonos de indicación de España.
defaultzone=es       # Zona de los tonos por omisión.

fxoks=1      # Señalización FXO Kewl start para el
             # canal No. 1, que es un puerto de tipo FXS.

fxsks=2-4    # Señalización FXS Kewl start para los
             # canales No. 2, 3 y 4 que son puertos
             # de tipo FXO
```

En este ejemplo, el parámetro “loadzone” indica que la interfaz será cargada con tonos de indicación españoles. Es posible cargar más de un grupo de zonas de tonos para permitir la generación de tonos de diferentes zonas, separándolos por comas.

De manera similar, el parámetro “defaultzone” se refiere a las zonas de tonos que se emplearán por omisión cuando no se especifique zona alguna.



Las zonas de tonos se encuentra definidas en el archivo *zonedata.c* dentro del código fuente de DAHDI.

Los parámetros de ejemplo “fxsks” y “fxsko” indican el tipo de señalización que se utilizará para cada canal analógico. Asterisk soporta los más populares protocolos de señalización analógica, entre los que se encuentran los siguientes[25]:

25 Asterisk Development Team. DAHDI Telephony Interface Driver. [en línea].

- fxogs : FXO Ground start.
- fxsgs : FXS Ground start.
- fxsls : FXS Loop start.
- fxols : FXO Loop start.
- fxsks : FXS Kewl start.
- fxoks : FXO Kewl start.

Puede parecer confuso, pero se debe definir una señalización opuesta al tipo de puerto que se está configurando. Por ejemplo, señalización *fx**o**ks* si el puerto es FXS, y señalización *fx**s**ks* si el puerto es FXO.

Una vez se haya terminado de editar el archivo /etc/dahdi/system.conf se debe correr el programa dahdi_cfg para validar la configuración e inicializar el hardware de telefonía:

```
# dahdi_cfg -vvv
```

Si el comando termina de forma silenciosa, la configuración es correcta. En caso contrario, se debe revisar la sintaxis y los parámetros configurados en el archivo. Se utilizan los parámetros "-vvv" para aumentar el nivel de información a mostrar a la salida del comando (consulte las opciones completas con el comando "dahdi_cfg -h").

A continuación se puede utilizar la herramienta dahdi_tool, que es un programa que sirve para verificar el estado del hardware de telefonía instalado en el servidor.

<http://docs.tzafrir.org.il/dahdi-tools/> [citado en 15 de febrero de 2012]

```
# dahdi_tool
```

Si todo se encuentra debidamente configurado y la comunicación es posible, dahdi_tool indicará un mensaje de “OK”, o de lo contrario se mostrará un estado de alarma.

Configuración de Asterisk

El archivo /etc/asterisk/chan_dahdi.conf se encarga de la comunicación entre Asterisk y el controlador de las interfaces de telefonía DAHDI. La estructura típica del archivo para canales analógicos se muestra a continuación:

```
[channels]
language=es          ;para escuchar los sonidos
                     ;pre-grabados en Español

group=1              ;Grupo lógico de canales
                     ;Puede ir de 0 al 63
signalling=fxo_ks    ;Señalización FXO Kewl start para el
                     ;canal No. 1, que es un
                     ;puerto de tipo FXS.
context=usuarios     ;La sección del archivo
                     ;“extensions.conf” donde se
                     ;procesarán las llamadas iniciadas
                     ;por este teléfono.
callerid=”Juan” <2003>
channel=> 1

group=2
signalling=fxs_ks    ;Señalización FXS Kewl start para los
             ;canales No. 2, 3 y 4 que son puertos
             ;de tipo FXO.
```

```
context= menu_bienvenida ;Las llamadas de la red telefónica
                          ;se enviarán al contexto del IVR de
                          ;operadora automática.
callerid=asreceived
channel=> 2-4
```

Nótese que el tipo de señalización definido en el archivo *chan_dahdi.conf* debe coincidir con el valor elegido en el archivo *system.conf* para cada canal.

Para utilizar un teléfono analógico conectado al puerto FXS del ejemplo anterior, se debe agregar lo siguiente al final del archivo "*extensions.conf*", en el contexto "*[usuarios]*":

```
[usuarios]
; Extensión para llamar al teléfono analógico de Juan
exten = 2003,1,Dial(DAHDI/1,25)    ; Marcar en el canal 1.
exten = 2003,n,VoiceMail(2003@default)
```

Para utilizar los puertos FXO del mismo ejemplo para enviar llamadas hacia la red telefónica pública, se puede crear una nueva extensión que ejecute la aplicación "Dial", y ubicarla en un nuevo contexto "troncales", que a su vez debe ser incluido en el contexto "usuarios" creado previamente para que los usuarios lo utilicen:

```
[troncales]
exten = 911,1,Dial(DAHDI/g2/911,25) ;Extensión para
                          ;marcar el número 911 a través
                          ;de los canales analógicos del
                          ;grupo No.2

[usuarios]
```

```
...
include => troncales
```

Nótese que se ha indicado a la aplicación “Dial” que se marque a través del recurso “DAHDI/g2”. De esta forma se puede marcar a través de cualquiera de los canales del grupo No. 2, tal como se definió en el archivo chan_dahdi.conf. La aplicación “Dial” soporta los siguientes tipos de marcación por grupo:

- g: Recorre el grupo en orden ascendente desde el primer canal, hasta encontrar uno disponible.
- r: Recorre el grupo en orden ascendente desde el canal usado la última vez, hasta encontrar uno disponible.
- G: Recorre el grupo en orden descendente desde el último canal, hasta encontrar uno disponible.
- R: Recorre el grupo en orden descendente desde el canal usado la última vez, hasta encontrar uno disponible.

Siempre se debe reiniciar Asterisk para que tengan efecto los cambios realizados en el archivo *chan_dahdi.conf*, ya que el comando para recargar el módulo de DAHDI no siempre aplica todos los cambios, e incluso un cambio importante puede hacer que el módulo no cargue del todo.

Telefonía Digital

Las señales eléctricas que representan la voz humana y que son transmitidas de un lado a otro son analógicas, y por lo tanto susceptibles de ser afectadas de forma no deseada a causa del ruido y la atenuación, causando variaciones de difícil recuperación en la información transmitida, lo cuál afecta en gran

medida al correcto funcionamiento y la calidad de la transmisión telefónica a grandes distancias.

Figura 5.9: Transmisión de señales analógicas

Mientras las magnitudes de una señal analógica se representan mediante valores continuos, las de una señal digital se representan por valores discretos. Por ejemplo, una señal analógica como el nivel de agua en un tanque tiene múltiples valores continuos en función del tiempo, mientras que una lámpara sólo puede tomar dos valores o estados: encendida o apagada.

Los sistemas digitales utilizan los dígitos uno (1) y cero (0) como valores discretos. Estos dígitos se representan generalmente por dos niveles de tensión eléctrica, uno alto y otro bajo. Esto no solo facilita la aplicación de la lógica y la aritmética binaria, sino que también permite que una señal digital sea menos susceptible de ser afectada por el ruido y la atenuación cuando se transmite a grandes distancias.

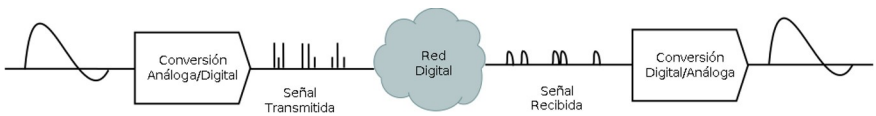


Figura 5.10: Transmisión digital

Se conoce entonces como Telefonía Digital a la conversión de la señal de voz de naturaleza analógica a una señal digital para transmitirla de un lado a otro y luego convertirla de nuevo a una señal que pueda ser interpretada por el oído humano.

Modulación MIC o PCM

La Modulación por Impulsos Codificados (MIC o PCM por sus siglas en inglés - Pulse Code Modulation), es un procedimiento de modulación utilizado para transformar una señal analógica de voz en una secuencia digital de bits. El procedimiento y los principales conceptos de digitalización presentes en la MIC se resumen en la figura 5.3[26].

El proceso de conversión de una señal analógica a digital es muy similar en las diferentes aplicaciones de procesamiento de señales que se pueden encontrar en el mundo de la tecnología, desde audio y voz, pasando por fotografías, hasta video, etc. Sin embargo, para la transmisión de voz sobre la Red Telefónica Pública se han definido los siguientes procedimientos que se rigen por reconocidos estándares y que hacen parte de la Modulación MIC: muestreo, cuantificación, compansión y codificación.

26 RENDÓN, Álvaro. Sistemas de Conmutación: Fundamentos y Tecnologías. Popayán: Universidad del Cauca, 2000. Capítulo 2.

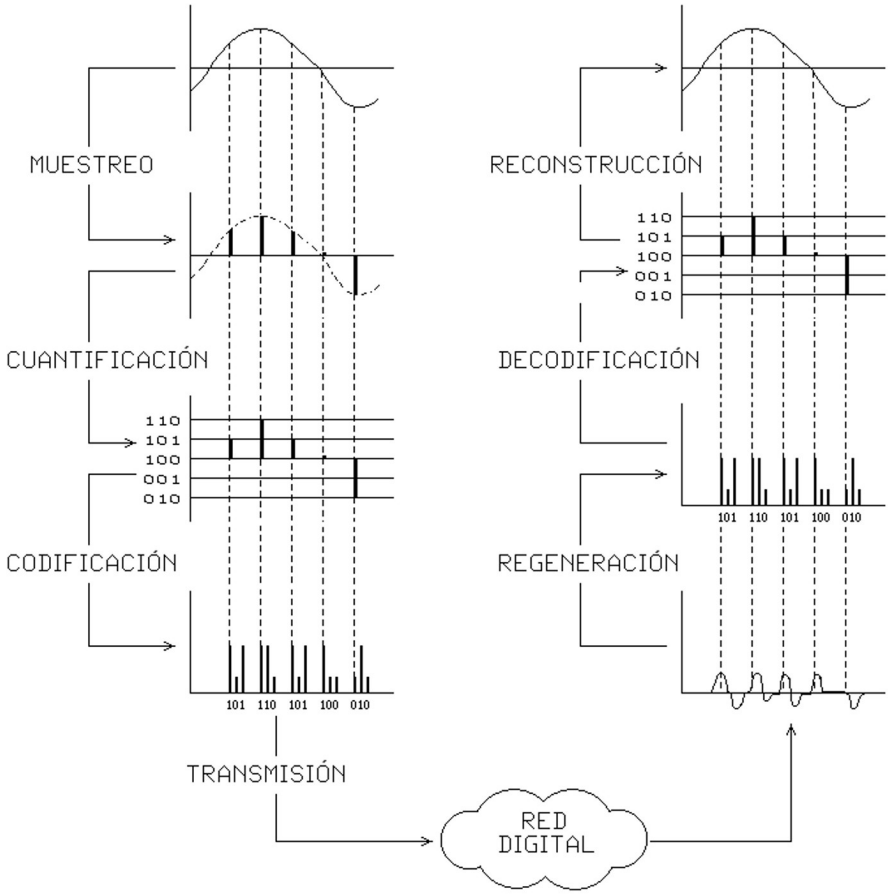


Figura 5.11: Modulación MIC o PCM

Muestreo

Consiste en tomar muestras periódicas de la amplitud de la señal analógica, a un intervalo entre muestras constante. El ritmo de este muestreo, llamado frecuencia o tasa de muestreo, determina el número de muestras que se toma en un intervalo de tiempo.

Para determinar el número correcto de muestras por segundo que deben tomarse de la señal analógica, se utiliza el Teorema del Muestreo[27], de acuerdo con el cuál, si se toman muestras de una señal eléctrica continua a intervalos regulares y con una frecuencia doble (F_m) a la frecuencia máxima que se quiera muestrear (*B*), dichas muestras contendrán toda la información necesaria para reconstruir la señal original:

$$F_m \geq 2 * B$$

La señal muestreada es luego procesada en el receptor a través de un filtro pasa-bajo, es decir, que deja pasar solo los componentes de la señal que sean menores a la frecuencia máxima de la señal original (B). El proceso completo de muestreo y reconstrucción se muestra a continuación[28]:

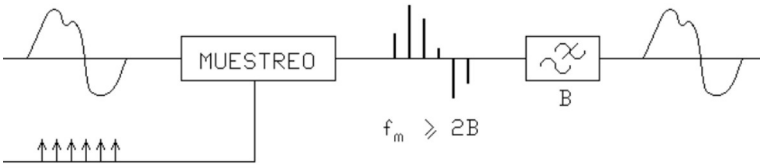


Figura 5.12: Teorema del muestreo

El espectro de frecuencias audibles va de 20 a 20.000 Hertz, sin embargo la frecuencia más alta que se puede encontrar en la voz humana a transmitir vía telefónica es de 3.400 Hz. En los canales telefónicos digitales, se transmiten frecuencias de hasta 4.000 Hz, de tal forma que con una frecuencia de muestreo de 8.000 Hz, sería posible transmitir la información suficiente para el canal telefónico de voz:

$$F_m = 2 * B = 2 * (4000\,Hz) = 8000\,Hz$$

27 El teorema del muestreo es conocido también como de Nyquist-Shannon, ya que fue formulado en forma de conjetura por primera vez por Harry Nyquist en 1928 ("Certain topics in telegraph transmission theory"), y fue demostrado formalmente por Claude E. Shannon en 1949 ("Communication in the presence of noise").

28 RENDÓN, Op. Cit.

Un muestreo realizado a una frecuencia menor a la que determina el teorema del muestreo (F_m), puede provocar que se presente el fenómeno conocido como «Aliasing». Cuando esto sucede, la señal original no puede ser reconstruida de forma correcta a partir de la señal digital recibida, ya que las muestras están muy separadas entre sí.

Cuantificación

El proceso de cuantificación consiste en convertir una sucesión de muestras de amplitud continua en una sucesión de valores discretos preestablecidos según el código utilizado. Durante el proceso de cuantificación se mide el nivel de tensión de cada una de las muestras, obtenidas en el proceso de muestreo, y se les atribuye un valor finito (discreto) de amplitud, seleccionado por aproximación dentro de un margen de niveles previamente fijado[29].

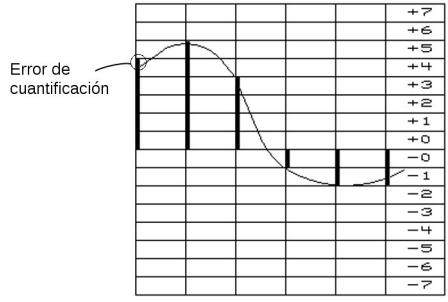


Figura 5.13: Cuantificación

Así pues, la señal digital que resulta tras la cuantificación es sensiblemente diferente a la señal eléctrica analógica que la originó, por lo que siempre va a existir una cierta diferencia entre ambas. Ésto se conoce como error de cuantificación, el cuál se produce cuando el valor real de la muestra no

29 Ibid.

equivale a ninguno de los escalones disponibles para su aproximación y la distancia entre el valor real y el que se toma como aproximación es muy grande. Un error de cuantificación se puede convertir en ruido cuando se reproduzca la señal tras el proceso de decodificación digital.

Compansión

Con el fin de disminuir la posibilidad de que se produzca un error de cuantificación perceptible, en MIC se realiza una cuantificación no lineal. Éste es un tipo de cuantificación digital en la que se lleva a cabo un paso previo a la cuantificación en donde se hace pasar la señal por un compansor logarítmico, formado por dos subprocesos: compresión y expansión (compressing y expanding en inglés respectivamente), a través del cuál la señal analógica a cuantificar se comprime de tal forma que presenta unas variaciones en la amplitud de voltaje menos abruptas[30].

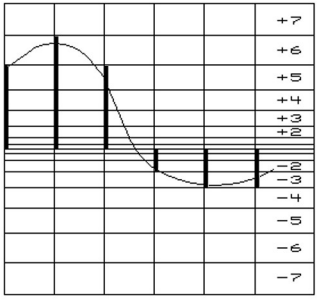


Figura 5.14: Cuantificación logarítmica

De esta forma, se tendrán pequeños pasos de cuantificación para los valores pequeños de amplitud y pasos de cuantificación grandes para los valores grandes de amplitud, lo que proporciona mayor resolución en señales débiles al compararse con una cuantificación uniforme de igual frecuencia, pero menor resolución en señales de gran amplitud.

Para telefonía digital existen dos algoritmos de compansión que se

30 Ibid.

aproximan a una función logarítmica, y son conocidos como Ley-μ (léase miu), usado en Norte América y Japón, y el Ley-A, usado en Europa y el resto del mundo).

Una vez cuantificada, la señal analógica se convierte en una señal digital, ya que los valores que están preestablecidos son finitos. No obstante, todavía no se traduce al sistema binario. La señal ha quedado representada por un valor finito que solo durante la codificación (siguiente proceso de la conversión analógico digital) será cuando se transforme en una sucesión de ceros y unos.

Codificación

La codificación consiste en la traducción de los valores de tensión eléctrica analógicos que ya han sido cuantificados al sistema binario, mediante códigos preestablecidos. La señal analógica va a quedar transformada como se ha mencionado previamente en un tren de impulsos digital (sucesión de ceros y unos).

El código preestablecido que se utiliza para la codificación/decodificación de los datos se conoce como códec (abreviatura de Codificador-Decodificador). El principal códec utilizado en telefonía digital es el estándar G.711 de la ITU-T para la compresión de audio. Este estándar fue liberado para su uso en el año 1972 y tiene variantes tanto para el algoritmo de Ley-A como para el de Ley-μ.

G.711 define que las muestras comprimidas de la señal de audio digital deben tener una tasa de muestreo de 8000 muestras por segundo, cada una de las cuales es cuantificada logarítmicamente en un código de 8 bits (también conocido como octeto)[31], proporcionando así un flujo de datos codificados de 64 kilo bits por segundo.

Conmutación de Circuitos

Ya se había mencionado previamente, que con el fin de compartir la

31 Con un código de 8 bits, cada muestra puede tener hasta 256 valores diferentes (2^8 = 256)

conexión que existe entre una central telefónica y otra para que sea utilizada por todos los abonados locales, se utilizan técnicas como la modulación y multiplexación de las señales digitales.

Multiplexación TDM

La multiplexación por división de tiempo (MDT) o (TDM), del inglés Time Division Multiplexing, es el método a través del cuál se envían varias señales digitales a través de un mismo canal de transmisión, repartiendo entre ellas el tiempo de uso del mismo.

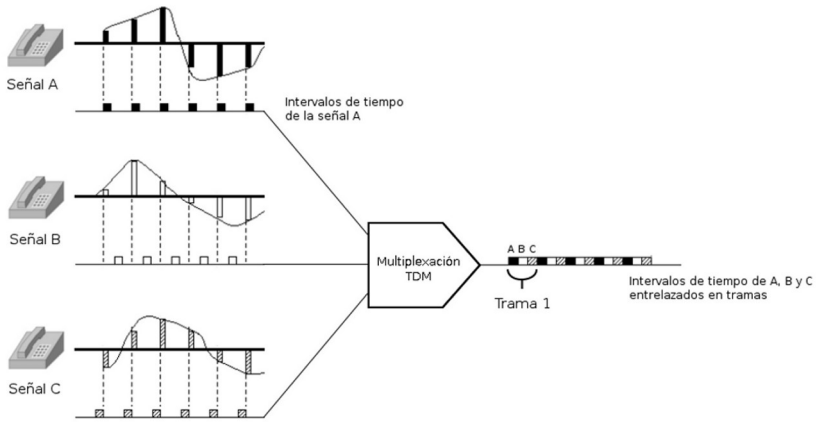


Figura 5.15: Multiplexación TDM

"Los octetos que representan las muestras tomadas en los tres canales son entrelazados, conformando una secuencia de impulsos. Tal conjunto de impulsos se denomina trama, y el tiempo de uso del canal por cada octeto de bits se denomina intervalo de tiempo."[32] en este ejemplo, cada trama tiene tres intervalos de tiempo.

De esta manera se puede utilizar una sola línea para establecer varios circuitos de comunicación simultáneamente, optimizando los recursos. Los

32 Ibid.

intervalos de tiempo serán transmitidos de una central telefónica a la siguiente, donde serán conmutados hacia su destino, estableciendo un circuito de comunicación, que mantiene la conexión digital durante toda la llamada.

Debido a que el tiempo es una variable tan importante en este tipo de multiplexación, se utilizan diferentes mecanismos para garantizar que los dos extremos de la comunicación estén sincronizados, como se verá más adelante.

Tipos de conexiones digitales

Como se mencionó anteriormente, el Intervalo de Tiempo (IT) está formado por una señal de audio codificada en formato MIC (PCM), bien sea con Ley-μ o Ley-A (G.711u y G.711a), es decir, a una tasa de muestreo de 8KHz, utilizando 8 bits, para un ancho de banda resultante de 64Kbps.

Estos intervalos de tiempo IT son las unidades básicas que componen todas las agrupaciones superiores de canales de telefonía digital, como los T1 y E1, de 24 y 32 ITs, respectivamente.

T1 y E1

Un T1 (MIC-24) es un canal digital utilizado en Estados Unidos y Japón que utiliza compansión Ley-A y a través del cuál se pueden enviar 24 canales básicos de voz (IT), de 64Kbps cada uno. Los 24 ITs utilizan 1,536 Mbps, y los restantes 0,008 Mbps son utilizados por bits de alineación de trama para sincronismo, formando un flujo único de 1,544Mbps. La estructura de una trama de T1 se muestra a continuación[33]:

33 Ibid.

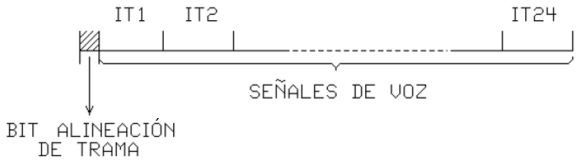


Figura 5.16: Estructura de la trama T1 (MIC-24)

Dependiendo de la forma como se organice la información de sincronismo y de señalización dentro de la trama, el formato de entramado de un T1 puede ser de dos tipos:

- D4 o Super Frame: Utiliza grupos de 12 T1 para formar un patrón de sincronismo repetitivo. Soporta únicamente señalización asociada al canal (CAS – Channel Associated Signalling), tomando uno de los bits de cada IT para enviar la información de señalización.

- ESF (Extended Super Frame): Similar a la anterior, pero utiliza un patrón de 24 T1 e incorpora un chequeo de bits usando un código de redundancia cíclica (CRC-6). Soporta tanto CAS, como señalización por canal común (CCS – Common Channel Signaling), en la cuál se toma un IT completo para enviar la señalización de todos los canales.

Un E1 (MIC-30) está compuesto por 32 ITs, para un total de 2,048Mbps, pero sólo utiliza 30 canales para transmisión de vos pues el primero se utiliza para información de sincronización y el No. 16 para información de señalización. Utiliza compansión Ley-A y es utilizado en Europa y Latinoamérica. La estructura de una trama de E1 se muestra a continuación[34]:

34 Ibid.

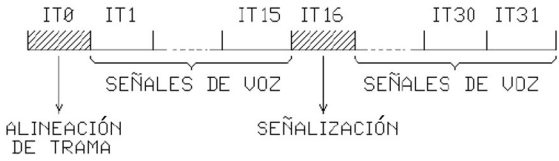


Figura 5.17: Estructura de la trama E1 (MIC-30)

El formato de entramado de E1 soporta información de señalización tanto CAS como CCS, e incorpora un mecanismo opcional de comprobación de bits por código de redundancia cíclica (CRC-4).

Otras Jerarquías (T-n y E-n)

Tanto los circuitos T como los E tienen la posibilidad de escalar en número de ITs[35]:

T-n (MIC-24)		
Orden	**Canales (ITs)**	**Tasa de Kbps**
T1	24	1.544
T2	96	6.312
T3 (EE.UU.)	672	44.736
T4 (EE.UU.)	4032	274.176
T3 (Japón)	480	32.064
T4 (Japón)	1440	97.728

Tabla 5.1: Jerarquías superiores de T1

35 Ibid.

E-n (MIC-30)		
Orden	**Canales (ITs)**	**Tasa de Kbps**
E1	30	2.048
E2	120	8.448
E3	480	34.368
E4	1920	139.264

Tabla 5.2: Jerarquías superiores de E1

SONETH, SDH

Tanto para los canales T-n como para los E-n fueron diseñadas tecnologías para ampliación de la capacidad de transporte de canales de voz a través de las nuevas redes de transmisión basadas en fibra óptica y radio enlaces de alta capacidad. Para esto fueron creadas la Jerarquía Digital Síncrona (SDH) y la Red Óptica Síncrona (SONET) para integrar el tráfico de las redes de telefonía a nivel internacional, que alcanzan velocidades de varios miles de Mbps y capacidad para decenas de miles de canales de voz.

Adaptación a la línea

El flujo de bits de la voz codificada, representada en niveles de voltaje alto (1) y bajo (0), puede ser mal interpretada cuando se transmite a grandes distancias, puesto que una secuencia de bits seguidos se puede confundir fácilmente[36].

36 Ibid.

Figura 5.18: Secuencias de bits seguidos pueden malinterpretarse

Para evitar esta situación, se utiliza un código de adaptación de línea, que traduce el flujo de bits de la voz codificada en voltajes bien diferenciados para enviar a través del medio de transmisión.

Una vez se ha adaptado la información a transmitir de tal forma que esté lista a viajar sobre el medio físico de transmisión, éste puede ser llevado a señales electromagnéticas sobre diferentes tecnologías, como pares de cobre, cables coaxiales, fibra óptica, radio enlaces, etc.

Los principales códigos de línea que se utilizan en telefonía digital se dividen en códigos polares (utilizan dos niveles de voltaje: positivo y negativo) y códigos bipolares (utilizan tres valores: positivo, negativo y cero).

Asterisk soporta los 3 tipos de codificación bipolar más difundidos: AMI, B8ZS y HDB3.

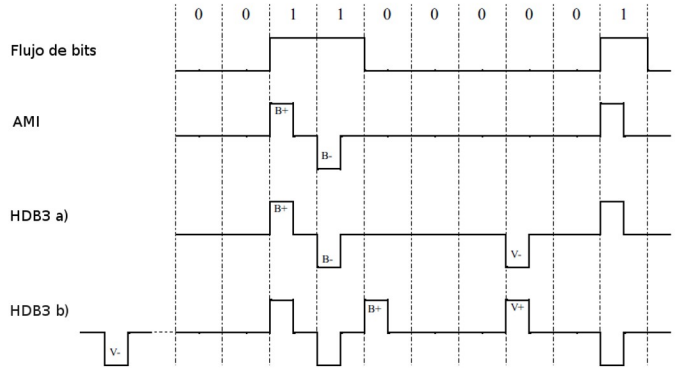


Figura 5.19: Ejemplos de AMI y HDB3

AMI (Alternate Mark Inversión)

Utiliza tres valores: positivo (B+), negativo (B-) y cero. Siempre se produce una alternancia entre los valores de amplitud positivo y negativo para representar los bits '1', aunque estos bits no sean consecutivos.

La gran desventaja de AMI es que una secuencia larga de ceros puede hacer que se pierda la sincronía de la transmisión. B8ZS y HDB3 son dos variaciones de AMI que sustituyen las secuencias largas de ceros, evitando tensiones constantes durante largo tiempo.

B8ZS (Bipolar 8-Zero Substitution)

Es utilizada principalmente en América del Norte. Cuando aparecen 8 ceros consecutivos, B8ZS introduce cambios artificiales (violaciones y transiciones de polaridad) en el patrón, basados en la polaridad del último bit '1' codificado:

- V: Violación, mantiene la polaridad anterior en la secuencia.
- B: Transición, invierte la polaridad anterior en la secuencia.

Los ocho ceros se sustituyen por la secuencia: 000VB0VB.

HDB3 (High Density Bipolar 3)

Es utilizada en Europa y Japón. Cuando aparecen cuatro ceros consecutivos, estos se sustituyen por una de las dos siguientes secuencias:

a) Si el número de unos es impar desde la última sustitución o se trata de la primera sustitución realizada, HDB3 los sustituye por la secuencia: 000V.

b) Si el número de unos es par desde la ultima sustitución. HDB3 los sustituye por la secuencia: B00V.

Señalización digital[37]

A continuación se presentan los principales protocolos de señalización que se utilizan para establecer, mantener y terminar una conexión en telefonía digital. El tipo de señalización digital más utilizada con Asterisk es RDSI-PRI, pero también se cuenta con soporte oficial para E&M y SS7. Es posible también utilizar librerías externas para habilitar el soporte para E1 con señalización MFC-R2, aunque cada vez son menos comunes este tipo de troncales.

Señalización E&M

Se utiliza en troncales urbanas largas y en troncales interurbanas cortas. Se requieren hilos independientes, conocidos como E(recepción) y M (transmisión).

Esta señalización puede ser utilizada tanto por sistemas de señalización asociada al canal, como de canal común.

37 RENDÓN, Álvaro. Conmutación Digital: Señalización. Popayán: Universidad del Cauca, 1999. Borrador.

Señalización MFC-R2

Multi Frequency Compelled Region 2 signaling: Señalización región 2 dirigida por medio de multi frecuencias (tonos) o de secuencia obligada. La señalización R2 es de tipo CAS (Asociada al Canal) y fue desarrollada en los años 60s, pero aún es usada en Europa, Asia, Latino América y Australia. Se encuentra definida por la ITU (International Telecommunications Union), pero cada país ha creado variaciones según sus necesidades.

MFCR2 es señalización "peer to peer". Lo que significa que ambos lados son iguales desde el punto de vista de la señalización, es decir, no existe un "cliente" y un "servidor". Aquí ambos lados del enlace operan de la misma forma.

Para la realización de una llamada, R2 utiliza 2 tipos de señales:

- Señales Supervisoras: Se encargan de supervisar el estado de la línea o enlace. Son enviadas por medio de 4 bits que son comúnmente conocidos como bits ABCD.

- Señales de inter-registro: Se encargan de la inicialización y control de la llamada. A través de estas señales se envían los dígitos de identificación del número origen (ANI[38]) y el número llamado (DNIS[39]). Estas señales son tonales, y es donde se transmiten las señales de Multi Frecuencia.

Señalización RDSI (ISDN)

La Red Digital de Servicios Integrados (RDSI o ISDN en inglés) es una red que facilita conexiones digitales extremo a extremo para proporcionar una amplia gama de servicios, tanto de voz como de otros tipos. Fue diseñada para la transmisión del tráfico de datos y multimedia que luego fue abarcado en su mayoría por Internet desplazada por otras tecnologías de Acceso a

38 En inglés, *Automatic Number Identification.*
39 En inglés, *Dialed Number Information Service.*

Internet.

RDSI utiliza el protocolo ITU-T Q.931, cuyo flujo de mensajes se muestra a continuación:

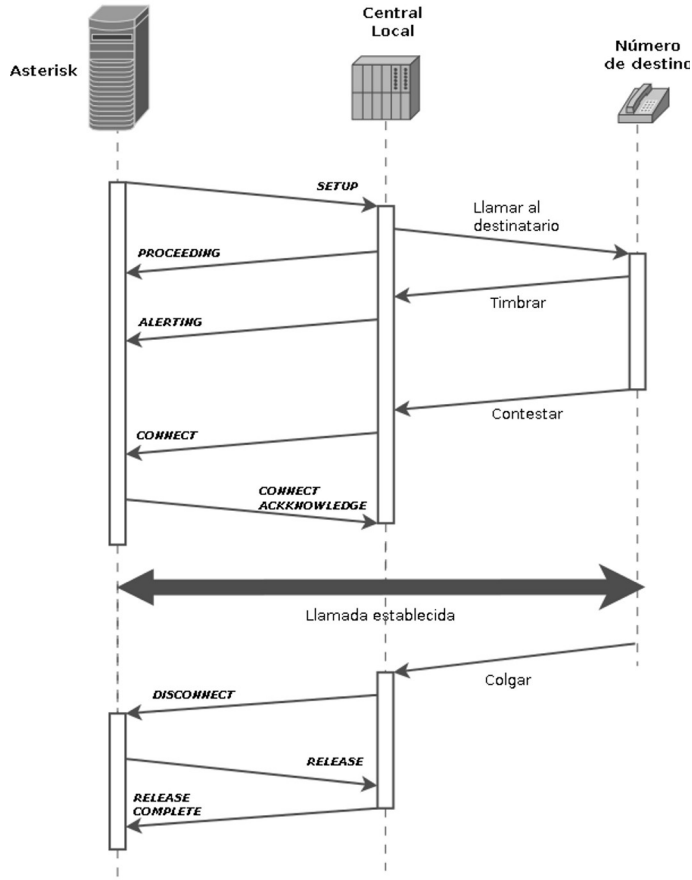


Figura 5.20: Flujo de mensajes Q.931 entre Asterisk y una troncal RDSI.

Para la transferencia de información y señalización se han definido los siguientes canales:

- Canal B: es el canal básico de usuario. Es el canal a 64 kbps para transporte de la información generada por el terminal de usuario.

- Canal D: es el canal de señalización a 16 ó 64 kbps.

- Canales H: son canales destinados al transporte de flujos de información de usuario a altas velocidades, superiores a 64 kbps.

De acuerdo al número de canales se definen dos tipos de troncales RDSI: BRI, y PRI

Acceso Básico - BRI

El acceso básico RDSI (Basic Rate Interface) consiste en dos canales B full-duplex de 64 kbps y un canal D full-duplex de 16 kbps. La división en tramas, la sincronización, y otros bits adicionales dan una velocidad total a un punto de acceso básico de 192 kbps:

- BRI: 2B+D+señalización+sincronización+mantenimiento

Acceso Primario - PRI

El acceso primario RDSI (Primary Rate Interface) está destinado a usuarios con requisitos de capacidad mayores, tales como oficinas con plantas (PBX) digitales o red local (LAN).

Estados Unidos, Japón y Canadá usan una estructura de transmisión basada en T1 (1,544 Mbps), mientras que en Europa la velocidad estándar es 2,048 Mbps. Típicamente, la estructura para el canal de 1,544 Mbps es 23 canales B más un canal D de 64 kbps y, para velocidades de 2,048 Mbps, 30 canales B más un canal D de 64 kbps:

- E1 PRI: 30B(64)+D(64)señalización+sincronización(64) -> 2048 Kbps

- T1 PRI: 23B(64)+D(64)señalización+sincronización(8) -> 1544 Kbps

Debido a que el tipo de señalización más extendido actualmente es el de RDSI, se suele asumir que una troncal E1 es siempre un primario (PRI), pero en muchos países, especialmente en latinoamérica, todavía se encuentran troncales E1 con señalización MFC-R2, por lo que se debe verificar con el operador telefónico el tipo de E1 que provee realmente.

Señalización SS7

Este sistema se señalización es el utilizado por los operadores telefónicos para las comunicaciones modernas entre centrales. Los protocolos del Sistema de señalización por canal común nº 7 (SS7) fueron desarrollados por AT&T a partir de 1975 y definidos como un estándar por el UIT-T en 1981 en la serie de Recomendaciones Q.7XX del UIT-T. SS7 utiliza un sistema de señalización fuera de línea fuera de banda, usando un canal de señalización separado. Esto evita los problemas de seguridad que tenían los sistemas anteriormente y los usuarios finales no tienen acceso a estos canales.

Usando puertos E1/T1 en Asterisk

Como se vio en el Capítulo 1, Asterisk puede interactuar con troncales digitales mediante tarjetas de telefonía que generalmente cuentan con 1, 2 o 4 puertos para E1 o T1.

La mayoría de modelos de tarjeta ofrece un mecanismo para seleccionar si sus puertos se utilizarán como E1 o como T1. En algunos es un jumper sobre la tarjeta, y otros es un parámetro que se especifica a la hora de cargar el módulo controlador de hardware en el núcleo de Linux.

Para configurar tarjetas digitales compatibles con Asterisk, se utiliza DAHDI, cuya instalación se cubre en el Capítulo 2.

Configuración de DAHDI

Es en este punto donde se aplicarán todos los conceptos de telefonía digital cubiertos en este capítulo. Un archivo /etc/dahdi/system.conf tiene la siguiente estructura típica para puertos de telefonía digital:

