Universidad Nacional Autónoma de Nicaragua

UNAN – León

Facultad de Ciencias y Tecnología

Departamento de Computación

Monografía para optar al título de Ingeniero en Sistemas de la Información

Tema:

Voz sobre IP implementada con Asterisk: Teoría y Práctica

Autores:

- *Br. Fernando Alfonso Larios Lira*
- *Br. Ada Luz Rivera Garmendia*
- *Br. Yesenia María Zapata Pulido*

Tutor:

Msc. Aldo René Martínez

León, Junio de 2010

Agradecimiento

A todos las personas de los foros que están siempre dispuestos a dar la mano, a la gran comunidad de Linux y a todos los que anónimamente aportan sus conocimientos.

CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

La Internet ha revolucionado la naturaleza de los negocios y el comportamiento social. Hoy en día la demanda de acceso crece desmesuradamente en relación a la demanda de la telefonía; esto se debe a las infinitas aplicaciones que ofrece, además de la facilidad de uso, accesibilidad y entretenimiento. Es por esto que la VoIP se hace cada día más apetecible para los fabricantes, los proveedores y usuarios ya sean de tipo corporativo o personal.

La tecnología Voz sobre IP básicamente es un conjunto de protocolos para transporte de voz sobre redes IP, considerada el futuro sustituto de las tecnologías de transporte de voz actuales, y va a provocar una revolución en cuanto a sus servicios, uso y sobre todo va a afectar de manera significativa a los precios actuales que los grandes operadores de voz (existentes en cada país) cobran por sus servicios.

A fin de competir con la telefonía clásica, las redes IP afrontan el desafío de ofrecer no sólo las mismas comunicaciones de voz de alta calidad, también los servicios avanzados de la misma.

La unión de la tecnología VoIP, sistema operativo Linux y software de libre distribución (Asterisk, softphone) permiten la producción de sistemas de telefonía privados de muy bajo costo, elevadas funcionalidades y flexibilidad de desarrollo.

Desarrollamos nuestro trabajo en base a software libre prestando atención en tener los servicios a bajos costos, no por ello descuidando la calidad del servicio, por el contrario, buscando tecnologías robustas que permitan la comunicación y optimización de recursos, además de compatibilidad entre dispositivos. La base de este trabajo es CentOS en su versión 5.2 y el software Asterisk también gratuito.

El documento consta de tres capítulos: el primer capítulo trata sobre los conceptos, componentes y algunas aplicaciones que comúnmente se encuentran en un sistema de telefonía convencional; el segundo capítulo abordan características de la tecnología VoIP, como son los tipos estándares, arquitectura, codecs de voz y una pequeña introducción de calidad de servicio para tráfico de voz en redes IP; el tercer capítulo comprende las particularidades de Asterisk y se documentan una serie de prácticas sobre configuración básica, interconexión de centralitas, lógica de marcado, administración de contextos, extensiones, buzón de voz, operadora automática, entre otros. Al final se encuentran los siguientes anexos que complementan la obra: manual de instalación del sistema base utilizada CentOS 5.2, proceso de instalación de Asterisk y voces en español.

ANTECEDENTES

Tradicionalmente, las grandes corporaciones han mantenido redes separadas para proporcionar los distintos servicios de voz, vídeo y datos que han precisado; dicha situación ha llevado a perpetuar elevados costes, no sólo por la contratación de los servicios de manera separada, sino también por la necesidad de mantener procedimientos de gestión y administración separados. Por esto surgen las **redes multiservicios** que tienen como objetivo la integración de los servicios de datos, voz y vídeo sobre una sola red. A esta tendencia se la conoce normalmente con el término "convergencia".

A partir de 1997 empiezan a aparecer nuevos dispositivos y métodos que nos llevan hoy en día a mantener el término XoIP (‘X’ over Internet Protocol) como la puerta hacia la convergencia de las redes. En este acrónimo X significa cualquier contenido susceptible de ser transmitido por una red (D = data, V = voz, F = fax, M = multimedia, etc.).

Ante la rápida extensión de los protocolos TCP/IP como estándares "de facto", en redes de comunicaciones se han desarrollado tecnologías y estándares para las redes multiservicios, que permiten resolver la satisfacción simultánea de los distintos requerimientos. Tal es el caso de la **tecnología para Voz sobre IP (VoIP).**

En la actualidad, se puede considerar que la tecnología VoIP se encuentra en una fase madura y ha sido adoptada por los principales proveedores del sector, para su soporte en sus principales sistemas hardware y software.

Actualmente nuestra sede no cuenta con suficiente material bibliográfico y no se ha realizado ningún trabajo similar por otros autores.

JUSTIFICACIÓN

La realización de este trabajo nace por la necesidad de contar con un documento teórico-practico de Telefonía y Voz sobre IP, que sea útil para estudiantes y profesores del departamento de computación, ya que hasta la fecha no se contaba con un trabajo de este tipo.

La telefonía IP y Voz sobre IP, en la actualidad, son áreas de mucha importancia en las empresas e instituciones, debido al ahorro económico que esta presenta en comparación con los costos de la telefonía tradicional. Por tal razón se pretende usar este trabajo como una asignatura electiva para la carrera de Ingeniería Telemática.

Se pretende introducir a los estudiantes conocimientos que le permitirán crear una central telefónica privada local teniendo como base la tecnología Voz sobre IP. Para ello, Linux y Asterisk se convierten en un factor esencial, ya que van a proporcionar la plataforma ideal para desarrollar e implementar un amplio abanico de posibilidades que no se podían ni imaginar antes, con el uso de sistemas telefónicos convencionales y propietarios.

OBJETIVOS

OBJETIVO GENERAL

- Desarrollar una documentación teórica que confronte la práctica del funcionamiento de un sistema de voz sobre IP basado en Asterisk.

OBJETIVOS ESPECÍFICOS

- Conocer las bases teóricas de la telefonía tradicional y de voz sobre IP.
- Exponer los protocolos de VoIP más utilizados.
- Explicar los aspectos más importantes del servidor Asterisk.
- Presentar la solución de prácticas con un apoyo teórico de las configuraciones más relevantes, usuales y útiles de Asterisk.

DISEÑO METODOLÓGICO

TIPO DE DISEÑO

La naturaleza del trabajo es documentativa y experimental. Durante la realización de éste se implementará parcialmente la tecnología en estudio. La integración con la PSTN no se realizará por la falta de recursos necesarios para adaptarse a ésta, por ello alternativamente se diseñaron prácticas con el objetivo que los usuarios no tengan que incurrir en costos.

Este trabajo muestra el funcionamiento de VoIP y la importancia que representa en el futuro de las telecomunicaciones.

FUENTE DE INFORMACIÓN

Debido a la naturaleza del trabajo la mayoría de las fuentes de información fueron constituidas por libros y documentos de la Internet.

RECOLECCIÓN DE INFORMACIÓN

El fuerte de la recolección de información ha sido la Internet, por su facilidad de acceso. En cuanto a esta fuente se presenta un problema: como se sabe, la red no cuenta con una regulación en la calidad de las publicaciones, causando así que mucha de la información flotante no sea precisa o inclusive errónea, lo cual representa un reto. La fuente se calificó como confiable si provenía de alguna institución y si no, era comparada en base a éstas con el propósito de evaluar su credibilidad.

TÉCNICAS E INSTRUMENTOS DE TRABAJO

Recursos de Software

- Sistema Operativo Windows XP y CentOS 5.2.
- Para la instalación se utilizó VMWare Server Console Versión 1.0.8.
- Asterisk, SoftPBX.
- SSH Secure Shell Client y Putty, para el acceso remoto a los servidores Asterisk.
- X-Lite y Zoiper, Cliente de Telefonía para Windows.
- Ekiga, Cliente de Telefonía para Linux.
- Microsoft Word 2007, Herramienta de apoyo para la edición de textos.
- Microsoft Power Point 2007, Herramienta de apoyo para la edición de textos.

Recursos de Hardware

- Portátil procesador Intel Core Duo T2350 con 1 GB en RAM, 120 Gb de espacio en disco duro y una tarjeta de red 3com. Se creó una máquina virtual con 8 Gb de disco.

CAPÍTULO. 1 TELEFONÍA TRADICIONAL

La Telefonía Tradicional también denominada PSTN (Public Switched Telephone Network) incluye redes conmutadas de dos tipos (analógica y digital), es decir, la Red Telefónica Básica (RTB[1]) y la Red Digital de Servicios Integrados (RDSI[2]).

RTB fue creada para transmitir la voz humana y tanto por la naturaleza de la información a transmitir, como por la tecnología disponible en la época en que fue creada (siglo XIX), es de tipo analógico, primero de conmutación humana (telefonistas) después de conmutación automática (electro-mecánica). Eran propensas al ruido, a pérdidas de conexión, y no facilitaban el establecimiento de llamadas a larga distancia.

A principios de los 60, fueron sustituyéndose gradualmente las primitivas y gigantescas centrales telefónicas convencionales por otras modernas de funcionamiento digital. No hay que confundir "línea analógica en central digital" con "línea digital". La primera, sigue siendo totalmente analógica, aunque esté conectada a una central digital donde los sistemas de conmutación ya no son de tipo electromecánico.

RDSI es una red que procede por evolución de la Red Digital Integrada (RDI) y que facilita las conexiones digitales de extremo a extremo para proporcionar una amplia gama de servicios, independientemente de la naturaleza de la información a transmitir y del equipo que la genere.

En este caso la central digital sólo proporciona algunas pequeñas ventajas adicionales, como lo es la posibilidad de marcar por tonos, llamadas en espera, transferencia de llamadas, facturación detallada, buzón de voz, entre otras. A las líneas analógicas solo se les pueden conectar dispositivos telefónicos de tipo

[1] También conocida como RTC (Red Telefónica Convencional).
[2] En inglés ISDN (Integrated Services Digital Network). Sistema para transmisión telefónica digital con línea y adaptadores especiales.

análogo, es decir, teléfonos, módems o máquinas de fax entre otros. La línea digital por contra, sólo transporta ceros y unos (mejor sería decir dos niveles de tensión o de luz) y por supuesto solo permite la conexión de dispositivos de este tipo.

Ambas redes (RTB y RDSI) están basadas en un sistema denominado "conmutación de circuito", el que establece un canal dedicado que se mantiene activo entre ambos puntos mientras dura la llamada, lo que hace posible la comunicación. Una vez terminada la llamada, se libera el canal.

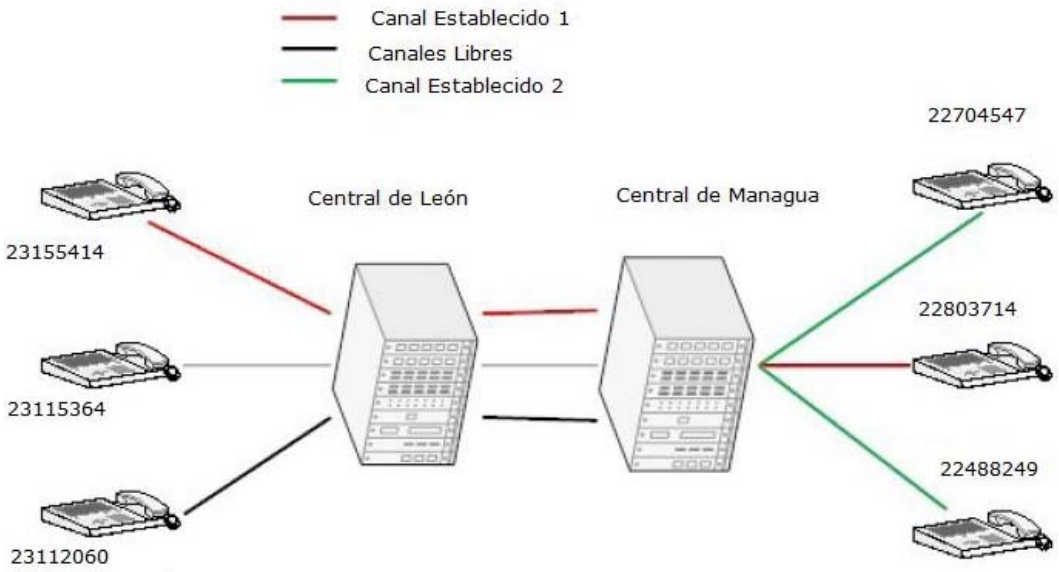

Figura 1. Conmutación de circuitos

En la figura, se logra apreciar que cada línea de teléfono tiene un número (dirección telefónica), las cuales se extienden desde la central telefónica hasta el teléfono (abonado). Cada central atiende las líneas de teléfono de un área geográfica determinada. A su vez la centrales telefónicas están unidas entre si y gracias a esto se constituye el sistema telefónico tradicional.

El sistema telefónico se compone de:

1. Circuitos locales (cables de par trenzado que van hacia las casas y empresas).
2. Troncales (fibra óptica digital que conecta a las oficinas de conmutación).
3. Oficinas de conmutación (donde las llamadas pasan de una troncal a otra).

1.1 FASE DE UNA LLAMADA EN LA TELEFONÍA TRADICIONAL

Durante una llamada se producen tres fases en la telefonía tradicional, estas son:

- **Establecimiento de la llamada**
  - En el Teléfono Tradicional se digita el número del destino y se envía la solicitud para realizar la conexión.
  - Verificar el destino
    - Si un suscriptor conectado a una oficina central determinada llama a otro suscriptor conectado a la misma oficina central, el mecanismo de conmutación dentro de la oficina establece una conexión eléctrica directa entre los dos circuitos locales.
    - Si el teléfono llamado está conectado a otra oficina central, se tiene que usar un procedimiento diferente. Cada oficina central tiene varias líneas salientes a uno o más centros de conmutación cercanos, llamado oficinas interurbanas (o si están dentro de la misma área local, oficinas en tándem). Estas líneas se llaman troncales de conexión interurbana. Si sucede que tanto la oficina central de quien llama como la de quien es llamado tiene una troncal de conexión a la misma oficina interurbano (algo muy probable si no están muy alejadas), la conexión se puede establecer dentro de las oficinas interurbanas.
    - Si el que llama y el que es llamado no tiene una oficina interurbana en común, la trayectoria se deberá establecer en un nivel más alto de la jerarquía. Hay oficinas primarias, seccionales

y regionales que forman una red de conexión a las oficinas interurbanas. Las centrales interurbanas primarias, seccionales y regionales se comunican entre sí mediante, troncales interurbanas de gran ancho de banda. La cantidad de tipos diferentes de centros de conmutación y su topología varían de país a país dependiendo de su densidad telefónica.

- La comunicación se establece si el extremo receptor acepta la petición de llamada de ser así se crea un canal dedicado, es decir, un canal que será permanente y exclusivo para ambos usuarios mientras dure la llamada.

- **Transferencia de voz:** Una vez que se establece la llamada o circuito se procede a convertir las señales acústicas en señales eléctricas, su transporte a través de los medios de comunicación, y la conversión de señales eléctricas nuevamente en acústicas para ser entregadas al destinatario.

- **Fin de la llamada:** Una vez que se deja de transmitir voz a través del canal, la conexión finaliza por orden de una de las dos estaciones (teléfonos) involucradas en la conversación. Cuando se produce la desconexión se liberan los recursos que se encontraban en uso al realizar la llamada, es decir se libera el canal.

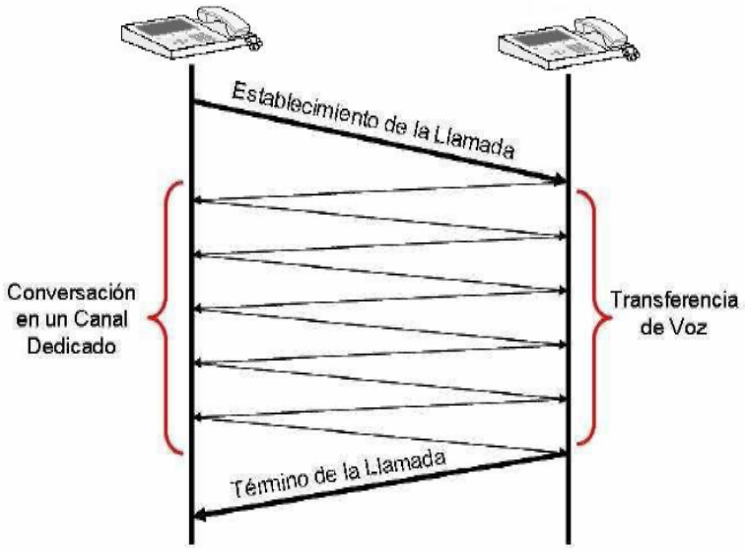

Figura 2. Establecimiento de una llamada telefónica tradicional

1.2 PBX

Un PBX[3], se refiere al dispositivo que actúa como un ramificación de la red primaria pública de teléfono, por lo que los usuarios no se comunican al exterior mediante líneas telefónicas convencionales, sino que al estar el PBX directamente conectado a la RTC, será esta misma la que enrute la llamada hasta su destino final mediante enlaces unificados de transporte de voz llamados líneas troncales. En otras palabras, los usuarios de una PBX no tienen asociada ninguna central de teléfono pública, ya que es el mismo PBX que actúa como tal, análogo a una central pública que da cobertura a todo un sector mientras que un PBX lo ofrece a las instalaciones de una compañía generalmente.

Una PBX no sólo permite compartir un conjunto de líneas con un grupo de usuarios, sino que también ofrece la posibilidad de crear servicios de valor añadido; como transferencia de llamadas, llamadas a tres, pasarela de voz a correo o servicios basados en una respuesta de voz interactiva (IVR), etc.

[3] Siglas en inglés de Private Branch Exchange o Private Business eXchange.

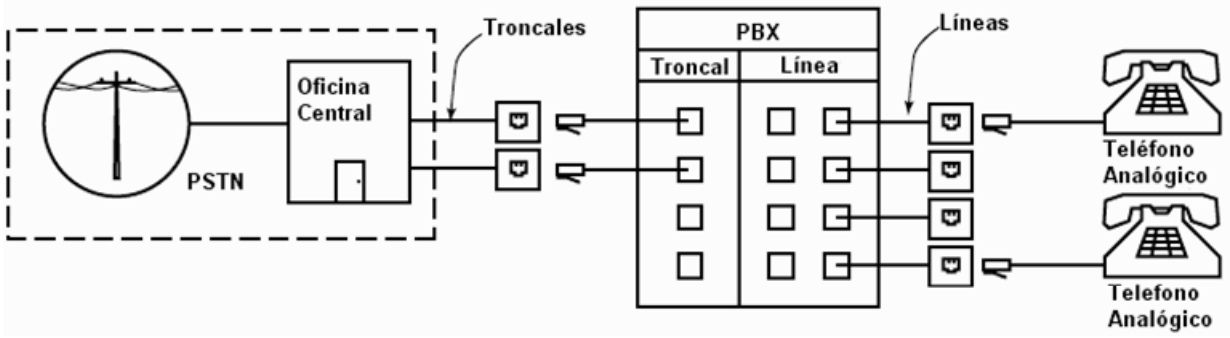

Figura 3. Escenario Convencional de una PBX.

Erróneamente se le llama PBX a cualquier central telefónica aunque no gestione las llamadas externas, bastando sólo con que conmute líneas exteriores pertenecientes a otra central que sí estaría conectada a la RTC. Estas serían centrales híbridas: gestionan llamadas y enlazan líneas internas o extensiones pero al momento de comunicarse a un destino exterior, tan sólo interconectaría el terminal con una línea convencional de la compañía de teléfono, mientras que un PBX se encargaría de procesar directamente el número marcado hacia el procesador central de la ciudad.

El método de conexión para pequeñas y medianas empresas no deja de ser por líneas comunes de la compañía telefónica, utilizando cuantas líneas quiera tener el usuario, y éstas a su vez conectadas a la central telefónica, que a pesar de que podría tratarse de un PBX, no estaría funcionando como tal, y tan sólo como una central privada híbrida. Esto se debe a que el tráfico de la llamada entrante o el inicio de la llamada saliente se originan en la central pública de la empresa de telefonía, probablemente al igual que otros abonados de la zona, mientras que si trabajase como PBX, el tráfico de llamadas culminaría o se iniciaría en la misma centralita.

1.3 SEÑALIZACION UTILIZADA EN TELEFONIA

En el uso de una línea telefónica se intercambian un conjunto de "señales" que se transmiten entre el FXS[4] y el FXO[5]. La señalización es el lenguaje que las redes telefónicas utilizan para comunicarse entre sí y para hablar con los equipos terminales de los abonados, así mismo es el elemento clave para transportar la información de comportamiento, rutas y fallas de circuitos. Algunos ejemplos de señalización son el envío del número telefónico, el tono de llamada o de ocupado y la información del número del que se llama.

En la RTB, voz y datos están separados

- Un "circuito" es para voz (la conversación).
- Un segundo "circuito" es para las señales de administración y supervisión (SS7).

Estos circuitos de información no tienen que usar el mismo canal físico.

1.3.1 MÉTODOS DE SEÑALIZACIÓN

Inicio de marcación: Asegura que el switch telefónico receptor está preparado para interpretar los dígitos (número de teléfono de destino) transmitidos por el switch telefónico emisor.

Transmisión de Dígitos: Los teléfonos y switches transmiten dígitos para representar las direcciones de destino y proporcionar así entradas de los usuarios a los sistemas automatizados como buzón de voz, distribuidores automáticos de llamadas (ACD) y sistemas de respuesta interactiva (IVR).

[4] Ver glosario Foreign eXchange Station (FXS).
[5] Ver glosario Foreign eXchange Office (FXO).

Marcación por Pulsos: Los teléfonos originales no tienen un método para transmitir dígitos. El esquema de pulsos se implemento con teléfonos de marcación giratoria. Cada número en el esquema de marcación por pulsos se señala como una serie de pulsos *make/break*, donde la posición make es el estado *off-hook* (descolgado), y la porción *break* es el estado *on-hook* (colgado).

Cada digito se representa por un número correspondiente al número de *break* en el circuito.

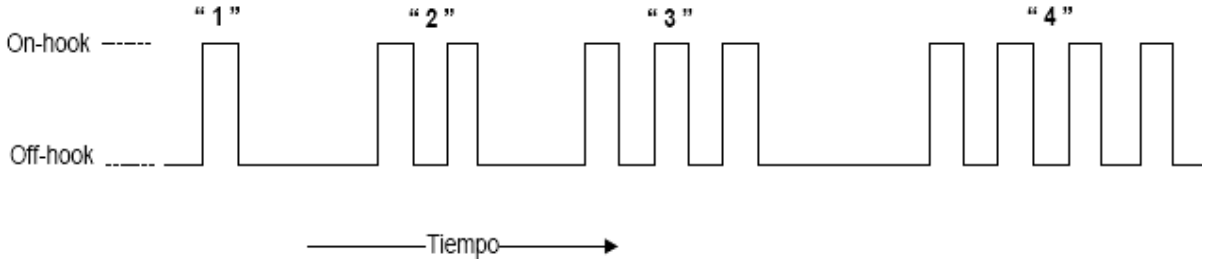

Figura 4. La marcación por pulsos.

Marcación por Tonos (DTFM): Son los sonidos que se escuchan al presionar las teclas de un teléfono, los sonidos son utilizados como señales para el switch, así éste puede identificar las direcciones de destino. La señalización DTFM reemplazo la marcación por pulsos.

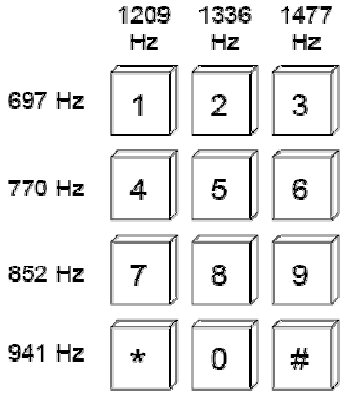

Figura 5. Teclado del teléfono DTFM

1.3.2 SENALIZACIÓN ENTRE CENTRALES TELEFÓNICAS

Sistema de Señalización #7 (SS7[6]): Es una colección de protocolos usada por los proveedores de servicios de telefonía para soportar la señalización de llamadas fuera de banda y las características más avanzadas de llamadas, se encargan de la gestión, del establecimiento de llamadas y su encaminamiento entre centrales telefónicas en la RTB. Es la norma internacional para la señalización en circuitos conmutados entre los proveedores de servicio de telefonía. Como el tráfico de SS7 se origina y termina dentro de las redes del proveedor, la inmensa mayoría de las empresas no necesitan preocuparse por el funcionamiento interno de SS7. Sin embargo los ISP requieren SS7 para la interconexión a gran escala con los proveedores de telefonía tradicional (LEC[7], IXC[8]).

[6] Es un grupo de estándares desarrollados originalmente por la AT&T y la UIT.
[7] Local Exchange Carriers o Proveedor de intercambio Local.
[8] Inter-eXchange Carriers, son operadores de conectividad entre Carriers o PSTN.

CAPÍTULO. 2 VOZ SOBRE IP

2.1 DEFINICIÓN DE VoIP

La voz sobre IP, es una tecnología que permite transportar llamadas de voz a través de las redes de datos, públicas o privadas.

2.2 FUNCIONAMIENTO DE UNA LLAMADA EN UN SISTEMA VOIP

- ***Inicio de sesión multimedia.*** Útil para determinar los contactos conectados, almacenar la dirección IP del cliente, identificar el tipo de conexión y determinar si el usuario estar detrás de un NAT o un Firewall.
- Se genera una señal desde la computadora o equipo que inicia la conversación. Se realiza una traducción de número de teléfono (o identificación de usuario), con las direcciones IP. Esta traducción se realiza automáticamente por el software, pero es necesaria para poder mapear los diferentes usuarios y/o números de teléfono a su correspondiente en la red de datos de Internet.
- Al contestar la persona en la otra computadora, la información que el micrófono recoja es ***digitalizada*** (conversión voz a datos – bits)***, codificada*** (para la transmisión o almacenamiento) ***y comprimida.*** Para ello se utilizan algoritmos matemáticos implementados en software llamados codecs acrónimo de codificador – decodificador, aunque principalmente se utilizan como compresores - descompresores. A mayor compresión menor calidad de audio.
- ***Encapsulamiento.*** La información se encapsula según el protocolo IP en un datagrama.
- ***Transmisión*** Se envía como un paquete IP cualquiera, permitiendo conexiones con otros sitios o programas a través de la red, al mismo tiempo. Cada paquete puede tomar caminos distintos.

- ***Decodificación.*** Cuando el paquete llega a su destino (dentro de una tolerancia de tiempo) es necesario restaurar la señal a la forma original para que el audio sea recibido por el usuario.
- Esto continúa en ambas vías, hasta que se corta la llamada, en cuyo caso se para el flujo de información entre ambas computadoras.

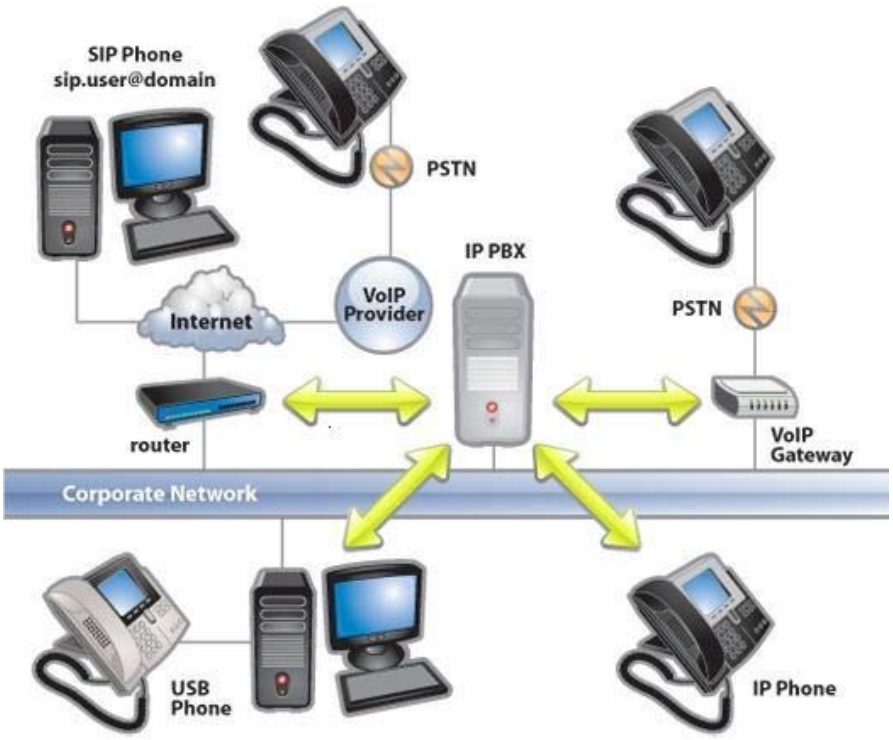

Figura 6. Funcionamiento de una llamada en un sistema VoIP.

2.2.1 CALIDAD DE SERVICIO

Uno de los grandes retos al implementar VoIP, especialmente en regiones en desarrollo, es garantizar que exista un ancho de banda constante para las conversaciones independientemente del estado del resto de las conexiones (incluso si la conexión a Internet está altamente ocupada), además de mitigar los retardos excesivos y así lograr una calidad de voz optima, siendo necesario implementar métodos de calidad de servicio o QoS[9].

[9] Capacidad de la red para ofrecer mejoras en las características de la entrega de paquetes de voz.

2.2.2 FACTORES EN LA TRANSMISIÓN DE VOZ

Latencia o retardo, es el tiempo que tarda un paquete en viajar de un punto a otro. Para mejorar la calidad de las conversaciones de voz sobre IP es necesario reducir los retardos al máximo, dando la máxima prioridad al tráfico de voz, esto significa que se les deja “saltarse la cola” y así ocupar una mejor posición que el resto de los paquetes que están esperando para ser transmitidos.

El retardo puede ser de dos tipos:

Constante. Siempre generan la misma cantidad de retardo, las más importantes son:

- *Codificación*, se da al tomar el audio y procesarlo por un codec específico.
- *Paquetización,* es generado al tomar el audio y convertirlo en paquetes IP.
- *Serialización,* producido al colocar los paquetes de voz desde las capas de aplicación hasta la interfaz por la cual será trasmitido.

Variable. Generan diferentes cantidades de retardo según las condiciones del medio, las más importantes son:

- *Encolamiento,* se genera cuando los paquetes de voz tienen que esperar en las colas de los equipos activos a ser trasmitidos.
- *Propagación,* ocurre 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.
- *Reproducción de Buffers:* Si el buffer es lo suficientemente grande para adecuar la variación del tiempo de llegada entre las muestras de audio recibidas, así este retardo podría ser reducido.

El retardo tolerado (suma de todos los retardos) por el oído humano esta entorno a los 200-250 ms, por lo que un valor apropiado debe ser menor de 200 ms.

Jitter – Fluctuaciones de velocidad, es la variación en el retardo. En términos simples, es la diferencia entre el tiempo en que llega un paquete y el tiempo en que se cree que llegara el paquete. Estas variaciones son debidas a la saturación de la red, la falta de sincronismo o los cambios dinámicos en las rutas.

Para corregir el Jitter (en redes con grandes cambios de velocidad) existen los "jitter buffer", este buffer es un registro intermedio donde se almacenan los paquetes hasta su procesamiento. La idea básica del "jitter buffer" es retrasar deliberadamente la reproducción del sonido para garantizar que los paquetes más "lentos" hayan llegado. Los paquetes se almacenan en el buffer, se reordenan si es necesario y se reproducen a una velocidad constante. La calidad de voz mejora al precio de incrementar la latencia total. Si la llegada de paquetes es demasiado desigual el buffer no la alcanza a controlar y perderá paquetes, deteriorando la calidad de la voz.

Muchos equipos de VoIP permiten ajustar el tamaño del "jitter buffer." El buffer es esa área donde los paquetes se almacenan para luego ser enviados al procesador de voz en intervalos constantes. El tamaño del buffer se mide en milisegundos. Si el buffer es de 200 ms significa que introducimos un retraso (esperamos ese tiempo) antes de reproducir la voz.

Existen dos tipos de jitter buffers: estático y dinámico. Un buffer estático está implementado como parte del equipo y configurado de manera fija por el fabricante. El dinámico se configura usando un programa y lo puede cambiar el usuario. Un valor común del jitter buffer es de 100 ms. Al incrementar el buffer vamos a mejorar la calidad de la conversación pero no olvide que lo que está haciendo es incrementar el retardo total (latencia). Debe buscar un valor de compromiso. Tenga en cuenta que un retraso total muy por encima de 300 ms hace muy difícil tener una conversación.

Pérdida de paquetes: Se producen principalmente por congestión en la redes o por fallos de comunicación. No nos referimos únicamente a la pérdida completa del paquete, que no llega a destino, si no a la llegada de paquetes después de un tiempo determinado, lo que provoca que el paquete sea inservible y es por tanto descartado.

Los diferentes codecs[10] pueden predecir los paquetes perdidos y reemplazarlos, pero cuando esta pérdida es superior al 5%, los codecs implementados no pueden predecir el valor del paquete perdido y se notará en la comunicación de voz que este paquete falta, disminuyendo la calidad de la comunicación.

Los codecs pueden:

- ***Interpolar***, cuando falta un paquete, el codec, toma el paquete anterior y el paquete siguiente y calcula el valor del paquete faltante.
- ***Sustituir,*** cuando el codec detecta un paquete faltante lo remplaza por un paquete igual al paquete anterior.

Eco: Se define como una reflexión retardada de la señal acústica original. Los principales productores de eco en la telefonía VoIP son los interfaces FXS y FXO, por lo que su calidad incidirá en la calidad de la voz. Los retardos en redes IP suelen ser mayores que en la red de telefonía tradicional, estos son directamente proporcionales a la intensidad del eco. Normalmente la señal de vuelta tiene menor potencia que la original siendo este otro factor que aumenta el eco.

Existen diferentes mecanismos para corregir el eco, tanto software como hardware, estos son:

- *Supresores de eco*: Consiste en evitar que la señal emitida sea devuelta convirtiendo por momentos la línea full-duplex en una línea half-duplex, de tal manera que si se detecta comunicación en un sentido se impide la

[10]Ver la sección 2.3.1

comunicación en sentido contrario. El tiempo de conmutación de los supresores de eco es muy pequeño.

- *Canceladores de eco:* Es el sistema por el cual el dispositivo emisor guarda la información que envía en memoria y es capaz de detectar en la señal de vuelta la misma información (tal vez atenuada y con ruido). El dispositivo filtra esa información y cancela esas componentes de la voz.

2.3 ESTÁNDARES Y PROTOCOLOS DE TRANSMISIÓN EN VOIP

2.3.1 CODECS DE AUDIO

Existen diferentes codecs de audio utilizados en VoIP, y dependiendo del algoritmo escogido en la transmisión, del ancho de banda necesaria, y la carga computacional variará la calidad de la voz. El objetivo principal de esta tecnología es encontrar un equilibrio entre eficiencia y calidad de voz.

Aunque el sistema auditivo humano es capaz de captar las frecuencias comprendidas entre 20 Hz y 20 KHz, la gran mayoría de codecs procesan aquella información dentro de la banda de 400 Hz a los 3.5 KHz, ya que con esto es suficiente para reconstruir la señal original.

A continuación se enumeran algunos de los codecs más comunes:

G.711: Principal codec de la PSTN estandarizado por la ITU (Internacional Telecommunication Union) en 1972. Este utiliza PCM (Pulse Code Modulation), la señal analógica de conversación es filtrada para eliminar los componentes de frecuencia alta y baja. La porción de bits es de 64Kbps (calidad de audio en el tramo de 3 KHz) lo que determina el tamaño de un canal de voz. El valor de cada muestra es codificado usando una de las técnicas de codificación:
μ-law: codifica cada 14 muestras en palabras de 8 bits. Usado en EE.UU. y Japón.

α-Law: Codifica cada 13 muestras en palabras de 8 bits. Usado en el resto del mundo.

Este es el algoritmo más simple y de menos carga computacional, ya que no realiza compresión en la codificación y es la base del resto de estándares.

G.722: Utiliza la codificación PCM proporcionando calidad de audio a 64 Kbits (en el tramo de 7 KHz).

G.723.1: Este algoritmo, estandarizado en 1995 por la ITU, utiliza dos algoritmos MP-MLQ (Multi-Pulse Maximum Likelihood Quantization) y ACELP (Algebraic Code Exited Linear Prediction) , puede operar a 6,3 kbps o 5,3 kbps. Este codec debe ser licenciado para poder ser usado.

G.726: Este Standard de la ITU, también conocido como ADPCM (Adaptive Differential Pulse Code Modulation, Modulación por impulsos codificados diferencial y adaptable), sustituyó al obsoleto Standard G.721 en 1990. Es un codec más avanzado que el PCM, en lugar de mandar los valores reales PCM, transmite una señal de error que es la diferencia entre la entrada real y la estimada. Su rendimiento y calidad a 40 Kbps son comparables a G.711. Permite conseguir un ancho de banda de 16 kbps, 24 kbps, y 32 kbps. La ventaja de este codec es la disminución del ancho de banda sin incrementar la carga computacional.

G.728: Utiliza el algoritmo CS-ACELP (Conjugate Structure Algebraic Codec Exited Linear Prediction) con un ancho de banda 16 Kbits a 3KHz

G.729[11]**:** Desarrollado por un consorcio de organizaciones: France Telecom, Mitsubishi Electric Corporation, Nippon Telegraph and Telephone Corporation (NTT) y la Universidad de Sherbrooke; patentado por la ITUT. Su algoritmo CS-

[11] El precio del codec es de 10 USD. Ver http://www.digium.com/en/products/voice/g729codec.php.

ACELP, codifica el audio cada 25 milisegundos con este algoritmo necesitaríamos un ancho de banda mínimo de 8Kbps (aprox. 30 kbps por conversación usando SIP), brinda una calidad de sonido impresionante. Debido a que esta patentado, no es posible utilizarlo sin el pago de las licencias, sin embargo su uso es muy popular y es soportado por varios dispositivos telefónicos, su carga computacional es elevada. No puede transportar tonos como DTMF, o fax, pero es el que menor tasa de bits proporciona (8 kbps).

GSM (RPE-LPT): Este codec aunque conocido popularmente por GSM, por ser usado en este tipos de redes (Codec Global System for Mobile Comunications, es utilizado en telefonía móvil digital en Europa, y en otras partes del mundo) su nombre original es: Regular Pulse Excitation-Long Prediction Term. Este codec codifica a 13 kbps con una carga computacional media, y no requiere el pago de licencia. Una buena opción por ser libre y tener una buena relación calidad/ancho de banda.

Speex: Codec de software libre diseñado para voz, con la idea de permitir la entrada al mercado de voz a más usuarios, al proveer una alternativa gratuita a los codec patentados. Entre sus aspectos fuertes es la capacidad de ofrecer una tasa de bits variable en la misma comunicación. Existen dos modos de funcionamiento para redes de banda ancha (SpeexWide, a 32 KHz) y de banda estrecha (Speex Narrow a 8 KHz), proporcionando diferentes tasas de bits en cada caso. Es un codec emergente que se está incluyendo en la mayor parte de las aplicaciones libres de VoIP, facilitando a demás módulos para aplicaciones Windows.

iLBC (Internet Low Bit-Rate Codec): Este codec muestrea cada 8 KHz, y utiliza para la codificación LPC y codifica a 15.2 kbps o 13.3 kbps. Este codec es libre, y no necesita ser licenciado.

Nombre	Estándar	Bit rate[12] (kb/s)	Sampling rate[13](KHz)	Frame size[14] (ms)	MOS[15] (Mean Opinion Score)
G.711	ITU-T	64	8	Muestreada	4.1
G.723.1	ITU-T	5.6/6.3	8	30	3.8-3.9
G.726	ITU-T	16/24/32/40	8	Muestreada	3.85
G.729	ITU-T	8	8	10	3.92
GSM	ETSI	13	8	22.5	3.5-3.7
Speex	-	8, 16, 32	2.15-24.6 (NB) 4-44.2 (WB)	30 (NB) 34 (WB)	-
iLBC	-	15.2 / 13.3	8	20/30	4.1

Tabla 1. Resumen con los codecs más utilizados actualmente.

2.3.2 DIRECCIONAMIENTO

DNS (Domain Name Service). Servicio de resolución de nombres de servidores de acuerdo a la dirección IP de destino a través de un servidor DNS.

2.3.3 TRANSMISIÓN DE VOZ

RTP[16] (Real Time Protocol). Intenta brindar cierta regularidad a las aplicaciones que utilizan Internet para el transporte del tráfico sensible al tiempo como la voz, datos y vídeo. Maneja los aspectos relativos a la temporización, marcando los paquetes UDP con la información necesaria para la correcta entrega de los mismos en recepción.

[12] Cantidad de información que se manda por segundo.
[13] Frecuencia de muestreo de la señal vocal. Cada cuanto se toma una muestra de la señal analógica.
[14] Cada cuantos milisegundos se envía un paquete con la información sonora.
[15] Calidad general del codec (valor de 1 a 5), refiriéndose principalmente a la calidad de la voz transmitida.
[16] Definido RFC 3550.

UDP. Es más atractivo para aplicaciones en tiempo real. Es no orientado a la conexión, no hace reenvió de paquetes al ocurrir errores lo que permite mayor aprovechamiento del ancho de banda es relación TCP.

TCP: Es orientado a la conexión lo que hace parecer el más indicado para VoIP, pero este siempre reenviará segmentos perdidos o descartados por la capa IP debido a errores, aumentando los retardos.

2.3.4 CONTROL DE LA TRANSMISIÓN

RTCP (Real Time Control Protocol). Va unido a RTP, proporciona información básica sobre los participantes de las sesiones y la calidad de servicio. Se utiliza principalmente para detectar situaciones de congestión de la red y tomar, en su caso, acciones correctoras.

2.4 PROTOCOLOS DE INICIO DE SESIÓN MULTIMEDIA DE VOIP

VoIP es una tecnología que aún no tiene un estándar universal para el inicio de sesión multimedia, de tal manera que los fabricantes han privilegiado el uso de protocolos propietarios que dificultan la interoperabilidad e integración entre dispositivos.

Las alternativas tecnológicas de VoIP se pueden dividir de una manera sencilla en dos grandes grupos tecnologías: cerradas propietarias y sistemas abiertos. En el primer grupo de tecnologías nos encontramos con el conocido Skype o el ya legendario Cisco Skinny. En el segundo grupo de tecnologías nos encontramos con los estándares abiertos basados en SIP, H.323, IAX, Megaco y MGCP, siendo los dos primeros los más utilizados.

2.4.1 PROTOCOLO H.323

El estándar H.323[17] especifica como proveer servicios multimedia sobre diversas redes. En concreto, proporciona la base para la transmisión de voz, datos y vídeo sobre red IP, incluida Internet. Aunque es un estándar para todo el rango de comunicaciones multimedia su mayor reconocimiento ha sido como fundamentos de soluciones de voz sobre IP. No garantiza una calidad de servicio, y en el transporte de datos puede, o no, ser fiable; en el caso de voz o vídeo, nunca es fiable. Además, es independiente de la topología de la red y admite pasarelas, permitiendo usar más de un canal de cada tipo voz, vídeo y datos al mismo tiempo.

Tiene como principal objetivo asegurar la supresión de silencios, codificación de la voz, direccionamiento e integrar nuevos elementos para permitir la conectividad con la infraestructura telefónica tradicional. Estos elementos se refieren básicamente a los servicios de directorio y a la transmisión de señalización por tonos multifrecuencia (DTMF).

PROTOCOLOS QUE UTILIZA H323

Comprende a su vez una serie de estándares y se apoya en una serie de protocolos que cubren los distintos aspectos de la comunicación:

[17] Recomendación de ITU-T (International Telecomunication Union)

<table>
<tr><td>Voz</td><td colspan="6">Control</td></tr>
<tr><td>Comprensión de audio G.7xx</td><td>DTMF</td><td colspan="2">Direccionamiento</td><td rowspan="2">Q.931 (H.225)</td><td rowspan="2">H.245</td><td>Direccionamiento</td></tr>
<tr><td>RTP</td><td>RTCP</td><td>H.225 (RAS)</td><td>DNS</td><td>DNS</td></tr>
<tr><td colspan="4">Transporte UDP</td><td colspan="3">Transporte TCP</td></tr>
<tr><td colspan="7">Protocolo de Red (IP)</td></tr>
</table>

Tabla 2. Pila de protocolos H.323

CODECS DE AUDIO

***Requeridos*:** G.711 y G.723.

Opcionales: G.728, G.729 y G.722.

SEÑALIZACIÓN

Q.931 (Señalización de llamadas): Protocolo de control de conexiones ISDN, que permite la señalización inicial de llamada y configuración de llamada.

H.225 Control de llamada (Señalización de llamadas): Señalización, registro, admisión y paquetización-sincronización del flujo de voz.

H.245 Protocolo de control (Control de llamadas): Para especificar mensajes de apertura y cierre de canales para streams de voz.

RAS (Registration, Admision and Status): Protocolo de comunicaciones que permite a una estación H.323 localizar otra estación H.323 a través del Gatekeeper.

COMPONENTES DE LAS REDES H.323

Terminales. Son puntos finales de la comunicación. Proporcionan comunicación en tiempo real bi-direccional.

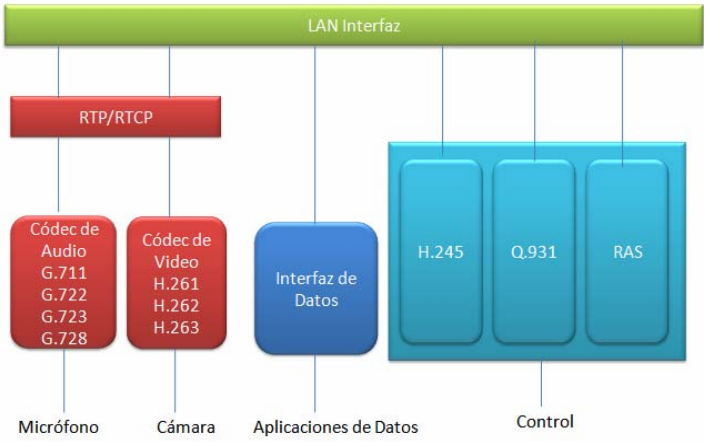

Figura 7. Componentes de las terminales.

Gateway (O Pasarela). Es un elemento opcional de una conferencia, necesario sólo si necesitamos comunicar con un terminal que está en otra red (por ejemplo RTC). Los Gateways proporcionan muchos servicios, el más común es la traducción entre formatos de transmisión (por ejemplo H.225.0 a H.221) y entre procedimientos de comunicación (por ejemplo H.245 a H.242). También traduce entre los codecs de video y audio usados en ambas redes, procesa la configuración de la llamada y limpieza de ambos lados de la comunicación. Es un tipo particular de terminal y es una entidad llamable (tiene una dirección).

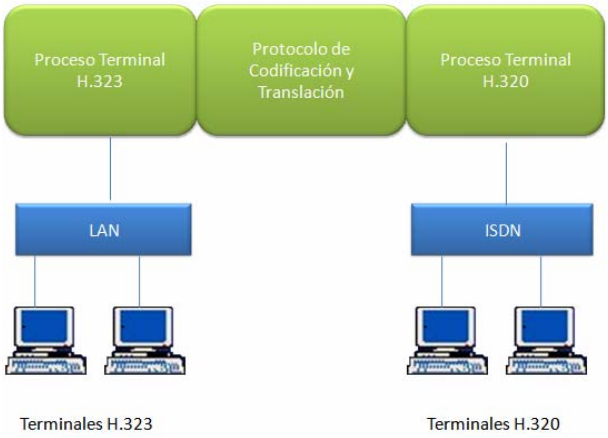

Figura 8. Gateway

Gatekeepers. Es el reemplazo de las actuales centrales telefónicas, normalmente son implementados en software. Son un elemento opcional en la comunicación entre terminales H.323. No obstante, son el elemento más importante de una red H.323. Actúan como punto central de todas las llamadas dentro de una zona y proporcionan servicios a los terminales registrados y control de las llamadas. De alguna forma, el gatekeeper H.323 actúa como un conmutador virtual.

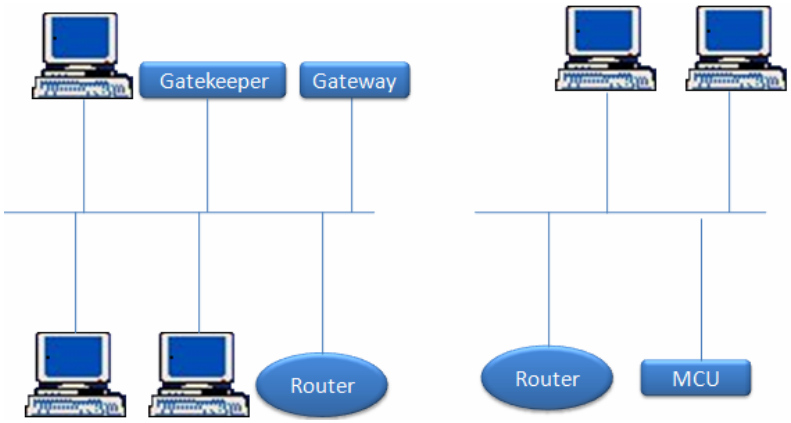

Figura 9. Zona H.323

MCU (Unidades Control Multipunto): Soporta conferencias entre tres o más extremos, se compone de: Controlador Multipunto (MC) que es obligatorio, y cero o más Procesadores Multipunto (MP). El MC gestiona las negociaciones H.245 entre todos los terminales para determinar las capacidades comunes para el procesado de audio y video; además controla los recursos de conferencias para

determinar cuáles de los flujos, si hay alguno, serán multicast[18]. El MC no trata directamente con ningún flujo de datos, audio o video. Esto se lo deja al MP, este mezcla, conmuta y procesa audio, video y/o bits de datos. Las capacidades del MC y MP pueden estar implementadas en un componente dedicado o ser parte de otros componentes H.323, en concreto puede ser parte de un Gatekeeper, un Gateway, un terminal o una MCU.

PROCEDIMIENTO DE UNA LLAMADA H.323

Setup

- El Terminal 1 (T1) se registra con gatekeeper usando el protocolo de RAS enviando un mensaje ARQ y recibiendo un mensaje ACF.
- Usando H.225 T1 envía un mensaje SETUP a T2 pidiendo una conexión. Este mensaje contiene la dirección de IP, puerto y alias del usuario llamado o la dirección de IP y puerto del usuario llamado.
- T2 envía un mensaje CALL PROCEEDING en el intento para establecer una llamada
- T2 se debe registrar en el gatekeeper como T1 previamente se registró.
- Alertar con un mensaje que indica el comienzo de la fase de generación de tono.
- El mensaje CONNECT indica el comienzo de la conexión.

Señalización de control

En esta fase de negociación usando H.245 es abierto el control de conferencia, el intercambio de los mensajes (request y answer) entre ambas terminales establece quien será el maestro y el esclavo, las capacidades de los participantes, los codecs tanto de audio y video para usarse. Cuando la negociación termina el canal de comunicación está abierto (direcciones de IP, puerto).

[18] Hace más eficiente el uso del ancho de banda de la red, pero supone una más alta carga computacional en los terminales.

La parte principal de los mensajes H.245 usado en este paso son:

- TerminalCapabilitySet (TCS). Se envía como mensaje para las capacidades soportado por las terminales que participan en una llamada.
- OpenLogicalChannel (OLC). Mensaje para abrir el canal lógico que contiene información para permitir la recepción y codificación de los datos. Contiene la información del tipo de datos que se enviará.

Audio

Las terminales empiezan la comunicación usando el protocolo de RTP/RTCP.

Liberación de Llamada

- T1 o T2 pueden iniciar el proceso de conclusión usando los mensajes (enviados usando H.245) CloseLogicalChannel y EndSessionComand.
- Empleando H.225 la conexión es cerrada con el mensaje de RELEASE COMPLETE.
- Las terminales se registran en el gatekeeper utilizando el protocolo de RAS.

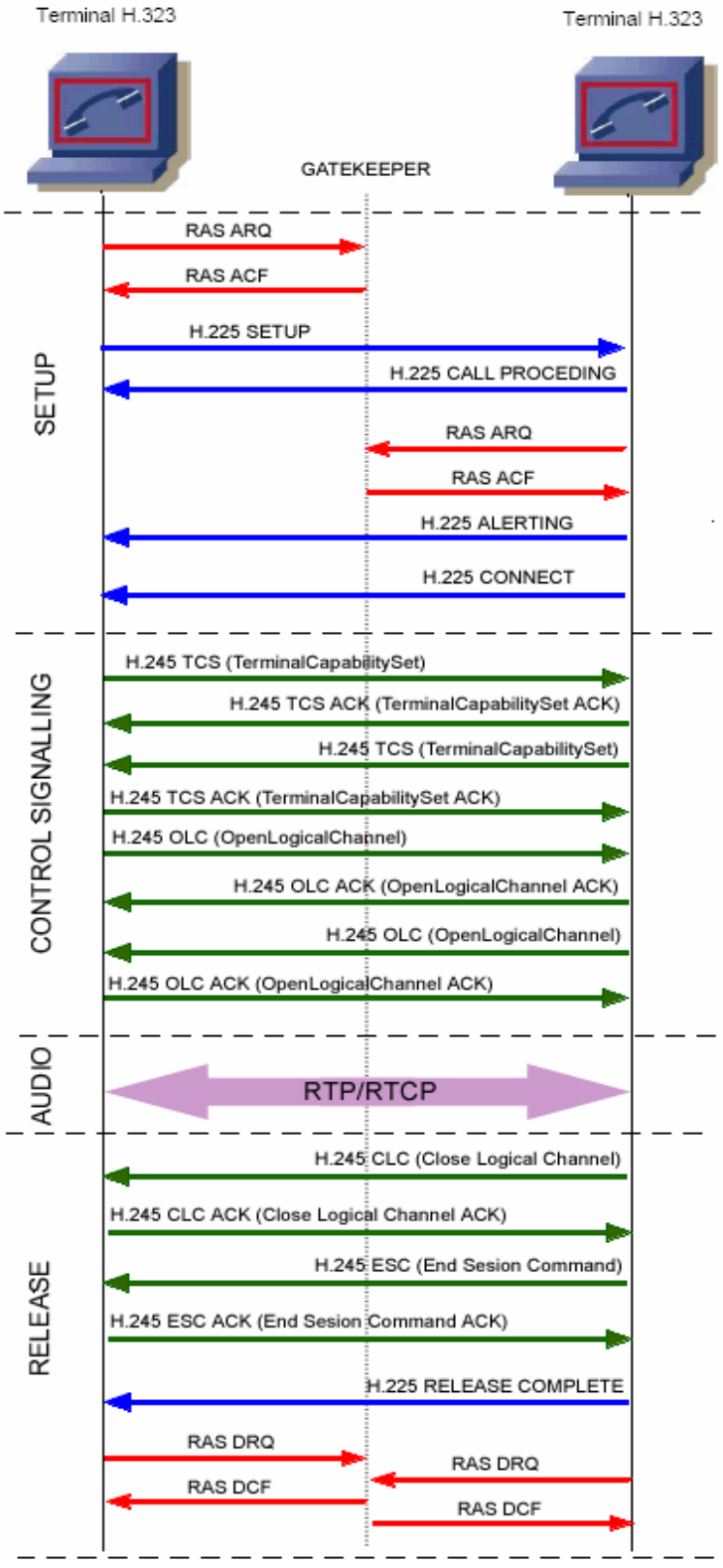

Figura 10. Etapas de una llamada H.323

2.4.2 PROTOCOLO SIP

Session Initiation Protocol[19] es un protocolo de señalización simple de la capa de aplicación, utilizado para telefonía y videoconferencia por Internet. Fue desarrollado por el IETF[20] con la intención de ser el estándar para la iniciación, modificación y finalización de sesiones interactivas de usuario donde intervienen elementos multimedia como el video, voz, mensajería instantánea, juegos online y realidad virtual.

NORMAS Y PROTOCOLOS USADOS EN SIP

XML (eXtensible Markup Language): Transmite información de eventos.

RTSP (Real Time Streaming Protocol): Para controlar el envió de streaming media.

MIME[21]: Describe el contenido en Internet. Agrega estructura a los mensajes y define reglas de codificación para mensajes no ASCII.

HTTP (Hypertext Transfer Protocol, Protocolo de Transferencia Hipertexto): Toma parte de la sintaxis y semántica, los mecanismos de autenticación, etc.

SAP[22]: SAP es usado para gestionar sesiones del tipo multicast entre un gran grupo de recipientes, asi mismo permite anunciar la sesión de multicast (en forma similar al un e-mail, newsgroup, página web). Utiliza mensajes UDP para multicast.