```
span=<número_de_puerto>,<fuente_de_temporización>,<LBO>,<entramado>
,<codificación>[,crc4][,yellow]

bchan=<canales B del enlace E1/T1 PRI>

dchan=<canales D (señalización) del enlace E1/T1 PRI>
```

Los valores de estos parámetros deben coincidir con la configuración entregada por el operador telefónico que provee la troncal digital. La descripción de cada parámetro se encuentra a continuación:

<número de puerto>: Número entero irrepetible que corresponde al número de la interfaz de telefonía a configurar. Una forma sencilla de obtener la lista de puertos y canales de cada uno es revisando el contenido de la carpeta "/proc/dahdi", creada cuando se carga el módulo de DAHDI. Por cada puerto se crea un archivo en esa carpeta, y el nombre de ese archivo es el número de puerto:

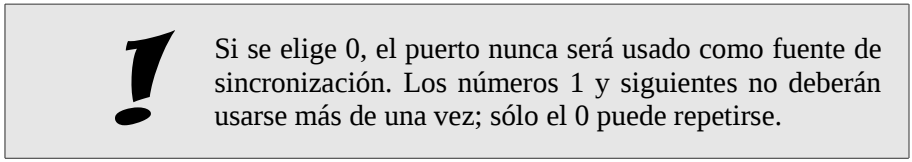
```
# cat /proc/dahdi/*
```

<fuente de temporización>: Todos los puertos T1 o E1 generan una señal de reloj en su lado transmisor. El parámetro <fuente de temporización> determina si la señal del reloj del extremo remoto de la línea T1/T1 será usada como reloj maestro de sincronización. Si así es, el reloj local se sincronizará con éste. La opción natural tratándose de un sistema Asterisk es sincronizarse con la señal de reloj provista por la central del operador telefónico, es decir como esclavo.

- 0: Usar este puerto como fuente de sincronización. Enviar la señal de reloj al otro extremo (se usa cuando se desea que Asterisk sea la fuente de reloj del enlace digital).

- 1: Usar la troncal digital como fuente primaria de sincronización (es la opción más usual y se usa para recibir la señal de reloj de la central telefónica del operador).

- 2: Usar como fuente secundaria de sincronización (y así sucesivamente...).



<LBO>: En inglés Line Build Out es un nivel de potencia que se ajusta sobre una línea de transmisión dependiendo de la longitud del cable que conecta a la interfaz T1/E1 y el CPE (Customer Premises Equipment), es decir el equipo Asterisk. Los valores admitidos se muestran a continuación:

Valor de LBO	Distancia
0	0 – 133 pies
1	133 - 266 pies
2	266 - 399 pies
3	399 - 533 pies
4	533 - 655 pies

Tabla 5.3: Valores de LBO

<entramado>: El entramado o formato de señalización de las tramas. Puede ser:

- Para E1: cas o ccs.

- Para T1: d4 o esf.

<codificación>: El código adaptación a la línea que se empleará para la comunicación:

- Para E1: ami o hdb3.
- Para T1: ami o b8zs.

<crc4>: Parámetro opcional. Sólo se debe especificar si el operador telefónico tiene activado el chequeo de bits por código de redundancia cíclica.

<yellow>: Parámetro opcional para generar una alarma *amarilla* cuando no esté abierto ningún canal.

A continuación se muestra un ejemplo de configuración de una tarjeta con dos puertos digitales, el primero de tipo E1 y el segundo de tipo T1, ambos configurados para señalización RDSI-PRI:

```
span=1,1,0,ccs,hdb3,crc4    # Primer puerto, de tipo E1.
bchan=1-15,17-31             # 30 Canales de voz.
dchan=16                     # 1 Canal de señalización PRI.

span=2,1,0,esf,b8zs          # Segundo puerto, de tipo T1.
bchan=32-54                  # 23 Canales de Voz.
dchan=55                     # 1 Canal de señalización PRI.
```

Una vez se haya terminado de editar el archivo /etc/dahdi/system.conf se debe correr el programa dahdi_cfg para validar la configuración e inicializar el hardware de telefonía:

```
# dahdi_cfg -vvv
```

Y a continuación se puede utilizar la herramienta dahdi_tool para verificar el estado de los canales digitales:

```
# dahdi_tool
```

En una línea T1/E1 PRI, dahdi_tool indicará una alarma *roja* si se ha establecido la conexión entre los extremos de comunicación, o una alarma *amarilla* si hay problemas de sincronía.

Configuración de Asterisk

Un archivo chan_dahdi.conf para la conexión de un E1 PRI RDSI tiene la siguiente estructura:

```
[channels]
language=es          ;para escuchar los sonidos
                     ;pre-grabados en Español

group=1
signalling=pri_cpe           ;Señalización RDSI PRI
switchtype=euroisdn          ;Variante de RDSI PRI para E1.
channel => 1-15,17-31        ;30 Canales del puerto E1
context= menu_bienvenida     ;Las llamadas de la red telefónica
                             ;se enviarán al contexto del IVR de
                             ;operadora automática.

signalling=pri_cpe           ;Señalización RDSI PRI
switchtype=5ess              ;Variante de RDSI PRI para T1.
channel => 32-54             ;23 Canales del puerto T1,
                             ; consecutivos al puerto E1.
context= menu_bienvenida
```

Los principales parámetros de configuración a tener en cuenta son:

signalling: Señalización. Permite configurar el tipo de señalización del canal. En el caso de una interfaz digital, éste parámetro puede tomar los siguientes valores:

- pri_cpe: Señalización RDSI PRI, extremo del usuario (CPE).
- pri_net: Señalización RDSI PRI, extremo dela red.
- em: Señalización E&M.
- em_e1: Señalización E&M para E1.
- ss7: Señalización Número 7.

switchtype: Indica el tipo de variante RDSI PRI que implementa el operador telefónico. Para E1 PRI, el valor debe ser “euroisdn” (EuroISDN). Para T1 PRI, los valores permitidos son:

- national: National ISDN 2 (EE.UU.)
- dms100: Nortel DMS100
- 4ess: AT&T 4ESS
- 5ess: Lucent 5ESS
- qsig: Q.sig.

resetinterval: Ajusta el tiempo de reinicio programado de los canales sin usar. Puede tomar un valor entre 60 y 3600 segundos, aunque never (nunca) también es un valor válido.

channel: Indica los canales de voz (canales tipo B) configurados en la interfaz T1/E1. Deben coincidir con los indicados en el system.conf. Los canales de señalización (canales tipo D) se omiten aquí.

Para utilizar los puertos digitales del mismo ejemplo para enviar llamadas hacia la red telefónica pública, se puede crear una nueva extensión que ejecute la aplicación “Dial”, y ubicarla en el contexto “troncales” del archivo extensions.conf:

```
[troncales]
exten = 911,1,Dial(DAHDI/g1/911,25) ;Extensión para
                                    ;marcar el número 911 a través
                                    ;de cualquiera de los 53
                                    ;canales digitales del
                                    ;grupo No.1

[usuarios]
...
include => troncales
```

Una vez reiniciado Asterisk para aplicar los cambios hechos al archivo chan_dahdi.conf, se puede iniciar la Consola o Interfaz de Línea de Comandos (CLI), y ejecutar los siguientes comandos para verificar la configuración y el estado de los dos puertos PRI del ejemplo:

```
*CLI> pri show span 1
*CLI> pri show span 2
```

Capítulo 6. Troncales de telefonía IP

> “La comprensión mutua sería enormemente facilitada por el uso de una lengua universal”
>
> Nikola Tesla (1856-1943) ingeniero e inventor Croata.

Telefonía IP

Debido a su alta dependencia del hardware, la implementación de nuevos servicios de valor agregado es lenta y costosa en la telefonía tradicional. Por ejemplo, si una empresa tiene una PBX tradicional y quiere agregarle capacidades para correo de voz o algún otro nuevo servicio, deberá hacer una fuerte inversión de dinero para adquirir el módulo hardware respectivo.

Por su facilidad de integración con la infraestructura informática existente, la telefonía a través de redes de datos ofrece a empresas y operadores telefónicos la capacidad de adaptarse a la evolución tecnológica y de implementar fácilmente servicios avanzados, como correo de voz, conferencia, sistemas de respuesta interactiva (IVR), operadoras automáticas, etc.

Adicionalmente, a pesar de que el diseño de la RTPC asegura un buen trabajo en el establecimiento de los circuitos o caminos necesarios para realizar una comunicación entre dos abonados, el hecho de que el circuito establecido debe ser exclusivo para una llamada durante el tiempo que ésta dure, hace que se subutilicen los recursos y el transporte de la voz a grandes distancias sea mucho más costoso que si se utilizaran redes de datos para llevarlo a cabo.

Conmutación de Paquetes

En las redes de datos la información que un extremo quiere enviar a otro es primero dividida en paquetes para hacer un uso eficiente de los enlaces físicos, ahorrando costos en el transporte a largas distancias de la voz.

Cada paquete contienen la dirección del nodo destino. En cada nodo intermedio por el que pasa el paquete se detiene el tiempo necesario para que éste analice la dirección de destino y seleccione el siguiente nodo del camino por el que deben retransmitirse los paquetes para hacerlos llegar a su destino. Este proceso se conoce como conmutación de paquetes.

Cuando todos los paquetes llegan al extremo receptor, son organizados y unidos de nuevo para reconstruir la información original.

A través de la historia de la computación, se han desarrollado muchos diferentes protocolos para definir la forma como se envían y conmutan los paquetes de datos, pero el protocolo más difundido y extendido hoy en día es el Protocolo de Internet IP.

Protocolo IP

El Protocolo de Internet (IP, de sus siglas en inglés Internet Protocol) es un protocolo no orientado a conexión usado tanto por el origen como por el destino para la comunicación de datos a través de una red de conmutación de paquetes.

Los conceptos más importantes de IP son el direccionamiento y el enrutamiento.

El direccionamiento se refiere a la forma como se asigna una dirección IP y como se dividen y se agrupan subredes de equipos.

El enrutamiento consiste en encontrar un camino que conecte una red con otra y es realizado principalmente por enrutadores que se especializan en recibir y enviar paquetes por diferentes interfaces de red, así como proporcionar opciones de seguridad, redundancia de caminos y eficiencia en

la utilización de los recursos.[40]

Pero el Protocolo IP es solo uno de los protocolos de la familia de protocolos en la que se basa Internet y que permiten la transmisión de datos entre redes de computadoras. A este conjunto de protocolos se le denomina Modelo TCP/IP, en referencia a los dos protocolos más importantes que la componen: Protocolo de Control de Transmisión (TCP) y Protocolo de Internet (IP), que fueron los dos primeros en definirse, y que son los más utilizados de la familia.

El Modelo TCP/IP tiene cuatro capas de abstracción según se define en el RFC 1122. Esta arquitectura de capas a menudo es comparada con el Modelo OSI[41] de siete capas:

Modelo TCP/IP	Modelo OSI	Ejemplos en Voz IP
4. Aplicación	7. Aplicación	*Servidores y servicios:* Asterisk, softphones, etc.
	6. Presentación	*Códecs:* G.711, G.729, GSM, H.264, etc.
	5. Sesión	*Protocolos de señalización:* SIP, SDP, IAX, MGCP, H.323, etc.
3. Transporte	4. Transporte	*Transporte de paquetes:* UDP, TCP, RTP, RTCP, etc.
2. Internet	3. Red	*El Protocolo IP*
1. Enlace de Datos	2. Enlace de datos	*Tipos de redes de datos:* Ethernet, 802.11, DSL, MPLS, etc.
	1. Física	

Tabla 6.1: Ejemplo de protocolos de voz por cada capa del Modelo TCP/IP

40 WIKIPEDIA. Internet Protocol. [en línea]. <http://es.wikipedia.org/wiki/Internet_Protocol> [citado en 29 de septiembre de 2011]

41 El modelo de referencia de Interconexión de Sistemas Abiertos (OSI, Open System Interconnection) lanzado en 1984 fue el modelo de red descriptivo creado por ISO para solucionar el problema de interoperabilidad entre los distintos tipos de tecnologías de internetworking existentes. Está dividido en siete capas: física, enlace de datos, red, transporte, sesión, presentación y aplicación.

Protocolos de Transporte

En el nivel de transporte de TCP/IP (según el modelo de referencia OSI) se pueden solucionar problemas como la fiabilidad de que los datos alcancen su destino y la seguridad de que los datos llegan en el orden correcto. En el conjunto de protocolos TCP/IP, los protocolos de transporte también determinan a que aplicación van destinados los datos. Los dos protocolos del nivel de transporte más utilizados son TCP y UDP, pero en Telefonía IP se utiliza principalmente RTP:

TCP

TCP (Transmission Control Protocol, en español Protocolo de Control de Transmisión) es un protocolo orientado a la conexión, estableciendo un circuito virtual (de manera análoga a la conmutación de circuitos de la telefonía tradicional) y desde este punto de vista no es utilizado para transmitir paquetes de voz, pues implementa diferentes mecanismos para asegurarse de que todos y cada uno de los paquetes lleguen a su destino generando una lentitud en el proceso de transmisión que no permite una buena calidad a la hora de tratar con tráfico multimedia en tiempo real.

UDP

User Datagram Protocol (UDP), a diferencia de TCP, está basado en el intercambio de unos paquetes de datos especiales llamados datagrama, sin que se haya establecido previamente una conexión, ya que el propio datagrama incorpora suficiente información de direccionamiento en su cabecera. Tampoco tiene confirmación, ni control de flujo, por lo que los paquetes pueden adelantarse unos a otros; y tampoco se puede asegurar que se reciban correctamente, ya que no hay confirmación de entrega o de recepción[42].

42 WIKIPEDIA. User Datagram Protocol. [en línea]. <http://es.wikipedia.org/wiki/User_Datagram_Protocol> [citado en 29 de septiembre de 2011]

RTP

El Protocolo de Transporte de Tiempo real (en inglés Real-time Transport Protocol) es un protocolo utilizado para la transmisión de información en tiempo real (audio, video, etc). Es considerado un protocolo de transporte en sí, aunque para su transmisión se encapsula dentro de datagramas UDP.

La información de voz y video debe ser codificada en paquetes de voz por medio de un códec antes de ser transmitida. RTP asigna un número de secuencia, que identifica la posición de cada paquete dentro del flujo de paquetes. También envía información sobre el tiempo, identificación y sincronización de los paquetes.

RTP cuenta con un protocolo asociado RTCP (Real Time Control Protocol) que se usa para enviar información sobre la calidad del flujo RTP, monitorear los paquetes, y medir variables como el retraso y variación de la frecuencia de la voz (Jitter).

RTP también soporta el envío de tonos DTMF a través de un tipo especial de paquete para eventos telefónicos, definido en el estándar RFC2833. Este mecanismo de transmisión de los tonos facilita su identificación cuando se utilizan códecs de compresión que eliminan las frecuencias altas que se emplean al transmitir los tonos en medio del flujo de voz (en banda).

Protocolos de Señalización

H323

H.323 es una recomendación ITU-T, que pertenece a la serie de protocolos H.32x, que describen las recomendaciones para implementar servicios multimedia sobre redes RDSI, SS7 y 3G[43]. H.323 se creó originalmente para proveer de un mecanismo para el transporte de aplicaciones multimedia en LANs (Redes de área local) pero evolucionó rápidamente para suplir las crecientes necesidades de las redes de VoIP, pese a que luego perdió adeptos y mercado cuando el protocolo SIP se convirtió en el estándar de-facto de la

43 WIKIPEDIA. H323. [en línea]. <http://es.wikipedia.org/wiki/H323> [citado en 29 de septiembre de 2011]

industria.

Principales características:

- Utiliza RTP para el transporte de audio y vídeo.
- No garantiza calidad de servicio en el transporte de datos.
- Está basado en el protocolo RDSI Q.931.
- Muy usado todavía en sistemas de videoconferencia sobre IP.

MGCP

MGCP (Media Gateway Control Protocol) es un protocolo de control de dispositivos, donde un gateway esclavo (MG, Media Gateway) es controlado por un maestro (MGC, Media Gateway Controller, también llamado Call Agent).

Es un protocolo muy utilizado en operadores de telefonía tradicional, que empezaron a ofrecer alternativas de interconexión entre suscriptores de telefonía IP y la red telefónica pública conmutada, y en donde es muy común encontrar softswitches con soporte para clientes MGCP.

SIP

El Protocolo de Inicio de Sesiones (en inglés Session Initiation Protocol) es un protocolo definido en el RFC 3261 del IETF. Es el estándar de-facto para la iniciación, modificación y finalización de sesiones interactivas de usuario donde intervienen elementos multimedia como el video y la voz.

Principales características:

- Independiente del transporte: puede transportarse sobre UDP o sobre TCP.

- Basado en texto y por lo tanto legible por seres humanos (al igual que otros protocolos como HTTP).

- SIP funciona en colaboración con otros protocolos pero solo interviene en la función de señalización al establecer y terminar la sesión de comunicación.

- SIP actúa como envoltura de SDP, protocolo que describe el contenido multimedia de la sesión, por ejemplo qué puerto IP y códec se usarán durante la comunicación, etc. Todas las comunicaciones de voz/video van sobre RTP (Real-time Transport Protocol) o RTCP (Real-time Transport Control Protocol).

- La definición de SIP es similar a la del protocolo HTTP (sintaxis, códigos de estado, orientado a esquemas de desafío-respuesta).

Mensajes y Respuestas

El protocolo SIP define varios tipos de mensajes:

- INVITE: Sirve para iniciar una sesión o cambiar una sesión actual (re-INVITE)

- ACK: Confirma el establecimiento de una llamada

- BYE: Termina una sesión

- CANCEL: Cancela una petición de inicio sesión.

- REGISTER: registra una localización con un servidor de registro SIP (SIP Registrar)

- Otros: OPTIONS, SUBSCRIBE, NOTIFY, MESSAGE

Para cada mensaje SIP, el destino puede enviar respuestas transitorias y luego una respuesta definitiva, o simplemente enviar una respuesta definitiva directamente.

Los códigos de la respuesta SIP se basan de los códigos de respuesta HTTP:

- 1xx: Provisorias: indican que el servidor contactado está realizando una cierta acción y todavía no tiene una respuesta definitiva (Ej: 100 Trying, 180 Ringing, etc.)

- 2xx: Éxito: la acción fue recibida, entendida, y aceptada con éxito. (Ej: 200 OK).

- 3xx: Redirección: Es necesario tomar acciones adicionales para terminar la petición (Ej: 302 Redirect).

- 4xx: Error de cliente: La petición contiene sintaxis errónea o no se puede llevar a cabo en este servidor (Ej: 404 Not Found, 480 Timeout).

- 5xx: Error de servidor: El servidor no pudo llevar a cabo una petición al parecer válida (Ej: 500 Internal Server Error).

- 6xx: Falla global: La petición no se pudo satisfacer en ningún servidor.

Elementos de una red SIP:

- ***User Agent:*** son los elementos que originan y reciben mensajes SIP. Tienen una parte cliente (UAC, User Agent Client) que envía peticiones y una parte servidor (UAS, User Agent Server) que las recibe y las procesa. Generalmente todo terminal SIP implementa este elemento.

- ***URI:*** Identificador Uniforme de Recursos (en inglés Uniform Resource Identifier). Es una cadena de caracteres empleada para identificar el nombre de un recurso, usuario o servicio en una red determinada (Ej: sip:alicia@dominio.com)

- ***Registrar/Location Server:*** Procesa las peticiones de registro de los User Agents que desean ser localizados en cierto dominio. El UA

envía un mensaje REGISTER incluyendo su URI y su información de contacto, para ser localizado luego dentro de la Red (Ver Figura X).

- ***Proxy:*** Un proxy SIP se encarga de enrutar las peticiones SIP a los UAS y las respuestas SIP a los UAC. Una petición puede atravesar varios proxys en su camino hacia un UAS. Cada uno tomará decisiones de enrutamiento, modificando la petición antes de enviarlo al elemento siguiente.

- ***Redirect Server:*** genera respuestas 3xx a las peticiones recibidas, ordenando al cliente entrar en contacto con una URI alterna.

En una red SIP normal, un UAC enviará la peticiones de inicio de llamada al servidor Proxy, el cuál utilizará la función de localización para dirigir la petición al UAS de destino, si está en su propio dominio, o para enviarlo a otro Proxy que sea responsable del dominio de destino. Esta topología se conoce como el Trapezoide SIP.

Sin embargo, a pesar de que implementa casi todas las funciones de los elementos SIP, Asterisk no actúa como Proxy, sino como un Back to Back User Agent (B2BUA), es decir, como un UAC y un UAS conectados internamente. Esto es lo que permite que Asterisk pueda comunicar tanto a dos terminales SIP, como a un terminal SIP y otro terminal que hable otro protocolo.

Ejemplo de registro de un usuario SIP en un servidor Asterisk:

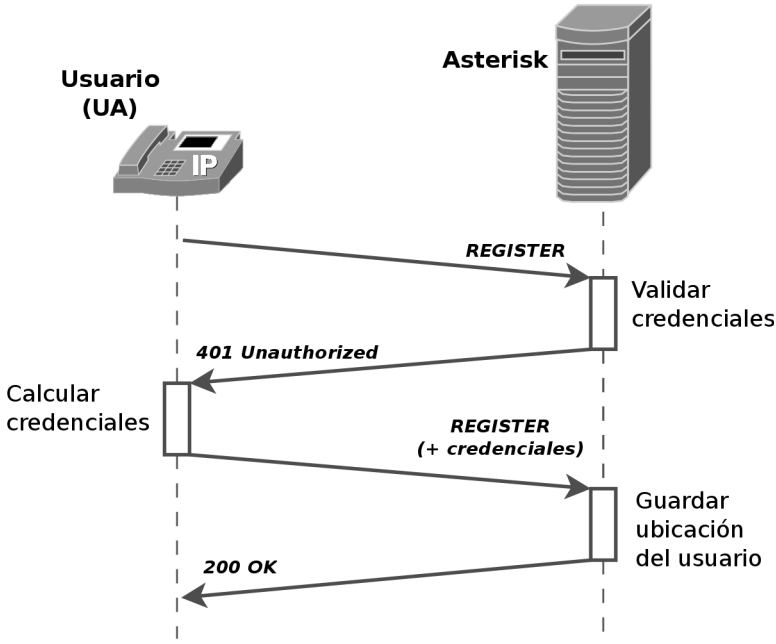


Figura 6.1: Secuencia de registro de un cliente SIP

El siguiente es el diagrama de una llamada típica usando el protocolo SIP:

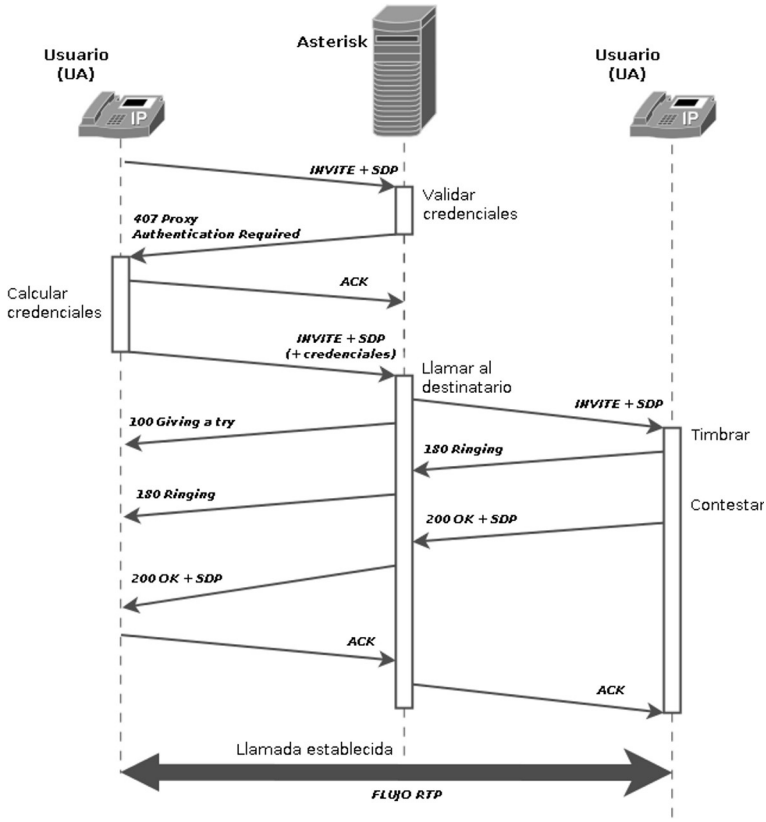


Figura 6.2: Flujo de mensajes en una llamada SIP

SDP

El protocolo de Descripción de Sesión (en inglés Session Description Protocol), es un formato para describir parámetros de inicialización de flujos multimedia y se encuentra definido en el RFC 4566.

SDP está enfocado hacia la descripción de el contenido multimedia de la sesión, por ejemplo qué puerto IP y códec se usarán durante la comunicación por parte de cada User Agent, etc. Normalmente la petición INVITE contiene la información SDP del UAC y la respuesta 200 OK contiene la información SDP del UAC.

Una sesión se describe con una serie de atributos, cada uno en una línea. Los nombres de estos atributos son un carácter seguido por '=' y el valor respectivo[44]:

```
v=  (Versión del protocolo)
o=  (Origen e identificador de sesión)
s=  (Nombre de sesión)
c=* (Información de conexión)
t=  (Tiempo durante el cual la sesión estará activa)
m=  (Nombre de medio y dirección de transporte)
a=* (Cero o más líneas de atributos de sesión)
```

En el Capítulo 11 se encuentra un ejemplo detallado de un mensaje SIP y problema que presenta en redes que implementan NAT (*Network Address Translation*).

IAX2

El protocolo de intercambio entre Asterisk (en inglés Inter-Asterisk eXchange protocol) se utiliza para manejar conexiones VoIP entre servidores Asterisk, y entre servidores y clientes con soporte para este protocolo.

El protocolo IAX tiene algunas ventajas sobre su contraparte (SIP), tales como:

- Trunking en las comunicaciones IAX (varios canales de comunicación comparten una misma cabecera de señalización,

44 WIKIPEDIA. Session Description Protocol. [en línea]. <http://es.wikipedia.org/wiki/Session_Description_Protocol> [citado en 29 de septiembre de 2011]

reduciendo el ancho de banda consumido por canal).

- El uso de un sólo puerto para señalización y envío del flujo de medios o datos (por lo general, este puerto es el 4569), a diferencia de SIP que requiere de un conjunto de puertos SIP y RTP para el envío de la voz.

- Mejor soporte para conexiones desde redes con NAT.

- El diseño de IAX se basó en muchos estándares de transmisión de datos, incluidos SIP (el cual es el más común actualmente), MGCP y Real-time Transport Protocol.

Ejemplo de una llamada IAX en Asterisk:

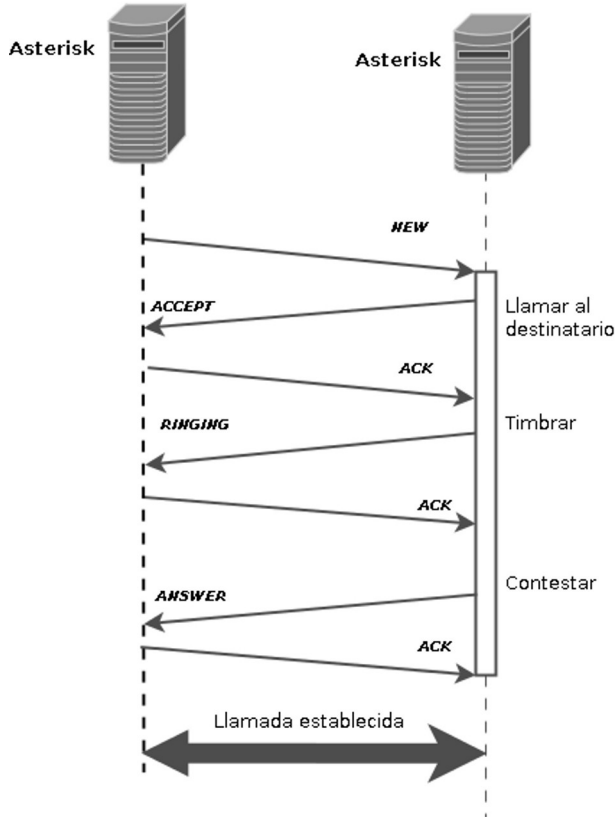


Figura 6.3: Secuencia de mensajes en una llamada IAX2

Códecs

Como se vio en el Capítulo 5, el proceso de digitalización de la voz involucra que ésta sea cuantificada y codificada antes de ser transmitida. En redes telefónicas tradicionales se utilizan códecs sin compresión, como G.711 pero para transmitir la voz a través de redes de datos se emplean

códecs con mayor capacidad de compresión, para optimizar el uso del ancho de banda de transmisión, así como con una alta tolerancia a pérdida de paquetes y errores en la red.

Algunos de los códecs más conocidos se presentan a continuación:

- ***G.711:*** Ofrece tasas de 64 Kbps en sus dos variantes, Ley-u y Ley-A. A pesar de su alto consumo de ancho de banda, es el códec básico que se encuentra en todos los equipos de telefonía.

- ***G.726:*** También conocido como ADPCM (MIC Diferencial Adaptativa), ofrece tasas de 16 Kbps, 24 Kbps, y 32 Kbps. Se basa en la modulación PCM, pero en lugar de transmitir cada muestra completa, solo se envía la diferencia entre la muestra actual y la anterior.

- ***GSM:*** Es un códec derivado de la telefonía móvil, como su nombre lo indica (Global System for Mobile Communications). Ofrece una muy buena relación de compresión versus utilización de CPU, ocupando un ancho de banda de 13Kbps.

- ***G.722:*** Al igual que G.711, es un estándar de la ITU-T, y ocupa un ancho de banda de 64Kpbs, pero con una calidad de la voz mucho mayor.

- ***Speex:*** Es un códec abierto y gratuito, publicado bajo licencia BSD. Es un códec VBR (Variable Bit Rate), lo que significa que puede cambiar su tasa de bits de acuerdo a las condiciones variables de la red de datos. El ancho de banda que ocupa está entre 2.15 y 22.4 Kbps.

- ***G.723.1:*** A pesar de que requiere licenciamiento, goza de creciente popularidad. Dependiendo del algoritmo utilizado, tiene una tasa de bits de 6.3Kbps o 5.3Kbps.

- ***G.729A***: Utiliza un complejo algoritmo de predicción fonética que le permite comprimir la voz hasta ocupar un ancho de banda de

8Kbps, aunque con un elevado costo de procesamiento de CPU. Debido a una patente, se requiere de una licencia para su uso, pero es muy popular entre los ITSP y los fabricantes de hardware.

- ***ILBC:*** Internet Low Bitrate Codec (iLBC), ofrece una atractiva calidad en conexiones de red de bajo ancho de banda, operando a 13.3 Kbps (tramas de 30-ms) y a 15.2 Kbps (tramas de 20-ms). Su uso requiere una licencia, pero tiene una versión gratuita.

Conectando Asterisk con VoIP

Protocolos y CODECS soportados

Asterisk tiene soporte completo para los protocolos IAX y SIP, y cuanta con un soporte parcial para los protocolos H323, MGCP y SCCP.

Asterisk puede conectar llamadas que utilicen diferentes códecs, haciendo la traducción (transcoding) entre ellos. Entre los códecs de libre uso que Asterisk soporta se encuentran: G.711 (ulaw y alaw), G.726 (a 32Kbps), GSM y Speex.

Para realizar transcoding usando iLBC, G.729 y G.723.1 se requiere de una licencia especial, sin embargo, Asterisk es capaz de establecer llamadas que usen este tipo de códecs en los dos extremos, siempre y cuando no se requiera usar servicios como voicemail o grabación de llamadas. Esto se conoce como modo "*Pass through*".

Users, Peers y Friends, Directiva Register

Asterisk maneja tres conceptos diferentes para las cuentas o canales que pueden definirse por cada tipo de protocolo de Voz IP[45]:

- ***User***: para permitir llamadas desde el exterior hacia Asterisk.

45 MADSEN, Leif, VAN MEGGELEN, Jim y BRYANT, Russell. Asterisk: The Definitive Guide. 3ra edición. O'Reilly Media, 2011

- ***Peer***: para enviar llamadas hacia el exterior.

- ***Friend***: se comporta como una combinación de “User” y “Peer”.

En los canales de tipo “***User***”, el nombre del contexto es el nombre de usuario que debe utilizar el origen de la llamada, junto con la contraseña especificada en el parámetro ”secret”. Si la información es correcta, el proceso de la llamada continúa en el contexto especificado en el plan de marcado. A continuación se muestra un ejemplo de la definición básica de un canal “User” en el archivo */etc/asterisk/sip.conf*:

```
[user_ejemplo]
type=user
secret=clave_secreta
context=menu_bienvenida
```

Se puede restringir la conexión desde ciertos rangos de direcciones IP, utilizando los parámetros “permit” y “deny”. Por ejemplo:

```
[user_ejemplo]
type=user
secret=clave_secreta
context=menu_bienvenida
deny=0.0.0.0/0       ; Prohibir la conexión desde toda IP,
permit=192.168.1.10  ; excepto desde esta dirección
permit=10.10.1.0/24  ; y desde este rango definido por la máscara
                     ; de red 255.255.255.0
```

De manera similar, se puede restringir el tipo de códecs que se permitirá al recibir llamadas por este canal, haciendo uso de los parámetros “allow” y “disallow”. El orden en que se especifican los códecs permitidos, será el orden de preferencia a la hora de negociar el códec con el extremo origen de la llamada. Por ejemplo:

```
[user_ejemplo]
type=user
secret=clave_secreta
context=menu_bienvenida
disallow=all          ; Deshabilitar todos los códecs,
allow=gsm             ; excepto gsm, que sería el preferido,
allow=ulaw            ; y también g.711 Ley-U
```

Los códecs por omisión para todos los canales se pueden definir dentro del contexto "*[general]*", y dentro de cada canal solo especificar las excepciones o particularidades de cada conexión.

Los canales de tipo "***Peer***" no tienen un contexto asociado, ya que no se usarán para procesar llamadas entrantes. En lugar de eso, tienen un parámetro "host" para indicar el extremo hacia donde se enviarán las llamadas salientes por este canal.

De igual manera, los canales de tipo "Peer" tienen un parámetro "qualify" que se usa para que Asterisk verifique periódicamente la disponibilidad del extremo de destino. Esto es muy útil cuando se requiere mantener la conexión abierta en ciertos tipos de redes con NAT.

```
[peer_ejemplo]
type=peer
fromuser=nombre_de_usuario
defaultuser=nombre_de_usuario
remotesecret=clave_secreta
host=192.168.1.10    ;Si el extremo remoto es el que se registra,
                     ; se usa host=dynamic.
                     ;Opcionalmente se puede especificar
                     ; un puerto de destino. Ej: port=5060
```

```
qualify=2000         ;Se puede poner qualify=yes, o igual al número
                     ; de milisegundos de latencia permitida.
```

Para realizar llamadas a través de un canal de tipo “Peer”, se utiliza la aplicación Dial, de la siguiente manera:

```
Dial(SIP/nombre_del_peer/extensión)
```

Comúnmente los proveedores de troncales SIP requerirán que Asterisk se registre periódicamente antes de aceptar llamadas provenientes de él. Para configurar esto, se utiliza la directiva “register” al final del contexto “[general]” tanto en el archivo sip.conf como en el iax.conf:

```
register=>usuario[:clave]@dirección_IP[:puerto][/extensión]
```

Un canal de tipo “***Friend***” es un alias para la mezcla de un “User” y un “Peer”. La mayoría de los canales se definen con este tipo de canal, pero hay algunos proveedores de servicio que requieren de la definición separada de un “User” y un “Peer”.

Para un canal de tipo “User”, Asterisk omitirá la autenticación si no se especifica el parámetro “secret”. Para un canal de tipo “Friend”, el parámetro “secret” se usa para las llamadas salientes, de modo que para omitir la autenticación en las llamadas entrantes se requiere de otro parámetro:

```
insecure=invite,port        ;
```

Usando Troncales SIP con Asterisk

En el Capítulo 3 se mostró la forma en que se configuran terminales SIP para su uso con Asterisk. A continuación se mostrará la configuración de na troncal SIP.

Cada troncal de Voz IP tiene parámetros específicos que pueden variar de un

operador a otro, pero en general, de deben solicitar los siguientes:

- Dirección IP o nombre de dominio del servicio. Ej: sip.ejemplo.com
- Nombre de usuario. Ej: mi_troncal
- Clave o contraseña. Ej: clave_secreta
- Códecs soportados. Ej: ulaw.

Si el proveedor de la troncal SIP requiere que Asterisk se registre periódicamente, se debe agregar la línea “register” al final del contexto “[general]” del archivo sip.conf:

```
[general]
...
register=>mi_troncal:clave_secreta@sip.ejemplo.com

[canal_1]
...
[canal_2]
....
```


Si la directiva “register” se define en otra parte del archivo, será ignorada.

Siguiendo el mismo ejemplo, la estructura del sip.conf para esta troncal SIP sería la siguiente:

```
[mi_troncal]
type=friend
fromuser=mi_troncal
```

```
defaultuser=mi_troncal
remotesecret=clave_secreta
host=sip.ejemplo.com
context=menu_bienvenida
disallow=all
allow=ulaw
insecure=invite,port
```

Para utilizar esta troncal para enviar llamadas hacia el proveedor SIP, se puede crear una nueva extensión que ejecute la aplicación “Dial”, y ubicarla en el contexto “troncales” del archivo extensions.conf:

```
[troncales]
exten = 011,1,Dial(SIP/011@mi_troncal,25) ;Extensión para
                                    ;marcar el número 011 a través
                                    ;de la troncal SIP

[usuarios]
...
include => troncales
```

Una vez reiniciado Asterisk para aplicar los cambios hechos a los archivos, se puede iniciar la Consola o Interfaz de Línea de Comandos (CLI), y ejecutar los siguientes comandos para verificar la configuración y el estado de la troncal SIP de ejemplo:

```
*CLI> sip show peers
```

Otros comandos útiles se muestran a continuación:

Para mostrar el estado de servidores sip donde asterisk se registra:

```
*CLI> sip show registry
```

Mostrar estado de una cuenta sip de tipo peer o friend:

```
*CLI> sip show peer <cuenta>
```

Mostrar estado de los canales sip (útil para validar códecs utilizados):

```
*CLI> sip show channels
```

Depurar mensajes SIP desde/hacia la dirección IP o la cuenta SIP:

```
*CLI> sip debug <ip/peer>
```

Mostrar la configuración global de los canales sip:

```
*CLI> sip show settings
```

Usando Troncales IAX con Asterisk

A pesar de que IAX2 es un protocolo con ventajas importantes frente a SIP, principalmente en cuanto a la solución de problemas relacionados con el NAT, hay muy pocos proveedores de voz sobre IP que ofrecen soporte para este protocolo. Sin embargo, su uso sí es muy popular cuando se trata de establecer conexiones entre dos o más servidores Asterisk.

Los canales IAX en Asterisk soportan tres tipos de autenticación: texto plano, MD5 y RSA. La autenticación en texto plano no es muy popular, debido a que es muy insegura, así que a continuación se presentan los dos métodos restantes:

Conexión mediante tipo de autenticación MD5

La autenticación mediante MD5 es similar a la utilizada por el protocolo SIP, se realiza un desafío y se espera una respuesta con el hash md5 de acuerdo a la clave, pero esta información viaja de manera clara por la red, lo cuál puede permitir que un atacante la intercepte.

Los parámetros para definir un canal IAX en el archivo iax.conf son muy parecidos a los de un canal SIP.

Para definir una troncal de ejemplo “troncal_iax”, se puede crear un canal de tipo “Friend” en el archivo iax.conf:

```
[troncal_iax]
type=friend
host=direccion.ip.otro.servidor
username=troncal_iax
context=menu_bienvenida
qualify=yes
auth=md5
secret= contraseñasecreta
```

También se podría definir un canal de tipo “user” y otro de tipo “peer”, y opcionalmente incluir una directiva “register” al final del contexto “[general]”, si se desconoce la dirección IP desde donde se conectará el otro extremo.

Si no se especifica el campo “secret”, el canal IAX toma las llamadas entrantes como invitados (guest) sin autenticación.

También cabe mencionar que se permite la definición de múltiples contextos dentro de cada canal IAX, lo cuál permite que el otro extremo pueda enviar llamadas a diferentes partes del plan de marcado adicionando “@contexto” al comando Dial.

Para utilizar esta troncal para enviar llamadas hacia el proveedor SIP, se puede crear una nueva extensión que ejecute la aplicación "Dial", y ubicarla en el contexto "troncales" del archivo extensions.conf:

```
[troncales]
exten = 011,1,Dial(IAX2/troncal_iax/011,60);Extensión para
                                    ;marcar el número 011 a través
                                    ;de la troncal IAX

[usuarios]
...
include => troncales
```

Conexión mediante autorización RSA

RSA (por Rivest, Shamir y Adleman) es un algoritmo para criptografía de llave pública. Cada extremo posee dos claves: una pública y otra privada. Cuando se quiere enviar un mensaje, el emisor cifra su mensaje con la clave pública del receptor, y una vez que el mensaje cifrado llega al receptor, este se ocupa de descifrarlo usando su clave privada[46].

RSA Fue el primer algoritmo que permitió ajustarse tanto para la firmas digitales como para el cifrado de información. Es ampliamente utilizado en protocolos de comercio electrónico y se considera seguro si se emplean llaves lo suficientemente seguras. En Asterisk, se emplean llaves RSA con mensajes de resumen SHA-1 para las firmas digitales.

Para crear las llaves se utiliza el script astgenkey, que se incluye en la instalación de Asterisk y se encarga de generar un par de llaves RSA en formato PEM (*Privacy-enhanced Electronic Mail*):

46 WIKIPEDIA. RSA. [en línea]. <http://es.wikipedia.org/wiki/RSA> [citado en 29 de septiembre de 2011]

Sintaxis:

```
astgenkey [ -q ] [ -n ] [ keyname ]
```

Por omisión, astgenkey creará llaves protegidas por una contraseña, y ésta deberá introducirse cada vez que se usen o una sola vez si se inicia Asterisk con la opción -i. Para evitar la solicitud de la contraseña, deberán crearse las llaves con la opción -n:

```
 astgenkey -n
```

astgenkey generará una llave pública (<llave>.pub) y una llave privada o secreta (<llave>.key)

La llave pública deberá copiarse en el directorio /var/lib/asterisk/keys del servidor Asterisk remoto con el que se quiera establecer la troncal IAX.

La llave privada deberá copiarse al directorio /var/lib/asterisk/keys del servidor Asterisk local.

Para revisar las llaves que se encuentran actualmente disponibles, se emplea el comando:

```
*CLI> show keys
```

Marcación empleando llaves RSA:

```
IAX2/[<user>:[nombre_llave@]<peer>[:<puerto>
[/<extension>[@<contexto>][/<opciones>]]
```

El nombre de la llave, excluye la extensión.

Archivo IAX.conf empleando llaves:

```
[troncal_iax_rsa]
type=friend
```

```
host=0.0.0.0
username=troncal_iax_rsa
context=entrada_iax
bandwidth=low
qualify=yes
auth=rsa
inkeys=pub_key
outkey=priv_key
```

Una alternativa a la declaración de un canal como de tipo friend es hacerlo como peer para la realización de llamadas y como user para la recepción de llamadas. De este modo, se emplea nuestra llave pública como outkey de un servidor Asterisk remoto y la llave pública de éste como inkeys nuestro.

Capítulo 7. El Plan de Marcado

> “La Auto-Educación es, estoy convencido, el único tipo de educación que existe”
>
> Isaac Asimov (1920 – 1992), escritor ruso.

En este capítulo se presentarán diferentes elementos que permitirán agregar mayor opciones de procesamiento al plan de marcado, muchos de ellos heredados de la programación estructurada.

Variables en Asterisk

Las variables le dan flexibilidad al plan de marcado de Asterisk. Permiten escribir el código de manera más sencilla y permiten la programación de tareas más complejas. Existen dos tipos de variables en Asterisk: variables globales y variables de canal.

De un lado, las variables de canal tienen valor únicamente durante la duración de la llamada y están asociadas al contexto del canal involucrado en la llamada. Las variables se asignan mediante la aplicación “Set” y el operador “=”. Por ejemplo:

```
Set(variable=valor)
```

Para recuperar o utilizar el valor almacenado en la variable, se utiliza la notación:

```
${variable}
```

Los nombres de las variables son sensibles a la capitalización. Las variables predefinidas por Asterisk están siempre en mayúsculas, así que se recomienda que las variables del usuario se definan en minúsculas o en combinación.

A continuación un ejemplo del uso de variables, haciendo uso de la aplicación “SayDigits” para escuchar los valores:

```
[servicios]
exten => 456,1,Answer()
exten => 456,n,Set(variable_A=1)
exten => 456,n,Set(variable_B=2)
exten => 456,n,Wait(1)
exten => 456,n,SayDigits(${variable_A}${variable_B})
```

Asterisk reemplaza el valor de cada variable, pasando los dígitos uno a lado del otro como argumento a la función: “SayDigits(12)”.

De otro lado, las variables globales afectan a todos los canales, contextos y extensiones del plan de marcado. Se utilizan a menudo para declarar constantes y se definen en la sección “[globals]” del archivo “/etc/asterisk/extensions.conf”. Por ejemplo:

```
[globals]
VariableGlobal1=valor
```

También se puede modificar su valor en cualquier lugar del plan de marcado, mediante la aplicación “Set” y la función “GLOBAL”:

```
Set(GLOBAL(VariableGlobal)=valor)
```

Herencia de Variables

Las variables de canal son específicas y tienen duración solamente en el contexto del canal actual de ejecución. Sin embargo, también es posible hacer que una variable sea heredada por un canal que sea creado por el canal actual, por ejemplo, cuando se ejecuta el comando Dial para conectar con otro canal.

Para esto, se declara la variable agregando un signo "_" al inicio de su nombre. Por ejemplo:

```
Set(_variable=valor)
```

También es posible declarar que la variable sea heredada a su vez por los canales creados por los canales hijos del canal inicial, por ejemplo, cuando una llamada es transferida por el destino. Para esto, se declara la variable agregando dos signos "_" al inicio de su nombre. Por ejemplo:

```
Set(__variable=valor)
```

La aplicación Read

Una forma especial de asignar un valor a una variable es obtener una serie de tonos por parte del usuario haciendo uso de la aplicación Read. La sintaxis es la siguiente:

```
Read(variable,archivo_de_sonido,dígitos,opciones,intentos,tiempo_li
mite)
```

Donde "variable" es el nombre de la variable a la que se asignarán los dígitos, "archivo_de_sonido" es el mensaje pre-grabado con las instrucciones para el usuario, "dígitos" es la cantidad máxima de dígitos que se leerán, "intentos" la cantidad de veces que se repetirá el archivo de sonido

si el usuario no marca nada y “tiempo_limite” la cantidad de segundos que se dará de espera para que el usuario ingrese los dígitos.

Cuando se ejecuta la aplicación, el usuario escuchará el mensaje e ingresará los dígitos. Si no se especificó una cantidad máxima de dígitos, deberá marcar la tecla de número “#” para terminar.

A continuación, un ejemplo donde se asignará a la variable “num” el valor de 5 dígitos ingresado por el usuario:

```
[servicios]
exten => 111,1,Answer()
exten => 111,n,Wait(2)
exten => 111,n,Read(num,vm-press&vm-extension,5)

exten => 111,n,SayDigits(${num})
```

Manipulación de Variables

Se pueden realizar varias operaciones sobre las variables, pero la más común es la de recortar la variable. La sintaxis para obtener una porción de una variable es la siguiente:

```
${variable:posición:largo}
```

Donde “posición” es el número de caracteres que Asterisk debe recortar desde el principio de la variable, y “largo” es el número de caracteres a retornar a partir de la posición. Si “posición” es negativo, se retorna el número de caracteres contando desde el final.

Por ejemplo, si la variable ${num} tiene el valor “12345”:

```
[servicios]
exten => 789,1,Answer()
exten => 789,n,Set(num=12345)
```

```
exten => 789,n,SayDigits(${num:2})        ; retorna “345”.
exten => 789,n,SayDigits(${num:1:3})      ; retorna “234”.
exten => 789,n,SayDigits(${num:-2}) ; retorna “45”.
exten => 789,n,SayDigits(${num:-2:1})     ; retorna “4”.
```

Variables Predefinidas

Asterisk proporciona un conjunto de variables predefinidas, que están disponibles para cada canal. Las más importantes son:

- ${CONTEXT}: Contexto actual de ejecución.
- ${EXTEN}: Extensión actual de ejecución.
- ${PRIORITY}: prioridad actual de ejecución.
- ${UNIQUEID}: Identificador único de la llamada.
- ${CHANNEL}: nombre del canal actual.
- ${HANGUPCAUSE}: Causa del cuelgue de la llamada.
- ${EPOCH}: La cantidad de segundos transcurridos desde la medianoche UTC (Tiempo Universal Coordinado) del 1 de enero de 1970.

Adicionalmente, muchas aplicaciones asignarán valores a determinadas variables dependiendo del resultado de su ejecución. Por ejemplo, la aplicación “Record” asigna a la variable “${RECORDED_FILE}” el nombre definitivo del archivo con la grabación realizada.

Otro ejemplo es el de la variable “${DIALSTATUS}”, a la cuál la aplicación “Dial” asigna el resultado de la llamada cuando termina, que puede ser:

- CHANUNAVAIL: canal no disponible.

- CONGESTION : la llamada no se pudo cursar.
- NOANSWER : no hay respuesta en el destino.
- BUSY : el destino está ocupado.
- ANSWER : la llamada fue contestada normalmente.
- CANCEL : la llamada fue cancelada por el llamante.
- DONTCALL, TORTURE: cuando se usan las opciones de privacidad.
- INVALIDARGS : argumentos inválidos.

En el Capítulo 4 se hizo una introducción al tema de los correos de voz, donde se mencionó que el usuario puede tener un mensaje diferente para cuando está ocupado y para cuando no está disponible. En el siguiente ejemplo, se utiliza el resultado de la aplicación “Dial” para especificar el mensaje a reproducir en los buzones de Alicia y Benito:

```
[usuarios]
exten = 2000,1,Dial(SIP/alicia,25)
exten = 2000,n,Goto(${DIALSTATUS}) ; Se salta a una etiqueta
                            ; de acuerdo al resultado de Dial.
exten = 2000,n(NOANSWER),VoiceMail(2000@default,u)
                                    ; Mensaje de “no disponible”
exten = 2000,n,Hangup()
exten = 2000,n(BUSY),VoiceMail(2000@default,b)
                                    ; Mensaje de “ocupado”
exten = 2000,n,Hangup()



exten = 2001,1,Dial(SIP/benito,25)
```

```
exten = 2001,n,Goto(${DIALSTATUS})
exten = 2001,n(NOANSWER),VoiceMail(2001@default,u)
exten = 2001,n,Hangup()
exten = 2001,n(BUSY),VoiceMail(2001@default,b)
exten = 2001,n,Hangup()
```

Funciones

Las funciones del plan de marcado permiten obtener y asignar valores sobre distintas propiedades de un canal. También sirven para enriquecer la complejidad del plan de marcado con muchas funcionalidades útiles.

Los nombres de las funciones siempre se escriben en mayúscula. La sintaxis de las funciones es similar al de las variables, pero con argumentos:

```
FUNCIÓN(argumentos)  ; para asignar un valor


${FUNCIÓN(argumentos)}      ; para obtener un valor
```

Al igual que con las variables, para signar un valor a una función debe usarse la aplicación “Set”. Por ejemplo, usando la función “CHANNEL” para cambiar el idioma del canal actual:

```
exten => 444,1,Answer()
exten => 444,n,Set(CHANNEL(language)=en)   ; Cambiar a inglés.
exten => 444,n,Playback(hello-world)
exten => 444,n,Set(CHANNEL(language)=es); Cambiar a español.
exten => 444,n,Playback(hello-world)
```

A continuación un ejemplo de cómo obtener o utilizar un valor de una función, usando la función “LEN” para calcular el tamaño de una cadena:

```
exten => 333,1,Answer()
```

```
exten => 333,n,Set(cadena=Hola Mundo)      ; Se define una cadena.
exten => 333,n,Set(largo=${LEN(${cadena})}); Se calcula el largo.
exten => 333,n,SayDigits(${largo})  ; Se reproduce el resultado.
exten => 333,n,Wait(2)
exten => 333,n,SayDigits(${LEN(Hola Mundo)})
                                    ; O todo en un solo paso.
```

De manera similar a la aplicaciones, para ver la lista de funciones disponibles en el sistema, se puede usar el siguiente comando:

```
*CLI> core show functions
```

Para ver el detalle de una función en particular, por ejemplo “CALLERID”:

```
*CLI> core show function CALLERID
```

Coincidencia de patrones

A veces se requiere hacer coincidir el número marcado con un rango mayor de extensiones sin tener que escribir lo mismo en el plan de marcado una y otra vez para todas las posibles combinaciones.

Cuando un usuario marca un número, Asterisk busca en el contexto actual de ejecución una extensión que coincida con el número marcado. Si no encuentra una coincidencia exacta, buscará el patrón que más fácilmente se ajuste al número marcado.

Los patrones se definen de igual manera que una extensión, pero antecedidos por un signo “_”, y pueden contener los siguientes elementos:

- **X**: Cualquier dígito del 0 al 9.
- **Z**: Cualquier dígito del 1 al 9.

- **N**: Cualquier dígito del 2 al 9.

Por ejemplo, para ejecutar una extensión que coincida con cualquier número de cuatro dígitos que empiece por 40:

```
exten => _40XX,1,Answer()
exten => _40XX,n,SayDigits(${EXTEN})
; ${EXTEN} toma el valor marcado.
```

De esta manera, coincidirán todos los número en el rango de 4000 a 4099.

El signo de punto es un comodín que coincide con uno o más dígitos. Es decir, si se quiere coincidir con todos los números que empiecen por 55, sin importar el tamaño, se podría escribir:

```
exten => _55.,1,Answer()
exten => _55.,n,SayDigits(${EXTEN})
```

No se recomienda el uso del comodín sin especificar un prefijo. Por ejemplo: "exten => **_.**,1,SayDigits(${EXTEN}) se ejecutará incluso para las extensiones especiales como "s", "i", "h", etc., lo cuál puede ocasionar serios problemas.

También se pueden usar corchetes para definir rangos de coincidencia. Por ejemplo:

- [2-5] coincidirá con cualquier número del 2 al 5.
- [34] coincidirá con el 3 o el 4.
- [1-367] coincidirá con 1, 2, 3, 6 o 7.

Así se pueden realizar combinaciones para obtener patrones más complejos. Por ejemplo, un patrón que coincida con cualquier número que empiece por 800 , 810 o 820, seguido de 7 dígitos del 0 al 9:

```
exten => _8[0-2]0XXXXXXX,1,Answer()
exten => _8[0-2]0XXXXXXX,n,SayDigits(${EXTEN})
```

Para un determinado contexto, es posible que existan varios patrones que coincidan para un número marcado. Por ejemplo, si se marca el número 4110, cualquiera de los siguientes patrones puede coincidir:

```
[servicios]
exten => _41XX,1,Answer()
exten => _41XX,n,SayAlpha(A) ; SayAlpha reproduce un archivo
                    ; pre-grabado correspondiente a la letra
exten => _411X,1,Answer()
exten => _411X,n,SayAlpha(B)

exten => _41ZX,1,Answer()
exten => _41ZX,n,SayAlpha(C)

exten => _4XX1,1,Answer()
exten => _4XX1,n,SayAlpha(D)
```

Para determinar cuál es la extensión que se debe ejecutar, Asterisk ordena los patrones siguiendo un conjunto de reglas:

- Evalúa dígito por dígito de izquierda a derecha.

- Para cada dígito, encuentra el patrón que tenga el menor número de posibilidades de coincidencia. Por ejemplo, X coincide con los números del 0 al 9, osea que tiene 10 posibles valores de coincidencia, y N coincide con los números del 2 al 9, osea que tiene 8 posibilidades de coincidencia.

- En caso de que haya empate entre dos patrones, se elige de acuerdo al orden alfabético según el código de caracteres ASCII[47].

En caso de duda sobre el orden asignado por Asterisk a los patrones, se puede utilizar el comando "dialpĺan show" de la CLI. Para el ejemplo anterior:

```
*CLI> dialplan show 4110@servicios
[ Context 'servicios' created by 'pbx_config' ]
  '_411X' =>        1. Answer()       [pbx_config]
                    2. SayAlpha(B)          [pbx_config]
  '_41ZX' =>        1. Answer()       [pbx_config]
                    2. SayAlpha(C)          [pbx_config]
  '_41XX' =>        1. Answer()       [pbx_config]
                    2. SayAlpha(A)          [pbx_config]
```

De esta manera, Asterisk ejecutará la extensión del patrón "_411X" cuando se marque la extensión 4110. Una característica que se debe tener en cuenta es que esto sucederá incluso cuando también exista una extensión 4110 explícitamente definida en el mismo contexto. Por ejemplo:

```
[servicios]
exten => 4110,1,Answer()
exten => 4110,n,SayDigits(${EXTEN}) ; Si no se llama a Hangup()
                                    ; en la extensión 4110
exten => _411X,1,Answer()
exten => _411X,n,SayAlpha(B)
exten => _411X,n,Playback(hello-world) ; al marcar 4110 también se
                                    ; ejecuta la siguiente prioridad
                                    ; del patrón coincidente.
```

47 Acrónimo en inglés de *American Standard Code for Information Interchange* - Código Estadounidense Estándar para el Intercambio de Información.

El resultado en este caso sería el siguiente:

```
   --   Executing   [4110@usuarios:1]   Answer("SIP/alicia-
0000001c", "") in new stack
  --   Executing   [4110@usuarios:2]   SayDigits("SIP/alicia-
0000001c", "4110") in new stack
 -- <SIP/alicia-0000001c> Playing 'digits/4.gsm' (language
'es')
 -- <SIP/alicia-0000001c> Playing 'digits/1.gsm' (language
'es')
 -- <SIP/alicia-0000001c> Playing 'digits/1.gsm' (language
'es')
 -- <SIP/alicia-0000001c> Playing 'digits/0.gsm' (language
'es')
  -- Executing [4110@usuarios:3] Playback("SIP/alicia-
0000001c", "hello-world") in new stack
  -- <SIP/alicia-0000001c> Playing 'hello-world.gsm'
(language 'es')
 -- Auto fallthrough, channel 'SIP/alicia-0000001c' status
is 'UNKNOWN'
```

Es por esto que se recomienda que siempre se ejecute la aplicación Hangup() de manera explícita si no se quiere continuar con la ejecución del plan de marcado.

Un importante ejemplo del uso de patrones es el de utilizar diferentes troncales para marcar hacia el exterior de la PBX. Por ejemplo, si un Asterisk tiene una troncal digital E1 y una troncal SIP, se pueden usar patrones para indicar que las llamadas locales se envíen a través del primario y que las llamadas de larga distancia se envíen a través de voz IP para que tengan un costo menor:

```
[troncales]
;Marcar números locales a través de los canales
```

```
;DAHDI del grupo No.1
exten = _NXXXXXX,1,Dial(DAHDI/g1/${EXTEN},25)

;Si se marca un número que empiece por 9 y tenga más de 8 dígitos,
; remover el 9 inicial y llamar al número restante a través
; de la troncal SIP
exten = _9XXXXXXXX.,1,Dial(SIP/${EXTEN:1}@mi_troncal,25)
```

Expresiones

Las expresiones en Asterisk permiten combinar variables, operadores y valores para obtener un resultado o comportamiento determinado dentro del flujo que sigue el plan de marcado. Son muy útiles para verificar datos, modificar variables, y realizar cálculos.

La sintaxis es la siguiente:

```
$[expresión1 operador expresión2]
```

Por ejemplo, dadas las variables “foo” y “bar”:

```
[servicios]
exten => 222,1,Answer()
exten => 222,n,Set(foo=$[1 + 2])           ;Expresión de suma.
exten => 222,n,Set(bar=$[2 * ${foo}]) ;Expresión de multiplicación.
exten => 222,n,SayDigits(${bar}) ;El valor final de “bar” será 6.
```

Los operadores usados en las expresiones pueden ser de varios tipos, como se menciona a continuación.

Operadores lógicos

Sirven para evaluar el nivel de verdad de una expresión. Pueden tomar dos posibles valores: Verdadero o Falso (uno o cero). Los operadores lógicos se resumen son los siguientes:

Multiplicación lógica (OR), el resultado será siempre 1 si al menos una de las expresiones es verdadera:

```
expr_1 | expr_2
```

Suma lógica (AND), el resultado siempre será 0 a menos que ambas expresiones sean verdaderas:

```
expr_1 & expr_2
```

Negación lógica, el resultado siempre es la inversión del valor de verdad de la expresión:

```
!expr
```

Operadores de comparación

Retornan el resultado de la comparación entre enteros, si las dos expresiones son numéricas, o la comparación de dos cadenas, de acuerdo al juego de caracteres del sistema.

```
expr_1 {=, !=, <, >, >=, <=} expr_2
```

Operadores matemáticos

Operadores de suma y sustracción:

```
expr_1 {+, -} expr_2
```

Operadores de multiplicación, división y módulo:

```
expr_1 {*, /, %} expr_2
```

Expresiones regulares

Asterisk permite utilizar expresiones regulares para evaluar una expresión sobre otra. La expresión regular debe ser el segundo argumento (expr2):

```
expr1 : expr2
```

La expresión será evaluada con un ancla al inicio de la cadena (un "^" explícito). Para evaluar la expresión sin el ancla al inicio:

```
expr1 =~ expr2
```

Condicionales

Los condicionales se utilizan para generar ciclos de decisión en el flujo del plan de marcado. En Asterisk hay dos estilos de estructuras de control o condicionales:

- Saltos y bucles con las aplicaciones GotoIf y GotoIfTime.
- Bucles con las aplicaciones While y EndWhile.

GotoIf

Evalúa una expresión y salta a otro lugar del plan de marcado dependiendo de su valor de verdad.

Sintaxis:

```
GotoIf($[expresión]?destino_1:destino_2)
```

Si la expresión es verdadera, el flujo salta a “destino_1”, y si es falsa, salta a “destino_2”. Si se omite cualquiera de los dos destinos, el flujo continua en la siguiente prioridad.

Los dos destinos se especifican de la misma manera que el argumento del comando “Goto”, es decir que pueden ser una combinación de “Contexto,Extensión,Prioridad” o una etiqueta[48].

Por ejemplo, saltar a un mensaje o colgar dependiendo de la evaluación de una variable:

```
[servicios]
exten => 222,1,Answer()
exten => 222,n,Set(foo=$[1 + 2])
exten => 222,n,Set(bar=$[2 * ${foo}])
exten => 222,n,SayDigits(${bar})
exten => 222,n,GotoIf($[${bar} = 6]?monos:colgar)
exten => 222,n(monos),Playback(tt-monkeysintro&tt-monkeys)
exten => 222,n(colgar),Hangup()
```

También se puede utilizar “GotoIf” para realizar bucles, por ejemplo:

```
[servicios]
exten => 555,1,Answer()
exten => 555,n,Set(contador=1)
exten => 555,n(bucle),SayDigits(${contador})
exten => 555,n,Set(contador=$[${contador} + 1])
exten => 555,n,GotoIf($[${contador} > 3]?:bucle)
exten => 555,n,Playback(vm-goodbye)
```

48 Ver más sobre “Goto” en el Capítulo 4.

GotoIfTime

Esta aplicación también permite hacer saltos, pero en lugar de evaluar una expresión, verifica si la hora del sistema coincide con un patrón de horario determinado y salta a un destino en el plan de marcado.

La sintaxis es similar a la de la aplicación “GotoIf”, pero la expresión es un patrón de tiempo:

```
GotoIfTime(horas,días_Semana,días_mes,meses?destino_1:destino_2)
```

- “horas” : Una hora o rango de tiempo en formato de 24 horas, desde las 00:00 hasta las 23:59. Por ejemplo: “08:00-18:00”.

- “días_semana” : Un día de la semana o rango de días, en el siguiente formato en Inglés: mon, tue, wed, thu, fri, sat y sun. También se pueden especificar días no consecutivos con el operador “&”. Por ejemplo: “mon-wed&fri”.

- “días_mes” : Un día del mes o rango de días, del 1 al 31. También se pueden especificar días no consecutivos con el operador “&”. Por ejemplo: “1-15&20”.

- “meses” : Un mes o rango de meses, en el siguiente formato en Inglés: jan, feb, mar, apr, may, jun, jul, aug, sep, oct, nov y dec. También se pueden especificar días no consecutivos con el operador “&”. Por ejemplo: “jan&mar-jun”.

- En cualquiera de estos ítems, se puede usar el signo “*” como comodín.

Por ejemplo, para saltar al menú IVR de operadora automática, de lunes a viernes de 8 am a 6 pm, todos los días del año:

```
GotoIfTime(08:00-17:59,mon-fri,*,*?
menu_bienvenida,s,1)
```

Retomando el ejemplo de operadora automática del Capítulo 4, evaluar si la llamada entra en un día festivo (1 de enero) o por fuera del horario laboral, y reproducir un mensaje diferente:

```
[menu_bienvenida]
exten => s,1,Answer()
exten => s,n,GotoIfTime(*,*,1,jan?no_laboral,s,1)
exten => s,n,GotoIfTime(08:00-17:59,mon-fri,*,*?menu)
exten => s,n,Goto(no_laboral,s,1)
exten => s,n(menu),Background(bienvenida)
exten => s,n,WaitExten(2)


[no_laboral]
exten => s,1,Answer()
exten => s,n,Playback(tt-monkeysintro&tt-monkeys)
exten => s,n,Hangup()
```

While y EndWhile

Otra forma de realizar bucles es a través de las aplicaciones “While” y “EndWhile”. La aplicación “While” evalúa una expresión y si es verdadera ejecuta las líneas de plan de marcado hasta la línea con “EndWhile”.

Por ejemplo:

```
[servicios]
exten => 888,1,Answer()
exten => 888,n,Set(contador=1)
exten => 888,n,While($[${contador} < 4])
exten => 888,n,SayDigits(${contador})
exten => 888,n,Set(contador=$[${contador} + 1])
exten => 888,n,EndWhile()
exten => 888,n,Playback(vm-goodbye)
```

Operadora Automática Mejorada

Aplicando todo lo visto en este capítulo, a continuación se presenta una versión mejorada de la operadora automática del Capítulo 4, donde se le da la oportunidad al llamante de volver a introducir una extensión válida tres veces antes de colgar:

```
[menu_bienvenida]
exten => s,1,Answer()
exten => s,n,Set(intentos=0)
exten => s,n,GotoIfTime(*,*,1,jan?no_laboral,s,1)
exten => s,n,GotoIfTime(08:00-17:59,mon-fri,*,*?menu)
exten => s,n,Goto(no_laboral,s,1)
exten => s,n(menu),Background(bienvenida)
exten => s,n,WaitExten(5)

exten => 1,1,Playback(digits/1)
exten => 1,n,Goto(s,1)

exten => 2,1,Playback(digits/2)
```

```
exten => 2,n,Goto(s,1)

exten => t,1,Playback(vm-pls-try-again)
exten => t,n,Set(intentos=$[${intentos} + 1])
exten => t,n,GotoIf($[${intentos} > 2]?colgar)
exten => t,n,Goto(s,menu)
exten => t,n(colgar),Hangup()

exten => i,1,Playback(invalid)
exten => i,n,Set(intentos=$[${intentos} + 1])
exten => i,n,GotoIf($[${intentos} > 2]?colgar)
exten => i,n,Goto(s,menu)
exten => i,n(colgar),Hangup()

include => usuarios
```

Capítulo 8. Configuración y programación avanzada de servicios de voz

> “Sólo es posible avanzar cuando se mira lejos”
>
> José Ortega y Gasset (1883 – 1957), filósofo y periodista español.

Macros

Retomando el ejemplo de configuración de una extensión con correo de voz expuesto en en el capítulo 7:

```
[usuarios]
exten = 2000,1,Dial(SIP/alicia,25)
exten = 2000,n,Goto(${DIALSTATUS})
exten = 2000,n(NOANSWER),VoiceMail(2000@default,u)
exten = 2000,n,Hangup()
exten = 2000,n(BUSY),VoiceMail(2000@default,b)
```

Si además de correo de voz se agregan otros servicios a la extensión, y si en un mismo Asterisk no se tienen pocas extensiones, sino decenas o cientos, el mantenimiento del plan de marcado puede llegar a ser bastante dispendioso.

La aplicación Macro se utiliza para invocar subrutinas o plantillas de tareas que se pueden escribir una sola vez y que luego se repiten a lo largo del plan de marcado. La sintaxis es la siguiente:

```
Macro(nombre[,argumento1[,argumento2[,...]]])
```

De esta manera, se puede invocar la subrutina muchas veces, incluyendo los argumentos o parámetros de acuerdo a cada extensión:

```
[usuarios]
exten = 2000,1,Macro(marcar-usuario,SIP/alicia)
exten = 2001,1,Macro(marcar-usuario,SIP/benito)
exten = 2002,1,Macro(marcar-usuario,SIP/carlos)
exten = 2003,1,Macro(marcar-usuario,DAHDI/1)
; y así sucesivamente ...
```

La definición de un macro es similar a la de un contexto, simplemente el nombre entre corchetes con el prefijo “macro-”, y se utiliza la extensión especial “s” para ejecutar las acciones correspondientes. A continuación se muestra cómo quedaría el macro de ejemplo “marcar-usuario”:

```
[macro-marcar-usuario]
exten = s,1,Dial(${ARG1},25)
exten = s,n,Goto(${DIALSTATUS})
exten = s,n(NOANSWER),VoiceMail(${MACRO_EXTEN}@default,u)
exten = s,n,Hangup()
exten = s,n(BUSY),VoiceMail(${MACRO_EXTEN}@default,b)
```

La variable *${ARG1}* representa al primer argumento con el que se invoque la aplicación, el segundo argumento sería *${ARG2}*, etc.

El contexto, extensión y prioridad desde donde se invoca el Macro son pasados a este como las variables *${MACRO_EXTEN}*, *${MACRO_CONTEXT}* y *${MACRO_PRIORITY}*, respectivamente. Por ejemplo, si se ejecuta el Macro para la extensión 2000, la variable *${MACRO_EXTEN}* tendrá el valor 2000, y se invocará el correo de voz del usuario Alicia.

Cuando la subrutina termina, la ejecución del plan de marcado continúa en el punto desde donde se invocó el macro.

Por la forma como está implementada la aplicación Macro, ésta tiene una limitación en cuanto al número de llamados recursivos que se pueden hacer, de hasta 7 subrutinas. Es por esto que se implementó una nueva aplicación llamada GoSub.

GoSub

Con la aplicación GoSub se puede saltar a otro punto del plan de marcado, de manera similar a la aplicación Goto, pero adicionalmente acepta argumentos y permite el retorno cuando se termina la subrutina como la aplicación Macro.

Su sintaxis es la siguiente:

```
Gosub([[contexto,]extensión,]prioridad[(arg1[,...][,argN])])
```

Continuando con el anterior ejemplo de los macros, si se utilizara GoSub el plan de marcado quedaría de la siguiente manera:

```
[usuarios]
exten = 2000,1,Gosub(marcar-usuario,s,1(SIP/alicia,${EXTEN}))
exten = 2001,1,Gosub(marcar-usuario,s,1(SIP/benito,${EXTEN}))
exten = 2002,1,Gosub(marcar-usuario,s,1(SIP/carlos,${EXTEN}))
exten = 2003,1,Gosub(marcar-usuario,s,1(DAHDI/1,${EXTEN}))
```

Nótese que en este caso, se debe indicar cuál es la extensión como un segundo parámetro *${ARG2}*, ya que no se cuenta con la variable *${MACRO_EXTEN}*.

Y el contexto de la subrutina quedaría de la siguiente manera:

```
[marcar-usuario]
exten = s,1,Dial(${ARG1},25)
exten = s,n,Goto(${DIALSTATUS})
exten = s,n(NOANSWER),VoiceMail(${ARG2}@default,u)
exten = s,n,Return(${DIALSTATUS})
exten = s,n(BUSY),VoiceMail(${ARG2}@default,b)
exten = s,n,Return(${DIALSTATUS})
```

A diferencia de la aplicación Macro, con la aplicación GoSub siempre que se desee retornar al punto desde donde se invocó la subrutina, se debe utilizar la aplicación Return, que adicionalmente permite retornar un valor.

GoSubIf, es el equivalente a GotoIf dentro de una subrutina. La diferencia es que GosubIf almacena el punto donde se debe retornar una vez termine la subrutina y se llame la función Return().

Adicionalmente, dentro de una subrutina invocada con GoSub se puede usar la función "Local()" para leer y escribir variables que tendrán valor sólo dentro de la subrutina, y que desaparecerán una vez se llame a la aplicación Return, como se verá en los ejemplos de la siguiente sección.

AstDB

Asterisk cuenta con una base de datos interna llamada AstDB que puede utilizarse para almacenar información de acceso rápido. Los datos se agrupan en familias y se identifican mediante una llave única dentro de esa familia.

Hasta Asterisk 1.8, la base de datos utilizaba el formato de base de datos de Berkley, muy similar al del registro de Windows, pero a partir de Asterisk 10 el formato cambió a SQLite 3[49].

49 Asterisk Project Wiki. SQLite3 astdb back-end. [en línea]. <https://wiki.asterisk.org/wiki/display/AST/SQLite3+astdb+back-end> [citado en 29 de noviembre de 2011]

Desde el plan de marcado, se cuenta con los siguientes elementos para manipular la base de datos:

- DB(familia/llave): Función para leer y escribir un valor en la base de datos.
- DB_DELETE(familia/llave): Función para borrar un valor de la base de datos.
- DBdeltree(familia): Aplicación para borrar toda una familia de la base de datos.

Con estos elementos es posible implementar muchos servicios. A continuación se muestra cómo se puede agregar el servicio de No-Molestar (DND - Do No Disturb) para los usuarios de una PBX:

Primero, se agrega la extensión *78 para que el usuario indique que no desea recibir llamadas:

```
[servicios]
exten => *78,1,Answer
exten => *78,n,Set(DB(DND/${CALLERID(num)})=True)
exten => *78,n,Playback(beep)
exten => *78,n,Wait(2)
exten => *78,n,Hangup
```

Luego se crea la extensión *79 para deshabilitar el No-Molestar:

```
exten => *79,1,Answer
exten => *79,n,NoOp(${DB_DELETE(DND/${CALLERID(num)})})
exten => *79,n,Playback(beep)
exten => *79,n,Wait(2)
exten => *79,n,Hangup
```

Y la subrutina de marcar al usuario donde se evalúa si éste tiene el DND

activo o no, quedaría de la siguiente manera:

```
[marcar-usuario]
exten = s,1,Set(LOCAL(dnd)=${DB(DND/${ARG2})})
exten = s,n,Gosubif($[${dnd}]?ocupado:marcar)
exten = s,n(ocupado),Return(DND activo)
exten = s,n(marcar),Dial(${ARG1},25)
exten = s,n,Goto(${DIALSTATUS})
exten = s,n(NOANSWER),VoiceMail(${ARG2}@default,u)
exten = s,n,Return(${DIALSTATUS})
exten = s,n(BUSY),VoiceMail(${ARG2}@default,b)
exten = s,n,Return(${DIALSTATUS})
```

También es posible modificar entradas en la base de datos desde la interfaz de línea de comandos:

Agregar entradas de AstDB:

```
*CLI> database put <familia> <llave> <valor>
```

Eliminar entradas de AstDB:

```
*CLI> database del <familia> <llave>
```

Eliminar todas las entradas de AstDB de la misma família:

```
*CLI> database deltree <familia>
```

Asterisk Manager Interface (AMI)

AMI (Asterisk Management Interface) es una interfaz de gestión remota de Asterisk que utiliza el modelo cliente/servidor sobre TCP, permitiendo que

aplicaciones externas puedan monitorear y controlar la PBX, realizar llamadas y ejecutar comandos[50].

Antes de enviar comandos a la interfaz AMI, la aplicación cliente debe iniciar sesión. Los usuarios de gestión se configuran en el archivo */etc/asterisk/manager.conf*, donde se especifican los permisos que tendrán para ejecutar diferentes acciones (write) y para monitorear ciertos eventos (read):

```
[general]
enabled = yes
port = 5038
bindaddr = 127.0.0.1

[usuario_manager]
secret = clavesupersecreta
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,originate
write = system,call,log,verbose,command,agent,user,originate
```

La interfaz AMI recibe comandos llamados “acciones”, las cuales generan “respuestas” por parte de Asterisk. Adicionalmente, Asterisk envía “eventos” que informan de diferentes cambios en la PBX.

Los mensajes que intercambian el cliente y el servicio de AMI utilizan un protocolo simple basado en lineas con “llave: valor”. Cada línea finaliza con un salto de línea (CR/LF), y para terminar el comando, se utiliza un doble salto de línea.

Para ver todos los comandos disponibles en AMI, se puede usar el siguiente comando en la consola de Asterisk:

50 Asterisk Project Wiki. Asterisk Manager Interface. [en línea]. <https://wiki.asterisk.org/wiki/display/AST/Asterisk+Manager+Interface+(AMI)> [citado en 29 de noviembre de 2011]

```
*CLI> manager show commands
```

Y se puede obtener más información de un comando específico con:

```
*CLI> manager show command <comando>
```

Por ejemplo, para iniciar sesión con AMI, se deber abrir una socket que se conecte por TCP al puerto de AMI y enviar un comando con la acción "Login":

```
Action: Login
Username:  usuario_manager
Secret: clavesupersecreta
```

Una vez iniciada la sesión, se puede iniciar una llamada desde el usuario "alicia" hacia el usuario "benito", utilizando la acción "Originate":

```
Action: Originate
Channel: sip/alicia
Exten: 2001
Context: usuarios
```

De esta manera es posible controlar Asterisk desde cualquier aplicación externa, sin importar el lenguaje en el que esté escrita. Hay también muchas herramientas disponibles en el mercado que hacen uso de la interfaz AMI para monitorear y controlar sistemas basados en Asterisk.

AJAM

Además de AMI, Asterisk incluye una interfaz javascript llamada AJAM (Asynchronous Javascript Asterisk Manger) que permite ejecutar los mismos comandos a través de HTTP. Esto permite que tanto navegadores como otras aplicaciones Web puedan monitorear y controlar la PBX.

Para utilizar AJAM, se debe configurar un servidor HTTP externo, o simplemente usar el servidor embebido que incluye Asterisk. Para esto, se debe editar el archivo */etc/asterisk/http.conf*:

```
enabled=yes          ;Habilitar el servidor embebido
enablestatic=yes     ;Generar contenido estático (imágenes, etc.)
bindaddr=0.0.0.0     ;Dirección IP donde escuchará el servicio.
Bindport=8080        ;Puerto
prefix=asterisk      ;Prefijo a usar en las URL de las peticiones.
```

También se debe modificar el archivo /etc/asterisk/manager.conf para habilitar el soporte de AJAM:

```
[general]
enabled = yes
port = 5038
bindaddr = 127.0.0.1
webenabled = yes
httptimeout = 3600   ;Duración máxima de la sesión HTTP en segundos
...
```

Una vez se finaliza esta configuración, se puede recargar o reiniciar asterisk, y empezar a enviar comandos a través del navegador. Por ejemplo, para iniciar sesión:

```
http://<direccion_ip_asterisk>:8080/asterisk/manager?
action=login&username=usuario_manager&secret=clavesupersecreta
```

Luego de eso, Asterisk guarda una cookie en el navegador que tiene validez hasta que expira el tiempo definido en “httptimeout”, y que permite continuar enviando comandos.

Para ver las respuestas en el mismo formato de texto plano de AMI, se deben enviar las peticiones cambiando “manager” por “rawman”:

```
http://<direccion_ip_asterisk>:8080/asterisk/rawman?action=status
```

Y para recibir las respuestas en formato AJAX:

```
http://<direccion_ip_asterisk>:8080/asterisk/mxml?action=status
```

Adicionalmente, habiendo habilitado la opción “enablestatic”, se puede utilizar una interfaz web de ejemplo, entrando en la siguiente URL:

```
http://<direccion_ip_asterisk>:8080/asterisk/static/ajamdemo.html
```

Finalmente, el código fuente de Asterisk incluye una librería javascript de ejemplo (astman.js), con el fin de facilitar la creación de interfaces de gestión Web.

Marcación por archivo

Otra forma de generar llamadas programadas es a través de una cola de archivos especiales que Asterisk procesa periódicamente en la carpeta */var/spool/asterisk/outgoing/*. Los archivos deben tener un formato especial y tener la extensión ”*.call*”[51].

El formato del archivo consisten en pares “llave: valor”, uno por línea. Por ejemplo, para programar una llamada entre el usuario “alicia” y la extensión 2001, se puede crear el siguiente archivo “ejemplo_*llamada.call*”:

```
Channel: SIP/alicia  ;Canal hacia donde se llamará, con el
                     ; mismo formato de la aplicación Dial.
WaitTime: 25         ;Tiempo a esperar a que el destino conteste.
MaxRetries: 3        ;Número de re-intentos de llamada.
RetryTime: 40        ;Tiempo para volver a intentar la llamada.
```

51 Asterisk Project Wiki. Asterisk Call Files. [en línea]. <https://wiki.asterisk.org/wiki/display/AST/Asterisk+Call+Files> [citado en 29 de noviembre de 2011]

```
Archive: yes        ;Si se quiere guardar una copia al terminar.

Context: usuarios   ;Contexto,extensión y prioridad a conectar
Extension: 2001     ; cuando la llamada sea contestada.
Priority: 1

; Alternativamente, se puede especificar una aplicación
; y sus argumentos:
;Application: Macro
;Data: marcar-usuario,SIP/benito
```

El archivo debe pertenecer a asterisk y tener permisos de escritura:

```
chown -R asterisk: prueba_llamada.call
chmod -R 777 prueba_llamada.call
```

Luego se debe copiar o mover a la carpeta de encolamiento, y la llamada será procesada por Asterisk:

```
cp -p prueba_llamada.call  /var/spool/asterisk/outgoing/
```

No se debe crear el archivo directamente en la carpeta “outgoing”, porque Asterisk hace un escaneo periódico y puede procesar el archivo solo parcialmente.

El procesamiento del archivo termina cuando la llamada se contesta, o cuando se terminan los re-intentos de marcación. Si se especificó la opción de “Archive”, se guarda una copia del archivo en la carpeta “*/var/spool/asterisk/outgoing_done*”, incluyendo al final del mismo, el resultado de la llamada:

```
Status: Expired/Completed/Failed
```

Finalmente, es posible indicarle a Asterisk el momento exacto en el que se quiere que se procese el archivo .call, en lugar de procesarlo inmediatamente. Para hacer esto, antes de copiarlo al directorio “outgoing” se debe usar el comando “touch” de linux para cambiar la fecha de modificación del archivo, la cuál será la fecha y hora en la que Asterisk inicie la llamada:

```
touch -t 201212312359 prueba_llamada.call
```

Programación con AGI

AGI (Asterisk Gateway Interface), es una interfaz abierta de comandos que permite que Asterisk pueda comunicarse con un programa externo para ejecutar acciones sobre las llamadas, extendiendo la potencia del plan de marcado al poder invocar programas que interactúen con Bases de Datos, Servicios Web y cualquier otro recurso externo.

Los scripts AGI se comunican con Asterisk a través de los flujos de entrada, salida y error, conocidos en programación como *STDIN*, *STDOUT*, y *STDERR*, de la siguiente manera:

- El script de AGI lee el flujo *STDIN* para obtener información desde Asterisk.
- El script de AGI escribe datos en el flujo *STDOUT* para enviar información hacia Asterisk.
- El script de AGI tiene la posibilidad de escribir datos en el flujo *STDERR* para enviar información de depuración hacia la consola de Asterisk.

La comunicación entre Asterisk y el script AGi sigue siempre la siguiente secuencia:

1. Cuando inicia el script, Asterisk le envía una lista de variables de entorno con sus valores.

2. Asterisk envía una línea en blanco para indicar que terminó de enviar las variables.

3. A partir de este punto, el script puede empezar a enviar comandos a Asterisk a través del flujo *STDOUT*.

4. Cada vez que el script envía un comando, Asterisk envía una respuesta que el script debe capturar.

5. Se repite este último paso hasta que termina la ejecución del script.

Los comandos que puede enviar el script de AGI pueden revisarse ejecutando los siguientes comandos en la consola de Asterisk:

```
*CLI> agi show commands
*CLI> agi show commands topic [commando]
```

Existen APIs para AGIs en los lenguajes de programación más populares, como python, java, php, etc. Pero a continuación se muestra un ejemplo con la manera más sencilla de scripting, usando el intérprete de comandos de Linux, *bash*.

Como ejemplo, se puede crear el siguiente archivo *"/var/lib/asterisk/agi-bin/prueba.agi":*

```
#!/bin/bash
# 1. Se leen las variables de entorno enviadas por Asterisk:
declare -a variables
while read -e ARG && [ "$ARG" ] ; do
      variables=(` echo $ARG | sed -e 's/://'`)
      export ${variables[0]}=${variables[1]}
done
```

```
# 2. Se imprimen las variables en STDERR, a modo informativo:
echo $agi_request >&2
echo $agi_channel >&2
echo $agi_language >&2
echo $agi_type >&2
echo $agi_uniqueid >&2
echo $agi_callerid >&2
echo $agi_dnid >&2
echo $agi_rdnis >&2
echo $agi_context >&2
echo $agi_extension >&2
echo $agi_priority >&2
echo $agi_enhanced >&2

# 3. Se define una función para procesar las respuestas a los
#    comandos enviados a Asterisk:
procesar_resultado() {
      while read line
      do
      case ${line:0:4} in
      # Resultado exitoso:
      "200 " ) echo $line >&2
               return;;
      # Comando inválido:
      "510 " ) echo $line >&2
               return;;
      # Comando no permitido en un canal terminado:
      "511 " ) echo $line >&2
               return;;
      # Fin de uso permitido:
```

```
      “520 ” ) echo $line >&2
               return;;
      *      ) echo $line >&2;;
      esac
      done
}
# 4. Se envía el comando para ejecutar sayNumber:
echo “Probando 'saynumber' ...” >&2
echo “SAY NUMBER 123456 \“\““
procesar_resultado

# 5. Se envía el comando para ejecutar Playback:
echo “Probando playback” >&2
echo “STREAM FILE hello-world \“\” “
procesar_resultado
```

Los scripts deben tener permisos de ejecución para el usuario “asterisk”:

```
chown -R asterisk: prueba.agi
chmod -R 755 prueba.agi
```

Para invocar un script AGI desde el plan de marcado, se debe utilizar alguna de las siguientes aplicaciones, provistas por el módulo “res_agi.so”:

- **AGI**: puede controlar el plan de marcado.
- **EAGI**: además de controlar la ejecución del plan de marcado, puede tener acceso al audio de la llamada.
- **FastAGI**: puede utilizarse para invocar scripts remotamente a través de la red de datos.
- **DeadAGI**: da acceso al canal, aún después de haber sido colgada la

llamada.

Asterisk busca los scripts en la carpeta "*/var/lib/asterisk/agi-bin*", pero también se puede especificar la ruta absoluta al script.

Utilizando la aplicación AGI(), el script se invocaría así:

```
[servicios]
exten =>999,1,Answer()
exten =>999,2,AGI(prueba.agi)
```

Una vez se recargue el plan de marcado, se puede habilitar la depuración de AGI para revisar su funcionamiento:

```
*CLI> agi set debug on
```

Al hacer la llamada a la extensión 456, se podrá observar la ejecución del script en la consola de Asterisk.

Configuración dinámica con ARA

La arquitectura de tiempo real de Asterisk (ARA por sus siglas en inglés - *Asterisk Realtime Architecture*) es un conjunto de módulos que permiten cargar la configuración desde una base de datos o desde otras fuentes dinámicas de información.

Este mecanismo presenta beneficios desde el punto de vista de flexibilidad y facilidad en la administración de Asterisk, pero supone también una disminución en la capacidad de procesamiento, debido a la carga adicional que implica el consultar una base de datos frente a los archivos de configuración.

ARA se configura en el archivo "/etc/asterisk/extconfig.conf", y soporta la carga dinámica de la siguiente información:

- **sippeers, sipusers**: usuarios y *peers* SIP.
- **sipregs**: usuarios SIP registrados.
- **iaxpeers, iaxusers**: usuarios y *peers* IAX.
- **voicemail**: cuentas de correo de Voz.
- **extensions**: contextos, extensions, prioridades y aplicaciones del plan de marcado.
- **meetme**: salas de conferencia.
- **queues**: colas de llamadas.
- **queue_members**: miembros de colas de llamadas.
- **musiconhold**: clases de música en espera.
- **queue_log**: registros de colas de llamadas.

Capítulo 9. Call Center

"El que espera, desespera"

Refrán.

Como se mencionó en el Capítulo 1, Asterisk permite contar con una completa plataforma de Centro de Llamadas (Call Center en inglés), para lo cual incluye características como encolamiento de llamadas, agentes, y supervisión y monitoreo de llamadas, entre otras.

Colas de llamadas

Una cola de llamadas, como su nombre lo indica, es una sala de espera telefónica en la que un conjunto de abonados llamantes esperan para ser atendidos por unos agentes o teléfonos configurados para la atención de la misma.

En general, una cola se compone de:

- Las llamadas que ingresan a la cola.
- Los miembros de la cola (que pueden ser agentes que inicien sesión o extensiones de usuarios).
- Una estrategia para repartir las llamadas.
- Música de fondo que se reproduce mientras la llamada es atendida.
- Anuncios a los miembros de la cola y a los abonados llamantes.

Las colas pueden ser estáticas (definidas en el archivo "*/etc/asterisk/queues.conf*") o dinámicas (en tiempo real empleando la

consola de comandos).

De igual manera, los agentes que responden las llamadas que ingresen a una cola pueden ser de dos tipos:

- Miembros estáticos, que no requieren de ningún inicio de sesión para recibir llamadas. Son canales (SIP, IAX, etc) que se definen en el archivo de configuración para responder las llamadas de una cola.

- Miembros dinámicos que inician una sesión de agente en Asterisk, llamando a una extensión que ejecuta la aplicación "AddQueueMember" o "AgentLogin", como se verá más adelante.

Ejemplo de configuración del archivo "queues.conf":

```
; Colas de llamada
[general]
persistentmembers = yes

; Cola de ventas
[cola-ventas]
member => SIP/alicia        ;SIP/alicia y benito son miembros
member => SIP/benito        ; estáticos de la cola y
                            ; siempre recibirán llamadas.
member => Agent/1001,10     ;Los agentes 1000 y 1002 son dinámicos
member => Agent/1001,20     ; como se verá más adelante.
music=default
context=ventas
strategy=ringall
joinempty=strict
leavewhenempty=strict
retry=10
timeout=25
```

```
; Soporte técnico
[cola-soporte]
music=default
context=soporte
strategy=ringall
joinempty=strict
leavewhenempty=strict
```

De esta forma se han definido dos (2) colas de llamadas: cola-ventas y cola-soporte, cada una con su respectivo contexto de entrada (que debe crearse también en el archivo "*extensions.conf*").

Para los miembros de la cola se puede especificar un peso o penalidad opcional luego de una coma, de tal forma que los agentes con pesos mayores son considerados como la última opción.

En la sección general, la opción "*persistentmembers=yes*", causará que las listas de los agentes se almacenen en la base de datos de Asterisk (astdb) y que por lo tanto sean recordados al reiniciar el servicio.

El parámetro "*retry*" define el tiempo en segundos antes de intentar contactar a algún miembro de la cola disponible.

El parámetro "*timeout*" define el tiempo máximo que permanecerá una llamada dentro de una cola de llamadas.

La estrategia ("*strategy*") define el modo en que se repartirán las llamadas entre los agentes disponibles y puede ser cualquiera de las siguientes:

- "*ringall*": llamar a todos los canales disponibles hasta que alguno conteste.
- "*leastrecent*": llamar al miembro que lleve más tiempo sin contestar.
- "*fewestcalls*": llamar al miembro de la cola que menos llamadas

haya contestado.

- "*random*": llamar en forma aleatoria.
- "rrmemory": Llamar uno por uno, empezando donde se quedó en la última vuelta.
- "linear": Llamar en el orden exacto en que se definieron los miembros en el archivo de configuración, o en el orden que que fueron agregados dinámicamente.
- "wrandom": Llama a un canal aleatorio, pero toma en cuenta la prioridad de cada agente.

La opción "*joinempty*" configurada como "strict" evitará que las llamadas entrantes sean ubicadas dentro de colas que carezcan de agentes para atenderlas. En este caso, la aplicación "Queue" retornará y el plan de marcado indicará qué hacer a continuación.

Si existen llamadas en la cola, y el último agente finaliza su sesión, las llamadas entrantes restantes serán retiradas de la cola y "Queue()" retornará, si "*leavewhenempty*" está configurado como "*strict*".

Enviar llamadas a la cola

Retomando el ejemplo de la operadora automática del Capítulo 7, se pueden agregar opciones para enviar la llamada al contexto de cada cola:

```
[menu_bienvenida]
exten => s,1,Answer()
exten => s,n(menu),Background(bienvenida)
exten => s,n,WaitExten(5)
; ...
exten => 3,1,Goto(ventas,s,1)
exten => 4,1,Goto(soporte,s,1)
```

```
include => usuarios
```

Y los contextos de las colas serían:

```
[ventas]
exten => s,1,Ringing()
exten => s,n,Set(CALLERID(name)=Ventas)
exten => s,n,Queue(cola-ventas,t)
exten => s,n,Set(CALLERID(name)=ColaVentasVacia)
exten => s,n,Goto(menu_bienvenida,s,1)