---

[19] Es definido completamente en la RFC 2543 ftp://ftp.isi.edu/in-notes/rfc2543.txt y en la RFC 3261.
[20] Fuerza de trabajo para la Ingeniería de Internet.
[21] **Multipurpose Internet Mail Extension** definido en RFC 2045 – 20. http://www.rfc-editor.org/rfcsearch.html
[22] **Session Advertisement Protocol**. Proyecto de Internet: http://search.ietf.org/internet-drafts/draft-ietf-mmusic-sap-v2-04.txt.

SDP[23]**:** Es un protocolo de descripción y se puede transportar sobre distintos protocolos como SIP, SAP, RTSP. Mediante SDP los extremos pueden negociar las capacidades multimedia y decidir cuales compondrán la sesión, la manera en que se establecerá, los tipos de medios multimedia correspondientes a dichos flujos (audio, video), que codecs son soportados y se van a emplear en cada uno.

<table>
<tr><td>Aplicación de Audio</td><td>Aplicación de Video</td><td colspan="3">Aplicación | Control de Llamada</td></tr>
<tr><td>G.711
G.729
G.723.1</td><td>H.261
H.263</td><td>SIP (SDP)</td><td>SAP</td><td>Direccionamiento</td></tr>
<tr><td colspan="2">RTP/RTCP</td><td colspan="2"></td><td>DNS</td></tr>
<tr><td colspan="5">TCP/UDP Convergence Layer (TUCL) – puerto 5060</td></tr>
<tr><td colspan="2">Transporte UDP</td><td colspan="3">Transporte TCP</td></tr>
<tr><td colspan="5">Protocolo de Red IP</td></tr>
</table>

Tabla 3. Pila de Protocolos SIP

DISEÑO DEL PROTOCOLO

Fue diseñado para que la telefonía se vuelva un servicio más en la Internet, es un protocolo abierto basado en estándares, ampliamente soportado y no es dependiente de un solo fabricante de equipos.

En caso de utilizar un protocolo seguro como SIPS el puerto a utilizar es el 5061. En la práctica se puede presentar el uso de puertos comprendidos entre el 5060 hasta el 5070. Además puede utilizar en su capa de transporte (Nivel 4 en el modelo OSI), TLS (Transport Layer Security. Refiriéndonos a TLS sobre TCP). TLS es utilizado para dar un cierto nivel de seguridad, encriptando la información que usualmente es vulnerable a ataques ya que se envía en texto plano.

[23] **Session Description Protocol** Definido en RFC 2327 ftp://ftp.isi.edu/in-notes/rfc2327.txt

En Internet, las conversaciones que usan señalización de tipo SIP resultan en flujo constante de paquetes de pequeño tamaño entre los comunicantes. Aunque el NAT permite conectar más fácilmente computadores a la red, lo hace al precio de no permitir una conexión puramente bi-direccional, no se pueden recibir conexiones iniciadas desde el exterior. El más común de los problemas es conocido como “audio en una sola dirección” (one-way audio). Una conversación está compuesta por dos flujos de paquetes RTP distintos. En presencia de un NAT, sólo el flujo de adentro hacia afuera no es bloqueado; el flujo de afuera hacia adentro no tiene la misma suerte y no puede atravesar el NAT. La consecuencia: el que inicia la llamada desde dentro del NAT no puede escuchar a la otra parte. Si los dos comunicantes se encuentran dentro de NATs las cosas se complican aún más, hasta el punto de que ningún flujo de audio llega a su destino final.

ARQUITECTURA SIP

La arquitectura de SIP define cuatro tipos de servidores:

Servidor Proxy. Se encarga de encaminar peticiones/respuestas hacía el destino final. El encaminamiento se realiza salto a salto de un servidor a otro hasta alcanzar el destino final. Para estos casos, existe un parámetro incluido en las peticiones/respuestas denominado *Vía* que incluye los sistemas intermedios que han participado en el proceso de encaminamiento. Esto evita bucles y permite forzar que las respuestas sigan el mismo camino que las peticiones. Esto afecta únicamente a la información de control pues el transporte de medios, salvo en el caso de requerir transcodificación intermedia, se realiza directamente entre origen y destino.

Servidor de Redirección. Realiza una función equivalente al servidor proxy, pero a diferencia de éste no progresa la llamada, sino que contesta a un INVITE con un mensaje de redirección, indicándole en el mismo como contactar con el destino.

Servidor de Registro. Mantienen la localización actual de un usuario. Se utiliza para que los terminales registren la localización en la que se encuentran. Este servidor facilita la movilidad de usuarios, al actualizar dinámicamente la misma.

Agente de Llamada (Call Agent). Existen dos tipos de Agentes:

- ***User Agent Client (UAC):*** Funciona como cliente iniciando peticiones SIP.
- ***User Agent Server (UAS):*** Funciona como servidor contactando al usuario cuando una petición SIP es recibida, y retornando una respuesta a favor del usuario.

Estos realizan las funciones de los tres servidores anteriores, además de poder realizar las siguientes acciones:

- Localizar a un usuario mediante la redirección de la llamada a una o varias localizaciones.
- Implementar servicios de redirección como reenvío si ocupado, reenvío si no contesta, etc.
- Implementar filtrado de llamada en función del origen o del instante de la llamada.
- Almacenar información de administración de llamadas.
- Realizar cualquier otra función de gestión.

MENSAJES DEL PROTOCOLO SIP

Direcciones SIP: Trabaja en una premisa simple de operación cliente-servidor. Los clientes o endpoints son identificados por direcciones únicas definidas como URL's, es decir las direcciones vienen en un formato muy similar a una dirección de correo electrónico, a fin de que las páginas Web puedan contenerlos, lo que permite hacer click en un vinculo para iniciar una llamada telefónica.

Las direcciones son identificadas mediante los denominados *URI (Uniform Resource Identifiers)*, siempre tienen el formato de user@host.

El user puede ser: nombre, número telefónico.

El host puede ser: dominio (DNS), dirección de red (IP).

Mensajes SIP: Usa mensajes para la conexión y control de llamadas. Hay dos tipos de mensajes:

Peticiones:

INVITE: Solicita el inicio de una llamada.

TRYING: Indica que el servidor Proxy está tratando de establecer la comunicación.

RINGING: Indicación de aviso de llamado.

BYE: Solicita la terminación de una llamada entre dos usuarios.

REGISTER: Informa a un servidor de registro sobre la ubicación actual del usuario.

ACK: Usado para facilitar un intercambio confiable de mensajes entre los pares. Confirmación de diferentes campos del mensaje INVITE.

CANCEL: Cancela una solicitud pendiente.

OPTIONS: Solicita información a una Host acerca de sus propias capacidades. Se utiliza antes de iniciar la llamada a fin de averiguar si ese host tiene la capacidad de transmitir VoIP, etc.

INFO: Usada para señalización de sesiones de media.

Respuestas:

1xx: Respuestas informativas, como 180, que significa teléfono sonando (ringing).

2xx: Respuestas de éxito, por ejemplo 200OK, sirve para enviar confirmaciones satisfactorias de diferentes sucesos

3xx: Respuestas de redirección.

4xx: Errores de solicitud.

5xx: Errores de servidor.

6xx: Errores globales.

ESTABLECIMIENTO DE LLAMADA

El flujo habitual del establecimiento de una sesión mediante el protocolo SIP es el siguiente, en este ejemplo todos los servidores actúan como proxy:

1. El UA A enviará una petición de INVITE con destino el UA B, con el objetivo de establecer una comunicación con él. Normalmente la petición con el método INVITE lleva un cuerpo donde viaja una descripción de la sesión que quiere establecer, esta descripción es realizada con el protocolo SDP. En ella se indica el tipo de contenido a intercambiar (voz, video, etc.) y sus características (codecs, direcciones, puertos donde se espera recibirlos, velocidades de transmisión, etc.). Esto se conoce como "oferta de sesión SDP".
2. Esta petición llega al proxy, que determina dónde se encuentra UA B y realiza dos acciones: En primer lugar envía un mensaje de 100 TRYING (intentando) que informa al agente de usuario que ésta petición ya le ha llegado al proxy y que, por tanto, no es necesario que vuelva a retransmitirla. Luego contacta con su servicio de localización y envía la petición generada en UA A hasta su destino.
3. El UA B realiza dos acciones: Devuelve al proxy una respuesta de tipo 100 para que no reenvíe la petición, si las descripciones fueran incompatibles la sesión debe terminarse (mediante una petición con el método BYE). Acepta la llamada 200OK y envía el mensaje al llamante (UA A). Este mensaje contiene también un paquete SDP describiendo las capacidades multimedia del terminal llamado.
4. El UA A reconoce la respuesta del UA B.
5. Se procede la comunicación propiamente dicha. En este punto, pueden comenzar el flujo de datos utilizando el protocolo RTP. El flujo de datos se controla mediante el protocolo RTCP.
6. El UA A decide terminar la comunicación, por lo que envía una petición de tipo BYE al UA B.
7. Éste notifica que la conexión ha terminado, y se finaliza la transacción.

Si bien se describió el caso de una sesión bi-partita, el protocolo permite el establecimiento de sesiones multi-partitas. También permite que un usuario esté registrado en diferentes ubicaciones pudiendo realizar la búsqueda en paralelo o secuencial entre todas ellas.

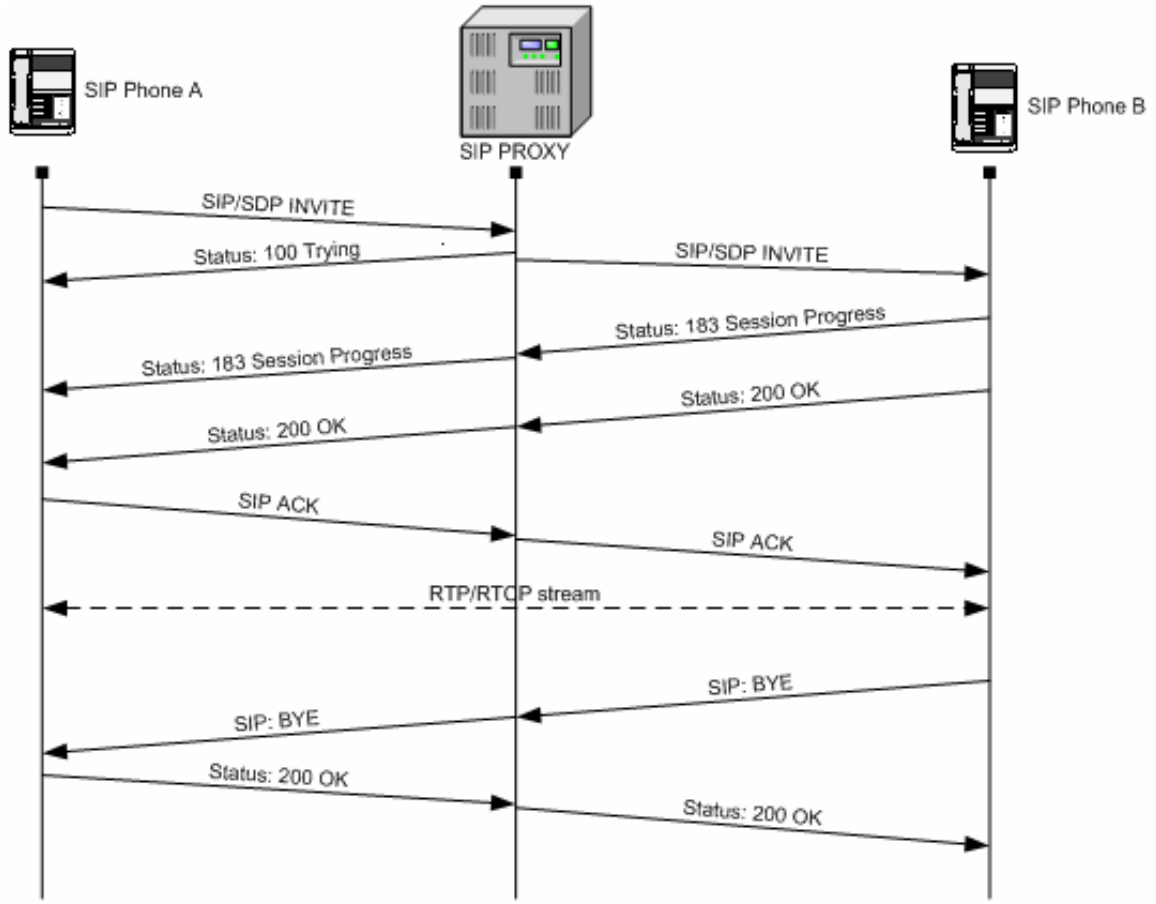

Figura 11. El Flujo de establecimiento de una sesión.

Los terminales SIP pueden establecer llamadas de voz directamente sin la intervención de elementos intermedios. Ejemplo de conexión entre *user1* con dirección IP 172.16.10.1 y user2 con dirección IP 172.16.1.2 mediante el envío de una petición *INVITE Request,* en la cual el user1 indica al user2 las capacidades de recepción de audio (codificación leym) y el puerto donde espera recibir dicho audio (port 12345). Al recibir la petición, el user2 puede inmediatamente

establecer el canal de voz y enviar la aceptación de conexión mediante el envío de *OK Response,* en la cual incluye la información complementaria para el establecimiento del canal opuesto (codificación GSM, puerto 54321 en nuestro ejemplo). Tras el intercambio de señal de audio, cualquiera de los participantes puede finalizar la llamada mediante el envío de mensaje *BYE Request* que debe ser asentido mediante un mensaje de confirmación *(OK).*

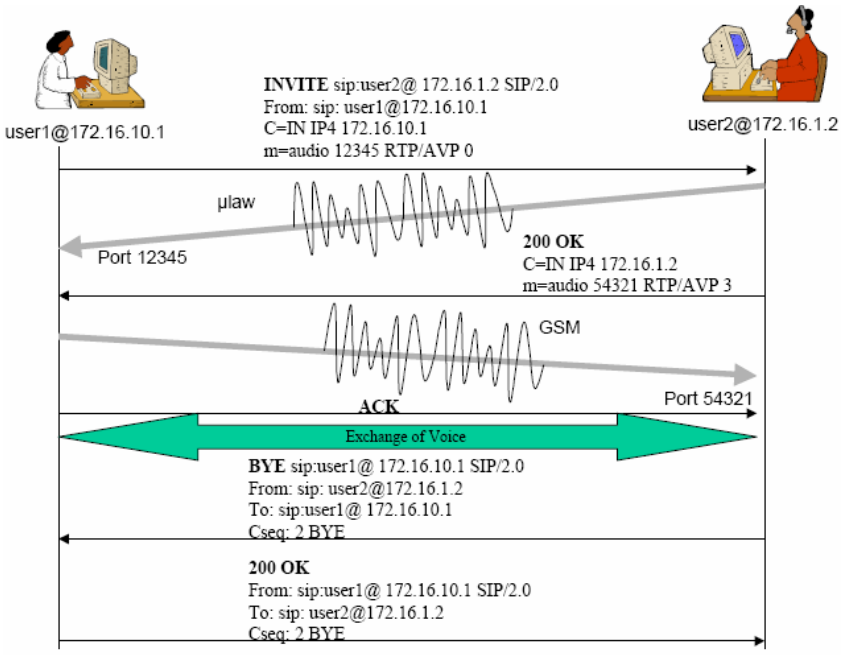

Figura 12. Establecimiento de llamada SIP

2.4.3 PROTOCOLO MGCP

MGCP es un protocolo que se utiliza para controlar gateways telefónicos desde elementos de control de llamada externos denominados agentes de llamada (call agents).

Una gateway telefónico es un elemento de red que provee una conversión entre señales de audio provenientes de circuitos telefónicos a paquetes de datos sobre Internet o cualquier red de paquetes y viceversa.

En soluciones de voz MGCP se separa la inteligencia de control de llamada (fuera del gateway) del manejo multimedia, funcionando como un protocolo interno entre componentes separados. MGCP entiende que los agentes de control, o agentes de llamada, se sincronizan entre ellos para enviar comandos de forma coherente a los gateways bajo su control. De este modo el modelo de comunicación empleado en un intercambio es de tipo maestro / esclavo, de forma que los gateways esperan a ejecutar los comandos que les envíen los call agents.

Especificando algo más, MGCP es un protocolo usado por elementos de control de llamada externos llamados Media Gateway Controllers (MGCs) para controlar Media Gateways (MGs). Así, diferentes gateways se agrupan lógicamente como si fuesen uno solo.

Ejemplos de este tipo de pasarelas incluyen:

- Gateways troncales que interconecta la RTB con redes VoIP.
- Gateways residenciales o de acceso que proveen interfaces analógicos tradiciones (RJ11) o digitales a redes VoIP.
- Servidores IVR (Interactive Voice Response) que proveen de servicios interactivos de voz a redes VoIP.

2.4.4 PROTOCOLO MEGACO

MEGACO[24] / H.248 es muy parecido a MGCP desde el punto de vista de la arquitectura y de la relación entre el elemento de control (de hecho se basa en él y en MDCP – Media Device Control Protocol) y el gateway, si bien soporta un rango mayor de redes, tales como ATM. Así el MG (Media Gateway) transforma cadenas provenientes de una red analógica a paquetes o celdas dentro de un protocolo como RTP (Real-Time Transport Protocol).

[24] Definido en RFC 3525

El objetivo inicial de MEGACO fue la utilización de redes de paquetes para la transmisión de tráfico de voz originado por redes tradicionales. Los operadores tradicionales fueron uno de los que mayor interés han mostrado en esta propuesta, pensando en integrar progresivamente sus redes de telefonía basadas en conmutación de circuitos y sus redes de datos basadas en conmutación de paquetes en una red homogénea que transportará ambos tipos de tráfico (voz y datos) y que fuera transparente a los usuarios finales.

MEGACO resuelve este problema dividiendo las pasarelas en tres entidades diferentes:

- **Controlador de Medios (Media Gateway Controller –MGC):** Proporciona la señalización.
- **Pasarela de Medios (Media Gateway –MG):** Proporciona la adaptación de medios y/o las funciones de transcodificación. Este bloque realiza las funciones de traslación de direcciones, cancelación de eco, envío/recepción de dígitos DMTF, etc.
- **Pasarela de Señalización (SG):** Proporciona funciones de mediación de señalización entre redes IP.

En un escenario habitual los tres elementos están físicamente separados de modo que pueden proporcionar ventajas como la concentración de muchos MG (conectados a usuarios finales) en algunos MGC controlados por un SG. La siguiente figura muestra la arquitectura de MEGACO.

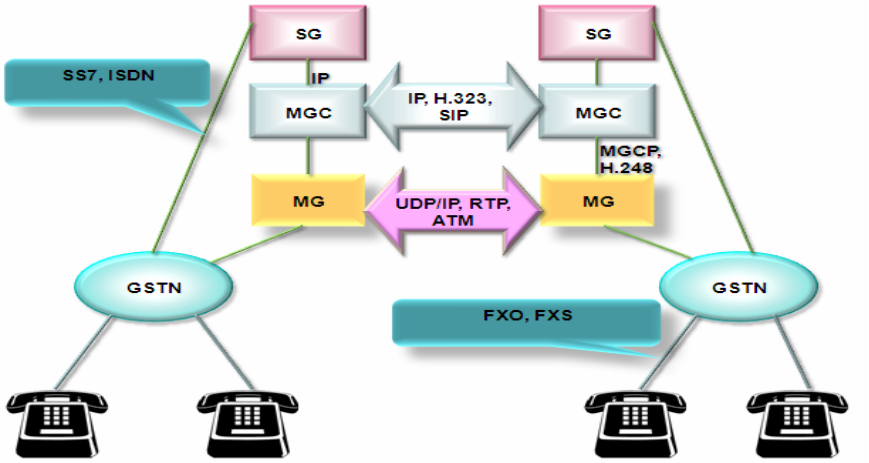

Figura 13. Arquitectura MEGACO.

2.4.5 IAX

El protocolo IAX (Inter-Asterisk eXchange protocolo) es una alternativa de SIP, fue desarrollado por Mark Spencer para comunicación entre servidores Asterisk[25], ya sea de modo cliente-servidor o entre servidores. IAX, ahora, más comúnmente se conoce como IAX2, la versión original ha quedado obsoleta.

Los principales objetivos de este protocolo son minimizar el ancho de banda que ocupa el tráfico de control y proveer transparencia a la utilización de técnicas NAT en la red.

Para conseguir estos objetivos utiliza un mecanismo denominado multiplexión o trunking que le permite el envió de la información de señalización y los datos de todas sus llamadas a través de un mismo canal utilizando un solo puerto UDP (4569[26]). Esta transmisión in-band hace que sea un protocolo casi transparente a los cortafuegos y realmente eficaz para trabajar dentro de redes internas (NAT).

[25] Ver el capitulo 3 que trata de ASTERISK página 53.
[26] El IAX1 usaba el puerto 5036.

En cuanto al formato de mensajes, es un protocolo binario, diseño que se realiza para mejorar la eficiencia en el uso de ancho de banda, especialmente en llamadas individuales.

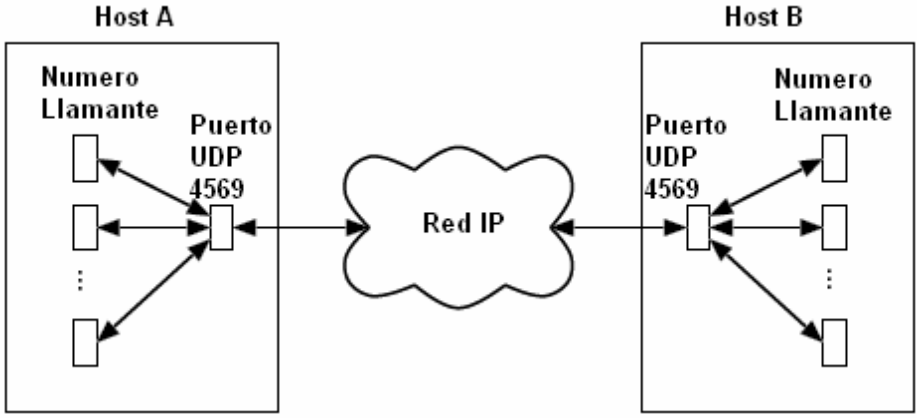

Figura 13. Múltiples llamadas multiplexadas en un único puerto UDP.

Permite manejar una gran cantidad de codecs y un gran número de streams, lo que significa que puede ser utilizado para transportar virtualmente cualquier tipo de datos. Esta capacidad lo hace muy útil para realizar videoconferencias o realizar prestaciones remotas. Además presenta funcionalidades interesantes como la posibilidad de enviar o recibir planes de marcado (dialplans).

Con respecto a la implantación de este protocolo en una instalación real cabe destacar que existen múltiples terminales de usuario software que son compatibles con el mismo, algunos de ellos de libre distribución. Sin embargo, el hecho de que no sea aún un estándar extendido, sino un protocolo propietario de Asterisk, lo hace incompatible con productos de otros fabricantes, principalmente gateways, lo que dificulta su aplicación en infraestructuras donde se desea tener interconexión con la red telefónica básica.

PROCESOS PARA UNA LLAMADA IAX

Registro para establecer una comunicación un terminal debe ser accesible (reachable) por otro terminal, para ello este debe conocer la dirección de red del otro. Esto se puede realizar manualmente, con un directorio compartido (ENUM[27]) o mediante el registro en el servidor IAX2.

Proporciona un mecanismo para que un terminal registre su dirección y credenciales con otro terminal, que es el registrante, el servidor de registro. El proceso de registro de un terminal, es muy sencillo:

1. Se pide registro al servidor. (REGREQ).
2. Este pide que se autentifique. (REGAUTH).
3. Se proporciona la autentificación. (REGREQ).
4. Se confirma el registro (REGACK).
5. Se confirma que se ha recibido (ACK).

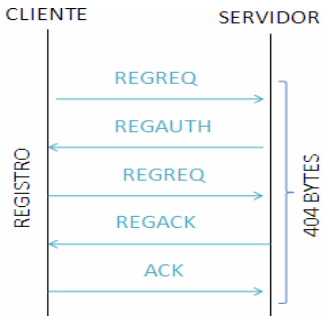

Figura 14. Registro de un peer

Establecimiento de la llamada

El Terminal A (TA) inicia una conexión y manda un mensaje NEW. El Terminal B (TB) responde con un ACCEPT y el llamante le responde con un ACK. A continuación el TB da las señales de RINGING (también se pueden producir los siguientes mensajes de control: ANSWER, BUSY, o PROCEEDING.) y el llamante

[27] ENUM, Proviene de tElephone NUmber Mapping. Se asocian los números telefónicos convencionales (E.164) a nombre DNS en *.e164.arpa*. El servidor DNS que alberga la entrada, tiene registrados los servicios (sip, mail, http) publicados para dicho número.

contesta con un ACK para confirmar la recepción del mensaje. Por último, el llamado acepta la llamada con un ANSWER el llamante confirma ese mensaje.

Flujo de datos o flujo de audio

Los mensajes de este protocolo se denominan tramas. Cada flujo de información de los que se generan se compone principalmente de Mini Tramas IAX (Mini Frames), que contienen una cabecera simple de 4 bytes y que permiten enviar la información con el mínimo overhead; suplementándose por Tramas Completas (Full Frames) de forma periódica, que incluyen la información de sincronización y la confiabilidad. Se mandan los frames M y F en ambos sentidos con la información vocal.

Figura 15. Ejemplo de un flujo IAX Media en un solo sentido

IAX2 soporta los siguientes tipos de mensajes, optimizados para cada uno de los tipos de comunicaciones que puede establecer.

- *DTMF Media Message:* Datos del protocolo DTMF.
- *Voice Media Message:* Transporte de Voz.
- *Video Media Message:* Transporte de Vídeo.
- *Text Media Message:* Transporta Texto.
- *Image Media Message:* Transporta Imágenes.
- *HTML Media Message:* Transporta HTML.
- *Comfort Noise Media Message:* Transporta información de calidad del enlace.

Liberación de la llamada o desconexión

La liberación de la conexión es tan sencilla como enviar un mensaje de HANGUP y confirmar dicho mensaje.

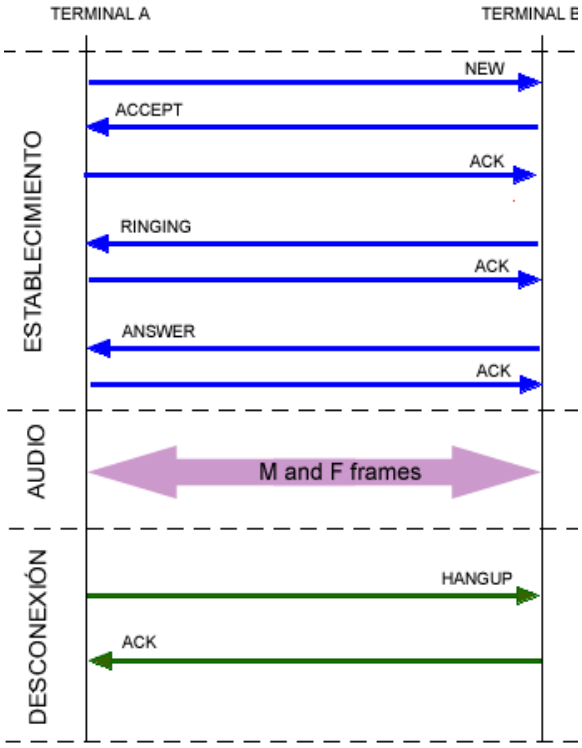

Figura 16. Ejemplo del flujo de datos de una comunicación IAX2.

2.4.6 SKINNY/SCCP

Acrónimo de Skinny Client Control Protocol. Es un sistema de telefonía móvil conectado a una planta de cableado común. Originalmente desarrollado por Selsius Corporation y ahora bajo el control y diseño de Cisco Systems, Inc. Uno de los clientes más famosos de Skinny es la serie Cisco 7900 de teléfonos IP. Es el protocolo de control por defecto para terminales con el servidor Cisco Call Manager[28] PBX, que es el similar a Asterisk PBX. El cliente Skinny usa TCP/IP para transmitir y recibir llamadas. Para el audio utiliza RTP, UDP e IP. Los mensajes Skinny son transmitidos sobre TCP y usa el puerto 2000. Skinny es un protocolo para terminales.

[28] Call Manager: Es el nombre con el que se comercializa la solución de VoIP de Cisco.

2.5 TIPOS DE ARQUITECTURAS

En el pasado, todas las redes de voz fueron construidas usando una arquitectura centralizada, en la cual los Endpoints (teléfonos) fueron controlados por los conmutadores centralizados. Sin embargo este modelo trabajó bien para los servicios de telefonía básica.

Uno de los beneficios de la tecnología VoIP, es que permite a las redes ser construidas usando una arquitectura centralizada o bien distribuida. Esta flexibilidad permite a las compañías construir redes caracterizadas por una administración simplificada e innovación de Endpoints (teléfonos), dependiendo del protocolo usado.

2.5.1 ARQUITECTURA CENTRALIZADA

En general, la arquitectura centralizada está asociada con los protocolos MGCP y MEGACO. Estos protocolos fueron diseñados para un dispositivo centralizado llamado Controlador Media Gateway o Call Agent, que maneja la lógica de conmutación y control de llamadas. El dispositivo centralizado comunica al Media Gateways, el cual enruta y transmite la porción audio/media de las llamadas (la información de voz actual).

En la arquitectura centralizada, la inteligencia de la red es centralizada y los dispositivos finales de usuario (endpoints) son relativamente mudos (con características limitadas). Sin embargo, muchas arquitecturas VoIP centralizadas usan protocolos MGCP o MEGACO.

Los defensores de la arquitectura VoIP centralizada, apoyan este modelo porque centraliza la administración, el aprovisionamiento y el control de llamadas. Simplifica el flujo de llamadas repitiendo las características de voz. Es fácil para los ingenieros de voz entenderlo. Los críticos de la arquitectura VoIP centralizada

demandan que se suprimen las innovaciones de las características de los teléfonos (endpoints) y que llegara a ser un problema cuando se construyan servicios VoIP que muevan mas allá de características de voz.

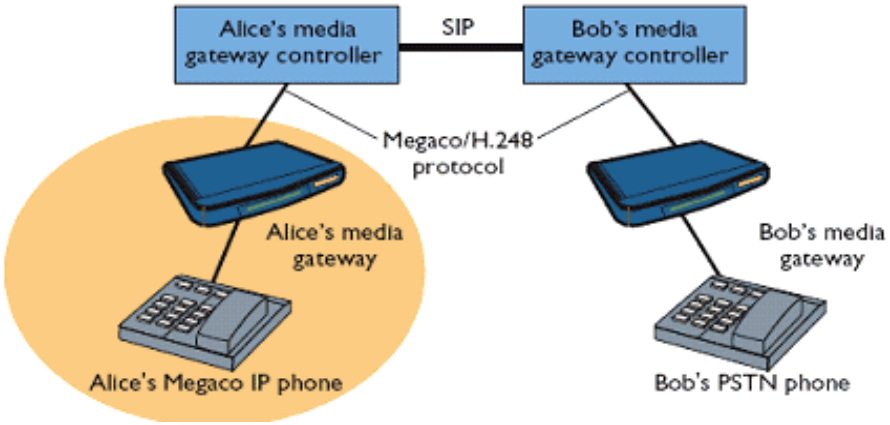

Figura 17. Arquitectura centralizada VoIP con protocolo MEGACO.

2.5.2 ARQUITECTURA DISTRIBUIDA

La arquitectura distribuida está asociada con los protocolos H.323 y SIP. Estos protocolos permiten que la inteligencia de la red sea distribuida entre dispositivos de control de llamadas y endpoints. La inteligencia en esta instancia se refiere a establecer las llamadas, características de llamadas, enrutamiento de llamadas, aprovisionamiento, facturación o cualquier otro aspecto de manejo de llamadas. Los Endpoints pueden ser Gateways VoIP, teléfonos IP, servidores media o cualquier dispositivo que pueda iniciar y terminar una llamada VoIP. Los dispositivos de control de llamadas son llamados Gatekeepers en una red H.323, y servidores Proxy o servidores Redirect en una red SIP.

Los defensores de la arquitectura VoIP distribuida apoyan este modelo por su flexibilidad. Permite que las aplicaciones VoIP sean tratadas como cualquier otra aplicación IP distribuida, y permite la flexibilidad para añadir inteligencia a cualquier dispositivo de control de llamadas o Endpoints, dependiendo de los requerimientos tecnológicos y comerciales de la red VoIP. La arquitectura

distribuida son usualmente bien entendida por los ingenieros que manejan redes de datos IP. Los críticos de la arquitectura distribuida comúnmente apuntan a la existencia de la Infraestructura PSTN como el único modelo de referencia que debiera ser usado cuando intentamos repetir los servicios de voz. Ellos también notan que las redes distribuidas tienden a ser más complejas.

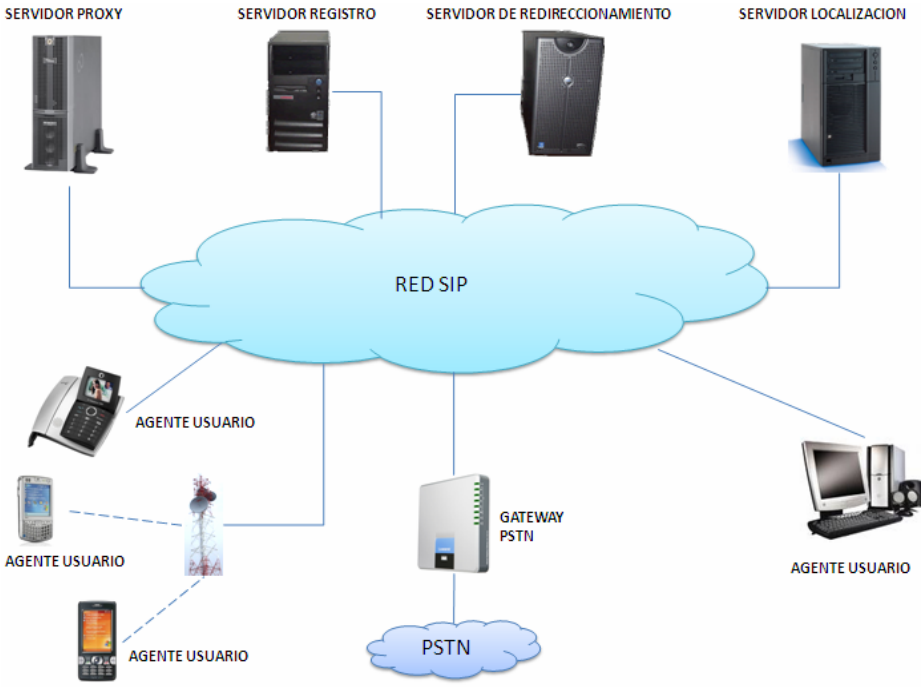

Figura 18. Arquitectura Distribuida en SIP

2.6 VENTAJAS DE LA UTILIZACIÓN DE VOIP

- La red IP permite un manejo inteligente del ancho de banda contra el ancho de banda fijo que posee cada conexión.
- La red IP, tiende a una estructura modular lo cual permite mejoras de bajo costo, contra la inercia económica que presenta la PSTN.
- El direccionamiento de la red IP es controlado y cuenta en general con varias rutas alternas al mismo destino, al contrario de la PSTN.
- El direccionamiento sobre IP, es un direccionamiento dinámico. La PSTN es fija absolutamente.

- El punto anterior nos lleva al hecho de que las regulaciones presentes en la PSTN, limitan mucho el progreso de la red, mientras que la red IP ha gozado de completa libertad, traduciéndose en una constante innovación tecnológica.
- Una red integrada permite un mejor sistema tarifario, el cual se mantiene en uno de los servidores. El método de facturación se facilita tanto como para el proveedor como para el usuario, quien puede monitorear sus propias llamadas.
- El servicio a proveer pueden ser paquetes de pago mensual, o prepagados, con la posibilidad de conexión de banda ancha y llamadas ilimitadas en todo el mundo.
- A pesar que la PSTN ya ofrece un QoS de alto nivel, los nuevos estándares de VoIP proveen la posibilidad de alcanzar los mismos niveles, y la posibilidad de mejorarlos, respecto a rutas alternas, mejores técnicas de compresión, etc.
- Permite el control del tráfico de la red, lo que mejora su rendimiento.
- Por último, esta tecnología ofrece la posibilidad de una transición suave, ya que puede integrar la PSTN a su sistema. Así que la implementación de ésta no excluye el uso de la telefonía tradicional.

Sin embargo dos aspectos de gran importancia deben de aclarar fuera de la sección anterior:

- La PSTN es independiente del sistema de alimentación eléctrica, y utiliza alimentación eléctrica en línea a las terminales, lo cual es de gran importancia en cuanto a la confiabilidad.
- El otro aspecto de gran delicadeza se refiere a la seguridad que ofrece la PSTN, la cual al ser exclusiva para la voz responde a estándares exigentes, sin embargo la VoIP ofrece una seguridad virtual, que vence no sólo intrusos por hardware, sino también por software.

CAPÍTULO. 3 ASTERISK

3.1 INTRODUCCION

Asterisk[29] el más poderoso, flexible y extenso software de telecomunicaciones disponible. En la actualidad es una solución probada y robusta, tanto para empresas que lo utilizan como base de usuario, para proveedores o carriers.

Su nombre viene del símbolo asterisco "*", que en ambientes UNIX y DOS representa un wildcard o símbolo comodín. Comenzó en 1999, desarrollado por Mark Spencer (creador de Digium, empresa fundada para tal fin) basado en las ideas y el trabajo previo de Jim Dixon (proyecto de telefonía Zapata). El programa, sus mejoras y correcciones, son el resultado del trabajo colectivo de la comunidad del software libre.

Aunque puede funcionar en muchos sistemas operativos, GNU/Linux[30] es la plataforma más estable y en la que existe un mayor soporte. Cuenta con un doble licenciamiento, GNU/GPL y licencia propietaria. Esta última es con el objeto de poder incluir soporte para el protocolo G.729, aunque el codificador correspondiente funciona indistintamente con una u otra versión.

Es una implementación de código abierto para central telefónica (PBX). La PBX Asterisk está diseñada para conectar cualquier hardware telefónico o cualquier tipo de software de telefonía de manera transparente y consistente. Permite que los teléfonos conectados a la centralita puedan hacer llamadas entre ellos y servir de pasarela a la red telefónica tradicional. Convergen aplicaciones de voz, datos y video. Es un soft-switch (un PBX-IP), realiza las funciones tradicionales de una PBX. A demás incluye todas las funcionalidades de las más costosas alternativas de código cerrado, como son correo de voz, llamada en conferencia, respuesta interactiva de voz y distribución automática de llamadas.

[29] Liberado con la licencia GPL2 y licencia comercial.
[30] Puede trabajar en base BSD o MacOSX también.

Es perfectamente posible construir de este modo aplicaciones de telefonía avanzadas, integración de telefonía con redes de datos, recepción y enrutado de faxes, colas de agentes ACD, y en suma todas aquellas funcionalidades que pueden esperarse de una solución de telefonía moderna. Asterisk le facilita su integración y despliegue.

3.2 ARQUITECTURA DE ASTERISK

Asterisk está diseñado para conseguir la máxima flexibilidad, la cual radica en su base, ya que contiene varios motores que juegan un rol crítico en la operación del software. Cuando Asterisk arranca el “Cargador Dinámico de Módulos” carga e inicializa los drivers que proveen los canales, formatos de archivos, codecs y aplicaciones enlazándolas con las apropiadas APIs internas. A partir de entonces la base del “conmutador PBX” acepta llamadas desde las interfaces direccionándolas según el plan de numeración, usando el “lanzador de aplicaciones” para llamar a los terminales, conectar al buzón de voz y otros.

La base también provee un “Planificador y Administrador de Entrada/Salida” de los que las aplicaciones y drivers pueden aprovecharse. El “Traductor de Codecs” permite a canales comprimidos con diferentes codecs hablar entre ellos.

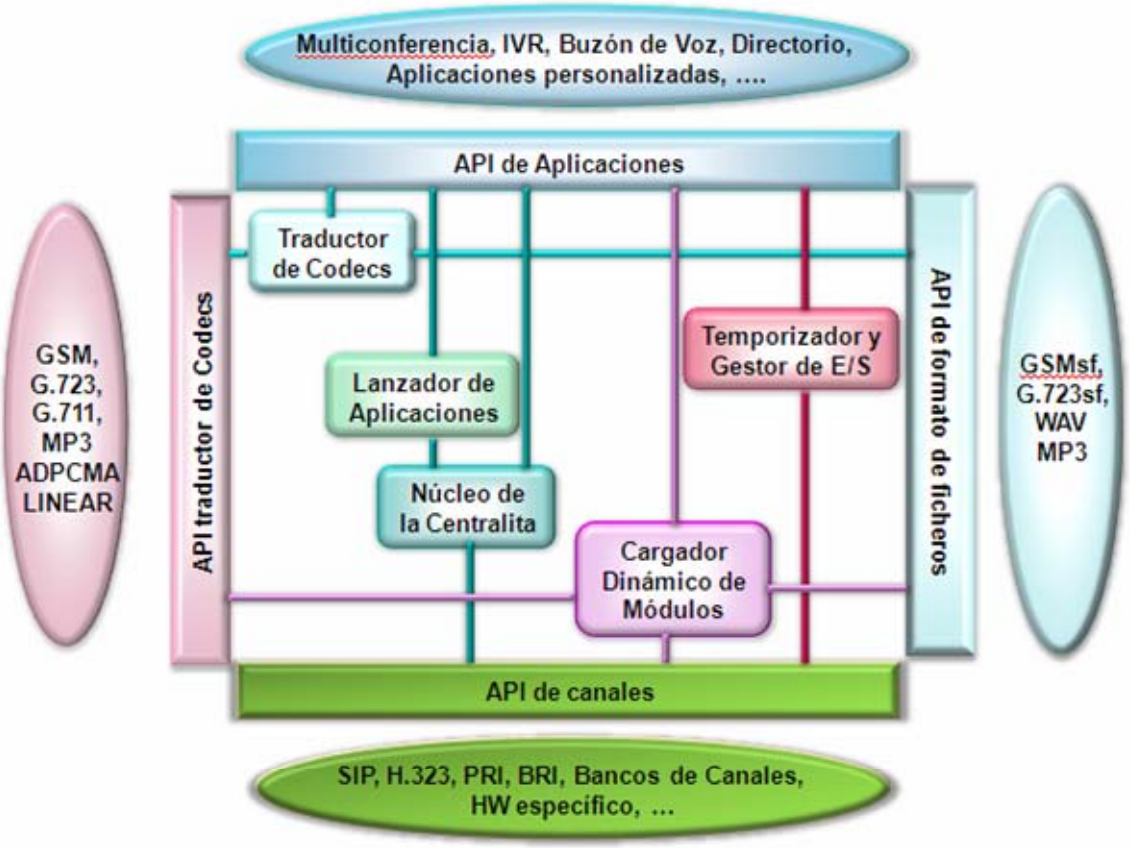

Figura 19. Arquitectura de Asterisk.

Hay 4 APIs definidas en módulos cargables permitiendo alcanzar una abstracción completa entre sus funciones de la base como sistema de servidor PBX y las tecnologías variadas que existen en telefonía. La estructura modular es lo que permite integrar el hardware de conmutación y las tecnologías de VoIP.

API de canales: Sirve para controlar todas las llamadas del sistema, sean VoIP, analógicas o cualquier otra tecnología pudiendo desarrollar nuevos canales.

API de Aplicaciones: Permite ejecutar varios módulos de tareas como buzón de voz, IVR, Multi Conferencia. Pudiendo desarrollar todas aquellas aplicaciones más mediante AGI (Asterisk Gateway Interface) en C, C++, perl, php, etc.

API de traducción de codecs: Carga los módulos de codec para soportar varios formatos de codificación de audio y permitir la comunicación entre los diferentes canales.

API de Formato de Ficheros: Maneja la lectura y la escritura de los diferentes formatos de archivo para el almacenamiento de datos en el sistema de archivos.

Las extensiones (físicas-teléfonos IP o lógicas-softphone) creadas pueden soportar distintos protocolos SIP, IAX, MGCP y H.323, su ventaja es la movilidad permitida al depender de la Red IP; estos usuarios pueden estar localizados en cualquier sitio siempre y cuando tengan acceso directo al servidor Asterisk.

Prácticamente se soportan todos los codificadores de audio. Y la conversión entre los mismos. Otra característica importante a tener en cuenta es la interconexión entre distintos Asterisk mediante los protocolos SIP e IAX, así como la interconexión con otros Sistemas de Voz IP (Operadores IP) mediante los protocolos SIP, IAX y H.323.

3.3 ARCHIVOS DE CONFIGURACIÓN DESCRIPCIÓN

Los archivos de configuración se encuentran ubicado en la carpeta **/etc/Asterisk** los más usados para nuestras prácticas se describen a continuación:

3.3.1 extensions.conf

El archivo extensions.conf es la parte central de toda la configuración, es donde se define el **Dial Plan** (plan de marcado) de Asterisk.

La creación del Dial Plan indica las acciones a tomar tanto para llamadas entrantes como para llamadas salientes. Define el comportamiento lógico de la PBX.

Figura 20. Arquitectura de Dial Plan

3.3.2 iax.conf y sip.conf

En primer lugar existe la sección **[general],** donde se definen variables globales y aspectos por defecto para todos los canales SIP e IAX y la sección de **canales** donde se define los tipo de canales que podrían ser: peers (es una llamada saliente), user (es una conexión que se autentifica con nuestra PBX -una llamada entrante) o friend (conexión saliente o entrante).

Cuando nos llega una conexión entrante del tipo “user” o “friend” tenemos que decidir qué hacer con la conexión. El término “contexto” se usa para definir qué reglas o grupo de reglas del plan de marcado (extensions.conf) se deben aplicar a esa llamada concreta. El “contexto” de una llamada entrante se encarga de asociarla con un conjunto de reglas presentes en el plan de marcado. El “contexto” representa el punto de entrada de la llamada en el plan de marcación.

3.3.3 Voicemail.conf

El archivo voicemail.conf es donde se configura todo lo relacionado con el contestador. Si recibimos una llamada y no contestamos o la línea está ocupada, entrará en función el contestador, grabará el eventual correo de voz dejado por quien llama y nos enviará un correo electrónico para avisarnos. Además podemos anexar el correo de voz al correo electrónico en el formato de audio que más nos guste. Este archivo se encuentra normalmente en la carpeta /etc/Asterisk (distribución CentOS).

3.4 INTERPRETE DE COMANDOS (CLI) DE ASTERISK

El intérprete de comandos de Asterisk es bastante potente, y permite controlar y monitorear nuestra PBX, como función adicional soporta el empleo de la tecla <tabulador> para ver un listado de todos los comandos disponibles, completar un comando o argumento, o ver posibles argumentos además se pueden ejecutar comandos sin estar dentro del intérprete:

```
/usr/sbin/Asterisk –rx "restart now"
```

Asterisk tiene dos componentes internos: un servidor y un cliente (CLI). Tanto el servidor como el cliente se invocan usando la orden *"Asterisk"* pero utilizando distintos argumentos.

3.4.1 Algunas órdenes básicas de Asterisk:

Arrancar/Parar *Asterisk* desde el arranque (run level)

```
#/etc/init.d/Asterisk (start|stop)
```

Arrancar *Asterisk* desde la línea de órdenes

```
# Asterisk
```

Arrancar el servidor en modo de depuración

```
# Asterisk -vvvc
```

Arranque en modo de depuración o “verbose” (-vvv) y abra un cliente en modo consola (-c). Con un cliente en modo consola (CLI) puede supervisar lo que está pasando en el servidor. Si el servidor está funcionando en segundo plano se puede conectar usando el cliente con el argumento (-r).

```
# Asterisk -r.
```

Activa el modo de depuración para SIP o IAX2

```
#CLI> IAX2 set debug
#CLI> SIP set debug
```

Desactiva el modo de depuración para SIP o IAX2

```
#CLI> IAX2 set debug off
#CLI> SIP set debug off
```

Muestra el estado de “usuarios”, “peers” y “canales” para SIP o IAX2

```
#CLI> sip show users
#CLI> sip show peers
#CLI> sip show channels
#CLI> iax2 show peers
#CLI> iax2 show users
#CLI> iax2 show channels
```

3.5 PRÁCTICAS PROPUESTAS

3.5.1 Práctica 1: Configuración de canales tipo SIP.

Objetivos

- Defina y configure los canales SIP.
- Defina reglas para sus extensiones (crear el dialplan).
- Configuración softphone.
- Analizar los mensajes que se producen al realizar una llamada.

Materiales usados

- 2 Pcs
- Vmware server
- Sistema operativo CentOS
- Asterisk
- 2 Softphone: X-lite
- Dos auriculares

Introducción Teórica

Configuración de sip.conf

Asterisk puede configurarse desde el Command Line Interface (CLI) o desde los ficheros de configuración (.conf) que habitualmente se encuentran en el directorio /etc/Asterisk. La configuración se carga al iniciar Asterisk, por lo que para aplicar cualquier cambio será necesario recargarla.

```
#CLI> reload
```

El fichero sip.conf contiene parámetros relacionados con la configuración SIP de Asterisk. En este fichero se definen variables generales, clientes y servidores SIP;

se estructura en secciones donde cada sección se define por un nombre entre corchetes seguido de las opciones de cada sección.

La primera sección, denominada general, define las opciones generales del servidor como la dirección IP y el puerto al que hacer el bind. Las siguientes secciones definen parámetros del cliente como el username, password u otras.

Palabras clave

Las siguientes palabras se definen en /etc/Asterisk/sip.conf. En la sección general:

Allowguest: yes|no. Permitir o rechazar las llamadas de invitados (guest) (por defecto está en yes).

Allowsubscribe: Esto permite a las extensiones controlar el estado de las demás.

bindport: El puerto que debe escuchar Asterisk para las conexiones SIP. Por defecto es 5060, de conformidad con las normas. Toma como argumento un número de puerto (que no debe estar en uso por cualquier otro servicio).

bindaddr: La dirección IP que Asterisk debe escuchar para las conexiones SIP. Si la máquina tiene múltiples direcciones IP o alias, se utiliza esta opción para seleccionar cual IP está escuchando Asterisk. El comportamiento por defecto es escuchar a todas las interfaces y alias disponibles. Toma como argumento una dirección IP (que debe ser un interfaz disponible en el sistema).

srvlookup: yes|no. Permitir las operaciones de búsqueda DNS SRV en las llamadas. Por defecto no.

context: El contexto de base que todas las extensiones utilizaran si no viene especificado diversamente. Este contexto se define en extensions.conf.

videosupport: yes|no. Habilitar el soporte para vídeo SIP. Por defecto no.

disallow:	Rechaza todos los codec (configuración global). Por defecto <all>.
allow:	Permite los codec en orden de preferencia (use disallow=all primero, antes de permitir otros codecs).
language:	Si hemos instalados mas de un idioma para las voces de la centralita aquí se puede definir cual idioma usará la extensión. Toma como valor es/en.

```
register=>usuario[:password[:authuser]]@hostservidor[:puerto]
[/extension]
```

usuario	Es el user-id para el servidor SIP.
authuser	Es el usuario de autorización opcional para el servidor SIP.
password	Es la contraseña del usuario.
hostservidor	Es el nombre del dominio o del servidor SIP. Este servidor SIP necesita tener una definición en una sección propia en el sip.conf.
puerto	Envía la petición register a este puerto en el Servidor. Por defecto al 5060.
/extension	Es la extensión de contacto del Asterisk, esta es utilizada por el servidor remoto del SIP cuando necesita enviar una llamada al Asterisk.

```
register => USUARIO_1:123@miAsterisk.com
```

Si el servidor usa un puerto que no es el 5060 tenemos que especificarlo al final de la línea de este modo:

```
register  =>  USUARIO_1:123@miAsterisk.com:5061
```

Opciones Cliente

type: La opción tipo establece el tipo de conexión para el cliente. Pueden ser:

peer: *Un dispositivo que recibe llamadas desde el servidor Asterisk.*

user: *Un dispositivo que hace llamadas a través del servidor Asterisk.*

Friend: *Un dispositivo que puede recibir y enviar llamadas a través del servidor Asterisk.*

secret: Establece la contraseña para el cliente. Toma una cadena alfanumérica.

host: Fija la dirección IP o el host name del dispositivo. Si se establece a 'dynamic' se espera que el host se conecte desde cualquier dirección IP. Esta es la opción más común, y normalmente necesario dentro de una red con DHCP.

canreinvite: Típicamente 'no' si se encuentra detrás de un NAT. De este modo se habilita que el tráfico RTP (voz) pase por el sistema Asterisk. En caso de activar este parámetro podría ocurrir que el RTP se estableciera entre los extremos sin pasar por el sistema Asterisk y podríamos tener problemas.

qualify: yes|no|milliseconds: Comprobar si el cliente esta accesible. Si está en “yes”, los chequeos ocurren cada 2000 milisegundos (2 segundos). Por defecto “no”.

port: El puerto que usará la extensión para conectarse al servidor, por defecto 5060.

context: El contexto que usará la extensión.

callerid: El nombre y número de identificación de la extensión. Ejemplo: callerid= Ada <2000>.

Configuración de extensions.conf

El archivo de configuración “extensions.conf” contiene el dialplan del Asterisk, el plan maestro de control o de flujo de ejecución para todas las operaciones.

Controla cómo se manejan y se encaminan las llamadas entrantes y salientes. Aquí es donde se configura el comportamiento de todas las conexiones con su PBX.

El contenido del “extensions.conf” se organiza en secciones, que pueden ser tanto para configuraciones estáticas y definiciones, o para los componentes ejecutables del dialplan a los cuales nos referimos con el nombre de contextos. Las secciones de configuraciones son ***general***, ***globals*** y los nombres de los contextos son definidos enteramente por el administrador de sistema. Un tipo especial de contextos son las macros, identificados por un nombre definido por el usuario el cual tiene como prefijo ***macro-***. Éstos son patrones reutilizables de ejecución, como los procedimientos en un lenguaje de programación. Cada sección del extensions.conf comienza con el nombre de la secciones contenida dentro de corchetes.

Contexto [general]

static: Indica si se ha de hacer caso a un comando "save dialplan" desde la consola. Por defecto es "yes". Funciona en conjunto con "writeprotect"

writeprotect: Si writeprotect=no y static=yes se permite ejecutar un comando "save dialplan" desde la consola. El valor por defecto es " no".

autofallthrough: Si está activado y una extensión se queda sin cosas que hacer termina la llamada con BUSY, CONGESTION o HANGUP. Si no está activada se queda esperando otra extensión. Nunca debería suceder que una extensión se quede sin cosas que hacer como explicaremos posteriormente.

clearglobalvars: Si está activado se liberan las variables globales cuando se recargan las extensiones o se reinicia Asterisk.

priorityjumping: Si tiene valor 'yes', la aplicación (función) soporta 'jumping' o salto a diferentes, se encuentran a una prioridad que

normalmente es n=+101 donde n es el numero de la línea que se está ejecutando.

En general estas opciones no son muy importantes y se pueden dejar tal y como aparecen por defecto.

Contexto [globals]

En este contexto se definen las variables globales que se van a poder utilizar en el resto de los contextos.

```
CONSOLE=Console/dsp      ;    indica    que    cuando    hagamos
referencia   a   la   variable   CONSOLE   estamos   llamando   a
/Console/dsp
```

Las variables suelen ponerse siempre en mayúsculas para diferenciarla posteriormente.

Resto de Contextos []

Esto es lo más importante de este fichero, es aquí donde se configura el verdadero dialplan. Para crear un contexto especifico y asignar un plan de numeración, todas las líneas de un determinado contexto tienen el mismo formato:

```
exten => extensión,prioridad,Comando(parámetros)
```

Extensión

Es una instrucción que será seguida por Asterisk, luego de ser disparada por una llamada entrante o bien por dígitos discados en un canal, definido en el marco de un contexto.

```
exten => 123, 1, Answer ()
```

Existen otras extensiones validas. La extensión s (start) es una extensión especial que es utilizada si una llamada entra a un contexto sin una extensión destino específica, la llamada trata de entrar automáticamente a la extensión s.

```
exten => s, 1, Answer ()
```

Una extensión puede ser un literal o un patrón. Una extensión literal puede ser un número, como 123, y puede también contener los símbolos estándar * y # que aparecen en los teléfonos ordinarios, así que 12#89* es una extensión válida.

Algunos teclados numéricos de teléfono tienen teclas especiales de DTMF etiquetadas A, B, C y D, y las extensiones también se pueden definir con estas letras. De hecho, el nombre de una extensión puede contener cualquier letra o numero así como algunos signos de puntuación. Observe que muchos teléfonos VoIP pueden marcar “números” de extensión que pueden ser cualquier string de texto arbitrario, tal como “oficina”. Esta perfectamente permitido definir una extensión con el nombre “**oficina**” en Asterisk.