[soporte]
exten => s,1,Ringing()
exten => s,n,Set(CALLERID(name)=Soporte)
exten => s,n,Set(QUEUE_MAX_PENALTY=10);
exten => s,n,Queue(cola-soporte,t)
exten => s,n,Set(QUEUE_MAX_PENALTY=0);
exten => s,n,Queue(cola-soporte,t)
exten => s,n,Set(CALLERID(name)=ColaSoporteVacia)
exten => s,n,Goto(menu_bienvenida,s,1)
```

El propósito de especificar la variable "*QUEUE_MAX_PENALTY*" es definir una prioridad en el orden de llamado a los agentes. Aquellos agentes con una prioridad superior a 10 serán llamados primero. Si nadie contesta, se intentará llamar de nuevo, pero incluyendo a agentes con menor prioridad[52].

Agentes

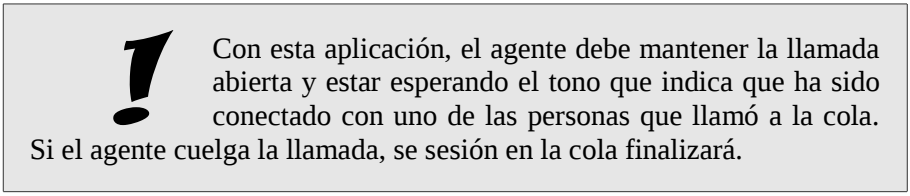
Un agente es un miembro de una cola que se encuentra preparado para atender llamadas de acuerdo a un conjunto de políticas bien definidas.

52 Asterisk Project Wiki. Asterisk Queues. [en línea]. <https://wiki.asterisk.org/wiki/display/AST/Asterisk+Queues> [citado en 29 de noviembre de 2011]

En Asterisk se pueden tener tres clases de agentes: canales fijos añadidos a una cola de llamadas y personas que inicien una sesión como agentes para la recepción de llamadas en Asterisk, bien sea con la aplicación *AgentLogin* o con *AddQueueMember.*

AgentLogin

Con AgentLogin(), el agente inicia sesión y empieza a escuchar música en espera mientras espera a que ingresen las llamadas. Cuando una nueva llamada ingresa a una cola de llamadas la música es interrumpida y el llamante es conectado con el agente.



Los agentes para usar con esta aplicación se definen en el archivo "*/etc/asterisk/agents.conf*":

```
[agents]
;Para cada agente se define un número de identificación,
; una clave y su nombre completo:
agent => 1001,1234,Agente 1
agent => 1002,1234,Agente 2
```

La sintaxis para invocar a AgentLogin es la siguiente:

```
AgentLogin([numero_agente[|opciones]]])
```

Por ejemplo, para crear una extensión donde el agente deba ingresar su número de agente para iniciar sesión, se agregaría al archivo "extensions.conf":

```
[servicios]
;Si se omite el número de agente, la aplicación
; se lo pide al usuario:
exten => 990,1,AgentLogin()
```

AddQueueMember

También es posible añadir agentes a una cola de llamadas de forma dinámica empleando la aplicación "AddQueueMember()", cuya sintaxis se muestra a continuación:

```
AddQueueMember(cola[|canal[|prioridad[|opciones]]])
```

La principal diferencia con AgentLogin, es que los agentes registran dinámicamente sus canales como miembros de la cola. Una vez inician sesión, cuelgan el teléfono y esperan a ser llamados de vuelta cuando ingresen llamadas a la cola. Empleando esta aplicación es posible crear una lógica de plan de marcado más compleja para la gestión de un centro de llamadas completo.

Continuando con el ejemplo anterior, se definirán unas extensiones especiales para iniciar y finalizar sesión en la cola. Para iniciar sesión como Agente a contactar se marca la extensión 991, introduciendo el número de agente. Para finalizar la sesión como agentes, se puede marcar la extensión 992:

```
[servicios]
exten=> 991, Answer()       ;Extensión para iniciar sesión.
exten=> 991,n,Read(NUMERO_AGENTE,agent-enternum) ;Pedir No. agente.
exten=> 991,n,Goto(agentes-colas,E${NUMERO_AGENTE},1)

exten=> 992,1,Answer()      ;Extensión para finalizar sesión.
exten=> 992,n,Read(NUMERO_AGENTE,agent-enternum)
```

```
exten=> 992,n,Goto(agentes-colas,S${NUMERO_AGENTE},1)


[agentes-colas]
;Colas asignadas al Agente 6010:
exten=>_[ES]6010,1,Macro(entrar_salir_cola,cola-ventas,20)
exten=>_[ES]6010,n,Macro(entrar_salir_cola,cola-soporte,20)
;Colas asignadas al Agente 6020:
exten=>_[ES]6020,1,Macro(entrar_salir_cola,cola-soporte,20)


[macro-entrar_salir_cola]
exten=>s,1,Goto(${MACRO_EXTEN:0:1}) ; Para saber si es 'E' (entrar)
                                    ; o 'S' (salir)
exten=>E,1,AddQueueMember(${ARG1},Local/${MACRO_EXTEN:1}@agentes,${ARG2}) ; Entrar a la cola
exten=>S,1,RemoveQueueMember(${ARG1},Local/${MACRO_EXTEN:1}@agentes) ; Salir de la cola
```

Nótese que en la aplicación AddQueueMember se ha definido como miembro de la cola el canal "*Local/<número_de_agente>@agentes*", esto quiere decir que cuando se intente llamar al agente, se ejecutará la extensión correspondiente dentro del contexto "agentes" del plan de marcado.

A continuación se definen la lógica necesaria para llamar al canal de cada agente de manera controlada:

```
[agentes]
exten=>6010,1,Macro(llamar-agente,SIP/alicia)
exten=>6020,1,Macro(llamar-agente,SIP/benito)


[macro-llamar-agente]
exten=>s,1,GotoIf($[${GROUP_COUNT(${MACRO_EXTEN}@agentes)}=0]llamar?saltar)
exten=>s,n(saltar),Busy()
```

```
exten=>s,n(llamar),Set(OUTBOUND_GROUP_ONCE=${MACRO_EXTEN}@agentes)
exten=>s,n,Dial(${ARG1},30,t)
exten=>s,n,Goto(s-${DIALSTATUS},1)
exten=>s,n,Hangup()
;Si el agente no contesta, es retirado automáticamente de la cola:
exten=>s-NOANSWER,1,Goto(agentes-colas,S${MACRO_EXTEN},1)
exten=>s-BUSY,1,Busy()
exten=>_s-.,1,Hangup()
```

Las funciones "*GROUP_COUNT*" y "*OUTBOUND_GROUP*" se utilizan para controlar el número de llamadas simultáneas que serán enviadas a cada agente. Por ejemplo, cuando el agente 6010 es llamado, se asocia el valor "*6010@agentes*" al canal de la llamada, de tal forma que cuando se llama a la función *GROUP_COUNT*, ésta retorna el valor 1 para ese grupo, causando que se invoque a la aplicación Busy para ese agente.

Supervisión en vivo de llamadas en curso

La supervisión es una actividad común en los centros de llamadas. Los coordinadores escuchan las llamadas de los agentes para controlar la calidad del servicio, e incluso pueden aconsejar lo que el agente debe decir a continuación, lo cuál se conoce como "susurrar".

La supervisión y susurro en Asterisk se realiza a través de la aplicación Chanspy, del módulo"app_chanspy.so", y su sintaxis es la siguiente:

```
ChanSpy([prefijo_de_canal][|opciones])
```

Donde "prefijo_de_canal" quiere decir que se interceptarán todos los canales cuyo nombre empiece por esa cadena. Las principales opciones de ChanSpy son:

- "q": Supervisión silenciosa (de lo contrario se escuchará un "bip")

- “g(grupo)”: supervisar canales con la variable ${SPYGROUP} igual a 'grupo'.
- “r[(nombre_de_archivo)]”: grabar la sesión de supervisión.
- “w”: Susurrar al oído del agente (para aconsejar lo que debe decir).

Adicionalmente, durante la supervisión se pueden enviar los siguientes tonos para controlar el comportamiento:

- “#”: cambia el volumen.
- “*” cambia de canal a supervisar.
- “XXX#”: Agrega los dígitos XXX a la cadena de prefijo de canal

Ejemplos:

```
exten=*43,1,Chanspy(SIP/alicia) ;Espiar las llamadas de Alicia.

exten=*44,1,Chanspy(Agent/|q)      ;Espiar a todos los agentes
                                   ; de manera silenciosa.
exten=*45,1,Chanspy(Agent/|w)      ;Espiar a todos los agentes,
                                   ; con susurro.
```

Monitoreo y grabación de llamadas

La grabación en caliente de las llamadas de los agentes puede configurarse haciendo uso de la característica “automon” del archivo “*features.conf*”. Este archivo contiene características predefinidas que se activan cuando las llamadas están en curso, por ejemplo: parqueo de llamadas, transferencia y captura de llamadas.

La característica de grabación en caliente se configura dentro del contexto “featuremap”:

```
[featuremap]
automon => *3
```

Y en el archivo “extensions.conf” se debe definir una variable de canal o global que permita el uso de estas características personalizadas:

```
[globals]
DYNAMIC_FEATURES=>automon
```

De esta manera, cuando un agente desee empezar a grabar una conversación, presiona los tonos “*3” durante la llamada, y el archivo se creará en la carpeta “*/var/spool/asterisk/monitor*”.

Adicionalmente se debe agregar la opción “w” y/o “W” a la aplicación *Queue* o *Dial* que llama al Agente.

Música en Espera

Muchas veces es deseable tener diferentes músicas de espera para cada cola. En Asterisk se pueden tener diferentes clases de música en espera, que representan directorios separados que contienen archivos en cualquier formato de audio reconocido por asterisk.

En el archivo “*musiconhold.conf*” se definen diferentes “clases” de música de espera, con la siguiente sintaxis:

```
[classname]
mode => modo de operación
directory => directorio
application => aplicación para reproducir
```

Por ejemplo:

```
[default]
;Archivos en formatos reconocidos por asterisk
; ubicados en el directorio /var/spool/asterisk/moh
mode=files
directory=moh

[musicaColaVentas]   ;Clase de música en espera personalizada.
mode=custom
directory=/usr/share/asterisk/mohmp3
;Se utiliza la aplicación externa “madplay” para ejecutar
; los archivos en formato mp3:
application=/usr/bin/madplay --mono -R 8000 --output=raw:-
```

De la siguiente manera, se puede seleccionar una clase por omisión de música en espera para un canal determinado. Cuando se activa la música en espera, esta clase será usada para seleccionar la música a reproducir:

```
[ventas]
exten => s,1,Ringing()
exten => s,n,Set(CALLERID(name)=Ventas)
exten => s,n,Set(CHANNEL(musicclass)=musicaColaVentas)
exten => s,n,Queue(cola-ventas,t)
exten => s,n,Set(CALLERID(name)=ColaVentasVacia)
exten => s,n,Goto(menu_bienvenida,s,1)
```

En la consola de Asterisk se pueden ver las clases de música en espera disponibles con el siguiente comando:

```
*CLI> moh classes show
```

Estadísticas

Finalmente, ya con el centro de llamadas en operación, se vuelve muy importante la tarea de revisar y analizar los registros de las llamadas, tanto para costear el servicio en base a los tiempos de atención, así como para mejorar la experiencia de las personas que llaman a la cola y disminuir los tiempos de espera.

El archivo que contiene los registros de actividad de las colas de llamadas es "*/var/log/asterisk/queue_log*". La información almacenada en este archivo para cada evento es la siguiente:

- "AGENTDUMP": El agente rechazó la llamada antes de contestar.
- "AGENTLOGIN(canal)": El agente inició sesión a través del canal especificado.
- "AGENTLOGOFF(canal|tiempo_de_sesion)": El agente terminó su sesión.
- "ENTERQUEUE(url|callerid)": Una llamada entró a la cola. Se muestran la url y el identificador de llamante, si fueron especificados.
- "ABANDON(posición_al_salir|posición_al_entrar| tiempo_de_espera)": El llamante abandonó la llamada en la cola.
- "EXITWITHKEY(tecla|posición)". el llamante decidió presionar una tecla para salir de la cola.
- "EXITWITHTIMEOUT(posición)": el llamante esperó por mucho tiempo y expiró el tiempo máximo.
- "CONNECT(tiempo_de_espera)": El llamante fue comunicado con un agente. Se muestra el tiempo que estuvo en espera.
- "SYSCOMPAT": se comunicó al llamante con el agente, pero la

llamada se cayó por incompatibilidad de los canales (códecs, etc)

- “TRANSFER(extensión,contexto)”: Se transfirió la llamada a otra extensión.

- “COMPLETEAGENT(tiempo_de_espera|tiempo_de_la_llamada| posición_al_entrar)”: La llamada fue atendida por un agente y éste colgó la llamada normalmente. Se graba el tiempo que estuvo esperando el llamante, y la posición original en la cola al momento de entrar.

- “COMPLETECALLER(tiempo_de_espera|tiempo_de_la_llamada| posición_al_entrar)”: Similar al anterior, pero cuando es el llamante quien cuelga.

- “QUEUESTART”: carga inicial del módulo de colas.

- “CONFIGRELOAD”: Se recargó la configuración del registro de queuelog (por ejemplo, con 'asterisk -rx reload')

Capítulo 10. Herramientas de administración y visualización de reportes.

> "Lo primero es conocerse, y lo segundo, cultivarse"
>
> Fernando González Ochoa (1895-1964) escritor y filósofo colombiano.

Registros de llamadas y estadísticas

Una de las ventajas de Asterisk respecto a plantas telefónicas tradicionales, es la facilidad que ofrece para obtener información sobre las llamadas realizadas, con el fin de analizar y controlar el consumo telefónico, optimizando los costos de las organizaciones.

Para esto, desde su creación Asterisk ha entregado unos registros de llamadas en formato CDR y, desde la versión 1.8, incluye también la generación de eventos telefónicos conocida como CEL (Channel Event Logging).

Registro Detallado de Llamadas (CDR)

Es el registro que deja Asterisk de todas las llamadas realizadas. Los campos que se incluyen por omisión son los siguientes:

Campo	**Descripción**
uniqueid	Identificador único del canal.
linkedid	Identificador de la llamada, que puede involucrar varios canales.
src	Número del identificador de llamante.
dst	Última extensión a donde fue remitido.
clid	Identificador de llamante.
start	tiempo de Inicio de Llamada.
answer	tiempo de inicio de la llamada.
end	tiempo Fin Llamada.
duration	Tiempo Total. Desde la marcación hasta que se cuelga.
billsec	Tiempo Total. Desde que se contesta hasta que se cuelga.
lastapp	Ultima aplicación ejecutada.
lastdata	parámetros de la última aplicación ejecutada.
channel	canal utilizado.
dstchannel	canal de destino.
dcontext	contexto de destino.
accountcode	código para tarificación.
disposition	resultado de la llamada.
userfield	Información adicional de libre uso.

Tabla 10.1: Campos del registro de llamadas - CDR

Todos los campos en el CDR pueden accederse desde el plan de marcado mediante la función “CDR”, Algunos son de solo lectura, y otros pueden cambiarse:

```
exten => s,1,Verbose(Llamada desde: ${CDR(clid)})
exten => s,n,Set(CDR(userfield)=TarifaPlana)
```

Los parámetros generales de la generación de registros de llamadas se configuran en el archivo "*/etc/asterisk/cdr.conf*", donde también se pueden configurar parámetros específicos de cada uno de los módulos que almacenan los CDRs.

Almacenamiento de los CDR

Asterisk ofrece módulos para para almacenar los CDR en diferentes medios. El más sencillo de ellos es el "*cdr_csv*", que almacena los registros en un archivo de texto en formato separado por comas (formato CSV, comma separated value).

El archivo es "*/var/log/asterisk/cdr-csv/Master.csv*", y el formato de cada línea es el siguiente:

```
<accountcode>,<src>,<dst>,<dcontext>,<clid>,<channel>,<dstchannel>,
<lastapp>,<lastadata>,<start>,<answer>,<end>,<duration>,<billsec>,<
disposition>,<amaflags>[,<uniqueid>][,<userfield>]
```

Los campos "*uniqueid*" y "*userfield*" sólo se registrarán si se habilita la opción correspondiente en el archivo "*cdr.csv*":

```
[general]
enable=yes


[csv]
loguniqueid=yes
loguserfield=yes
```

Otros módulos disponibles para almacenar los CDR se listan a continuación:

Módulo	Medio en que almacena los CDR
cdr_custom	En archivo CSV, con campos personalizados.
cdr_manager	Como eventos en la interfaz de gestión AMI.
cdr_syslog	Envía los CDR al servicio de registro de Linux, Syslog.
cdr_radius	Envía un servidor RADIUS, que son muy populares para tarificación en proveedores de servicios.
cdr_mysql	Almacenamiento nativo en base de datos MySQL.
cdr_adaptive_odbc	Almacenamiento en cualquier base de datos compatible con el estándar ODBC, con campos personalizados
cdr_pgsql	Almacenamiento nativo en base de datos PostgreSQL.
cdr_sqlite	Almacenamiento nativo en base de datos SQLite versión 2.
cdr_sqlite3_custom	Almacenamiento nativo en base de datos SQLite versión 3, con campos personalizados.

Tabla 10.2: Módulos para almacenamiento de CDR

Los CDR funcionan bien en escenarios de llamadas sencillas, pero cuando una llamada involucra varios canales, es transferida, o algún otro escenario especial, los registros tienden a presentar información incompleta o errónea. Es por esto que a pesar de que los CDR son el formato más popular para analizar los registros de llamadas en la mayoría de las aplicaciones y herramientas disponibles en el mercado, se creó un sistema alternativo llamado CEL (Channel Event Logging, Registro de Eventos de Canal).

Registro de Eventos de Canal (CEL)

Es un medio alternativo que ofrece Asterisk para generar y almacenar registros de llamadas, que no genera un solo registro por canal, sino un registro por cada evento que ocurre en el sistema para toda la llamada[53].

53 Asterisk Project Wiki. Channel Event Logging. [en línea].

Evento	Descripción
CHAN_START/END	Tiempo de inicio/terminación del canal
ANSWER	Tiempo en que el canal fue contestado.
HANGUP	Tiempo en que el canal fue colgado.
CONF_ENTER	Tiempo de entrada a una sala de conferencia.
CONF_EXIT	Tiempo de salida de una sala de conferencia
CONF_START	Tiempo en que inició la conferencia.
CONF_END	Tiempo en que terminó la conferencia.
APP_START/END	Tiempo de inició/fin de una aplicación monitoreada.
PARK_START/END	Tiempo en que la llamada fue parqueada/desparqueada.
BRIDGE_START/END	Tiempo de inicio/fin del “puente” (bridge) interno entre dos canales.
BRIDGE_UPDATE	Cambio en el puente entre canales.
3WAY_START/END	Tiempo de inicio/fin de una conferencia tripartita entre canales (Por ejemplo, una transferencia asistida).
BLINDTRANSFER	Cuando se inicia una transferencia directa.
ATTENDEDTRANSFER	Cuando se inicia una transferencia asistida.
PICKUP	El canal capturó otro canal que estaba parqueado.
FORWARD	Este canal está siendo redirigido.
HOOKFLASH	Cuando se cuelga un canal DAHDI.
LINKEDID_END	El último canal para el que el campo “linkedid” es válido.
USER_DEFINED	Evento personalizado por el usuario en el plan de marcado.

Tabla 10.3: Eventos registrados por CEL

Cada uno de estos eventos se acompaña de campos muy similares a los del CDR, como: *uniqueid*, *userfield*, *channame*, *appname*, *appdata*, etc. Pero sus valores serán relativos sólo al evento correspondiente, ya que durante la llamada estos valores pueden cambiar.

<https://wiki.asterisk.org/wiki/display/AST/Channel+Event+Logging+(CEL)> [citado en 22 de agosto de 2012]

Así mismo, y de manera análoga que con los CDRs, los registros generados por CEL pueden ser personalizados en el archivo "*cel.conf*", y se cuentan con módulos similares para el almacenamiento de los eventos en diferentes medios: CSV, ODBC, PGSQL, Sqlite3 y Radius.

Herramientas web de administración y configuración

Se han desarrollado muchas aplicaciones relacionadas con Asterisk por parte de terceros. A continuación se encontrará un listado de las herramientas y proyectos más conocidos, sin entrar en detalles sobre su instalación y configuración, pues el objetivo de este libro se limita exclusivamente al dominio de Asterisk.

Configuración

Las herramientas de configuración web más populares son las siguientes:

Nombre	Sitio Web	Descripción
FreePBX	http://www.freepbx.org	Interfaz gráfica de usuario (GUI) completa para configurar Asterisk como una PBX.
AsteriskGUI	http://www.asterisk.org	Es una arquitectura de referencia que incluye módulos para crear interfaces gráficas para Asterisk. Incluye una GUI de ejemplo.
ViciDIAL	http://www.vicidial.org	Es un conjunto de programas que permiten usar Asterisk como un completo sistema de Call Center para llamadas entrantes y salientes.
A2Billing	http://www.asterisk2billing.org	Es una completa herramienta de configuración y administración de Asterisk como un Softswitch para operadores de llamadas de larga distancia con Voz IP.

Tabla 10.4: Herramientas Web de administración

Registros y Monitoreo

Las herramientas de monitoreo y estadísticas más populares son las siguientes:

Nombre	Sitio Web	Descripción
Flash Operator Panel (FOP)	http://www.asternic.org	Panel web que emula una consola de operadora. Permite ver qué extensiones están libres y cuales tienen llamadas, así como hacer transferencias, colgar, etc.
CDR-Stats	http://www.cdr-stats.org	Herramienta para analizar y visualizar reportes de registros de llamadas (CDR)

Tabla 10.5: Herramientas de monitoreo y registros

Distribuciones

Una distribución es un completo sistema operativo basado en Linux, que incluye todas la aplicaciones requeridas para determinado propósito, pre-instaladas y listas para configurar y utilizar.

Se volvieron muy populares debido a que instalar y configurar manualmente es una tarea dispendiosa para la mayoría de usuarios principiantes, aunque para usuarios más avanzados la cantidad de aplicaciones que incluyen puede resultar muy pesada en términos de un óptimo desempeño.

Las distribuciones basadas en Asterisk más populares son las siguientes:

Nombre	Sitio Web	Descripción
Trixbox	http://fonality.com/trixbox	Una de las primeras distribuciones basadas en Asterisk, que incluye FreePBX pre-instalado como interfaz de configuración, así como muchas otras utilidades.
Elastix	http://www.elastix.org	Una completa distribución que incluye herramientas que buscan complementar a Asterisk como una solución de Comunicaciones Unificadas (Ver Capítulo 12).
AsteriskNOW	http://www.asterisk.org	Es la distribución oficial de Digium, la empresa detrás del proyecto de Asterisk.
ViciBOX	http://www.vicibox.com	Distribución que incluye ViciDIAL para un completo sistema de Call Center para llamadas entrantes y salientes.

Tabla 10.6: Distribuciones de Linux especializadas en Asterisk

Capítulo 11. Problemas frecuentes y Diagnóstico

"El talento se educa en la calma, y el carácter en la tempestad"

Johann Wolfgang von Goethe (1749 – 1832), escritor Alemán.

Herramientas y procedimientos de diagnóstico

Herramientas para encontrar la solución a los problemas más frecuentes a la hora de administrar una PBX IP basada en Asterisk.

Opciones de depuración del CLI

La interfaz de línea de comandos cuenta con diferentes opciones que permiten aumentar el nivel de información. Esto permite poder identificar posibles problemas con la configuración de alguno de los módulos de Asterisk o con el plan de marcado.

Por ejemplo, para aumentar el nivel de detalle de la información que se muestra:

```
*CLI> core set <debug|verbose> <nivel>
```

Donde el número especificado en <nivel> equivale a "-vvv..." o "-ddd..." al arrancar Asterisk.

Para depurar los mensajes SIP desde o hacia una dirección IP, una cuenta SIP, o en todos los canales:

```
*CLI> sip set debug <ip|peer|on>
```

Para depurar los mensajes IAX:

```
*CLI> iax2 set debug <ip|peer|on>
```

Para depurar la señalización de un puerto de primario:

```
*CLI> pri debug span <número_de_span>
```

Para guardar en un archivo la salida de la consola de asterisk, puede utilizarse el programa “tee”, que envía la entrada estándar de un programa hacia un archivo:

```
asterisk -rvvv | tee grabando_cli.txt
```

y para terminar, se presiona “Ctrl+C”, luego de lo cuál puede revisarse el contenido del archivo.

Archivos de log

Asterisk utiliza el archivo “/var/log/asterisk/messages” para registrar todos los eventos en el sistema. El nivel de información que se muestra en este archivo, o en la consola de asterisk, puede alterarse editando el archivo “/etc/asterisk/logger.conf”.

Por ejemplo, para activar los mensajes de depuración (debug) en el archivo “messages”, pero no en la CLI:

```
[logfiles]
console => error,warning,notice
messages => notice,warning,error,debug,verbose
```

Luego de lo cual se debe recargar el proceso de logger desde la consola, con el comando:

```
*CLI> logger reload
```

Captura de paquetes de red

La mayoría de los problemas a nivel de comunicación en redes de Voz sobre IP tienen que ver con una mala señalización o con problemas de red (audio en un solo sentido, fallo al registrarse, etc.). Para ese tipo de problemas lo más conveniente es utilizar herramienta para monitorear directamente el tráfico y determinar las causas que los originan.

Una de las herramientas más conocidas para este propósito es "***tcpdump***", la cual tiene muchas opciones para capturar los paquetes de red. Por ejemplo, para monitorear si existe tráfico de cualquier tipo con un equipo remoto:

```
tcpdump -i eth0 host <ip_equipo_remoto>
```

Donde "eth0" en el nombre de la interfaz de red que se quiere interceptar para capturar los paquetes.

Para saber cuál es el nombre de la interfaz de red, se puede utilizar el comando "ifconfig".

Para monitorear tráfico RTP en un rango específico, por ejemplo entre los puertos diez mil a veinte mil:

```
tcpdump -i eth0 udp portrange 10000-20000 and host
<ip_equipo_remoto> -v
```

Generalmente se va a querer monitorear el tráfico y guardar la captura en un archivo en formato "pcap" para su posterior análisis. Para esto, se puede utilizar el siguiente comando:

```
tcpdump -i eth0 host <ip_remota> -w -s0 captura.pcap
```

Otra herramienta similar a "tcpdump" es "***ngrep***". Ngrep permite utilizar expresiones regulares para filtrar el tráfico y soporta igualmente el formato de archivo "pcap":

```
ngrep -d eth0 -N -p -q -W byline port 5060 -O captura.pcap
```

Analizador de protocolos

Una vez almacenadas las capturas de paquetes en archivos “pcap”, éstos pueden ser utilizados en un programa analizador de protocolos como ***Wireshark***, que cuenta con todo un menú de opciones especializadas para telefonía.

Wireshark tiene la opción de auto detectar las llamadas que se pueden ver en los paquetes capturados, mostrando el flujo de mensajes SIP y RTP. También da la opción de escuchar la conversación, si ésta utiliza un códec sin compresión, y analizar el flujo RTP para identificar problemas de retardos y variaciones, problemas con tonos DTMF, etc.

Core-dumps

Puede ocurrir en algunas ocasiones que Asterisk se estrelle debido a algún error irrecuperable o a algún fallo de software, sobretodo cuando se utilice una versión de prueba que todavía no sea estable. Para identificar la causa exacta de ese tipo de problemas, se puede analizar el volcado de la información del proceso al momento de terminarse (core-dump).

Para esto se debe invocar el proceso de asterisk con la opción “-g'", con lo cuál habilita la opción para almacenar un archivo de core-dump con el que se puede solicitar soporte en las listas de correo de asterisk.

Por omisión, el archivo se almacenará en “/tmp”, pero en algunas ocasiones es mejor cambiar esta ubicación, ya que el archivo de core-dump puede pesar varios giga bytes. Esto se puede lograr editando el archivo de inicio del proceso en "*/etc/init.d/asterisk*" agregando las líneas:

```
echo '/var/log/asterisk/core' > /proc/sys/kernel/core_pattern
```

El resultado de core-dump si asterisk se estrella se encontrará entonces en

“/var/log/asterisk/core”.

Una vez Asterisk se estrelle, se debe tomar una traza de depuración del archivo de core, con el siguiente comando:

```
gdb -se "asterisk" -c //var/log/asterisk/core
```

Y una vez se termine de cargar el depurador, ejecutar la siguiente instrucción:

```
(gdb) backtrace
```

La información mostrada se puede ahora reportar a los desarrolladores de Asterisk.

Reportando fallos

Antes de reportar un fallo, se debe constatar que no haya sido reportado previamente. Los fallos (bugs) de Asterisk pueden ser buscados y reportados en la página http://issues.digium.com/

Si el fallo ya se encuentra reportado, puede agregar información sobre cómo se le presentó este fallo en particular, para que esta información se sume y ayude a llegar a la solución a los desarrolladores.

La información que se debe tener a la mano para crear o agregar información a fallos es:

- Versión exacta de Asterisk que está utilizando. Si está utilizando una copia de SVN, debe especificar el número de la revisión ("svn info"). Es mejor probar con la última versión del SVN antes de reportar el problema, pues puede ser que ya haya sido solucionado allí.

- Sistema operativo o distribución de linux.

- Información sobre cómo reproducir el problema.
- Información de depuración para anexar al reporte del bug.

Problemas Frecuentes

Eco

Uno de los problemas frecuentes, especialmente cuando se utilizan líneas telefónicas de telefonía tradicional, es cuando se escucha eco en la comunicación. Por ejemplo, una persona en una extensión se escucha a sí misma, o una persona de afuera de la PBX es quien escucha el eco.

Si el problema se presenta sólo en algunos softphones, sólo a través de troncales de Voz IP, se debe revisar primero el dispositivo de audio utilizado (micrófono/parlantes/auriculares), así como la calidad del cable y la conexión de red.

Si el eco se escucha en todas las llamadas hacia o desde la RTPC, se deben activar las opciones de cancelación de eco en DAHDI, el cual cuenta con módulos que se pueden especificar por cada canal.

Para esto, se edita el archivo “/etc/dahdi/system.conf” para que incluya alguno de los canceladores de eco que incluye DAHDI por omisión:

```
echocanceller=mg2,1-15      ;Usar el cancelador MG2
echocanceller=hwec,17-31    ;Usar el cancelador hardware incluido
                            ; en la tarjeta de interfaz.
```

Finalmente se reinicia “dahdi” y se reconfiguran los spans:

```
/etc/init.d/dahdi restart
dahdi_cfg -vvv
```

Si el eco persiste, se recomienda utilizar el cancelador OSLEC[54], que debe ser instalado por aparte, pero que da unos resultados muy buenos.

Pérdida de Paquetes

La pérdida de paquetes es inevitable en un 100% debido a múltiples factores que pueden afectar los diferentes medios de transmisión utilizados en las redes de datos, además de factores como la congestión, equipos defectuosos y sobrecarga de procesamiento.

La forma más sencilla de medir la pérdida de paquetes es ejecutando el comando “ping” entre el servidor Asterisk y el extremo remoto de la comunicación, bien sea un softphone o una troncal.

Los diferentes códecs utilizados y en especial los más comunes pueden predecir los paquetes perdidos y remplazarlos, de esta manera el oído del usuario no se da cuenta de que faltó un paquete. Pero cuando ésta perdida es superior al 5%, ninguno de los códecs más populares puede predecir el valor del paquete perdido y se notará en la voz que este paquete hace falta.

Esto es lo que ocasiona que en ocasiones la Voz sobre IP no opere de forma óptima sobre enlaces inalámbricos (802.11b/g), o a través de canales de Internet compartidos y muy congestionados.

Jitter

El Jitter es la diferencia entre el tiempo en que efectivamente llega un paquete y el tiempo en que se creía que llegaría el paquete. Puede causarse por congestión en la red o sobrecarga en los procesos de codificación y decodificación, o por la variación en las condiciones físicas de un medio de transmisión.

Como se vio en el funcionamiento de TCP/IP y en especial del protocolo UDP (Capítulo 6.), los paquetes no llegan a su destino en orden y mucho menos a una velocidad constante, pero el audio tiene que tener una

54 <http://www.rowetel.com/ucasterisk/oslec.html>

velocidad constante. Para resolver esto existen los *jitter buffer*, que pueden almacenar unos 300 milisegundos y controlar esta variación para que el audio se escuche a velocidad constante. si la llegada de paquetes es demasiado desigual el búfer no la alcanza a controlar y perderá paquetes, deteriorando la calidad de la voz. Y si esta pérdida es superior al 5% el usuario la notará.

Asterisk incluye opciones para configurar el uso de jitter buffer tanto en canales SIP (jbenable=yes) como en canales IAX (jitterbuffer=yes), así como la opción de usar un búfer en los canales de tipo LOCAL, agregando "/nj" al comando "Dial":

```
exten=123456,1,Dial(Local/123456@otro_contexto/nj)
```

Retardo

El retardo en la diferencia que existe entre el momento en que una señal es trasmitida y el momento que una señal llega a su destino. Además del efecto de espera en la comunicación percibida por los interlocutores, el retardo puede dar pie a la generación de otro problema más molesto como lo es el ECO.

En las redes de telefonía tradicional, el retardo suele ser de apenas 15 ms, mientras en telefonía IP el retardo puede alcanzar 20-50ms en una LAN y centenas de milisegundos a través de WAN/Internet. El máximo retardo tolerable en una comunicación puede estar entre 150ms y 300 ms.

El retardo puede tener dos tipos de fuentes: constantes y variables.

Retardo constante

Dentro de las fuentes de retardo constante están todas aquellas que siempre generaran la misma cantidad de retardo, las más importantes son:

- Codificación, es el retardo generado al tomar el audio y procesarlo por un códec especifico.

- Paquetización, es el retardo generado al tomar el audio y convertirlo en paquetes IP.

- Serialización, es el retardo generado al ubicar los paquetes de voz, desde las capas de aplicación hasta la interfaz por la cual será trasmitido.

Retardo variable

Las fuentes de retardo variable son todas aquellas que generan diferentes cantidades de retardo según las condiciones del medio, las más importantes son:

- Encolamiento: el que se genera cuando los paquetes de voz tienen que esperar en las colas de los equipos activos al ser trasmitidos.

- Propagación: es el retardo que se genera al pasar los paquetes por los diferentes cables hasta llegar a su destino, o en el caso de las comunicaciones por satélite, el tiempo de ir y volver al satélite.

Para los cálculos de retardo se debe tomar en cuenta la suma de todos los retardos.

Cuando la pérdida de paquetes es inferior al 5% los diferentes códec utilizados pueden corregir el error, los métodos utilizados para corregir este error son básicamente dos:

- Intrapolar: cuando falta un paquete, el códec toma el paquete anterior y el paquete siguiente y calcula el valor del paquete faltante.

- Sustitución: cuando el códec detecta un paquete faltante lo remplaza por un paquete igual al paquete anterior.

Calidad de Servicio

La falta de calidad de servicio en la comunicación de Red puede afectar las

comunicaciones de Voz sobre IP. La calidad de servicio se puede lograr empleando técnicas para dos conceptos diferentes que muchas veces se confunden entre sí, que son: Clase de Servicio (CoS) y Nivel de Servicio (QoS).

Los sistemas de CoS priorizan el tráfico en un segmento de la red más no en todos los saltos de una comunicación, a diferencia de QoS. Los sistemas CoS no pueden garantizar una priorización de tráfico de un extremo a otro de la comunicación, pues solamente se ocupan de una porción de la misma.

En general, los sistemas de CoS son ideales para redes privadas de alta velocidad y VLAN, pero no para canales de ancho de banda limitado, como el Internet. Los dos tipos de sistemas de CoS son 802.1p y DiffServ. Estas tecnologías deben ser implementadas en cada uno de los nodos de la red, de tal forma que la comunicación tenga la política de calidad de servicio a través de todo el camino de la comunicación.

QoS lleva un paso más allá las tecnologías de CoS. Las dos tecnologías principales de QoS son RSVP y MPLS. Estas tecnologías en general caen bajo la responsabilidad del proveedor del canal o ISP. El administrador de los enrutadores de cada uno de los extremos debe configurar clases de servicios (ToS) para los cuales se garantizan determinados niveles de servicio.

Asterisk incluye soporte para marcar los paquetes salientes a nivel de ToS o CoS, de tal forma de los demás elementos de transporte de red puedan darle prioridad a los paquetes si tienen soporte para ello[55]. Los valores recomendados a utilizar en el archivo “sip.conf” son los siguientes:

```
tos_sip=cs3          ; ToS para señalización.
tos_audio=ef         ; ToS para paquetes RTP de audio.
tos_video=af41       ; ToS para paquetes RTP de video.
```

55 Asterisk Project Wiki. IP Quality of Service. [en línea]. <https://wiki.asterisk.org/wiki/display/AST/IP+Quality+of+Service> [citado en 22 de agosto de 2012]

```
cos_sip=3           ; Prioridad 802.1p para SIP.
cos_audio=5         ; Prioridad 802.1p para RTP de audio.
cos_video=4         ; Prioridad 802.1p para RTP de video.
```

NAT

La traducción de direcciones de red o NAT (por sus siglas en inglés - Network Address Translation), es un procedimiento mediante el cuál una red privada puede compartir una única dirección IP pública para acceder a Internet. Fue la solución que se dio al problema de la limitación en el número de direcciones IP versión 4 que habían disponibles a nivel mundial en la década de los noventa.

Haciendo uso de NAT, una red privada de computadores con un rango de direcciones definido en el RFC1918 (10.X.X.X, 172.16.X.X ó 192.168.X.X), puede acceder a Internet a través de un enrutador que se encarga de compartir la dirección IP pública, haciendo una traducción entre la IP privada de origen y un puerto TCP o UDP de la dirección IP pública.

Sin embargo, este mecanismo crea inconvenientes para muchos protocolos, incluyendo SIP, ya que éste utiliza la dirección IP privada del origen en varias partes de las cabeceras del mensaje. Lo cuál hace que para el extremo receptor, la información no tenga sentido, pues éste solo conoce o tiene acceso a la dirección IP pública compartida.

Esta situación puede generar varios problemas en la comunicación SIP:

- Problemas de registro: el terminal SIP envía el mensaje REGISTER, pero el servidor no puede enviar de regreso la confirmación o los mensajes para refrescar la sesión (OPTIONS), haciendo que expire el registro.

- Problemas en las llamadas: algunos servidores aceptan llamadas sin necesidad de registro previo, pero cuando hay problemas de NAT, la llamada entra, pero no queda completamente establecida y luego se cuelga.

- Audio en un sólo sentido: cuando el extremo remoto envía su dirección IP privada en las cabeceras SDP, indicando el puerto erróneo para el envío del audio, causando que los paquetes RTP nunca lleguen a su destino.

Existen varias implementaciones de NAT, que dependiendo de su tipo, pueden generar problemas diferentes[56]:

Tipo de NAT	Necesita enviar datos antes de poder recibir	Es posible determinar IP:puerto para los paquetes de respuesta	Restringe los paquetes entrantes por IP:puerto
Cono completo	No	Si	No
Cono restringido	Si	Si	Sólo IP
Cono de puerto restringido	Si	Si	Si
Simétrico	Si	No	Si

Tabla 11.1: Tipos de NAT

Solucionar los problemas relacionados con NAT no es tarea sencilla, pero existen diferentes alternativas que se pueden combinar para dar una solución completa.

Lo primero que se debe hacer es tener claro qué tipo NAT se tiene, tomando capturas de paquetes de la comunicación, y revisando las cabeceras del mensaje SIP que son afectadas por el NAT: Contact, Via, Route, Record-Route, y en las cabeceras del contenido SDP. Por ejemplo:

```
INVITE sip:benito@asterisk.org SIP/2.0
Via: SIP/2.0/UDP 192.168.99.1:5060
From: Alicia <sip:alicia@asterisk.org>
To: Benito <sip:benito@asterisk.org>
```

56 GONCALVES, Flavio. Building Telephony Systems with OpenSIPS 1.6. Packt Publishing, 2010

```
Call-ID: 12345600@asterisk
CSeq: 1 INVITE
Contact: Alicia <sip:alicia@192.168.99.1>
Content-Type: application/sdp
Content-Length: 147
v=0
o=Alicia 2890844526 2890844526 IN IP4 asterisk.org
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

Muchos Firewalls (cortafuegos) y enrutadores realizan modificaciones de los mensajes SIP, que en varias ocasiones resuelven automáticamente el problema de NAT, pero otra veces causan un efecto no deseado, y se debe desactivar esa función para poder resolver el problema de otra manera.

Cuando quién está detrás de NAT es el terminal SIP, lo que se recomienda es implementar primero una solución del lado del cliente como STUN[57], que es un protocolo que implementan la mayoría de terminales SIP, y se encarga de detectar la dirección IP pública compartida, y la utiliza para escribir los mensajes SIP con la información correcta desde el principio.

STUN es efectivo para la mayor parte de tipos de NAT: cono completo, cono restringido y cono de puerto restringido. Sin embargo, no soluciona el tipo de NAT simétrico, ya que en éste se utiliza un puerto diferente en cada conexión saliente hacia Internet, haciendo que la detección realizada por STUN no sirva para los siguientes mensajes de la comunicación.

Del lado del servidor, existen diferentes opciones que se pueden utilizar en el archivo /etc/asterisk/sip.conf bien sea para todas las cuentas sip, o por cada usuario[58]:

57 *Simple Traversal of User Datagram Protocol* (UDP) *through Network Address Translators* (NATs)
58 Asterisk Guru. SIP with NAT or Firewalls [en línea].

- “nat=route”: Asterisk enviará el audio a la dirección IP y al puerto desde donde reciba el audio, ignorando los valores que se encuentren en el mensaje SDP.

- “nat=rfc3581”: éste es el valor por omisión si no se incluye la opción “nat”. Lo que hace es agregar el parámetro “rport” a la cabecera “Via” de los mensajes SIP, indicándole al cliente SIP que Asterisk le puede enviar las respuestas a la misma dirección IP y puerto desde donde se reciban las peticiones, omitiendo la información de la cabecera “Contact”.

- “nat=yes”: Es la combinación del comportamiento de “route” y “rfc3581”.

- “nat=never”: deshabilita las opciones de NAT.

- “qualify=yes”: mantiene abierta la sesión de NAT, evitando que expire mediante el envío periódico de mensajes SIP OPTIONS. Si el terminal no contesta al mensaje, Asterisk considera que se encuentra inalcanzable, y dejará de enviarle llamadas.

- “qualify=número_de_milisegundos”: para controlar el tiempo máximo que se espera la respuesta por parte del terminal. El valor por omisión es 2 segundos.

- “qualifyfreq=número_de_segundos”: para controlar el intervalo entre los mensajes OPTIONS.

- “directmedia=no/nonat”: esta opción controla si Asterisk realiza un RE-INVITE para alterar la sesión establecida en una llamada, de tal forma que los terminales envíen el audio directamente el uno al otro, en lugar de pasar por el servidor.

Cuando es Asterisk el que se encuentra detrás de NAT, se pueden definir en el contexto “[general]” del archivo “sip.conf” las siguientes opciones:

<http://www.asteriskguru.com/tutorials/sip_nat_oneway_or_no_audio_asterisk.html> [citado en 25 de agosto de 2012]

- “externip=dirección_IP_pública”: define la dirección IP que se incluirá en todos los mensajes SIP desde Asterisk hacia las redes externas.

- “localnet=192.168.0.0/255.255.255.0”: define el rango de direcciones IP que hacen parte de la red interna, para los cuales no se incluirá la IP pública definida con “externip”. Se pueden incluir tantas líneas “localnet” como rangos de IP privadas existan en la red interna.

A partir de la versión 11, Asterisk incluirá soporte para ICE (Interactive Connectivity Establishment), un nuevo mecanismo que combina STUN con otras soluciones del lado del servidor para determinar automáticamente la mejor solución al problema de NAT en cada caso.

Existen soluciones alternativas al problema del NAT, como el uso del protocolo IAX en lugar de SIP, o el establecimiento de túneles IP entre los extremos de la comunicación, pero en cualquier caso, elegir la solución correcta depende de las condiciones específicas de cada escenario.

Caída de llamadas

En llamadas a través de troncales de voz sobre IP, generalmente la caída de llamadas está relacionada con problemas de red (bien sea de enrutamiento, o de una tarjeta de red defectuosa), o de NAT, como se mencionó en el apartado anterior.

Sin embargo, en troncales digitales (E1 o T1), la caída de llamadas tiene causas diferentes. El problema puede estar del lado del servidor Asterisk, a nivel de hardware (tarjetas de telefonía) o a nivel de configuración (fallos en DAHDI); pero también puede ser un problema de sincronización del lado del operador telefónico. Generalmente cuando hay problemas de este tipo, se verán errores relacionados con la palabra “PRI” en los archivos de log y en la consola.

La manera más práctica de descartar tanto problemas de hardware como del operador telefónico o de la calidad del cable como tal, es usando un conector

de bucle (loopback), que se puede construir fácilmente utilizando un conector RJ45 y dos alambres de cable de Red. Se ponchan los alambres de tal manera que queden conectados el pin 1 con el pin 4, y el pin 2 con el pin 5.

Una vez terminado se instala el conector en el puerto de E1/T1 de la tarjeta en el servidor Asterisk, simulando la comunicación con el operador, y luego se puede verificar la conexión con el comando de linux “dahdi-tool”.

Otro problema frecuente con tarjetas de interfaz digital es la asignación de interrupciones de la CPU del computador (IRQ). En algunos servidores que cuentan con muchas tarjetas PCI conectadas, el sistema operativo tiene a asignar varios dispositivos a la misma IRQ, o en otros casos se asignan todas las IRQ a un sólo procesador.

Para verificar la asignación de IRQs en Linux, se puede usar el siguiente comando:

```
# cat /proc/interrupts
```

Algunos computadores soportan la asignación de IRQ exclusiva para la tarjeta de telefonía desde la BIOS, pero en otros casos esto debe realizarse desde el sistema operativo, o simplemente se logra cambiando de ranura PCI la tarjeta, por ejemplo, a una ranura que no esté compartida físicamente.

En muchas ocasiones el problema puede estar del lado del operador. Por ejemplo, algunas centrales telefónicas bloquean los puertos de primario cuando hay muchas caídas de conexión seguidas (durante un mantenimiento), por eso se debe tener a la mano siempre los datos de contacto del proveedor para llamar en caso de soporte.

Capítulo 12. Comunicaciones unificadas

> "Cualquier tecnología lo suficientemente avanzada es indistinguible de la magia"
>
> Tercera Ley de Arthur C. Clarke, escritor británico.

Las Comunicaciones Unificadas son el siguiente paso en la evolución de las telecomunicaciones. Se trata de poder integrar todas las diferentes tecnologías disponibles con el fin de acelerar las comunicaciones y mejorar la colaboración entre las personas[59]:

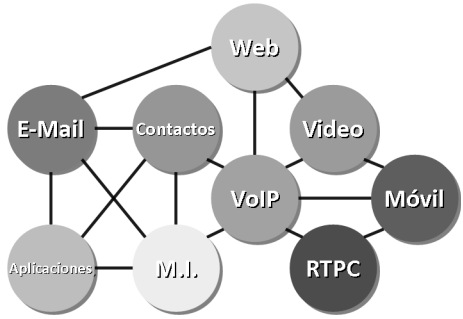


Figura 12.1: Integración de tecnologías en las Comunicaciones Unificadas

Características clave:

- Habilidad de conocer el estado de disponibilidad de otros usuarios en la red (presencia).
- Un mensaje enviado a canal de un tipo, puede ser recibido en otro.

59 ROJANO, Elio. Comunicaciones Unificadas en grandes infraestructuras. En: VoIP2DAY: Septiembre de 2009. Disponible en <http://www.sinologic.net>

De esta manera las personas no tienen que preocuparse por acordar un medio de comunicación antes de ponerse en contacto.

- Alertas en tiempo real y mensajería instantánea. El mundo de hoy demanda movilidad e inmediatez.

- Permitir a los individuos definir sobre la marcha cuando, dónde y cómo quieren ser contactados.

Actualmente, llevar las comunicaciones unificadas a la realidad enfrenta barreras como el altísimo costo de las soluciones propietarias, incompatibilidades entre fabricantes, complejidad para la empresa que lo implementa y elevados requerimientos de infraestructura.

Es por esto que alrededor de Asterisk se han venido desarrollando diferentes extensiones y proyectos complementarios que buscan aprovechar e integrar otras herramientas de software libre y propietarias con el fin de entregar una solución de Comunicaciones Unificadas completa.

Fax

Es curioso como en gran parte del mundo el servicio de FAX continúa vigente a pesar de que el correo electrónico, los escáners y las impresoras lo reemplazan con muchas ventajas.

Debido a que el FAX emplea frecuencias diferentes a las de la voz humana para el envío de las imágenes codificadas, los códec empleados para transmitir la voz sobre IP afectan e incluso imposibilitan su transmisión dentro del flujo RTP. Es por esto que la Unión Internacional de Telecomunicaciones (ITU por sus siglas en inglés - *International Telecommunications Union*) definió el estándar T.38 para el envío de faxes a través de Internet en tiempo real.

Utilizando T.38, Asterisk puede enviar y recibir faxes a través del protocolo SIP. Para recibir llamadas desde troncales analógicas o digitales se puede utilizar el programa Hylafax. Los escenarios que se pueden implementar, son entre otros:

- Recibir un fax desde una troncal, convertirlo a PDF, enviarlo por correo electrónico; y viceversa.

- Recibir una llamada desde una *gateway* con puertos FXS, donde se conecta una máquina tradicional de FAX, y enviarla a través de una troncal hacia la red pública o Internet.

Mensajería Instantánea

El principal protocolo abierto de mensajería instantánea es XMPP (*Extensible Messaging and Presence Protocol*) originalmente conocido como Jabber. Es un estándar basado en XML, muy poderoso y popular, utilizado por infraestructuras de chat tan complejas como las de Gtalk y Facebook.

Asterisk cuenta con el módulo “res_jabber.so”, que le permite conectarse a cualquier servidor XMPP y enviar mensajes a contactos, o conocer y cambiar su estado de presencia. También cuenta con módulos para comunicación de voz en redes XMPP utilizando el protocolo Jingle (desarrollado por Google).

Existen muchas implementaciones *open source* de servidores Jabber. Una de ella, Openfire, cuenta con un *plugin* que permite conectarse con Asterisk para actualizar la presencia de un usuario cuando está ocupado en una llamada, y para que los usuarios puedan llamar haciendo clic cobre los contactos de mensajería.

Correo Electrónico

La forma más simple de integración entre Asterisk y el correo electrónico es a través de la función de notificación de correo de voz. En el archivo “voicemail.conf” se puede configurar un programa de envío de correo como *Sendmail* para enviar notificaciones de un nuevo correo de voz, e incluso se puede adjuntar el mensaje en formato de audio.

Adicionalmente, se puede configurar un servidor IMAP (protocolo para

acceso a cuentas de correo electrónico) como repositorio de almacenamiento de los correos de voz. Lo cuál permite una integración transparente; ya que cuando se leen o se borran los mensajes de voz desde el correo electrónico, éstos se marcan como leídos o borrados también en Asterisk, y viceversa.

Web

Hay dos proyectos *open source* que implementan servicios de conferencias Web (compartir escritorio, tablero virtual, chat, etc.) y que se integran con Asterisk: BigBlueButton y OpenMeetings.

Los dos proyectos utilizan “red5sip”, que es un teléfono SIP que se puede usar en un navegador utilizando Flash, sin embargo ya se encuentran disponibles algunas implementaciones de teléfonos SIP basados en HTML5, lo cuál abre las puertas a una mayor integración de Asterisk con la Web.

Video

Asterisk cuenta con soporte nativo para video llamadas punto a punto, agregando la opción “videosupport=yes” en el archivo “sip.conf” y habilitando los códecs de video respectivos.

Los códecs incluidos son H.261, H.263 y H.264. El soporte no incluye traducción de los mismos, por lo que los dos teléfonos que se quieran comunicar deben tener el mismo códec.

Las aplicaciones del plan de marcado que soportan video son: *Record*, *Playback*, *Voicemail* y *Echo*.

Adicionalmente, se encuentran módulos de terceros que permiten implementar aplicaciones avanzadas de video como:

- Video Conferencia: app_conference, que soporta miembros tanto de video como de sólo audio.

- Video móvil 3G: app_h324m, app_rtsp, app_transcoder.

- Servidor DiaStar de Dialogic: video conferencia, *transcoding*, *rescalling*, IVR, *gateway* SS7 y H324m.

Otras posibilidades

Tal y como se mencionó en la introducción de este libro, es gracias a la increíble flexibilidad de Asterisk y a la disponibilidad de su código fuente, que es posible desarrollar gran cantidad de escenarios, donde con suficiente tiempo y conocimiento, la única limitación restante es la imaginación. Es así como otras de las muchas posibilidades de Comunicación Unificada que ofrece el mercado se mencionan a continuación:

- La empresa Lummenvox ofrece la posibilidad de integrar sus motores de reconocimiento de Voz y sintentizadores (*Text to Speech*) con Asterisk.

- Asterisk incluye soporte para enviar y recibir mensajes de texto SMS con los módulos “chan_sebi” y “chan_mobile”.

- Digium ofrece un módulo para integración con Skype.

- La empresa Phonetag ofrece un servicio que permite la transcripción de los mensajes de correo de voz a texto para envíalos por correo electrónico.