Los nombres de extensiones son sensibles a mayúsculas en el sentido que cuando el Asterisk está intentando machear la extensión que un usuario marcó con las extensiones definidas para un contexto, la extensión debe machear exactamente. Así que si un usuario marca la extensión “OFICINA” en su teléfono VOIP, el Asterisk no comenzará a ejecutar los comandos que ha definido para una extensión llamada “oficina”.

Por otra parte, los nombres de extensión no son sensibles a mayúsculas en el sentido que no puedes definir diversas extensiones (en el mismo contexto) que tengan los mismos nombres diferenciados solamente en las mayúsculas. No puedes definir un set de comandos para la extensión “oficina” y otro set de comandos para la extensión “OFICINA”.

Nombres predefinidos de la extensión:

Asterisk utiliza algunos nombres de extensión para propósitos especiales:

- i: Inválido
- s: Start
- h: Hangup
- t: Timeout
- T: Timeout Absoluto
- o: Operador

Prioridad

Establece la secuencia de comandos a ejecutar se enumeran 1, 2, 3…. Aunque la prioridad primera puede ser 1, en la segunda, tercera,.. línea el número de la prioridad se puede definir ***n*** (next) y sirve para no tener que escribir en cada línea el número progresivo de la prioridad. Esto es muy cómodo cuando se modifica el dialplan porque cuando vamos a insertar una línea no tendremos que renumerar todas las demás.

```
exten => 123,1,Answer()
exten => 123,2,Hangup()
```

```
exten => 123,1,Answer()
exten => 123,n,hagoalgo
exten => 123,n,Hangup()
```

Aplicaciones

Las aplicaciones realizan una acción determinada en el canal actual, controlando el comportamiento de la llamada y del sistema en sí.

Answer (): Answer es la aplicación que permite contestar la llamada.

Hangup (): Colgar la llamada.

Monitor (): Comenzar la grabación a disco de la llamada.

Dial ([destino], [tiempo de timeout], [opciones]): Realizar una llamada saliente.

Destino: SIP/extensión

Tiempo de timeout: Dado en segundos para abandonar la llamada

Opciones:

"T" permite al usuario llamante transferir la llamada pulsando #

t" permite al usuario llamado transferir la llamada pulsando #

"m" indica que vamos a oír una música especial mientras esperamos a que el otro conteste.

Para tener una idea de los comandos que podemos usar, escribimos:

```
CLI> help
```

Enunciados

Plan de marcación:

Cada número perteneciente a una PBX dentro del laboratorio constará de 4 dígitos. Los dos primeros dígitos identificarán al servidor Asterisk correspondiente al número de grupo asignado, el posterior representara la cuenta sip. Por lo tanto cada número tendrá la siguiente forma: NNXX. Donde NN es la representación del grupo, XX es la extensión de la PBX.

Por ejemplo: Para definir una extensión 1, en el servidor del número de grupo 11, esta estaría dada por: 1101.

1. ***Configurar en el servidor Asterisk dos canales SIP identificados por los nombres de los participantes, que tengan tanto la opción de recibir como efectuar llamadas. Realizar la configuración vía IP dinámica. Recordar hacer la recarga de la configuración cada vez que se efectúen cambios sobre la misma.***

2. ***Verificar vía la CLI que ambos canales se encuentran configurados tanto como clientes como proveedores. Utilizar los comandos: sip show peers y sip show users.***

3. ***Configure el plan de marcación, en el que se maneje las aplicaciones básicas de una llamada.***

4. ***Configurar una cuenta en cada softphone. Utilice nombres descriptivos para identificar fácilmente en qué PBX está configurado el usuario.***

5. ***Realice llamadas entre los softphones y verifique vía la CLI que las llamadas están siendo llevadas a cabo. Utilizar: sip show channels / show channels.***

6. ***Repetir todos los puntos utilizando canales de tipo IAX2.***

3.5.2 Práctica 2: Ampliación de la funcionalidad del Plan de Marcación

Objetivos

- Implementar patrones para la creación de extensiones.
- Utilizar aplicaciones más comunes en el dial plan.
- Implementar macros para la reutilización de código.

Materiales usados

- 2 Pcs
- Vmware server
- Sistema operativo CentOS
- Asterisk
- 2 Softphone: X-lite
- Dos auriculares

Introducción Teórica

Patrones

- X - Acepta un número de 0 al 9 no valen caracteres.
- Z - Acepta un número de 1 al 9 y cualquier cosa.
- N - Acepta un número de 2 al 9.
- [1,5-7] - Acepta el 1, el 5, el 6 o el 7 y cualquier cosa.
- . (punto) acepta uno o más caracteres.

Ejemplos del uso de patrones

```
exten => _9.,1,Dial(SIP/${EXTEN:1},30,r)
```

Observar que la variable ${EXTEN: 1} obtiene todos los dígitos de la extensión actual menos el primer carácter, en este caso: 9 + los dígitos siguientes.

```
exten => _20XX,1,Dial(SIP/${EXTEN:2},90,Tt)
```

Deberíamos marcar 20 y dos números (no valen caracteres)

```
exten => _20ZZ.,1,Dial(SIP/${EXTEN:2},90,Tt)
```

Deberíamos marcar 20, dos números del 1 al 9 y cualquier cosa

```
exten => _20[1-3].,1,Dial(SIP/${EXTEN:2},90,Tt)
```

Deberíamos marcar 20, un numero del 1 al 3 y cualquier cosa

Otras opciones para definir extensiones incluyen una opción designada comúnmente como la lógica de la ex-novia. Esta lógica macheara la extensión marcada, haya venido del exterior o del interior, basada en el callerid de la persona que llama.

```
exten => 1234/100,1,Answer()
exten => 1234/100,2,Playback(tt-weasels)
exten => 1234/100,3,Voicemail(123)
exten => 1234/100,4,Hangup()
```

Esto macheara la extensión 1234 y realizará las opciones siguientes SOLAMENTE si el número de callerid del usuario que llama es 100. Esto se puede también lograr mediante patrones:

```
exten => 1234/_30NXXXXXX, 1, Answer ()
```

Y así sucesivamente...

Esto macheara solamente con 1234 si el número de identificación de llamada es algo que comience con 30.

Otras aplicaciones

Goto(): Saltar a otra extensión, prioridad o contexto.

```
exten => 123,1,Goto(contexto,extension,prioridad)
```

PlayBack(sonido): Reproducir un fichero de sonido. Los archivos de audio lo busca en /var/lib/asterisk/sounds, por defecto.

Echo(): Se ejecuta el test de eco.

AGI(): Llamar a una aplicación externa (stdin, stdout, stderr).

Voicemail(buzón): (prioridad + 101) Salta al contestador para dejar un mensaje. Es importante que por cada rama siempre se cierre el camino y se cuelgue la llamada con un ***Hangup***.

Background(): Similar a playback, pero si el usuario presiona dígitos, la aplicación lo captura y trata de enviarlo a la extensión presionada.

```
exten => 123, 1, Background (hello-world)
```

Variables

En el dialplan de Asterisk existen variables, que pueden ser modificadas por el propio Asterisk en su ejecución lógica o por comandos expresos (aplicaciones) del dialplan. Las variables reducen la escritura, agregan claridad al dialplan y le aportan lógica.

Los tipos de variables son:

Globales: Declaradas en extensions.conf (o por comando).

Canal: Asociadas con un canal particular.

Entorno: Variables de entorno (UNIX Like).

La sintaxis de una variable es:

```
${variable}
```

Manejo de variables

Asignación de variables:

- Set(Variable=valor)
- Global(Variable=valor)

Manejo de cadenas

Subcadenas: ${Variable:offset:longitud}

Devuelve la subcadena de variable que comienza en offset y con la longitud especificada.

```
${123456789:2:3}  devuelve 345
```

Longitud: ${LEN (Variable)}

Concatenación: ${Variable1} ${Variable2}

Variables globales

Permite que se pueda hacer referencia a ellas en todos los contextos, en todas las extensiones, a diferencia de las variables convencionales que sólo tienen validez en el canal actual. Es útil para tener claridad y manejabilidad en el dialplan. Se pueden definir en el contexto [globals] al inicio de extensions.conf

```
[globals]
YESENIA=Sip/yesenia
FERNANDO=IAX2/2000
```

Variables de canal definidas automáticamente

Listado de variables más importantes:

${CALLERID}: caller ID actual, nombre y número.

${CONTEXT}: contexto actual.

${EXTEN}: extensión actual.

${CHANNEL}: canal actual.

${DIALSTATUS}: estado de la llamada: unavailable, congestion, busy, noanswer, answer, cancel, hangup.

${DATETIME}: hora actual.

Un comando útil para ver el contenido es NoOp:

```
NoOp (${VARIABLE})  mostrará en el CLI el valor.
```

${EXTEN}: Permite saber cuál es la extensión que fue marcada. Se utiliza comúnmente para eliminar dígitos marcados: ${EXTEN: x}

```
exten => _XXX, 1, SayDigits (${EXTEN: 1})
```

Si x es positivo, quita los primeros x dígitos marcados

```
exten => _XXX, 1, SayDigits (${EXTEN:-1})
```

Si x es negativo, devuelve los últimos x dígitos marcados

Include

Puede utilizarse un contexto dentro de otro contexto a través de la directiva include, lo que permite habilitar derechos de acceso a las diferentes secciones del dialplan.

```
include => context
```

Primero trata de encontrar las extensiones en el contexto actual. Si no la encuentra, trata de encontrarla en el primer contexto incluido, y después en el segundo y así sucesivamente. Por ejemplo, que los dispositivos del contexto [default] puedan hacer llamadas a los dispositivos con cuentas IAX.

```
[default]; Context "default"

exten => 101, 1, Dial (SIP/101)
exten => 101, 2, Hangup ()
exten => 102, 1, Dial (SIP/102)
exten => 102, 2, Hangup ()

[local]; Context "local"
exten => _4XXX, 1, Dial (SIP/${EXTEN: 1})
exten => _4XXX, 2, Hangup ()
include => default
```

Usando macros para crear extensiones.

Macro: Ejecuta una Macro. Las Macros son Contextos especiales, en los cuales solo caben reglas para la extensión "s" las cuales son ejecutadas en orden para posteriormente devolver el flujo de las acciones al siguiente paso en la llamada a esta aplicación Macro. Usa todas las variables además de MACRO_EXTEN, MACRO_CONTEXT and MACRO_PRIORITY.

Los argumentos dentro del contexto macro se identifican con ${ARG1} ${ARG2}...

```
Macro(nombremacro,arg1,arg2,...,argN)
```

Parámetros:

nombremacro: Nombre de la Macro.

Argumentos: Los Argumentos se separan por comas en caso de existir más de uno.

```
[globals]
ADA=SIP/100
YESENIA=IAX2/200

[macro-miAsterisk]
exten => s, 1, Dial (${ARG1}, 20, t)
exten => s, 2, Hangup

[local]
exten => 100, 1, Macro (miAsterisk, ${ADA})
exten => 200, 1, Macro (miAsterisk, ${YESENIA})
```

Manipulación de expresiones y variables

Las expresiones son una combinación de variables, operadores y valores que arrojan un resultado.

Sintaxis:

$[expr1 operador expr2]

Operadores lógicos: | (OR), & (AND)

Operadores de comparación: =!,=, <, >, <=, >=

Operadores aritméticos: +, -, *, /, % [...],

```
exten => 4003,1,Set(COUNT=3)
exten => 4003,2,Set(NEWCOUNT=$[${COUNT}+1])
exten => 4003,3,SayNumber(${NEWCOUNT})
```

Bifurcación condicional

Permite tomar decisiones dentro del dialplan. Aplicación GotoIf ()

```
GotoIf (expresion1?destino1:destino2)
```

Si la expresión evaluada es verdadera, la llamada es enviada a destino1, de lo contrario es enviada a destino2. Una cadena vacía y el número 0 son evaluados con falso, cualquier otro valor es verdadero. Cualquiera de los destinos puede ser omitido, pero debe estar alguno de los 2. Si el destino omitido es el camino que debe seguir la llamada, el flujo que se sigue es la siguiente prioridad dentro de la extensión actual.

Bifurcación condicional basada en tiempo

Verifica la hora actual del servidor, permitiendo tomar decisiones basadas en tiempo. Se utiliza cuando se quiere dar una bienvenida diferente en horarios de trabajo y fuera de trabajo.

```
GotoIfTime(hora,dias_de_semana,dias_del_mes,meses?etiqueta)
```

Envía la llamada a etiqueta si la fecha y hora actual concuerdan con el criterio especificado por los parámetros.

- hora. Lista de uno o más rangos de horario en formato de 24 horas. 09:00-17:00
- dias_de_semana. Lista de uno o más días de la semana en ingles (mon, tue)

- dias_del_mes. Día numérico del mes 7-12,15
- meses. Lista de uno o más meses del año jun, apr, jul
- Matchea con cualquier valor
- Etiqueta puede ser una prioridad dentro de una misma extensión, una prioridad y extensión dentro del mismo contexto o un contexto, extensión y prioridad.

```
Exten => s,1,GotoIfTime(*,*,2,nov?open,s,1); se envía al contexto open, extensión s, prioridad 1.
```

Enunciados

1. ***Defina 2 cuentas sip, cada una de 3 dígitos. Defina 2 cuentas iax, cada una de 4 dígitos. Configure cada una de las cuentas de modo que todos puedan hacer y recibir llamadas, que sea Asterisk que administre el flujo de llamadas, habilite los codec necesarios.***

2. ***Verifique a través del CLI los users y peers que se crearon en cada una de los canales.***

3. ***Defina en el dial plan:***
    a. ***Dos contextos, el primero para las cuentas sip, el segundo para los canales iax. Utilice patrones en ambos contextos. Para llamar a las extensiones sip se debe utilizar el prefijo 1 y 2 para las iax, después del prefijo solo se debe permitir números del 0 al 9. Reproduzca el archivo de audio demo-congrats, seguido se reproducirán los números de la extensión marcada y por ultimo ejecute Dial, se le dará 20 segundos para que el usuario conteste.***
    b. ***Sustituya el uso de prioridades 1, 2,3… por next.***

4. ***Agregue un nuevo contexto, en el se definirá una nueva extensión que al ser marcada llamara simultáneamente a las extensiones 200(SIP) y 0001(IAX).***

5. ***En la sección [globals] defina nombres para los canales sip, haga los cambios necesarios en el contexto correspondiente.***

6. ***Dentro de un solo contexto defina una extensión cualquiera la cual, al ser marcada, pedirá la extensión con la cual desea comunicarse, esperará durante 20 segundos y transferirá la llamada a la extensión***

correspondiente, si la extensión no es válida reproduzca un archivo que indique el error, por ultimo si ninguna extensión fue marcada durante los 20 segundos despedirse de usuario y colgar. Utilice las aplicaciones BackGround Y WaitExten, para ayuda consulte en el CLI core show application nombreaplicacion.

7. ***Realice lo siguiente:***
    a. ***Defina una macro que pueda ser utilizada para las cuentas IAX y SIP. Controle el estado de la extensión o ${DIALSTATUS}) que puede ser: NOANSWER, BUSY, CHANUNAVAIL, CONGESTION, ANSWER. Para cada uno de los casos reproduzca un archivo de sonido indicando al llamante el estado de la misma, a excepción del estado ANSWER.***
    b. ***Defina nombres para cada una de las cuentas y realice los cambios necesarios en cada uno de los contextos para poder utilizar la macro.***

3.5.1 Práctica 3: Interconexión entre servidores Asterisk mediante el Protocolo IAX

Objetivos

- Configurar las cuentas IAX en cada servidor.
- Configure los contextos de entrada y salida en cada servidor permitiéndole realizar y recibir llamadas externas.
- Verificar en el CLI la interconexión entre servidores
- Realizar pruebas llamando a extensiones internas
- Realizar pruebas llamando a extensiones externas comprobando la interconexiones entre servidores

Materiales usados

- 2 PCs
- 2 servidores Asterisk
- 2 auriculares

Introducción Teórica

La interconexión entre diferentes servidores Asterisk se establece de la siguiente manera:

Configurando en ambos servidores el archivo iax.conf como peer y como user el cual permite recibir y realizar llamadas respectivamente, también se modificara el dialplan para que se puedan efectuar llamadas desde el user hacia el peer de cada servidor.

Como ejemplo tenemos dos servidores Asterisk, A y B, y queremos conectarlos entre ellos, usaremos el protocolo IAX, para llamar desde el servidor A las extensiones del servidor B y desde el servidor B las extensiones del servidor A. Las extensiones locales del servidor A y del servidor B no pueden ser iguales.

Ejemplo: en el servidor A las extensiones son de 3 cifras y empiezan por 3, en el servidor B también son de 3 cifras y empiezan por 4 configurando correctamente todos los archivos involucrados podremos realizar llamadas externas entre servidores y desde diferentes ubicaciones geográficas.

Enunciados

1. ***Configurar las cuentas IAX en cada servidor:***
   a) ***Defina una cuenta de tipo peer que le permita realizar llamadas.***
   b) ***Defina una cuenta de tipo user a través de la cual pueda recibir llamadas.***
   c) ***Defina cuentas para los usuarios.***

2. ***Configurar el dialplan de cada servidor:***
   a) ***Defina un contexto en el que se definan reglas para controlar las extensiones locales.***
   b) ***Defina un contexto que controle las llamadas externas.***

3. ***Realizar pruebas llamando a extensiones internas.***

4. ***Realizar pruebas llamando a extensiones externas comprobando las interconexiones entre servidores.***

5. ***Verificar en el CLI la interconexión entre servidores.***

3.5.2 Práctica 4: Voicemail

Objetivos

- Interactuar con el contestador automático.
- Enviar mensaje de voz cuando el usuario no se encuentre disponible.
- Escuchar mensaje de voz recibidos por parte de otro usuario.
- Manipular los mensajes de voz.

Materiales usados

- Un pc.
- Un servidor Asterisk.
- Un auricular.

Introducción Teórica

El Voicemail o buzón de voz es un sistema centralizado de manejo de mensajes telefónicos para un número determinado de personas. Permite a los usuarios recibir, almacenar y gestionar mensajes de voz de las personas que le llaman cuando se encuentra ausente o con la línea ocupada.

La configuración del voicemail se realiza en el archivo voicemail.conf, dentro del cual se establecen los parámetros de funcionamiento para correo de voz y se crean los buzones de correo o mailboxes. Cada buzón hace referencia al número de extensión definido en el dialplan, dentro del archivo extensions.conf. La sintaxis básica de un mailbox es la siguiente:

```
mailbox => password,nombre,email
```

A continuación se explica cada una de las partes que conforman la definición de un mailbox.

Mailbox: Es el número del buzón de voz y corresponde al número de extensión al cual se encuentra asociado.

Password: Es la contraseña que permite al dueño del buzón de correo acceder a su correo de voz.

Nombre: El nombre del propietario del mailbox

E-mail: Es la dirección de e-mail del dueño del mailbox, hacia el cual Asterisk envía la notificación de la existencia de un voicemail.

A continuación se muestra una línea de configuración del archivo voicemail.conf aplicada al diseño propuesto.

```
601 => 4242,Fernando Larios,lariosli2@mydomain.com
```

Para conseguir esto se usa la aplicación VoiceMail(), la cual dirige la llamada al buzón de correo específico siempre y cuando los usuarios de las extensiones no respondan o la linea este ocupada.

```
exten => extensión,prioridad,VoiceMail(mailbox@contexto)
```

La función VoiceMail() forma parte de las instrucciones del dialplan presentes en el archivo extensions.conf. A continuación se muestra un ejemplo:

```
exten => 601,1,Dial(SIP/gerencia_desk)
exten => 601,n,VoiceMail(601@buzones,u)
```

Para que los usuarios puedan acceder a sus mensajes de voicemail, cambiar sus opciones y grabar los saludos que están en sus casillas, es necesario definir una extensión dentro del dialplan y hacer uso de la aplicacion VoiceMailMain(), tal como se muestra a continuación:

```
exten => 800,1,VoiceMailMain ()
```

Correo de voz Configuración

[general]

***Format**:* gsm|wav ; el codec audio utilizado para grabar los correos de voz dejados en el contestador.

Serveremail: Asterisk@voztovoice.org; el remitente del correo electrónico que nos avisa de un nuevo correo de voz.

Attach Si attach está en yes el correo de voz se enviará como anexo al correo electrónico.

maxmsg=100 ;número máximo de correos de voz para cada casilla configurada.

maxsecs=300 ;número máximo de segundos por cada correo de voz.

minsecs=3 ;número mínimo de segundos para que un correo de voz sea reconocido como tal y enviado a la casilla del destinatario.

skipms=3000 ;cuando escuchamos los correos de voz podemos usar el teclado numérico para adelantar o atrasar el mensaje. Por ejemplo: presionando el número 8 nos adelantamos 3000 milisegundos, es decir 3 segundos, con el 9 nos devolvemos 3 segundos.

maxsilence=10 ;si mientras se graba el correo de voz hay un silencio de 10 segundos, la llamada se termina y también la grabación.

silencethreshold=128 ;esto es un nivel (decibel) que sirve para definir que se considera silencio. Más bajo el número, más sensible al ruido.

maxlogins=3 ;número máximo de veces que nos podemos equivocar insertando la contraseña para entrar a nuestra correo de voz.

usedirectory=yes ;Los correos de voz que recibimos podemos reenviarlos a otro usuarios/extensiones del servidor Asterisk. También podemos dejar directamente mensajes de voz en determinadas casillas. Esta opción permite buscar en el directorio la persona a la cual queremos dejar o reenviar el correo de voz.

odbcstorage=Asterisk ;podemos guardar los mensajes de voz en una base de datos usando el driver ODBC. Aquí es donde hay que configurar esta opción.

odbctable=voicemessages ;el nombre de la tabla de la base de datos donde guardar los mensajes de voz.

review=yes ;Si es igual a yes permite, a quien está dejando un correo de voz, escucharlo antes de enviarlo.

operator=yes ;permite, a quien llama, presionar 0 antes/después/mientras está dejando un correo de voz para buscar una operadora.

envelope=no ;creo que antes de reproducir el mensaje audio, reproduce los datos del mensaje.

delete=yes ;Si es igual a yes, está activado que una vez que se notifique la llegada de un correo de voz, éste se borrará del servidor.

volgain=0.0 ;si el correo de voz se grabó con un volumen muy bajo con esta opción podemos mejorar su calidad. Para que se pueda utilizar debemos tener instalado sox.

listen-control-forward-key=# ; tecla numérica para adelantar el mensaje que se está escuchando.

listen-control-reverse-key=* ;tecla numérica para ir atrás en el mensaje que se está escuchando.

listen-control-pause-key=0 ;tecla numérica para poner en pausa el mensaje.

listen-control-restart-key=2 ;tecla numérica para volver a escuchar el mensaje desde el inicio.

listen-control-stop-key=13456789 ;teclas numérica para detener el mensaje y volver al menú del contestador.

backupdeleted=100 ; número máximo de mensajes en la carpeta "borrados".

```
[zonemessages]
colombia=America/Bogota|'vm-received' aebY 'digits/at' HM
eastern=America/New_York|'vm-received' Q 'digits/at' IMp
central=America/Chicago|'vm-received' Q 'digits/at' IMp
central24=America/Chicago|'vm-received'  q  'digits/at'  H  N
'hours'
military=Zulu|'vm-received'   q   'digits/at'   H   N   'hours'
'phonetic/z_p'
european=Europe/Copenhagen|'vm-received' a d b 'digits/at' HM
```

En el bloque de arriba definimos las zonas horarias que podemos usar en el contestador. Si por ejemplo tenemos usuarios de distintos continentes, podemos

definir por cada uno su huso horario y de esta forma configurar la exacta fecha y hora de los correos de voz de su casilla

[default] ; aquí empieza la configuración de las casillas para los usuarios pertenecientes al contexto default

maxmsg=50 ; define el número máximo de correos de voz por cada carpeta de un determinado contexto (en este caso el contexto es default)

Cada casilla sigue estas reglas, opciones separadas por el signo |:

```
número extensión => contraseña, nombre apellido, correo electrónico, correo pager,
```

Enunciados

1. ***Crear una cuenta por cada uno de los integrantes tanto iax como sip.***

2. ***En el archivo voicemail.conf configure una cuenta correo de voz para cada usuario, con los parámetros correspondientes.***

3. ***Crear una macro que asocie cada cuenta con el correo de voz:***
   a) ***Configurar reproducción de anuncio automático después de 10 segundos si el usuario no se encuentra disponible o su línea este ocupada.***
   b) ***Configurar una extensión de acceso general a la casilla de correo para cada usuario donde podrán escuchar y administrar sus mensajes.***

4. ***Dejar mensajes en la casilla de usuario y luego acceder para escucharlos.***

3.5.3 Práctica 5: IVR

Objetivos:

- Personalizar los menús interactivo.
- Interactuar con el usuario mediante menús de voz.
- Presentar mediante los menús las acciones a tomar.
- Efectuar llamadas desde los Menús Interactivo.

Materiales usados

- Un pc
- Un servidor Asterisk
- Un auricular
- 1 máquina virtual
- Vmware permite crear las máquinas virtuales

Introducción Teórica

IVR son las siglas de Interactive Voice Response, que se traduce del inglés como Respuesta de Voz Interactiva. Es un sistema automatizado de respuesta interactiva, orientado a entregar y/o capturar información a través del teléfono, además permite que la persona que llama transfiera la llamada a la extensión deseada haciendo uso de menús interactivos, activados mediante la marcación del teclado del teléfono. En términos generales representa un sistema automatizado que está en capacidad de atender múltiples llamadas de manera simultánea.

Funcionamiento de esta aplicación

El usuario realiza una llamada a un número de teléfono, el sistema de audio respuesta contesta la llamada y le presenta al usuario una serie de acciones a realizar, esto se hace mediante mensajes (menús de opciones) previamente grabados en ficheros de audio (Por ejemplo "Pulse uno para ventas, dos para

administración"). El usuario elige la opción a realizar introduciendo un número en el teclado del teléfono y navega por los diferentes menús hasta encontrar la información solicitada o que el sistema enruta la llamada al destinatario elegido.

El modo del funcionamiento básico de un IVR sigue el siguiente comportamiento:

- Cuando una llamada se realiza, la aplicación WaitExten() permite que la central se quede a la espera de una extensión marcada por el usuario.
- Cuando el usuario dentro de un tiempo especificado o timeout no ha marcado, dentro del flujo de llamada programado, la central realiza un salto a la extensión '**t**' si es que ésta existe.
- Si el usuario ha digitado una opción la central Asterisk busca la extensión asociada a la opción y realiza un salto hacia ella para ejecutarla; en caso de no existir tal extensión el salto se lo realiza hacia la extensión '**i**' si se encuentra definida.

Las extensiones "**t**" e "**i**" son nombres de extensiones reservadas; algunos tipos de dichas extensiones se detallan a continuación:

s (start): Una llamada que no tiene algún dígito asociado con ella; por ejemplo una línea analógica loopstart.

t (Timeout): Cuando la persona que llama a un menú de voz interactivo no ingresa el número correcto de dígitos, la extensión ejecuta el timeout .

T (absolute timeout): Cuando una llamada excede el valor llevado a cabo en una variable absoluta de timeout.

i (invalid): Se ejecuta cuando quien llama, ingresa una extensión inválida 144

h (Hangup): Se ejecuta al final de una llamada cuando quien la realiza cuelga. Las aplicaciones ejecutadas en esta extensión no pueden tener acceso al canal cerrado, es muy útil para registrar o ejecutar comandos.

Un menú IVR en la mayoría de casos se emplea haciendo uso de la aplicación **BackGround** la cual empieza a reproducir un archivo de audio e inmediatamente ingresa en espera, para que el usuario pueda introducir extensiones sin tener que esperar.

Enunciados

1. ***Crear una extensión para grabar un mensaje y seguidamente reproducir el mensaje grabado.***

2. ***Grabar un mensaje de bienvenida para un sistema de IVR y que dicte las opciones del mismo.***

3. ***Agregar al dialplan un sistema de IVR, agregando una extensión especial para el mismo***

4. ***Agregar al dialplan un sistema de IVR, agregando una extensión de marcado directo para los internos tal que:***
   a) ***Utilice el mensaje de bienvenida grabado anteriormente.***
   b) ***Marcando una extensión cualquiera se comunique con ella.***
   c) ***El timeout entre dígitos sea de 5 segundos (hint: asignar 5 a la variable especial TIMEOUT(DigitTimeout)).***
   d) ***El timeout de ingreso de una opción sea de 10 segundos (hint: asignar 10 a la variable especial TIMEOUT(ResponseTimeout)).***
   e) ***Al presionar una opción inválida, reproduzca un mensaje de error (por Ej., pbx-invalid) y regrese al mensaje de bienvenida.***
   f) ***Al cumplirse el timeout, regrese al mensaje de bienvenida.***

3.5.4 Práctica 6: Configuraciones adicionales para tecnología de VoIP

Objetivos

- Configurar el Dialplan para la utilización de servicios adicionales
- Realizar Video llamadas entre las cuentas del dialplan
- Efectuar llamadas automáticas
- Manipular colas
- Utilizar música en espera
- Crear salas de conferencias

Introducción Teórica

Video llamada

Con terminales especiales es posible añadir soporte de video a sus llamadas. El envío de video usando Voz sobre IP se realiza añadiendo unos codecs especiales (ya incluidos en sus instalación de Asterisk) al protocolo. Estos codecs[31] permiten comprimir el video a fin de transportarlo sobre la red de datos sin consumir un exceso de recursos. Actualmente, es posible transmitir una conversación entre dos personas con un consumo de ancho de banda de tal solo 32 Kbits/seg. (Con una calidad reducida), el consumo puede crecer hasta 1 Mb/seg. para la calidad máxima.

Los canales que tienen soporte para video son SIP e IAX. El canal H.323 (chan_h323, chan_oh323, chan_ooh323) no permiten las llamadas de video en este momento; sin embargo, el ChangeLog de ooh323 en su release para 0.6: " añade el codec H.263 para negociación de video H.263.

[31] H.261, H.263, H.263p (Asterisk 1.4), H.264 (Asterisk 1.4)

La orden para habilitar video en Asterisk es modificando en el sip.conf o iax.conf

```
[general]
videosupport=yes
```

Comando Mixmonitor

Utlizado para grabar el audio de las conversaciones de algún canal en particular.

Descripción

```
MixMonitor(<file>.<ext>[|<options>[|<command>]])
```

Opciones validas:

b - Graba el audio en el archivo cuando la llamada se establece.

a - Agrega la grabación al mismo archivo en vez de sobrescribirlo. Cuando el archivo tiene el mismo nombre.

v(<x>) - Ajuste del volumen de escucha usando el factor <x> -4/4.

V(<x>) - Ajuste del volumen de habla usando el factor <x> -4/4.

W(<x>) - Ajuste de todo el volumen usando el factor <x> -4/4.

Los archivos de audio quedaran por defecto en:

```
/var/spool/Asterisk/monitor
```

Marcado automático en una extensión

Archivos .call se utilizan para iniciar llamadas desde una aplicación externa. Son archivos de texto que al copiarse en el directorio /var/spool/Asterisk/outgoing, Asterisk notará su presencia e inmediatamente activará la extensión en la prioridad especificada en el archivo .call. Generalmente, se combinan con el programador de tareas de Linux: el cron. Algunos ejemplos de uso son: soluciones de contestaciones automáticas de llamadas, despertadores telefónicos, anuncios automáticos.

Manejo de colas

El sistema de colas en Asterisk se compone de:

1. **Llamadas entrantes** que son ubicadas en una cola.
2. **Miembros** son aquellos canales disponibles que están activamente atendiendo la cola. Pueden ser tanto agentes como también canales regulares (SIP/4101) La configuración de los agentes se define en el archivo agents.conf. Estos deben realizar un **login** indicando que está listo para tomar llamadas.
   - ***Penalty:*** Se le asigna una penalidad a cada agente, de manera tal que primero se derivan las llamadas (vía la estrategia definida) a los agentes con el menor valor de penalidad. En el caso de estar todos ocupados, se continúa con el siguiente penalty y así sucesivamente.
   - ***Priority****:* Se le asigna una prioridad a cada llamada entrante, permitiendo situarla en un lugar más adelante de la cola (no siempre al final).
3. Algunas **estrategias** sobre cómo manejar la cola y repartir las llamadas entre los miembros:
   - ***ringall:*** Hace sonar todos los canales disponibles hasta que alguno responda (configuración por defecto).
   - ***roundrobin:*** Hace sonar cada interfaz disponible por turnos.
   - ***leastrecent:*** Hace sonar la interfaz que fue menos recientemente llamada por esta cola.
   - ***fewestcalls:*** Hace sonar la interfaz con la menor cantidad de llamadas completas.
   - ***random:*** Hace sonar una interfaz al azar.
   - ***rrmemory:*** Igual que el round robin pero recuerda cual fue el último teléfono que atendió una llamada y continúa con el siguiente.
4. **Música** que se reproduce durante la espera en la cola.
5. **Anuncios** para miembros y emisores de llamadas.

La configuración de las colas se define:

- **Estáticamente:** En el archivo queues.conf .
- **Dinámicamente:** La configuración se almacena en una BD, poniendo a disposición los cambios sin necesidad de realizar un reload.

Aplicaciones principales, utilizadas en extensions.conf:

Queue: Aplicación utilizada para encolar una llamada (toma como parámetro las colas definidas en queue.conf).

AddQueueMember: Agrega dinámicamente un miembro a la cola.

RemoveQueueMember: Remueve dinámicamente un miembro de la cola.

AgentLogin: Login de un agente a una cola.

Comandos relacionados de la CLI:

show agents: Muestra los agentes.

show queues: Lista todas las colas.

show queue: Muestra datos de una cola en particular.

queue add member: Agrega un miembro a la cola.

queue remove member: Elimina un miembro de la cola.

Registro de llamadas

Asterisk permite llevar un control exhaustivo de todas las llamadas que se han realizado o recibido. Es interesante para control propio de facturación, independiente del proveedor (si no se es uno de ellos). Permite realizar estadísticas. Este control se denomina CDR: Call Detail Record.

El registro del CDR se escribe por defecto en el archivo

```
/var/log/Asterisk/cdr-csv/Master.csv
```

Existen extensiones al cdr: cdr_mysql por ejemplo, que permiten almacenar los registros en una base de datos. cdr_mysql está disponible en Asterisk-addons El CDR se configura en el archivo cdr.conf, para el módulo de MySQL, se utiliza cdr_mysql.conf

Para confirmar el estado del CDR desde el CLI, se puede ejecutar:

```
CLI> cdr status
```

Algunos de los campos más importantes para un CDR (registro) son:

accountcode: Código de la cuenta a utilizar.

src: Número del caller ID.

dst: Extensión destino.

dcontext: Contexto destino.

start: Comienzo de la llamada (fecha/hora).

answer: Respuesta de la llamada (fecha/hora).

end: Fin de la llamada (fecha/hora).

duration: Duración de la llamada en segundos, desde que fue discada hasta el corte.

billsec: Duración de la llamada en segundos, desde que fue atendida hasta el corte.

disposition: Estado de la llamada (atendida, no atentendida, ocupado, fallida).

Existen muchas aplicaciones que permite gestionar el CDR. Desarrollar una propia no es realmente muy complejo. Algunas aplicaciones open source son:

- **Astbill:** Es una de las mejores aplicaciones opensource para tarificación, control de cuentas y llamadas.
- **Areski Stat v2:** Se trata de una aplicación para listar y realizar estadísticas de las llamadas realizas o enviadas.
- **A2Billing**

Sistema de logs

En el archivo /etc/Asterisk/logger.conf se encuentra la configuración del sistema de logging de Asterisk.

Los distintos niveles de información a capturar en los logs son:
Verbose: Mensajes generales sobre lo que está ocurriendo en el sistema (por Ej., si el valor de verbosity es mayor a 3, muestra las instrucciones del plan de marcación).
Debug: Mensajes con información extendida, en general utilizados por programadores.
Notice: Notificaciones no críticas.
Warning: Mensajes de alerta posiblemente críticos.
Error: Mensajes indicando que ocurrió algo grave.

En el contexto [logfiles] del archivo logger.conf se indican los archivos y mensajes a loguear en cada uno y, la sintaxis es: *archivo => nivel1,...,niveln*

Los archivos de log se crean por defecto en /var/log/Asterisk/ (esto se puede cambiar /etc/Asterisk/Asterisk.conf).

```
debug => debug
full => notice,warning,error,debug,verbose
```

Para enviar a la consola, hay que definir el archivo especial console:

```
console => notice,warning,error,debug
```

También se pueden enviar al syslog:

```
syslog.local0 => debug,warning,error,notice,verbose
```

Configurando además en /etc/syslog.conf:

```
local0.*                        @ip_servidor
```

Los comandos relacionados con el manejo del log del CLI son:

logger reload: Reabre los archivos de log del Asterisk y recarga la configuración del logger .

logger rotate: Rota los archivos de log y luego hace un logger reload.

core set verbose: Cambia el nivel de información a mostrar en la consola.

Comandos CLI, se revisarán algunos otros adicionales:

Canal de consola (console channel)

console dial: Permite hacer un llamada desde la consola.

console answer: Permite contestar una llamada desde la consola.

console hangup: Cuelga la llamada en curso en la consola.

Administración del servidor:

stop/restart gracefully: Parar/recomenzar Asterisk cuando no haya llamadas en curso y sin aceptar nuevas llamadas.

stop/restart now: Parar/recomenzar inmediatamente.

stop/restart when convenient: Parar/recomenzar Asterisk cuando no haya llamadas en curso.

reload: Recarga toda la configuración.

module load/unload: Cargar/descargar un módulo específico.

module show: Mostrar todos los módulos levantados.

Generales:

show applications: Mostrar aplicaciones registradas en Asterisk.

show channels: Listar los canales definidos.

show codecs: Mostrar información sobre los codecs instalados.

show translation: Mostrar un cuadro de doble entrada con los tiempos de conversión entre formatos de codecs.

soft hangup: Colgar una llamada en un canal.

debug channel: Realizar un debug de un canal.

Plan de marcación:

dialplan show: Mostrar el plan de marcación actual.

dialplan save: Guardar los cambios realizados.

dialplan add/remove extension: Agregar / eliminar una extensión en un contexto dado al plan de marcación.

dialplan add/remove include: Incorporar / eliminar un include en un contexto dado en el plan de marcación.

Salas de conferencias

El archivo meetme.conf permite configurar salas de conferencias simples para Asterisk. El archivo es leído cada vez que la aplicación de Asterisk ejecuta meetme() invocado desde el extensions.conf , está dividida en 2 contextos:

```
[general]
;audiobuffers=32         ; El numero de 20 ms de buffers de
audio para usarse
 [rooms]
;El  modo de usar
; conf => confno[,pin][,adminpin]
;Confno=numero de asignado a la conferencia
```

Music on Hold

```
; las opciones de modo válidas:
;  files             -- lee los archivos de un directorio,
cualquiera que soporte los formatos media
; quietmp3      -- por defecto
; mp3           -- loud
; mp3nb         -- unbuffered
; quietmp3nb    -- quiet unbuffered
; custom        -- ejecuta una aplicación personalizada
; =========
; Con base en archivo (nativo) music on hold
; =========
;
```

```
; este reproduce los archivos directamente de los archivos
especificados, no externos son procesados y requeridos. Los
archivos son reproducidos en el orden normal  (algunos son
ordenados según la lista del directorio),  no está disponible
ningún ajuste al volumen. Si el archivo está disponible en el
mismo formato del codec del canal será reproducido sin
transcoding. Los archivos pueden tener varios formatos, pero
se elegirá el mejor formato al reproducirse

; Si no está usando la "carga automática" en modules.conf,
entonces se debe asegurar que los módulos de formato para
cualesquiera formatos que desea usar esta cargado _before_
res_musiconhold. Si usted no hace esto, res_musiconhold
saltará los archivos no será capaz de comprender cuando
carga.

[default]
mode=files
directory=/var/lib/Asterisk/moh
;
[native-random]
mode=files
directory=/var/lib/Asterisk/moh
random=yes      ; Reproduce los archivos en orden aleatorio


; =========
; Otro (no-nativo) métodos de reproducción de una grabación
; =========

[manual]
mode=custom
; Note que con mode=custom, un directorio no es requerido,
tal como al leer de un stream.
directory=/var/lib/Asterisk/mohmp3
application=/usr/bin/mpg123 -q -r 8000 -f 8192 -b 2048 --mono
-s

[ulawstream]
mode=custom
application=/usr/bin/streamplayer 192.168.100.52 888
format=ulaw

; mpg123 en Solaris siempre no sale correctamente el madplay
puede ser una mejor elección

[solaris]
```

```
mode=custom
directory=/var/lib/Asterisk/mohmp3
application=/site/sw/bin/madplay -Q -o raw:- --mono -R 8000 -
a -12
```

Materiales usados

Softphone: X-lite

Vmware server

Sistema operativo CentOS

Asterisk

2 Diademas

2 Pc

Enunciados

1. ***Configurar en el servidor Asterisk dos canales SIP identificados por los nombres de los participantes, que tengan tanto la opción de recibir como efectuar llamadas. Realizar la configuración vía IP dinámica. Recordar hacer la recarga de la configuración cada vez que se efectúen cambios sobre la misma.***

2. ***Configurar en el servidor Asterisk dos canales IAX con nombres de cuentas user1, user2, que tengan tanto la opción de recibir como efectuar llamadas. Realizar la configuración vía IP dinámica. Recordar hacer la recarga de la configuración cada vez que se efectúen cambios sobre la misma.***

3. ***Activar en los canales (SIP e IAX) el soporte para video llamada.***

4. ***Realizar video llamada entre los participantes utilizando las cuentas configuradas en el inciso 1.***

5. ***Configurar en el servidor Asterisk para grabar las llamadas realizadas provenientes de todas cuentas creadas en el inciso anterior.***
    a) ***Cree una macro llamada grabar, guardar las conversaciones con el formato del archivo YYYY-mm-dd-extension.wav.***

    5.1. Invoque desde su plan de marcación para cada una de sus extensiones la macro.

    5.2. Defina una extensión para escuchar los archivos guardados por el llamante.

6. ***Configurar una extensión que pertenezca a un contexto [mensaje-salida], hacer que reproduzca un anuncio y se despida.***
    a) ***Crear un archivo de marcado automático, que llame a la extensión creada, darle prioridad 1.***

7. ***Configurar una cola de modo que se reproduzca una música antes de ser atendido.***

8. ***Configurar un agente para atender la cola.***

9. ***Configurar una extensión para el acceso del agente que atienda la cola en este caso será un softphone dedicado a administrar la cola.***

10. ***Configurar una extensión para agregarse a la cola.***

11. ***Cree una sala de conferencia con el numero "4000", seguido defina una nueva extensión "4100"(en el dialplan) para que cualquier usuario se agregue a la conferencia.***

12. ***Defina una extensión para reproducir el Music on Hold.***

3.5.5 Práctica 7: Añadiendo funcionalidad a nuestra centralita utilizando AGI en tecnología de VoIP.

Objetivos

- Adaptar nuestro plan de marcación para la implementación de los distintos programas propuestos.
- Crear un sistema que permita validar a usuarios para realizar sus llamadas.

Introducción Teórica

AGI

Conforme crece el uso de Asterisk, surgen también escenarios que presentan niveles de complejidad cada vez mayor en los IVR. En dichos escenarios, las prácticas comunes en los dialplan no son útiles, y se hace necesario utilizar lógica externa. Es ahí donde entra AGI (Asterisk Gateway Interface).

La AGI provee una interfaz estándar para que programas externos puedan controlar el plan de marcación. Generalmente, los scripts AGI se utilizan para realizar lógica avanzada, comunicarse con base de datos relacionales, etc. Los lenguajes más comunes de programación de scripts AGI son: PHP, Python y Perl, aunque se puede utilizar cualquier otro lenguaje.

El intercambio de información del script con Asterisk se realiza vía los canales de comunicación STDIN (Standard Input, para obtener información), STDOUT (Standard output, para enviar información) y STDERR (para enviar información de debugging). Desde el punto de vista del script AGI, todo flujo de datos desde Asterisk se considera STDIN mientras que el envió hacia él es STDOUT. El script AGI envía comandos a Asterisk escribiendo en el STDOUT. Seguidamente Asterisk envía una respuesta por cada uno de ellos que es leída por el script.

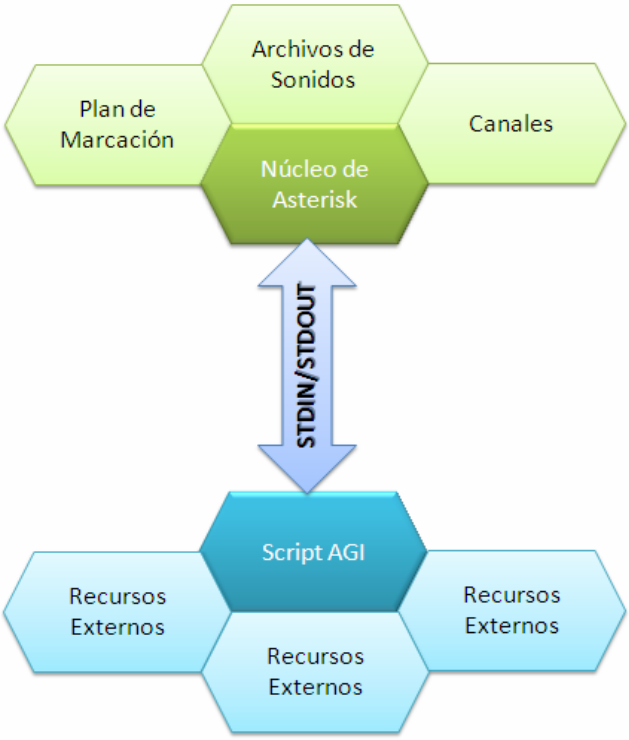

Figura 21. Funcionamiento de AGI

Para invocar un script AGI desde el dialplan hacemos (puede pasar argumentos a un script de AGI):

```
AGI(script.agi|arg1|arg2|...)
```

En las nuevas versiones de Asterisk, el formato para pasar parámetros a un script de AGI es el siguiente:

```
exten => _X.,1,AGI(miscript.agi,param1,param2,param3)
```

Como podrán apreciar es exactamente igual que invocar cualquier otra aplicación nativa en Asterisk. La mayor diferencia entre una aplicación nativa de Asterisk (ejemplo: Background) y un script AGI, es que la primera utiliza recursos internos de Asterisk mientras la última puede usar recursos externos (ejemplo: bases de datos).

Algunos ejemplos de comandos son:

ANSWER: Atiende.

HANGUP: Cuelga.

SAY [NUMBER | DIGITS | ALPHA | PHONETICS]: Dice un número, dígito, carácter o una cadena fonéticamente.

SET [CONTEXT | EXTENSION | PRIORITY]: Establece un nuevo contexto, extensión o prioridad luego de finalizada la ejecución de script.

VERBOSE: Imprime un mensaje en el log.

WAIT FOR DIGIT: Espera que se presione un dígito.

[SET | GET] VARIABLE: Asigna u obtiene el valor de una variable del plan de marcación.

El programa debe tener derechos de ejecución y presentar un intérprete válido. Ejemplo yum –y install php; chmod 755 mi_script.php. Además ser localizado por defecto en /var/lib/Asterisk/agi-bin

AGI tiene tres variantes:

EAGI

Esta variante es un poco más avanzada que AGI, permite al script AGI interactuar con el flujo de audio entrante a través de un descriptor de archivo. Fundamentalmente, EAGI se puede utilizar para crear aplicaciones que aprovechan un flujo de audio entrante, lo analizan y luego realizan tareas de acuerdo con dicho flujo de datos.

DeadAGI

AGI para poder ejecutarse requiere un canal abierto, está pensando para interactuar con el usuario. Esto significa, que si el canal sobre el cual opera un script AGI se cierra antes de completarse los comandos restantes no serian ejecutados y esto puede llegar a ser un problema. Por fortuna, existe DeadAGI precisamente salva dicha situación permitiendo la ejecución normal del script

antes o después de cerrarse el canal. Cabe resaltar, que es válido para las versiones 1.0.X y 1.2.X de Asterisk, la versión 1.4.X produce una alerta en la ejecución de DeadAGI cuando se establece el canal y aun este no ha sido respondido. Asterisk 1.6.X supuestamente incluirá un mecanismo para habilitar esta variante en AGI al usar de manera automática por lo que DeadAGI queda obsoleto.

Ahora veamos como invocar un script DeadAGI desde el dialplan de Asterisk.

```
exten => h,1,DeadAGI(some_script_name.agi,param1,param2,param3)
```

Es similar a un script AGI regular, con la diferencia que solo puede ser ejecutado por la extensión 'h'.

FastAGI — Ejecución AGI vía TCP socket

FastAGI se distingue de las demás variantes en que es ejecutado por un proceso completamente diferente a Asterisk. Mientras que AGI, EAGI y DeadAGI utilizan una comunicación STDIN/STDOUT entre Asterisk y un script FastAGI se basa en TCP socket. Entonces el script AGI, ahora servidor AGI, puede operar en un servidor completamente distinto, dando la posibilidad de crear una lógica separada del dialplan.

Un script FastAGI es invocado por el dialplan de Asterisk de la siguiente manera:

```
exten => _X.,1,AGI(agi://IP_NUMBER:PORT/some_script_name.agi)
```

Como en las otras variantes, en FastAGI podemos pasar parámetros al script pero esto varía según la versión de Asterisk.

Asterisk 1.2.X y 1.4.X

En estas versiones los parámetros son similares a un HTTP GET request:

```
exten => _X.,1,AGI(agi://192.168.2.1:1048/TestAGI?exten=${EXTEN})
```

En este caso, el servidor FastAGI debe enviar varios argumentos, parsearlos y ordenar cada uno.

Asterisk 1.6

En la versión Asterisk 1.6.X es igual que con un script AGI:

```
exten => _X.,1,AGI(agi://192.168.2.1:1048/TestAGI|${EXTEN}|{VAR2})
```

En este ejemplo, los argumentos están disponibles mediante variables AGI nombradas como agi_arg_1 and agi_arg_2 respectivamente. Aunque es posible usar la sintaxis anterior, para la versión 1.6.X de Asterisk se recomienda la nueva sintaxis por asuntos de compatibilidad.

FastAGI frameworks

Como indicamos arriba, FastAGI basa su comunicación en TCP socket, creando un ambiente cliente/servidor basado en open source por ello existen multitud de Frameworks.

Language	**Framework**	**URL**
.NET	NAsterisk	http://www.codeplex.com/nAsterisk
ActiveX	AstOCX	http://www.pcbest.net/astocx.htm
Erlang	ErlAst	http://tools.assembla.com/erlast
Python	FATS StarPy	http://fats.burus.org/ http://www.vrplumber.com/programming/starpy/
Java	Asterisk-Java	http://www.voip-info.org/wiki/view/Asterisk-java
Ruby	Adhearsion	http://www.adhearsion.com

Tabla 4. Algunos Frameworks de FastAGI

Impacto de los lenguajes de programación.

Debido a que la comunicación entre un AGI script y Asterisk se produce vía STDIN y STOUT (o TCP socket), es posible realizar la misma desde cualquier lenguaje de programación. No obstante esto, es necesario elegirlo cuidadosamente ya esto puede provocar un impacto importarte en el rendimiento provocando lentitud en el mismo.

Familia de lenguaje	**Miembros**	**Detalles**
Binarios	C, C++, Pascal	Los ejecutables pueden optimizarse enormemente, y aunque son ideales para AGI suelen ser tediosos de implementar
Máquinas Virtuales	Java, C#, Mono	Las Máquinas Virtuales por sí mismas consumen mucha memoria y esto puede ser un problema; por otro lado aunque JAVA permite el rápido desarrollo de aplicaciones es preferible limitar su uso a FastAGI
Interpretados	PERL, PHP, Python, Ruby	Los lenguajes Interpretados son mucho menos eficientes que los binarios pero consumen mucho menos memoria los basados en Máquina Virtual; mas del 80% de los script AGI están programados en PERL o PHP, y pueden utilizar de manera sencilla AGI y FastAGI

Tabla 5. El cuadro explica el impacto según la familia del lenguaje

AGI scripting frameworks

La cantidad de frameworks para el desarrollo de script AGI es sorprendente tomando en cuenta que el API de AGI cuenta con menos de veinte métodos.

Language	Framework	URL
PERL	Asterisk PERL Library	http://Asterisk.gnuinter.net/
PHP	PHPAGI	http://sourceforge.net/projects/phpagi/
Python	py-Asterisk	http://py-Asterisk.berlios.de/py-Asterisk.php
C	LibagiNow	http://www.open-tk.de/libagiNow/
.NET	MONO-TONE	http://gundy.org/Asterisk

Tabla 6. Framework para AGI script más populares.

Materiales usados

Softphone: X-lite

Vmware server

Sistema operativo CentOS

Asterisk

2 Diademas

2 Pc

php5

php5-mysql

php5-cli

mysql-server

phpagi v2.

Enunciados

1. ***Crear una base de datos para los usuarios. Donde se almacenara el número que realiza la llamada y la clave de acceso al sistema.***

2. ***Desarrollar un script en php con la función md5 para realizar la encriptación de una clave "pruebita"***

3. ***Una vez generada la tabla para los usuarios, agregar un registro de prueba con clave de usuario "pruebita" encriptado con md5 seria***

4. ***Realizar el programa para que los mismos usuarios realicen sus propios cambios de claves, de este modo, la acción quedara transparente***

5. ***Habilitar este servicio generando una nueva entrada al contexto definido para los usuarios.***

6. ***Realizar un script en php (cualquier otro lenguaje) este contendrá la lógica que procesara los nombres de usuarios y a su vez les solicitara la clave correspondiente.***

7. ***Se desea que se restrinja las llamadas a un determinado rango de horas entre las 7 de la mañana a las 6 de la tarde de lunes a sábado Crear una nueva entrada en el dialplan (extensions.conf) que verifique las horas para que cualquier numero marcado y determine si se elige validar al usuario o que realice la llamada.***

3.5.6 Práctica 8: Implementación de AMI en Asterisk

Objetivos

- Crear dialplan.
- Utilizar AMI
- Ejecutar comandos a través de sesiones remotas
- Desarrollar un AGI para la utilización del AMI

Introducción Teórica

AMI

Asterisk Manager Interface Permite que programas cliente se conecten a Asterisk mediante TCP/IP y sean capaces de ejecutar comandos y leer eventos. Por cada cosa que Asterisk realiza se generan eventos que pueden ser leídos mediante una sesión de manager, y el usuario puede tratarlos a su gusto. Además permite la ejecución de comandos, lo que proporciona la posibilidad de alterar el comportamiento de Asterisk desde un programa hecho a medida.

Generalmente, se utiliza el puerto 5038. Utiliza un protocolo en modo texto que consiste en líneas de tipo "clave: valor". Conjunto de líneas: paquete.

Funcionamiento

Para trabajar con AMI es necesario tener un usuario definido en el fichero *manager.conf*. A partir de aquí hay que establecer una comunicación, y una vez conectado y autentificado, se puede comenzar a leer los eventos o ejecutar comandos.

Activar la interfaz de Asterisk Manager mediante el establecimiento enabled=yes yes en el [general] en la sección manager.conf

```
[general]
enabled=yes

[admin] ; nombre de usuario
secret = claveadmin ; clave o contraseña
deny=0.0.0.0/0.0.0.0 ; sólo se permiten conexiones para este
permit=127.0.0.1/255.255.255.0 ; usuario vía localhost
read   =   system,call,log,verbose,command,agent,user      ;
establecen permisos (r,w,r/w)
write  =  system,call,log,verbose,command,agent,user  ;  para
cada clase (system, call, etc.)
```

Las opciones después de read y write permiten definir el tipo de comando para este usuario. Esta generosa cesión de derechos es sólo para propósitos de prueba. Los derechos en el nivel command significa que el usuario puede hacer en Asterisk (como Asterisk 1.4) cambios del dialplan atraves AMI - significa que es posible ejecutar comandos con privilegios de root utilizando System().

Después de reiniciar el Asterisk nos podemos conectar a la AMI, desde el sistema de depósito usando **telnet**

Ahora puede introducir comandos, por lo general consta de varias líneas. El tipo de paquete está dado por las siguientes claves:

Action: Paquete originado en el cliente requiriendo llevar a cabo una acción particular. Contiene el nombre de la acción y los parámetros de la misma.

Response: La respuesta del Asterisk a la Acción requerida por el cliente.

Event: Datos correspondientes a un evento generado dentro del núcleo de Asterisk o módulo.

Por ejemplo: Enviar un mensaje con acción "Login", junto con el usuario y la clave como parámetros.

```
Action: login
Username: admin
Secret: adminclave
```

```
Events: off[32]
```

Todos los paquetes de comandos están cerrados con dos retornos de carro.