Capítulo 13. Gestionando un proyecto de adopción de Telefonía IP

> "Si conoces el lugar y la fecha de la batalla, puedes acudir a ella aunque estés a mil kilómetros de distancia."
>
> Sun Tzu, estratega militar y filósofo de la antigua China.

Entre los años 2003 y 2010, la empresa Avatar Ltda., ahora liquidada, realizó la implementación de más de 70 proyectos de telefonía IP, en empresas de diferentes industrias y de diferentes tamaños, desde micro y pequeñas empresas hasta operadores de telecomunicaciones.

A partir de esa amplia experiencia, se llegó a la documentación de las mejores prácticas y elementos a tener en cuenta para tener éxito en los proyectos de implementación de soluciones de telecomunicaciones basadas en Asterisk. A continuación se hace un resumen de los puntos más importantes:

Planeación

En todo proyecto de instalación de un sistema de comunicaciones, se parte de una etapa en la que se identifican y se documentan las necesidades que tiene la organización.

Generalmente los departamentos de tecnología de las empresas están concentrados en la operación diaria que soporta sus procesos críticos y no tienen mucho conocimiento específico del tema de telefonía, por lo que buscan una asesoría externa con diferentes proveedores con el fin de evaluar las opciones que ofrece el mercado y generar un documento que recoja sus requerimientos.

No todas las organizaciones tienen requerimientos iguales, ni la misma cultura organizacional de cara al uso de nuevas tecnologías. Por ejemplo, algunas áreas de una misma empresa pueden ser más abiertas a cambiar su teléfono de escritorio por una diadema y un softphone que otras, o algunas

empresas simplemente no cuentan con la infraestructura de red LAN o de Internet suficiente para un adecuado funcionamiento de la Telefonía IP.

Viabilidad

En algunos casos se deberá hacer primero un análisis de retorno de la inversión, en la que se calcule el costo actual de las comunicaciones de la empresa, para comparar los ahorros que se tendrán en llamadas entre sedes o de larga distancia. Otro costo importante, pero que es difícil de calcular, es el de las llamadas de clientes que se pierden porque nunca son atendidas y que impactan directamente sobre el flujo de ingresos y la imagen de la empresa.

Es conveniente realizar un análisis de costos y el tiempo de retorno de la inversión realizada en el proyecto. Para hacerlo, se deben analizar tanto el presupuesto del proyecto como los costos en los que está incurriendo actualmente la organización.

La siguiente tabla es un ejemplo de la elaboración de un presupuesto para un proyecto típico de adopción de telefonía IP[60]:

Ítem	Costo mensual recurrente	Costo inicial	Propiedad	Costo Adicional
Teléfonos IP	0	XXXXX	Sí	No
Gateways	0	XXXX	Sí	No
PBX IP	0	XXXXX	Sí	No
Interfaz PRI	0	XXXX	Sí	No
Consumo telefónico	XXX	0	No	No
Servicios profesionales	XXX	XXXX	No	Sí
Canales WAN	XXX	XXXX	No	No
Costo Total	*$XXXXX*	*$XXXXX*		

Tabla 13.1: Presupuesto del proyecto

60 WALLINGFORD, Theodore. Switching to VoIP. O'Reilly Media, 2005

La siguiente tabla es un ejemplo de la forma como se puede estimar el ahorro mensual que se obtendrá con Telefonía IP:

Ítem	Costo sin VoIP	Costo con VoIP	Ahorro mensual	Razón del ahorro
Larga distancia entre sedes	XXXX	xxxxxx	XXXXXXX-xxxxxxx	Utilización de Gateways en sedes
Larga Distancia Internacional	XXXX	xxxx	XXXXXXX-xxxxxxx	Utilización de un proveedor de minutos VoIP
Celular	XXXX	xxxx	XXXXXXX-xxxxxxx	Utilización de un proveedor de minutos VoIP
Líneas directas	XXXX	xxxx	XXXXXXX-xxxxxxx	Reducción de DID (MDE)
Impacto de llamadas perdidas	XXXX	xxxx	XXXXXXX-xxxxxxx	Manejo flexible de llamadas
Llamadas no autorizadas	XXXXX	xxxx	XXXXXXX-xxxxxxx	Restricción de llamadas
Total	*$XXXXXX*	*$xxxxx*	*$YYYYYYYY*	

Tabla 13.2: Calculo del ahorro mensual

Se puede determinar entonces el número de meses que tomará el retorno de la inversión de la siguiente manera:

Retorno de la inversión = Costo Inicial total / Valor de ahorro mensual

Análisis de Requisitos

En la Tabla I.1 se muestra un ejemplo del tipo de información que se debe recopilar para poder adelantar el diseño de una solución de telefonía IP

basada en Asterisk. La tabla incluye respuestas de ejemplo para dos escenarios diferentes, en base a los cuales se analizarán posibles alternativas de solución (diseño Hardware y Software).

Tema	**Preguntas**	**Ejemplo Escenario 1**	**Ejemplo Escenario 2**
Sedes	Número de sedes y extensiones requeridas	*2 sedes*	*1 sede*
	Número y tipo de troncales por sede. Número de canales entre sedes.	*1 PRI en la sede principal*	*1 PRI y 4 líneas analógicas*
	Existe una PBX en la sede? Cuantas extensiones tiene actualmente?	*Sí, pero muy deteriorada*	*Sí*
Plan de Marcado	Esquema de numeración actual. Directorio telefónico	*3 dígitos*	*3 dígitos*
	Identificar las extensiones con permisos de larga distancia y celular, etc.	*Sólo gerentes*	*Con clave*
	Numeración (prefijos) entre sedes o áreas.	*Dígito adicional*	
	DIDs (MDE) o directos		
Colas de Llamadas	Agentes por cola	*1 cola con 3 agentes*	*2 colas, con 10 agentes*
	¿Números 018000?	*No*	*Sí*
	Número de recepcionistas	*2*	*1*
IVR	¿Mensaje actual disponible?	*No*	*Sí*
	¿Música en espera actual?	*No*	*Sí*

Tema	Preguntas	Ejemplo Escenario 1	Ejemplo Escenario 2
Hardware, LAN, WAN, Energía y Espacio	Conexión del primario en RJ45 o coaxial (requiere Balun[61]). Distancia PRI-servidor (longitud del cable de primario)	*Coaxial con Balun*	*RJ45*
	¿Se reutilizará el cableado de voz a 8 hilos y LAN aparte (con DHCP) o el cableado de datos será el mismo para los teléfonos?	*LANs separadas (PoE, DHCP)*	*No hay puntos de red libres*
	Existen canales WAN entre las sedes, o la comunicación es por Internet? utilizan VPN?	*Internet, con VPN*	*No.*
	¿En los puestos hay tomas de energía/UPS para los teléfonos? (se recomienda UPS al menos para la recepción o un pasillo)	*No*	*No*
	Espacio en el Rack y puertos de switch para todos los equipos.	*No*	*Sí*
Otros	Se tiene alguna planta de celufijo?	*No*	*Sí*
	Se tienen cuentas de minutos VoIP con algún proveedor?	*No*	*Sí*
	Se tienen máquinas de Fax?	*No*	*Sí*

Tabla 13.3: Formato de captura de requerimientos para dos escenarios de ejemplo diferentes

Definición del alcance

Una vez recopilada esta información, se procede a elaborar el plan del proyecto y cronograma, previo a la reunión de arranque del mismo. Dentro del Plan se deben cubrir los siguientes elementos:

- Detalle del alcance definitivo acordado.
- Planeación y gestión del riesgo (posibles problemas con equipos, recursos, personal, locaciones o conectividad).

61 Un Balun (del inglés *Balance-Unlabance*) es un adaptador de impedancias para conectar un cable de primario de RJ45 a un par de cables coaxiales, de acuerdo al estándar G.703.

- Gestión de las adquisiciones del proyecto (validación de órdenes de compra y seguimiento a entrega por parte de proveedores o cliente). Normalmente esta tarea hace parte de la ruta crítica y puede dilatar todo el proyecto.

- Plan de Capacitación (coordinación de horarios y grupos de usuarios y administradores).

Diseño

Los criterios de diseño para una solución Asterisk dependen de muchos factores, pero en general se deben tener en cuenta los siguientes puntos:

Troncales requeridas

En el Capítulo 1 se cubrieron los diferentes tipos de troncales soportadas por Asterisk, algunas con más capacidad de líneas telefónicas que otras. Para dimensionar con exactitud el número de troncales que requiere el sistema para procesar todas las llamadas generadas por los usuarios que demandan el servicio sin perdidas por insuficiencia de canales, se puede hacer uso de algunos conceptos básicos de la teoría de tráfico.

Para esto se debe tener en cuenta el tráfico que el sistema debe cursar en la hora de mayor actividad, aunque en el resto del día algunas de las troncales estén libres. Se conoce como HORA PICO al periodo de 60 minutos consecutivos en el cual el tráfico telefónico es el más alto del día dependiendo del sistema; por ejemplo la hora pico en una organización podría ser de las 4 p.m. a las 5 p.m., pero para otra podría ser de 10 a.m. A 11 a.m.

La intensidad de tráfico, comúnmente llamada simplemente tráfico, se define como el promedio de las llamadas establecidas en la hora pico por la duración promedio de cada llamada. Es una cantidad adimensional a la cuál se le ha asignado una unidad llamada ERLANG, en honor a A.K. Erlang, ciudadano Danés pionero de la teoría de tráfico telefónico.

Se calcula de la siguiente manera:

$$A = Y * h / T$$

dónde:

A = Tráfico en Erlangs.

Y = Número de ocupaciones.

h = Promedio de duración de ocupaciones.

T = Tiempo de observación.

El máximo valor de tráfico que puede cursar una troncal es de 1 Erlang lo que significa que tiene una ocupación del 100%. En condiciones normales el valor del tráfico por circuito es siempre menor que uno, puesto que entre una ocupación y otra transcurre algún tiempo en cual el circuito esta libre. La única posibilidad de obtener 1 Erlang es haciendo que el circuito se encuentre ocupado todo el tiempo de observación, lo que ocurre muy rara vez en telefonía, donde el tiempo de observación es de una hora.

Para determinar el número de troncales necesarias para un sistema a partir del número de Erlangs de tráfico que genera, es necesario correlacionar el resultado con la siguiente tabla, denominada Tabla B:

Troncales requeridas	Congestión de 1% (Erlangs)	Congestión de 5% (Erlangs)
1	0.01	0.05
2	0.15	0.4
3	0.5	0.9
4	0.9	1.5
5	1.4	2.2
6	1.9	3.0
7	2.5	3.7
8	3.0	4.5
9	3.8	5.4
10	4.5	6.2
11	5.2	7.1
12	5.9	8.0
13	6.6	8.8
14	7.4	9.7
15	8.1	10.6
16	8.9	11.5
17	9.7	12.5
18	10.4	13.4
19	11.2	14.3
20	12.0	15.3
21	12.8	16.2
22	13.7	17.1
23	14.5	18.1
24	15.3	19.0

Tabla 13.4: Tabla B de Erlang

Ejemplo: En una empresa se reciben en la hora pico 60 llamadas entrantes y se generan 70 llamadas salientes. La duración promedio de las llamadas salientes es de 5 minutos y el de las entrantes es de 7 minutos. Calcular el tráfico total y número de líneas troncales requeridas:

- El tráfico saliente: *As = 70*5/60 = 5.83 Erl.*
- El tráfico entrante: *Ae = 60*7/60 = 7 Erl.*
- El tráfico total sería: *As + Ae = 12.83 Erlangs.*
- Según la Tabla B, para cursar esta cantidad de tráfico con una congestión deseada no mayor de 1%, se deben dimensionar al menos *21* líneas troncales.

No solo se debe tener en cuenta el número de canales que se usará para conectar con la RTPC, sino también con otras plantas telefónicas o troncales de voz sobre IP.

Dimensionamiento del Servidor

A la hora de dimensionar el hardware que va a soportar el servidor de telefonía, se deben tener claros los siguientes puntos:

- Número de ranuras PCI requeridas y cables de poder para adición de interfaces de telefonía y de red . No se recomienda más de 3 tarjetas por servidor.
- Manejo de interrupciones de la tarjeta madre (*motherboard*). Si se usan interfaces PCI debe consultarse al fabricante sobre la compatibilidad con la MotherBoard del servidor, ya que hay algunos problemas conocidos[62].

62 Se puede ver un ejemplo de esto en <http://www.digium.com/en/docs/misc/pci_slot.php>

- Asterisk requiere aproximadamente 40 Mhz de procesador por cada canal de voz (G.711) en i386. El procedimiento de compresión de la voz eleva el consumo de CPU (pero ahorra ancho de banda). Se puede estimar que cada canal G.729 equivale a más de 5 canales G.711.

- La cancelación de eco por software puede llegar a duplicar el procesamiento.

- La aplicación de conferencia (*MeetMe*) tiene un elevado consumo de recursos hardware.

- La invocación a programas externos (*scripting*) tiene un serio impacto en los tiempos de lectura y escritura al disco duro.

- Otros procesos que consuman recursos en el servidor. Generalmente se recomienda que el servicio de telefonía ocupe un servidor de manera exclusiva, sin compartir con otros servicios.

Recomendaciones sobre requerimientos mínimos:

Canales Simultáneos	**CPU y RAM**
Menos de 10	400 Mhz, 256 MB
10 a 50	1 Ghz, 512 MB
Hasta 100	3 Ghz, 1GB
Hasta 300	Doble o cuádruple procesador
Más de 300	Balancear la carga con otros servidores.

Tabla 13.5: Dimensionamiento del servidor

Redes y Terminales

En el Capítulo 1 se mencionaron los diferentes tipos de terminales soportados por Asterisk. Sin embargo, no todos los terminales son susceptibles de ser utilizados en todos los proyectos. Generalmente la elección de los terminales va ligada a las condiciones en que se encuentra la red de datos y la red eléctrica de la organización:

- Si existe un cableado estructurado, el jack telefónico debe estar cableado con 8 hilos de cobre. Esto abre la posibilidad para que se pueda utilizar como cable de datos, para conectar teléfonos IP.

- En un caso ideal, se prefiere que los teléfonos IP tengan su propia red de datos, independiente de la red de los usuarios, bien sea con switches de datos diferentes, o mediante LANs virtuales (VLANs). Esto permite el uso de un servidor DHCP para aprovisionar y actualizar automáticamente los teléfonos.

- En el caso de no existir puertos de red independientes para los teléfonos IP, se pueden utilizar teléfonos con doble puerto de red, que permiten hacer una derivación al PC, a modo de un Hub de datos.

- Si no se cuenta con conexiones eléctricas suficientes para los teléfonos IP, se pueden emplear switches con soporte para PoE (*Power over Ethernet*). Esto permite que sea el switch el que provea la alimentación eléctrica para el teléfono, haciendo más fácil el uso de UPS para contingencia en casos de cortes de electricidad.

- Para no incurrir en costosos teléfonos IP *wireless*, se pueden utilizar ATAs para conectar los teléfonos inalámbricos tradicionales.

- No todos los PC tienen una buena tarjeta de sonido para utilizar softphones. Se recomienda el uso de diademas USB para obtener una mejor calidad de audio.

Implementación

Con todos los criterios de diseño descritos previamente, se tienen elementos de juicio suficientes para realizar un diseño detallado de la solución. Continuando con el ejemplo, a continuación se presentan los diseños planteados para los escenarios de ejemplo 1 y 2:

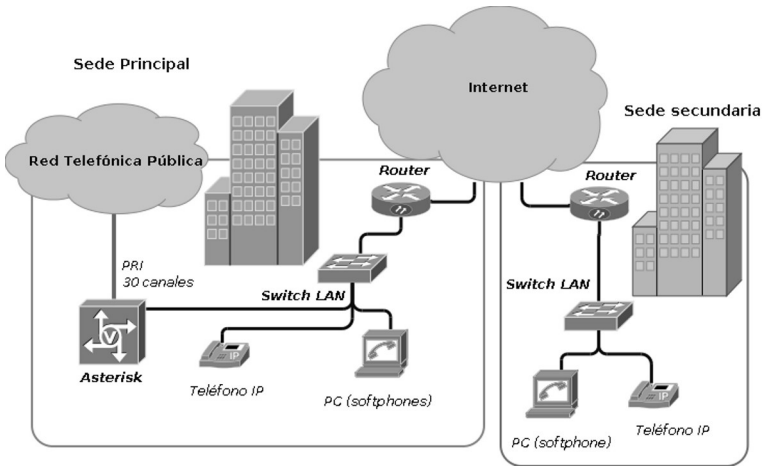


Figura 13.1: Posible diseño para el escenario de ejemplo 1

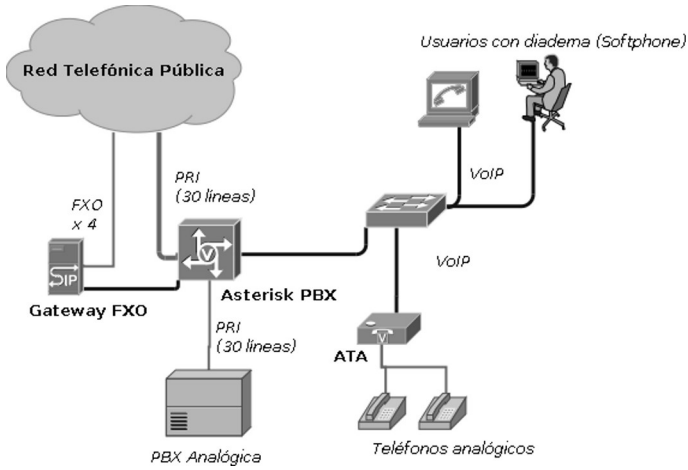


Figura 13.2: Posible diseño para el escenario de ejemplo 2

Una vez definido el diseño y puesto en marcha el plan del proyecto, se procede a la etapa de adquisición y compra de los equipos.

Mientras se realiza la ejecución del plan de compras, la cuál puede tomar un tiempo, se puede adelantar el desarrollo del Plan de Marcado, que consiste en la definición de la numeración de las extensiones que se van a tener en el sistema y de los principales servicios telefónicos de que van a hacer uso. También incluye la definición de los menús de operadora automática, para lo cuál se debe definir la forma como se desean enrutar las llamadas entrantes, desde que inician hasta que terminan, con las diferentes opciones que se le presentarán al llamante.

También es importante realizar unas pruebas previas de la conexión física del cable de primario y el Balun, ya que en algunos casos se deberá acudir al soporte técnico del operador telefónico para ajustar los parámetros correctamente. Se deben realizar pruebas de enganche (sincronización), duración de llamada y detección de dígitos DTMF.

Una vez se cuente con el hardware a la mano, las tareas de implantación se pueden resumir en las siguientes:

- Instalación física de las interfaces de telefonía (tarjetas PCI).
- Instalación de Sistema Operativo y utilidades básicas.
- Configuración de la solución según el tipo de requerimiento (plan de marcado para PBX, Call Center, etc).
- Aprovisionamiento de los terminales telefónicos del proyecto (automático o manual).
- Configuración de las Gateways y demás equipos de telefonía requeridos para el proyecto.

Pruebas y puesta en producción

Una vez realizada la instalación del hardware y software, se realiza la ubicación física de los equipos en el sitio definitivo: montaje en rack, conexión eléctrica/protecciones y conexión de red.

Posteriormente se realiza la activación de los servicios de telefonía: conexión de líneas troncales y conexión de otros equipos, ejecutando el plan de pruebas, previamente a la puesta en producción del sistema. A continuación se presenta un ejemplo de plan de pruebas para una PBX IP basada en Asterisk:

No.	Categoría	Descripción de la Prueba	Resultado
1	IVR	Cuando la PBX responde con el saludo de bienvenida, se puede marcar una extensión de la PBX.	
2	IVR	Si no se digita ninguna extensión u opción, al finalizar el saludo la llamada es conducida al teléfono de la recepcionista.	
3	Marcación entrante	Al pasar a una cola de atención de llamadas, una llamada rota entre las extensiones que componen la cola según la estrategia configurada.	
4	Marcación saliente	Se puede marcar desde una extensión con salida de llamadas (ej. Local, nacional, celular, etc.)	
5	Marcación saliente	Al marcar a un número local en donde responde una operadora automática que solicita digitar números a través de un teléfono, estos son capturados correctamente.	
6	Marcación entrante	Es posible marcar directamente a extensiones marcando números configurados como DID (Direct Inward Dialing)	
7	Buzón	Al marcar a una extensión, si ésta no responde después de n timbres, contesta el diálogo del buzón de mensajes de dicha extensión (en español) y permite dejar un mensaje.	
8	Buzón	Al marcar a una extensión, si ésta se encuentra ocupada, contesta el diálogo del buzón de mensajes de dicha extensión (en español) y permite dejar uno.	
9	Buzón	Si se marca la extensión del buzón de mensajes, responde el diálogo del correo de voz en español que permite consultarlos.	
10	Transferencia	Es posible hacer transferencia directa y asistida marcando un código de transferencia o utilizando la función incorporada por los teléfonos IP	
11	FAX	Se pueden enviar mensajes de FAX correctamente hacia la Red Telefónica Pública	

12	FAX	Se pueden recibir mensajes de FAX correctamente a través de la Red Telefónica Pública.	
13	Presencia	Se ejecutan correctamente las acciones al marcar los códigos de habilitar/deshabilitar redirección de llamadas	
14	Presencia	Se ejecutan correctamente las acciones al marcar los códigos de habilitar/deshabilitar buzón	
15	Presencia	Se ejecutan correctamente las acciones al marcar los códigos de habilitar/deshabilitar no-molestar.	
16	Administración Web	Se pueden crear, modificar y eliminar cuentas de usuario SIP y asignarles extensiones.	
17	Reportes Web	Se pueden consultar los registros de llamadas por origen, destino, entre fechas y por duración y estado de la llamada.	
18	Desempeño	Ejecutar prueba de llamadas simultáneas	

Tabla 13.6: Ejemplo de Plan de Pruebas para una PBX

Cierre del Proyecto

Una vez desplegado y probado el sistema, se procede a realizar la capacitación del personal técnico que administrará el sistema. Se entregan las guías de uso de los terminales telefónicos y servicios, y se lleva a cabo la capacitación de los usuarios finales, según el plan de horarios y grupos coordinados previamente.

Finalmente se realiza un acta mediante la cuál se entrega el proyecto a satisfacción y se inicia una etapa de soporte técnico y mantenimiento.

Índice

Referencias

- Asterisk Development Team. DAHDI Telephony Interface Driver. [en línea]. <http://docs.tzafrir.org.il/dahdi-tools/> [citado en 15 de febrero de 2012]

- Asterisk Guru. SIP with NAT or Firewalls [en línea]. <http://www.asteriskguru.com/tutorials/sip_nat_oneway_or_no_audio_asterisk.html> [citado en 25 de agosto de 2012]

- Asterisk Project Wiki. Asterisk Call Files. [en línea]. <https://wiki.asterisk.org/wiki/display/AST/Asterisk+Call+Files> [citado en 29 de noviembre de 2011]

- Asterisk Project Wiki. Asterisk Manager Interface. [en línea]. <https://wiki.asterisk.org/wiki/display/AST/Asterisk+Manager+Interface+(AMI)> [citado en 29 de noviembre de 2011]

- Asterisk Project Wiki. Asterisk Queues. [en línea]. <https://wiki.asterisk.org/wiki/display/AST/Asterisk+Queues> [citado en 29 de noviembre de 2011]

- Asterisk Project Wiki. Asterisk Versions. [en línea]. <https://wiki.asterisk.org/wiki/display/AST/Asterisk+Versions> [citado en 22 de agosto de 2012].

- Asterisk Project Wiki. Call Flow and Bridging Model. [en línea]. <https://wiki.asterisk.org/wiki/display/AST/Call+Flow+and+Bridging+Model> [citado en 22 de agosto de 2012].

- Asterisk Project Wiki. Channel Event Logging. [en línea]. <https://wiki.asterisk.org/wiki/display/AST/Channel+Event+Logging+(CEL)> [citado en 22 de agosto de 2012]

- Asterisk Project Wiki. IP Quality of Service. [en línea].

<https://wiki.asterisk.org/wiki/display/AST/IP+Quality+of+Service> [citado en 22 de agosto de 2012]

- Asterisk Project Wiki. SQLite3 astdb back-end. [en línea]. <https://wiki.asterisk.org/wiki/display/AST/SQLite3+astdb+back-end> [citado en 29 de noviembre de 2011]

- Asterisk Project. Architecture. [en línea]. <http://www.asterisk.org/architecture> [citado en 14 de septiembre de 2011]

- GONCALVES, Flavio. Building Telephony Systems with OpenSIPS 1.6. Packt Publishing, 2010

- MADSEN, Leif, VAN MEGGELEN, Jim y BRYANT, Russell. Asterisk: The Definitive Guide. 3ra edición. O'Reilly Media, 2011

- RENDÓN, Álvaro. Conmutación Digital: Señalización. Popayán: Universidad del Cauca, 1999. Borrador.

- RENDÓN, Álvaro. Sistemas de Conmutación: Fundamentos y Tecnologías. Popayán: Universidad del Cauca, 2000.

- ROJANO, Elio. Comunicaciones Unificadas en grandes infraestructuras. En: VoIP2DAY: Septiembre de 2009. Disponible en <http://www.sinologic.net>

- VoIP Wiki. GSM Gateways. [en linea] <http://www.voip-info.org/wiki/view/VOIP+GSM+Gateways> y <http://www.voip-info.org/wiki/view/GSM#PCIadapters> [citado en 2 de octubre de 2012]

- WALLINGFORD, Theodore. Switching to VoIP. O'Reilly Media, 2005

- WIKIPEDIA. H323. [en línea]. <http://es.wikipedia.org/wiki/H323> [citado en 29 de septiembre de 2011]

- WIKIPEDIA. Internet Protocol. [en línea]. <http://es.wikipedia.org/wiki/Internet_Protocol> [citado en 29 de septiembre de 2011]

- WIKIPEDIA. RSA. [en línea]. <http://es.wikipedia.org/wiki/RSA> [citado en 29 de septiembre de 2011]

- WIKIPEDIA. Session Description Protocol. [en línea]. <http://es.wikipedia.org/wiki/Session_Description_Protocol> [citado en 29 de septiembre de 2011]

- WIKIPEDIA. User Datagram Protocol. [en línea]. <http://es.wikipedia.org/wiki/User_Datagram_Protocol> [citado en 29 de septiembre de 2011]