Se recibirá por parte del servidor

```
Asterisk Call Manager/1.0
Response: Success
Message: Authentication accepted
```

O, en caso de error:

```
Asterisk Call Manager/1.0
Response: Error
Message: Authentication failed
```

Después de una autenticación exitosa, los paquetes pueden ser enviados en ambas direcciones. El tipo de paquete es siempre está determinada por la primera línea. El cliente envía el paquete Action, el servidor responde con Response o puede enviar paquete Event. De lo contrario, el orden de las líneas en un paquete es irrelevante.

Se pueden proveer parámetros adicionales (por ejemplo, un número a llamar o canal a desconectar). En el caso que la acción determine la ejecución de una entrada del plan de marcación, también se pueden proveer variables.

Formato:

```
Action: <action type><CRLF>
<Key 1>: <Value 1><CRLF>
<Key 2>: <Value 2><CRLF>
...
Variable: <Variable 1>=<Value 1><CRLF>
Variable: <Variable 2>=<Value 2><CRLF>
...
<CRLF>
```

[32] Indica que la conexión no recibirá eventos por parte del Asterisk.

Algunos ejemplos de acciones

Command: Ejecuta un comando (por ejemplo reload) (privilege: command, all).

Events: Controla el flujo de los eventos.

Hangup: Colgar canal (privilege: call, all).

IAXpeers: Lista los peers IAX (privilege: system, all).

ListCommands: Lista los comando disponibles del manager.

Logoff: Logoff del manager.

MailboxCount: Verifica la cantidad de mensajes en el mailbox (privilege: call, all).

MailboxStatus: Verifica el status del mailbox (privilege: call, all).

Originate: Origina llamada (privilege: call, all).

ParkedCalls: Lista las parked calls.

QueueAdd: Agrega un miembro a la cola (privilege: agent, all).

QueueRemove: Remueve un miembro de la cola (privilege: agent, all).

SIPpeers: Lista los peers SIP (privilege: system, all).

Status: Status (privilege: call, all).

EJEMPLO DE UNA LLAMADA

El cliente envía:

```
ACTION: Originate
Channel: SIP/12345
Exten: 1234
Priority: 1
Context: default
```

El cliente recibe, en caso de éxito:

```
Event: Newchannel
Channel: SIP/12345-ed8f
State: Down
CallerID:
Uniqueid: 1124982019.19157

Event: Newchannel
```

```
Channel: SIP/12345-ed8f
State: Ringing
CallerID:
Uniqueid: 1124982019.19157

Event: Newstate
Channel: SIP/12345-ed8f
State: Up
CallerID:
Uniqueid: 1124982019.19157

Event: Newexten
Channel: SIP/12345-ed8f
Context: default
Extension: 1234
Priority: 1
Application: SetVar
AppData: extension=1234
Uniqueid: 1124982019.19157

Response: Success
Message: Originate successfully queued
```

El cliente recibe, en caso de error:

```
Event: Newexten
Channel: OutgoingSpoolFailed
Context: default
Extension: failed
Priority: 1
Application: SetVar
AppData: extension=failed
Uniqueid: 1124981514.58775

Event: Hangup
Channel: OutgoingSpoolFailed
Uniqueid: 1124981514.58775
Cause: 0

Response: Error
Message: Originate failed
```

PROBLEMAS

La documentación sobre el protocolo y la funcionalidad del manager está incompleta.

Tiene problemas con el manejo de varias conexiones a la vez (> 5). Es recomendada la utilización de un Proxy (por, ejemplo ProxyAstMan), para sistemas que hagan un uso intensivo del manager, como pueden ser sistemas de monitoreo y campaña telefónica.

Si cree que este tipo de carga, vale la pena considerar un AMI proxy, como el "Administrador de Asterisk simple Proxy"[33] (un script en Perl), que puede manejar muchas conexiones y los bloques en una sola conexión. Esto es completamente transparente para el script de acceso a la AMI.

Por supuesto, están más interesados en la automatización de esta interacción con los scripts. El Administrador no es precisamente famoso por su habilidad para manejar múltiples conexiones simultáneas con gracia (aunque esto ha mejorado enormemente en la versión 1.4).

AJAM

De la mano de Asterisk 1.4 viene AJAM (Asyncronous Javascript Asterisk Manager), un nuevo manager, que permite conectar con Asterisk por medio de HTTP. Para poder trabajar con AJAM es necesario configurar los ficheros *manager.conf* y *httpd.conf*.

Ejemplos de funcionamiento:

Para ver lo que devuelve el comando “status” del manager puedes ejecutar (en una sola línea):

[33] http://www.popvox.com/simpleproxy.pl

```
http://IP_de_Asterisk:8088/Asterisk/manager?action=login&user
name=nombre_de_usuario&secret=contraseña
```

Esto abrirá una sesión de Asterisk Manager. Ahora ejecuta:

```
http://lIP_de_Asterisk:8088/Asterisk/rawman?action=status
```

Observa la salida del comando.

Materiales usados

Softphone: X-lite

Vmware server

Sistema operativo CentOS

Servidor Asterisk

2 Diademas

2 Pc

Enunciados

1. ***Configurar un usuario en el manager y permitir el acceso desde el localhost.***

2. ***Recargar la configuración en el CLI.***

3. ***Verificar el usuario creado.***

4. **Listar los comandos permitidos por el cli del Asterisk.**

5. ***Tomar cuatro de los comandos y ver sus descripciones.***

6. ***Interacción vía telnet con el manager.***

7. ***Ejecutar un login con el usuario recientemente creado.***

8. ***Listar los canales activos.***

9. ***Listar los peers IAX y SIP.***

10. ***Ejecutar un reload de la configuración.***

11. ***Ejecutar un logoff vía telnet.***

12. ***Interacción vía PHP.***
    a) ***Repetir las mismas acciones anteriores, utilizando un archivo en php que ejecute todos los comandos ejecutados a través de telnet.***

b) Realizar una llamada en algún canal SIP / IAX y que reproduzca music on hold al responder la llamada, seleccione su canción favorita.

CONCLUSIONES

El objetivo central de esta monografía es el estudio de la Voz sobre IP, una nueva tecnología en el mundo de las telecomunicaciones, para su implementación utilizamos un servidor Asterisk. De los temas abordados inferimos lo siguiente:

La PSTN a pesar del cambio de centrales manuales a analógicas y luego a digitales, unas cuantas mejoras a las terminales y nuevas aplicaciones ha mantenido prácticamente el mismo principio desde su invención, la conmutación de circuitos para el establecimiento de una llamada. Las aplicaciones de VoIP, como ya se mencionó, son ilimitadas y apenas se empiezan a implementar, en contraste con las aplicaciones de la PSTN que llegan a su tope tecnológico.

Los estándares, tales como el H.323, SIP e IAX, fueron establecidos para mantener la operación de la red IP multi-proveedor y multi-fabricante en contraste con la naturaleza propietaria de la PSTN.

Para la implementación de esta tecnología se necesitó de Asterisk, una solución probada y flexible que nos permitió garantizar el control del proceso de aprendizaje y desarrollar prácticas que finalmente se adaptaron a nuestras necesidades. Las funcionalidades aquí configuradas son sólo una pequeña parte de los servicios que Asterisk permite, y de la riqueza de posibilidades que esta presenta, tanto en módulos opcionales como el uso de AGI, AMI, IVR, como en módulos de terceros como los programas externos para el manejo de tarificación de llamadas, así como en hardware.

RECOMENDACIONES

Por lo antes expuesto y en base a nuestras conclusiones se recomienda:

Promover el uso e implementar la telefonía de voz sobre IP en base a Asterisk en todos los sectores públicos (gobierno, empresarial, educativo, etc.) por su accesibilidad, bajo costo y servicios adicionales.

Las facultades tecnológicas agreguen a su pensum académico una asignatura que incluya el manejo de Asterisk.

Complementar este trabajo con administradores gráficos de la PBX, el uso de Asterisk, estos se configuran sin necesidad de conocer Linux ni Asterisk de manera avanzada.

- Trixbox, es una de las distribuciones de Asterisk bajo Linux, basada en CentOS (instala sólo los componentes básicos), de sencilla y rápida instalación
- FreePBX se utiliza como entorno gráfico de configuración de la PBX.

.

ANEXOS

MANUAL DE INSTALACION

INTRODUCCCION

Se presenta dentro de esta sección una serie de pasos que le permitirá la instalación de Asterisk en el sistema Operativo CentOS 5.2, distribución de estándar abierto y código libre.

Es muy difícil imaginar un desarrollo sostenible sin transferencia de conocimiento y reapropiamiento tecnológico. Una solución basada en estándares abiertos y código libre no es sólo una buena solución desde un punto de vista puramente técnico sino que además permite la posibilidad de adaptación a la realidad local.

Las características del computador utilizado: procesador Intel Core Duo T2350 con 1 GB en RAM, 120 Gb de espacio en disco duro y una tarjeta de red 3com. Para la instalación se utilizo VMWare Server Console Versión 1.0.8 build 126538. Se creó una máquina virtual con 8 Gb de disco.

Para la instalación sobre una computadora con otro sistema operativo es buena recomendación respaldar y borrar el disco, aunque de todas formas puede convivir con otro Sistema Operativo (este no es el caso).

QUÉ PASOS DEBO SEGUIR ANTES DE INSTALAR LINUX

Para aquellas personas nuevas al mundo Linux, debe tener en cuenta dos puntos muy importantes: ***Linux como Sistema Operativo:*** posee programas diseñados específicamente para su ambiente y un sistema de archivos propio, entre otras variantes. Lo anterior, significa que debe **ajustar su PC o Servidor a nivel de disco duro** para poder realizar su instalación, no será capaz de instalar Linux estando dentro de Windows como lo haría para cualquier otro programa.

Existen diversas distribuciones de Linux

Linux como tal es el núcleo del sistema operativo, lo que técnicamente es conocido como el Kernel, sin embargo, sobre este mismo componente han evolucionado una serie de distribuciones que intentan satisfacer distintos grupos o usuarios, algunas de estas distribuciones son: Debian, Gentoo, Knoppix, Mandrake, Fedora, Red Hat y Slackware, entre muchas más; cada una de estas variantes incluye ciertas opciones como lo serían programas o ambientes gráficos, cada distribución puede que resulte más atractiva para ciertas tareas que otra.

Liberación de Espacio / Generación de Particiones

Dependiendo de la distribución y los componentes que desee instalar, es necesario que libere espacio en su disco duro que puede variar desde 300 MB

hasta 3 GB. El proceso para liberar espacio es conocido técnicamente como particionar o generación de divisiones lógicas. Salvo haya comprado su disco duro por separado, éste ya vendrá pre-particionado de fabrica dependiendo del vendedor.

Antes de incursionar en cualquiera de los métodos de partición aquí descritos, tome en cuenta que este proceso es considerado de alto riesgo ya que se realiza a un nivel muy bajo de software; modificaciones erróneas de particiones pueden hacer su disco duro inaccesible o pueden eliminar toda la información que en él reside.

Herramienta comercial para Windows

Esta opción es la más amigable para liberar espacio si su PC o Servidor ya contiene una instalación activa de Windows. Aunque este tipo de herramientas tienen un costo, le permitirán redimensionar el espacio de su disco duro empleando herramientas gráficas. Este proceso simplemente implica modificar la geometría del disco duro, dejando el suficiente espacio libre para su distribución Linux.

Utilería de instalación Linux

Si no tiene instalado Windows o simplemente desea un proceso más directo, puede optar por utilizar las herramientas incluidas en su misma distribución Linux. Al momento de inicializarse su PC, si realiza el "Boot" de los mismos discos de la distribución (algunas variantes de Linux) detectarán que no existe espacio suficiente en disco duro y lo redireccionarán a utilerías especificas para llevar a cabo el proceso de partición.

Este mecanismo, aunque más directo que utilizar una herramienta dentro de Windows puede resultar más complejo por la interfaz o comandos que deben utilizarse, no obstante, logra los mismos resultados.

SISTEMA BASE: CENTOS 5.2

Introducción

CentOS (Community ENTerprise Operating System) es un clon a nivel binario de la distribución Linux Red Hat Enterprise Linux RHEL, compilado por voluntarios a partir del código fuente liberado por Red Hat. Su núcleo es monolítico.

Red Hat Enterprise Linux se compone de software libre y código abierto, pero se publica en formato binario usable (CD-ROM o DVD-ROM) solamente a suscriptores pagados. Como es requerido, Red Hat libera todo el código fuente del producto de forma pública bajo los términos de la Licencia pública general de GNU (GPL) y otras licencias.

Los desarrolladores de CentOS usan ese código fuente para crear un producto final que es muy similar al Red Hat Enterprise Linux y está libremente disponible para ser bajado y usado por el público, pero no es mantenido ni asistido por Red Hat.

Trabajos derivados
Asterisk@Home está basado en CentOS.
Trixbox está basado en CentOS.

Por qué CentOS para el sistema operativo
Es una de las distribuciones preferidas por los proveedores de Internet a nivel mundial ofrece servicios tales como: Servidor web Apache (SSL), Servidor de correo Postfix con SMTP-AUTH y TLS, Servidor DNS BIND, Servidor FTP ProFTP, Servidor MySQL, Servidor POP3/IMAP, Firewall, etc.

Instalación

La instalación requiere de lo siguiente:

Requisitos de Hardware

- Memoria RAM: 192 MB (mínimo).
- Espacio en Disco Duro: 1 GB (mínimo) - 2 GB (recomendado).
- Procesador: CentOS soporta casi las mismas arquitecturas que Red Hat Enterprise Linux: Intel x86-compatible (32 bit) (Intel Pentium I/II/III/IV/Celeron/Xeon, AMD K6/II/III, AMD Duron, Athlon/XP/MP). Advanced Micro Devices AMD64 (Athlon 64, etc) e Intel EM64T (64 bit). También se tiene soporte para dos arquitecturas no soportadas por Red Hat Enterprise Linux: Alpha procesador (DEC Alpha), SPARC.

Requisitos software:

- Bajar el DVD de CentOS 5.2[34] o los 6 CDs de CentOS 5.2 CDs de un servidor espejo cerca a tu país (la lista de servidores espejo esta aquí: http://isoredirect.CentOS.org/CentOS/5/isos/i386/).
- Una conexión a Internet rápida.

Instalación mínima de CentOS
Instalación tipo CUSTOM:
Idioma inglés.
Desactivar SELinux.
Seleccionar paquete Base.
Por último, actualizar los paquetes con:

```
yum[35] update
```

[34] La última versión fue liberada el 24 de junio de 2008, para las arquitecturas i386 y x86_64.
[35] CentOS usa yum para bajar e instalar las actualizaciones, herramienta también utilizada por Fedora.

Proceso de instalar y configurar un servidor en CentOS de modo gráfico

Arranque de CD-ROM: Primeramente debe asegurarse que su BIOS[36] se encuentre configurado para leer el CD-ROM al momento de inicializarse su PC o Servidor, esto es necesario para que se ejecute el "shell" de instalación CentOS en lugar del sistema operativo que tiene instalado en su disco duro.

Arranque de CentOS: Inicialice su sistema con el CD de CentOS 5.2 CD (CD 1) o el DVD colocado en su lector óptico. Si configuró correctamente su BIOS para leer CD-ROM's al arranque.

Figura 22. Proceso inicial de Instalación de CentOS 5.2

Proceso inicial y detección de Hardware: Simplemente esperando u oprimiendo Enter, iniciará el proceso para detección de Hardware y los controladores apropiados para el mismo. Durante este proceso de instalación se abrirá una consola gráfica, esta secuencia puede durar entre 10 o 15 segundos, mientras observa mensajes descriptivos sobre su sistema. Si desea realizar su proceso de instalación mediante línea de comandos, entonces deberá introducir la secuencia linux text.

Figura 23. Verificación de medios

[36] El acceso al BIOS varía dependiendo del Hardware que utilice, sin embargo, las opciones más comunes son las teclas ESC, F2 o DEL.

Verificación de medios (CD-ROM's): La opción de realizar una prueba de integridad sobre los CD-ROM's de instalación CentOS, esta prueba dura entre 10 y 15 minutos para los 6 CD's de instalación. Si no desea realizar esta prueba de errores físicos en el CD o DVD que has grabado, asumimos que está libre de defectos, seleccione la opción “Skip" (Saltar o Pasar).

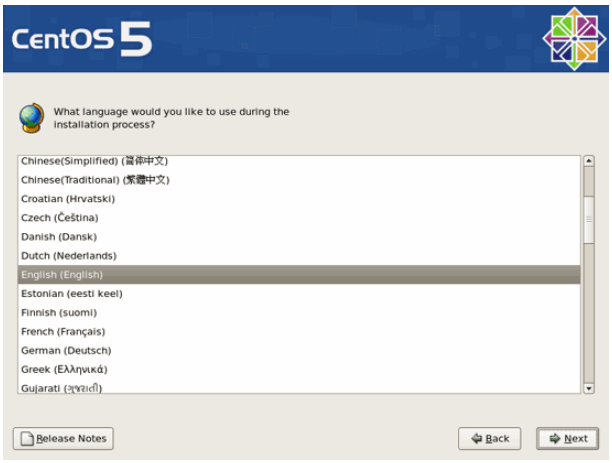

Figura 24. Selección del lenguaje de la instalación

La pantalla de bienvenida de CentOS, presiona "Next" (siguiente). Escoge el idioma, por cuestiones de compatibilidad con caracteres, sugerimos el idioma de tu teclado.

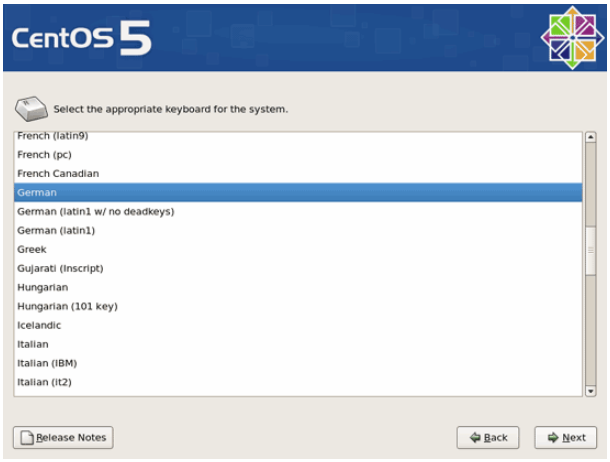

Figura 25. Selección del teclado de la instalación

Selecciona el tipo de teclado, (español) o ingles dependiendo de qué tipo de teclado uses.

Recuerde: En este manual la instalación es desde cero, es decir, que solo será usado como servidor dedicado y que no será usado para otro sistema operativo o sea Windows, por lo cual podemos eliminar todas las particiones del disco duro, para instalar CentOS.

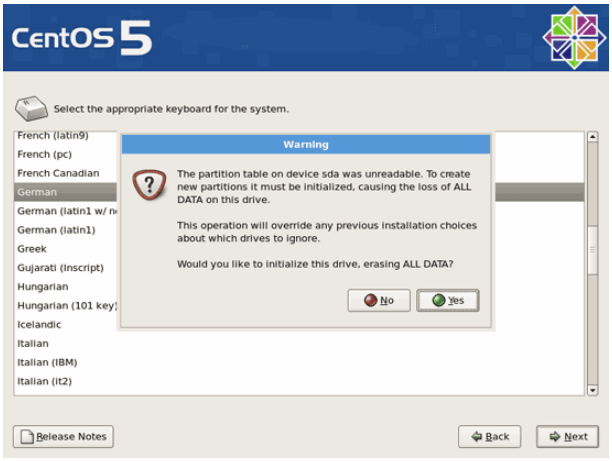

Figura 26. Confirmar la instalación en nuestro disco duro

Al seleccionar el tipo de particionamiento que debes usar, en la mayoría de casos la propuesta por defecto está bien, sin embargo para otro tipo de instalaciones especificas se requiere otro tipo de partición.

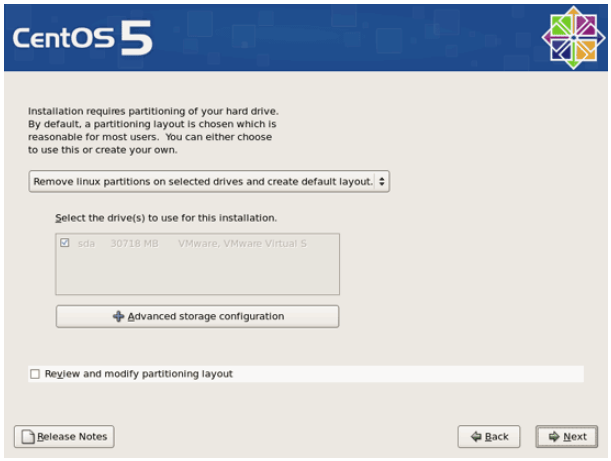

Figura 27. Selección del tipo de particionamiento

Confirma que estás de acuerdo con eliminar las particiones existentes.

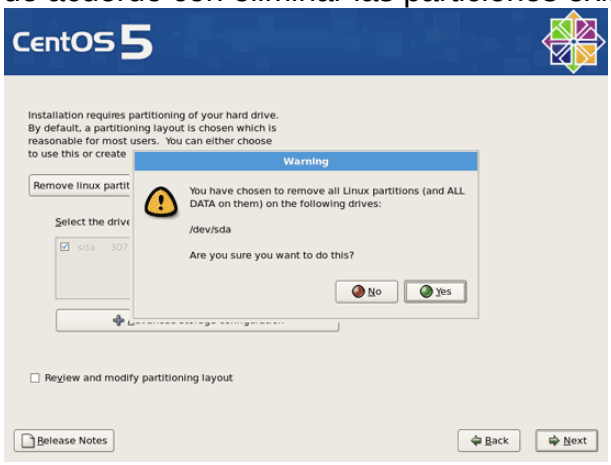

Figura 28. Confirmar la eliminación de particiones existentes

Luego se configura la red, por defecto CentOS propone DHCP, pero debido a que este va a ser un servidor dedicado seria convenientes que le configures una IP de tu red interna. Presiona "Edit".

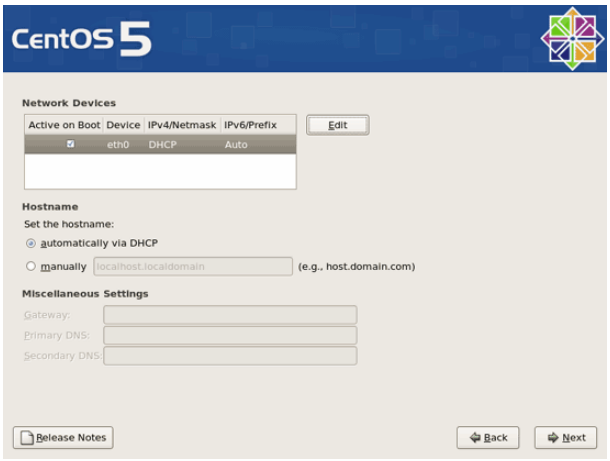

Figura 29. Configuración de la red

En la ventana que aparece configura los parámetros de red de acuerdo a tu red interna.

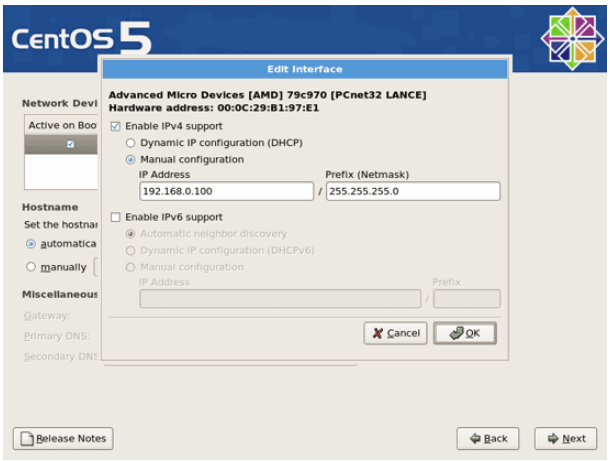

Figura 30. Edición de la configuración de la red del servidor

Configura el nombre de tu servidor, para este caso hemos escogido server1.example.com, configura la puerta de enlace y coloca los DNS, para este efecto usaremos los de opendns (http://www.opendns.org) 208.67.220.220 y 208.67.222.222 respectivamente.

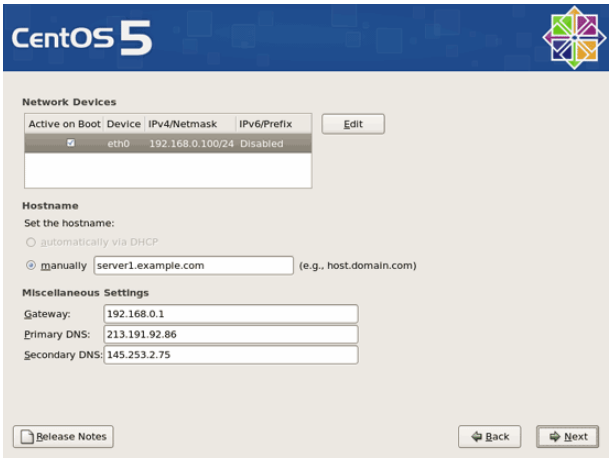

Figura 31. Asignar el DNS al servidor

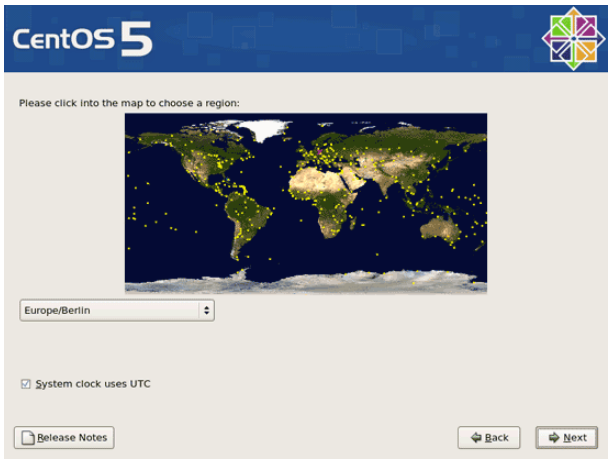

Figura 32. Selecciona tu zona horaria.

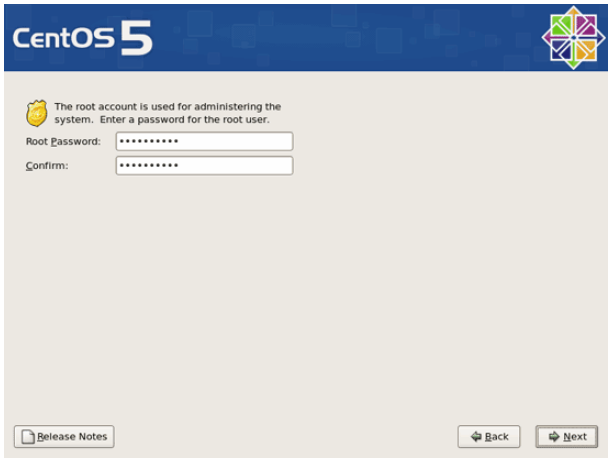

Figura 33. Selecciona un password de 8 dígitos entre números y letras.

Seleccionamos que tipo de instalación queremos realizar, en este caso seleccionamos "Server" (quítale el check a las demás casillas) adicionalmente no es necesario darle check a "Packages from CentOS Extras", Luego dale "Customize Now" (Personalizar ahora) y de ahí "Next" o Siguiente.

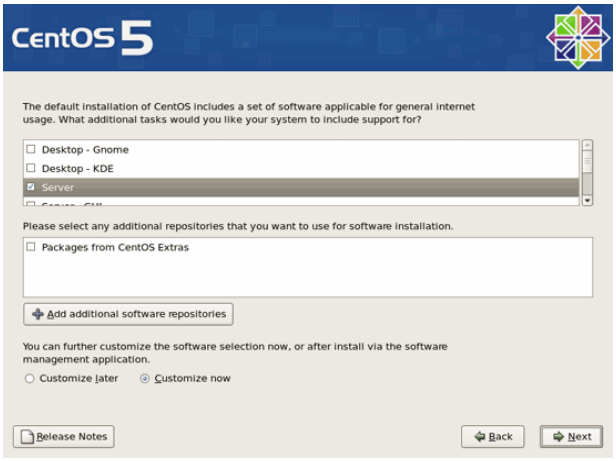

Figura 34. Selecciona el tipo de instalación

Debemos seleccionar los grupos de paquetes que queremos instalar. Selecciona: Editors, Text-based Internet, DNS Name Server, Mail Server, Server Configuration Tools, Web Server, Administration Tools, Base (Deselecciona otros grupos de paquetes) y dale click a "Next" o Siguiente.

Figura 35. Selecciona el grupo de paquetes a instalar.

Figura 36. El instalador revisa las dependencias.

Figura 37. Presiona "Next" para iniciar la instalación.

Figura 38. El disco duro está siendo formateado.

Figura 39. La instalación comienza, tomara unos minutos.

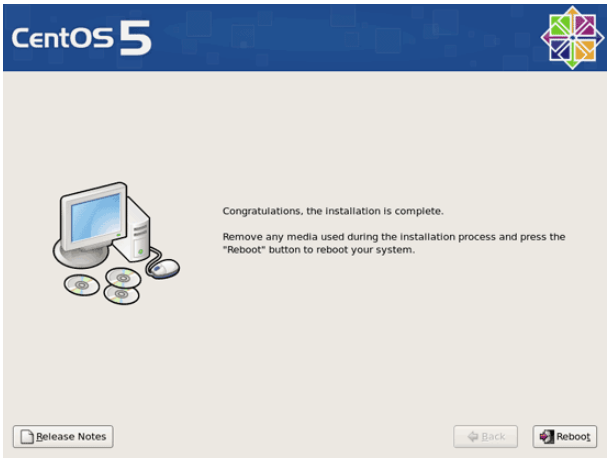

Figura 40. Instalación completada, quitar el CD o DVD y reiniciar el sistema.

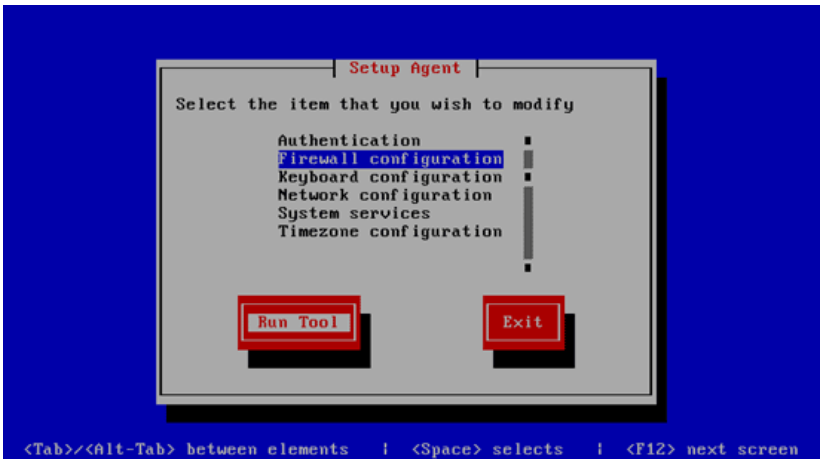

Figura 41. Ejecución de herramientas configurables

Después de reiniciar el servidor, veras "Firewall Configuration" (Configuración de Firewall), presiona "Run Tool" (Ejecutar herramienta).

Debido a que se instalaran medidas de seguridad a medida de la implementación, deshabilitaremos el firewall, por ahora será mejor deshabilitado debido que el mismo firewall podría interferir con la configuración e instalación de otras aplicaciones. Deshabilitar también SELinux (Security Enhanced Linux) es una extensión de seguridad de CentOS que provee seguridad adicional al sistema. La implementación de SELinux en Red Hat Enterprise Linux está diseñada para mejorar la seguridad de varios demonios de servidor y reducir al mínimo el impacto en el día a día las operaciones de su sistema.

Figura 42. Configurar el firewall y SELinux

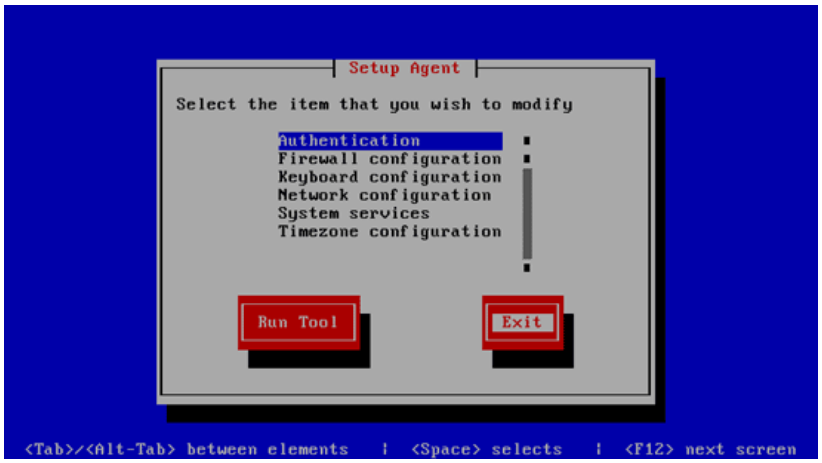

Figura 43. Luego selecciona "Exit" o Salir.

INSTALANDO ASTERISK

Como cualquier programa libre, existen dos formas principales de instalarlo. El primer método consiste en descargar el código fuente de la red y compilar su propia versión binaria. El segundo método consiste en descargar una versión ya compilada en forma de paquete. Si decide compilar Asterisk desde su código fuente los siguientes consejos le pueden ser de utilidad:

1. Descargue el código fuente de Asterisk de http://www.Asterisk.org, se da click en la pestaña Download ahí se podrán ver los mas recientes release, actualmente la versión 1.4 y 1.6

2. Para una versión básica no necesita bajar los paquetes de “addons” o “sounds”.

3. Para poder compilar Asterisk desde el código fuente es necesario tener un entorno de compilación en su sistema.

Asegúrese de que tiene los siguientes paquetes instalados:

kernel-headers (cabeceras del núcleo del kernel)
kernel-devel (fuentes del kernel)
bison y bison-devel (un generador de analizadores sintácticos)
zlib y zlib-devel (bibliotecas de compresión – desarrollo)
ncurses, ncurses-devel (bibliotecas de utilidades de consola - desarrollo)
openssl, openssl-devel (libssl-devel) (SSL – bibliotecas de desarrollo)
libc6-devel (cabeceras y bibliotecas de desarrollo GNU C)
gcc y make (el compilador C de gnu y la utilidad make)
gnutls-devel
gcc-c++
libtermcap-devel

Comandos de utilidad:
Para comprobar si están es

```
rpm –q [nombre del paquete]
```

Si alguno de estos paquetes faltase lo instalamos con

```
yum install nombre-del-paquete
```

Si se desea soporte para Realtime con Mysql

ODBC y MySQL.

```
yum -y  install mysql-server mysql-devel newt-devel unixODBC
unixODBC-devel   libtool-ltdl   libtool-ltdl-devel   mysql-
connector-odbc
```

La compilación de Asterisk no es diferente de otro programa de código libre en Linux:

- Compilar: # make
- Instalar: # make install
- Instalar los “scripts” de arranque: # make config
- Instalar los archivos de configuración de ejemplo: # make samples
- Instalar la documentación de desarrollo: # make progdocs

Si quiere usar una tarjeta Digium Wildcard(tm) con Asterisk

Va a tener que compilar e instalar un controlador llamado zaptel[37] (módulo del kernel):

[37] El propietario de Zaptel hace un par de años le comunicó a Digium que la marca estaba registrada, casualmente por una empresa dedicada a la telefonía (una empresa de venta de tarjetas para llamar por teléfono) y en parte se quejaba de que cuando alguien buscaba por término ‘Zaptel’ no aparecía su página y en cambio sí que aparecen otras tarjetas más famosas.

- Descargue el código fuente del Zaptel de http://www.Asterisk.org. Por desgracia, el controlador de zaptel no forma parte del núcleo (kernel) de Linux y tiene que crear sus propios módulos.

Información importante DAHDI (antes llamado Zaptel)

Se a *"renombrado"* de todo el paquete Zaptel[38] y se han realizando ciertas modificaciones bastante llamativas de manera que tendrá todas las funcionalidades de la versión Zaptel 1.4 y dejará de darse soporte para kernels de Linux 2.4 y sistemas de gestión de dispositivos DevFS (en favor del uDev), por lo que los drivers actuales de Zaptel pasarán a estar desfasados y no seguirán recibiendo actualizaciones una vez lanzada la versión DAHDI[39] 2.0.0.

Asterisk 1.2 únicamente será compatible únicamente con Zaptel.
Asterisk 1.4 tendrá que adaptarse a utilizar tanto el paquete Zaptel como el nuevo DAHDI.
Asterisk 1.6 únicamente soportará DAHDI.

Ahora nos encontramos con estos cambios:
Para empezar, los módulos de las tarjetas se encuentran en el directorio:

```
cd /lib/modules/2.6.XX-XXX/dahdi
```

Por otra parte, una vez instalado DAHDI habrá que iniciarlo manualmente con:

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

Luego tenemos los siguientes comandos:

```
dahdi_hardware
```

Es para detectar el tipo de tarjeta de que estamos usando, al mismo tiempo para ver si la reconoce.

```
dahdi_cfg -v
```

Es para verificar que el archivo de configuración está correcto, y por consecuencia muestra los canales bien configurados. Este comando hace lo que hacia el ztcfg -v

Pero no todo es virtud para DAHDI ya que todavía tienes unos problemas, ejemplo: no hay un comando para detener el demonio dahdi, como antes lo tenía zaptel. Esto es muy útil si queremos apagar el servicio dahdi sin que se nos queden canales bloqueados.

[38] Digium pondrá a disposición de todos los usuarios una página de información para lograr que la migración de Zaptel a DAHDI sea lo más cómoda posible. http://www.asterisk.org/zaptel-to-dahdi
[39] DAHDI: Digium Asterisk Hardware Device Interface.

Finalmente tenemos dos opciones o posibilidades, de acuerdo a la estructura de configuración de estas últimas versiones.

Si usted utiliza Zaptel: Con esta opción Asterisk 1.4.22 o superior, no traerá por defecto zapata.conf por lo se tiene que crear tomando como base el archivo chan_dahdi.conf aunque Asterisk seguirá buscando el archivo zapata.conf

Si usted utiliza DAHDI: Con esta opción Asterisk 1.4.22 o superior, se deberá configurar en /etc/dahdi/system.conf con una configuración prácticamente igual a la del zaptel.conf y seguidamente /etc/Asterisk/chan_dahdi.conf para definir los canales que Asterisk va a utilizar.

Está claro que de ahora en adelante, DAHDI va a tener que hacerse paso en medio de lo que queda de Zaptel, al final Zaptel irá perdiendo soporte. El cambio es irreversible y solo queda actualizarse y detectar los bug que pueda tener DAHDI, esperando que se vaya mejorando a medida que pasa el tiempo.

Descargando Asterisk

También es posible descargar Asterisk ya compilado. El programa compilado (binario ejecutable) se obtiene en la forma de “paquete.” Un paquete contiene todos los archivos necesarios para ejecutar un programa. Dependiendo de la distribución de Linux que esté usando existen paquetes en distintos formatos (rpm, deb o tgz). Si está usando una distribución basada en Debian, puede descargar e instalar Asterisk usando la utilidad “apt”. Ejecute la orden apt-get install con los siguientes paquetes:

Paquete Descripción

Asterisk-classic (obligatorio) PBX en código libre – versión original de Digium
Asterisk-config (sugerido) archivos de configuración para Asterisk
Asterisk-dev (opcional) archivos de desarrollo para Asterisk
Asterisk-doc (sugerido) documentación para Asterisk
Asterisk-sounds-extra (opcional) archivos adicionales de sonido para Asterisk
Asterisk-sounds-main (opcional) archivos de sonido para Asterisk

Al día de hoy, no existe una versión binaria (compilada) del controlador del núcleo “zaptel”. No tiene más opción que seguir el método descrito en la sección anterior. Descargue el código fuente del controlador del núcleo (zaptel kernel drive) y cree el módulo con la utilidad make y make install. No olvide que antes de compilar el controlador necesita tener instaladas las cabeceras del núcleo (kernel) de Linux.

zaptel (obligatorio) Utilidades para Zaptel
zaptel-source (obligatorio) Código fuente del controlador del núcleo Zaptel linux-headers-2.6.15-25-386 (depende su distribución)

El proyecto Asterisk se divide en las siguientes partes principales:

Asterisk: núcleo del sistema.
Asterisk-addons: módulos adicionales que incluyen soporte de almacenamiento de detalle de llamadas en base de datos.
libpri: librería para gestionar enlaces ISDN con tarjetas digitales.
Zaptel: módulos y herramientas.
dahdi-linux: módulos del kernel para acceder a tarjetas de comunicaciones para líneas analógicas.
dahdi-tools: herramientas para configurar y diagnosticar las tarjetas de hardware.
Versión estable y de desarrollo (stable, head)

Los archivos necesarios se descargan en /usr/src:

Descargar los módulos necesarios: zaptel, libpri, Asterisk y Asterisk-addons desde los repositorios públicos de Digium y guardarlos en nuestro sistema, así:

```
cd /usr/src
wget -c http://downloads.digium.com/pub/Asterisk/Asterisk-1.4-current.tar.gz
wget -c http://downloads.digium.com/pub/Asterisk/Asterisk-addons-1.4-current.tar.gz
wget -c http://downloads.digium.com/pub/libpri/libpri-1.4-current.tar.gz
wget -c http://downloads.digium.com/pub/zaptel/zaptel-1.4-current.tar.gz
wget -c http://downloads.digium.com/pub/telephony/dahdi-tools/dahdi-tools-current.tar.gz
wget -c http://downloads.digium.com/pub/telephony/dahdi-linux/dahdi-linux-current.tar.gz
wget -c http://downloads.digium.com/pub/telephony/dahdi-linux-complete/dahdi-linux-complete-2.0.0+2.0.0.tar.gz
```

dahdi-linux 2.0.0: Este paquete contiene los módulos del kernel necesarios para poder utilizar las tarjetas de comunicaciones.

dahdi-tools 2.0.0: Este paquete contiene las aplicaciones necesarias para cargar la configuración, hacer pruebas a algunas tarjetas y otras cosas herramientas para hacer comprobaciones.

dahdi-linux-complete 2.0.0 + 2.0.0: Este paquete es la unión de los dos anteriores, se hizo con la intensión de no tener que descargar dos paquetes independientes.

Ahora debemos averiguar la versión exacta de nuestro kernel:

```
[root@Asterisk~]# uname -a
Linux Asterisk.Asteriskclub.org 2.6.18-8.el5 #1 SMP Thu Mar 15 19:57:35 EDT 2007 i686 i686 i386 GNU/Linux
```

Apuntamos si la versión del kernel es 2.4 o 2.6.

CentOS no instala las fuentes del kernel en el directorio /usr/src/linux como esta escrito en los Makefile de Zaptel y Astersik, para ello nos vamos a /usr/src/ y creamos un link llamado linux hacia el directorio con las fuentes del kernel el cual esta ubicado en /usr/src/kernels/mi_version_del_kernel

```
[root@Asterisk ~]# cd /usr/src/
[root@Asterisk src]# ln –s kernels/2.6.18-8.el5-i686/ linux
```

A continuación debe descomprimirse el código fuente:

```
tar -zxf Asterisk-1.4-current.tar.gz
tar -zxf Asterisk-addons-1.4-current.tar.gz
tar -zxf libpri-1.4-current.tar.gz
tar -vxzf zaptel-current.tar.gz
tar -zxf dahdi-linux-current.tar.gz
tar -zxf dahdi-tools-current.tar.gz
tar xvzf dahdi-linux-complete-2.0.0+2.0.0.tar.gz
```

Make: compilar todos los archivos fuente necesarios.

```
cd dahdi-linux-current
make
make install

cd dahdi-tools-current
make clean
./configure
make
make install
make config

cd dahdi-linux-complete-2.0.0+2.0.0
make
make install
make config

cd ../libpri-1.4.7 (opt)
make clean
make
make install

cd ../zaptel-1.4.2.1
y ejecutamos
make clean
make
make install
```

```
make config (para hacer q el modulo zaptel se cargue al
tiempo de boot)

cd ../Asterisk-1.4.22
make clean
./configure
make menuconfig (opt)
make install
make samples
make config

cd ../Asterisk-addons-1.4.7
make clean
./configure
make menuselect (opt)
make install
make samples
```

Opcionalmente podemos poner a nuestra central soporte para codec g729, descargando una versión opensource de este codec.

```
cd /usr/var/lib/Asterisk/modules
#revisar el codec correspondiente al procesador en
http://Asterisk.hosting.lv/
wget    http://Asterisk.hosting.lv/bin/codec_g729-ast14-gcc4-
glibc-core2.so
mv codec_g729-ast14-gcc4-glibc-core.so codec_g729.so
chmod +x codec_g729.so
```

Directorios importantes

/etc/Asterisk: Que contiene todos los archivos de configuración de Asterisk

/usr/lib/Asterisk/modules: Todos los módulos que podremos usar en la centralita

/var/lib/Asterisk: AGIs, voces, música en espera. Que a su vez contiene las siguientes carpetas

- ***agi-bin/*** donde poner nuestros script
- ***firmware/*** donde van los firmware de las tarjetas en uso
- ***images/*** donde las aplicaciones irán a buscar las imágenes cuando se comuniquen con canales que suportan esta función.
- ***keys/*** claves públicas y privadas que Asterisk puede necesitar para conectarse con otros servidores o servicios (Ej.: dundi)
- ***mohmp3/*** carpeta con nuestros archivos mp3 que podremos usar como música en espera si hemos instalado Asterisk-addons
- ***sounds/*** contiene todos las voces que podremos usar en nuestro dialplan
- ***moh/*** la música en espera que viene con la instalación de Asterisk (no mp3)

licences/ las licencias de uso que hemos comprado (Ej.: para el codec g729)

/var/spool/Asterisk: voicemail, llamadas programadas (despertador). Que contiene:

dictate/ donde se guardaran los archivos audio creados con la aplicación Dictate()

meetme/ donde se guardarán las eventuales grabaciones de la conferencias

monitor/ si se graba una llamada esta es la carpeta donde se guardarán los archivos audio

outgoing/ donde hay que mover los archivo de llamadas (call files). Véase el archivo callfiles.txt en la carpeta doc de nuestra distribución Asterisk

system/ carpeta para archivos temporáneos creados por la aplicación System()

tmp/ carpeta donde se guardan los archivo temporáneos creados por algunas aplicaciones (Ej. contestador)

voicemail/ donde se guardaran los archivos audio de los correo de voz dejados en el contestador y los mensajes audio personalizados de cada usuario

/var/run que contiene la ID del proceso de Asterisk cuando esté corriendo

/var/log donde se guardaran todos los registros de Asterisk (llamadas, mensajes y eventos). De revisar cuando se tengan problemas con la centralita.

/var/log/Asterisk/cdr-csv: detalle de llamadas

Verificación de la instalación

Binario Asterisk: ls /usr/sbin/Asterisk
Módulos de Asterisk: ls /usr/lib/Asterisk/modules
Voces pregrabadas: ls /var/lib/Asterisk/sounds
Voces pregrabadas en castellano: ls /var/lib/Asterisk/sounds/es
Archivos de configuración de ejemplo: ls /etc/Asterisk/ *.conf

Iniciar Asterisk

Debería estar levantado el daemon de zaptel:

```
service dahdi start
```

En CentOS, como servicio, el arranque de Asterisk se hace con el comando:

```
service Asterisk start
```

Si todo ha salido bien, se está en condiciones de comenzar la configuración de Asterisk.

Deberíamos poder cargar Asterisk con:

```
Asterisk –vvvvvvvvcg
```

Manual de Instalación de Sonidos Asterisk 1.4

Para instalar los sonidos en castellano, basta descomprimir los paquetes core y extra dentro de la carpeta /var/lib/Asterisk/sounds.

```
cd /var/lib/Asterisk/sounds/
wget  http://www.asterio.com.ar/resources/downloads/ThaisaC-core-sounds-sln-1.4.12.tar.gz
wget  http://www.asterio.com.ar/resources/downloads/ThaisaC-extra-sounds-sln-1.4.12.tar.gz
 tar xzvf ThaisaC-core-sounds-sln-1.4.12.tar.gz
 tar xzvf ThaisaC-extra-sounds-sln-1.4.12.tar.gz
```

Si es la primera vez que se instalan sonidos en castellano, es necesario configurar el parámetro "language" en el archivo /etc/Asterisk/zapata.conf en "es".

```
;
; Zapata telephony interface
;
; Configuration file

[trunkgroups]

[channels]

language=es
defaultzone=es
context=from-zaptel
signalling=fxs_ks
...
```

Luego de haber editado zapata.conf, reiniciar Asterisk.

```
# amportal restart
```

Otras voces en español

```
wget   -c   http://www.voipnovatos.es/voces/voipnovatos-core-sounds-es-gsm-1.4.tar.gz
wget   -c   http://www.voipnovatos.es/voces/voipnovatos-extra-sounds-es-gsm-1.4.tar.gz
tar    xzf    voipnovatos-core-sounds-es-gsm-1.4.tar.gz    -C /var/lib/Asterisk/sounds/
tar    xzf    voipnovatos-extra-sounds-es-gsm-1.4.tar.gz    -C /var/lib/Asterisk/sounds/
```

GLOSARIO

Ancho de banda: Indica la cantidad de información por segundo que se puede transmitir por un enlace o interfaz determinado en una transmisión es analógica.

Correo de Voz: Es una de las aplicaciones mas (beloved) del sistema telefónico, esencialmente graba un mensaje de voz para un destinatario que no está presente en el momento que se realiza la llamada.

Fax sobre IP (FoIP): Servicios de transmisión de fax prestados sobre redes IP.

Foreign eXchange Office (FXO): Es cualquier dispositivo que, desde el punto de vista de la central telefónica, actúa como un teléfono tradicional, este debe ser capaz de aceptar señales de llamada o ring, ponerse en estado de colgado o descolgado, y enviar y recibir señales de voz. Supone que es como un "teléfono" o cualquier otro dispositivo que "suena" (como una máquina de fax o un módem).

Foreign eXchange Station (FXS): Es lo que está situado al otro lado de una línea telefónica tradicional (la estación), este envía el tono de marcado (elemento activo), la señal de llamada que hace sonar los (elemento pasivo) teléfonos y los alimenta. En líneas analógicas, un FXS alimenta al FXO. Utiliza alrededor de 48 voltios DC para alimentar al teléfono durante la conversación y hasta 80 voltios AC (20 Hz) cuando genera el tono de llamada (ring). Necesita estar conectado a un FXO (como una línea telefónica necesita estar conectada un teléfono) o viceversa.

Llamada en conferencia: Permite que tres o más personas se comuniquen simultáneamente. Existen dos tipos de conferencia de audio: espontánea o planificada. El tipo espontánea permite que una llamada sea convertida en una llamada en conferencia mediante la agregación de otros números en un momento dado. Las conferencias planificadas se realizan, por lo general, sobre equipos especializados que utilizan un "número puente" al cual se deben de conectar todos los participantes y un código que identifica esa conferencia.

Llamadas Retenida: Es el proceso de poner en estado ocioso a una conexión activa, sin desconectar la llamada. En ese momento es posible recibir la llamada, pasarla a otra línea o realizar alguna acción mientras el extremo contrario espera. La llamada retenida es implementada a veces acompañada de una música en espera.

Multimedia sobre IP (MoIP): Servicios multimedia (vídeo, audio, imagen, etc) prestados sobre redes IP

Notificación de mensajes: (pager, email, etc.) algunos servidores de correo de voz agregan la facilidad de notificar a los usuarios cuando tiene mensajes de voz sin revisar. Mediante el envío de un correo electrónico es posible realizar esta función.

Softphone: (en inglés combinación de Software y de Telephone) es un software que hace una simulación de teléfono convencional por computadora. Es decir, permite usar la computadora para hacer llamadas a otros softphones o a otros teléfonos convencionales. Normalmente, un Softphone es parte de un entorno Voz sobre IP y puede estar basado en el estándar SIP/H.323 o ser privativo. Son realmente parte de un grupo tecnológico mayor, el CTI (Integración Computadora Telefonía). Algunos softphones están implementados completamente en software, que se comunica con las PABX a través de la (LAN) Red de Área Local - TCP/IP para controlar y marcar a través del teléfono físico. Generalmente se hace a través de un entorno de centro de llamadas, para comunicarse desde un directorio de clientes o para recibir llamadas. En estos casos la información del cliente aparece en la pantalla de la computadora cuando el teléfono suena, dando a los agentes del centro de llamadas determinada información sobre quién está llamando y cómo recibirlo y dirigirse a esa persona.

Tarificación de llamadas: Es una aplicación importante en un entorno empresarial pues permite detallar la factura del consumo de llamadas, permite conocer detalles tales como quien, cuando y a donde se realizaron las llamadas. Permite a los administradores obtener información para crear políticas para evitar el abuso en el uso del sistema de telefonía[40].

Telefonía: Servicios de telecomunicación prestados sobre la RTC ya sea RTB o RDSI, a excepción de comunicación de datos.

Telefonía IP: Servicios de telefonía prestados sobre Redes IP "privadas" en interconexión con la RTC.

Transferencia de Llamada: Permite desviar hacia otro usuario una llamada activa sin desconectar. Esta funcionalidad es utilizada por los recepcionistas en una central telefónica. De esa manera el usuario que realiza la llamada no tiene que marcar otro número para poder contactar a su destinatario.

Voz en Internet: Servicios de telefonía prestados sobre la red pública global formada por la interconexión de redes de conmutación de paquetes basadas en IP.

Voz sobre ATM (Asyncronus Transfer Mode) (VoATM): Servicios de telefonía prestados sobre redes ATM donde existe posibilidad de ofrecer una calidad de servicio (Qos).

Voz sobre Frame Relay (VoFR): Servicios de telefonía prestados sobre redes soportadas por circuitos Frame Relay, orientados a la transmisión de datos.

Voz sobre IP (VoIP): Es la tecnología que presta servicios de telefonía sobre redes IP "privadas" sin interconexión a la RTC.

---

[40] Principalmente en entornos empresariales y en ocasiones en el hogar.

BIBLIOGRAFIA

[1] *Tanenbaum, Andrew S. Redes de Computadoras. Pearson Educación de México. Cuarta edición, 2003.*

[2] *Keagy, Scott. Integración de Redes de Voz y Datos, 1ª edición, Editorial Cartone, 2001.*

[3] *La red telefónica conmutada.*
www.celiclub.org/.../la%20red%20TE%20conmutada%20(AlejandroCorletti)*.pdf*

[4] PBX.
es.wikipedia.org/wiki/PBX

[5] *Voz sobre IP.*
digeset.ucol.mx/tesis.../Pdf/Maybelline%20Reza%20Robles.pdf

[6] *Protocolos de Voz sobre IP.*
www.lairent.com.ar/pdf/ART0002%20-%20Protocolos%20en%20VoIP.pdf –

[7] *VoIP para el desarrollo.*
www.it46.se/courses/voip/.../es/.../es_voip4d_it46_release_web.pdf

[8] *VoIP una puerta hacia la convergencia.*
www.cesga.es/component/option.../lang,gl/

[9] *La capa de control*
info.telefonica.es/.../pdf/publicaciones/.../07_la_capa_de_control.pdf –

[10] *Tecnología de voz sobre IP: Terminal H 323*
iie.fing.edu.uy/ense/asign/telef/TelefoniaIPH323.pdf

[11] *Asterisk en español.*
http://itaki.net/espanol/asterisk_espanol.pdfProyecto de documentación de Asterisk.

[12] Instalación de Dahdi.
http://Asterisknic.com/2008/11/25/guia-de-como-instalar-dahdi-antes-llamado-zaptel/

[13] Archivos de configuración de Asterisk.
voipenespañol.com

[14] *Diseño de un sistema de telefonía IP y plan de empresa para su comercialización.*
upcommons.upc.edu/pfc/bitstream/2099.1/3819/2/40377-2.pdf

[15] Modelo de configuración básica de Asterisk 1.2 con la plataforma de Voztelecom.
www.voztele.com/linea.../asterisk.../Asterisk-1.2-VozTelecom.pdf

[16] *Arquitectura de SIP*
http://www.voipforo.com/SIP/SIParquitectura.php

[17] *Configuración de sip.conf*
http://www.voipforo.com/asterisk/configuracion-sip-conf.php

[18] *Diseño e Implementación de un Agente de Usuario SIP*
sua.sourceforge.net/Presentacion_DIAUS.ppt

[19] *SIP: Session Initiation Protocol.*
www.docstoc.com/docs/134773/sip-gt2003

[20] Inter-Asterisk eXchange.
en.wikipedia.org/wiki/Inter-Asterisk_eXchange

[21] *Arquitectura de IAX*
http://www.voipforo.com/IAX/IAX-arquitectura.php

[22] *SCCP: Skinny Client Control Protocol.*
es.wikipedia.org/.../Skinny_Client_Control_Protocol