EN ESPAÑOL

1RA. EDICIÓN
Asterisk 11

Asterisk PBX

Instalación, Configuración y Puesta en marcha

- Instalación, Actualización y Configuración inicial
- Integración con Sistema Telefónico Análogo, VoIP y Digitales.
- Conectando Asterisk con el mundo.
- dialplan interactivo, conceptos avanzados.
- opciones de Correo de Voz de Asterisk.
- Construir Sistema de Menú y agregar aplicaciones que actúan sobre las entradas de los usuarios.
- Distribución Automáticas de llamadas.
- Direccionando sus llamadas a la Operadora automática .

PRIMERA EDICIÓN

Asterisk[PBX]: La Guía en Español

Bernard Pérez

SNC IT
R.D.

Asterisk PBX: Instalación, Configuración y puesta en marcha, Primera Edición.
Por Bernard Pérez

Impreso en República Dominicana.

Este libro puede ser adquirido para uso educacional únicamente.
Edición disponible solo en formato electrónico (PDF), descargable en línea.
Para obtener más información, póngase en contacto openbookrd.com / info@sncsit.com.

Primera Edición: Enero 2014

Visite https://payhip.com/b/2Qld o http://sncstore.blogspot.com/si precisas más información.

Tabla de Contenido

Capítulo I

Introducción a la Telefonía

Breve historia de la telefonía

Lo que hoy conocemos como teléfonos se originó con un interesante movimiento ha mediado del siglo 19.

En 1849 un médico de origen Italiano llamado Antonio Meucci, considerado por muchos como el inventor del teléfono, llevo a cabo una demostración de un dispositivo capaz de transmitir la voz en La Habana. Un tiempo después, en 1854, en la ciudad de Nueva York, Meucci hace una nueva demostración de su invento.

Pero la idea de construir un telégrafo parlante no era exclusiva de Meucci, y es así como 1860 el alemán Johann Philipp Reis construye un dispositivo capaz de transmitir voz basado en la idea original de Charles Bourseul, quien en 1854 había descrito la construcción de dicho dispositivo pero nunca lo construyó. Reis continuó mejorando su aparato y un año más tarde ya estaba transmitiendo voz a más de 100 metros de distancia.

Un par de años más tarde Innocenzo Manzetti construye el esperado "telégrafo parlante", pero no se interesa en patentarlo.

Las Patentes

Hasta este momento ya existían prototipos de teléfonos pero nadie lo había patentado.

Meucci fue el primero en tratar de patentar su invento, quien en 1871 suscribió un documento de "aviso de patente" pero por sus condiciones económicas nunca pudo pagar el dinero para terminar este trámite y su "aviso de patente" expiró pocos años después.

Para 1875, un año después de expirar el trámite de patente de Meucci, el escocés Alexander Graham Bell, radicado en los Estados Unidos, logra patentar un aparato similar y es el primero en hacerlo.

Gracias a la patente Bell pudo hacer de la idea del teléfono un negocio rentable y tiene el mérito de haber desarrollado la idea y convertirla en algo práctico para la sociedad.

Ya para 1886, existían 150,000 abonados telefónicos en los Estados Unidos.

A partir de entonces la telefonía se empezó a convertir en un servicio básico de la sociedad actual.

Evolución de la tecnología telefónica

Al principio, para que un abonado telefónico se comunicara con otro este tenía que solicitarle la llamada a una operadora, quien manualmente conectaba los cables para conmutar un punto con otro. En 1891 se inventó un teléfono "automático" que permitía marcar directamente.

Bell en un principio fue casi exclusivamente la única compañía en explotar la tecnología debido sus patentes. Sin embargo, cuando estas expiraron surgieron cientos de pequeñas compañías que empezaron a dar servicios, la mayoría en sitios rurales donde Bell aún no llegaba. Poco a poco estas compañías empezaron a crecer y ya a inicios del siglo 20 tenían en su conjunto más abonado que la propia Bell.

A finales de la segunda guerra mundial el servicio telefónico llegaba a millones de abonados.

En 1947, científicos de Bell inventaron el transistor y cambian el curso de la historia de la humanidad. En 1948 ganan el Premio Nobel por su trabajo.

En los años 60s se lanzan los primeros satélites de comunicaciones y las comunicaciones entre continentes se facilitan. Todo esto fue posible gracias a la invención del Transistor.

Principios y transmisión de la voz humana

La voz humana está compuesta por ondas acústicas que viajan a través del aire a la velocidad del sonido, esto es a 1,244 Km/h (o 340 m/s). Más rápido que un avión comercial. Pero esta rapidez no significa que me pueda comunicar fácilmente con puntos distantes pues la voz humana se atenúa rápidamente, perdiendo energía a medida que viaja. Luego de unos pocos metros ya no podemos escuchar una conversación.

La voz humana es de la misma naturaleza que el resto de ondas acústicas y estos ya se conocía desde antes de la invención del teléfono.

También antes de la invención del teléfono se conocía que existían otros tipos de ondas llamadas ondas eléctricas que podían ser transmitidas a través de un conductor metálico como un cable de cobre. Las ondas eléctricas son de naturaleza diferentes a las ondas acústicas y viajan a la velocidad de la luz, es decir aproximadamente 300,000 km/s. La atenuación de esta onda es fácil de controlar y transmitir a grandes distancias.

Con estos hechos conocidos ya a mediados del siglo 19 es más fácil comprender que muchos persiguieran la idea de transformar las ondas acústicas en ondas eléctricas para así poder transmitirlas luego a grandes distancias a través de conductores metálicos. La cuestión es que había que inventar un dispositivo para hacer dicha transformación y allí estaba la clave del asunto. Este dispositivo, conocido como micrófono en nuestros días es una parte importante de cualquier aparato telefónico.

Rango de frecuencia de la voz humana

Otra característica importante de la voz humana es que las cuerdas vocales modulan la voz en un amplio espectro de frecuencias que van de graves a agudos en un rango aproximado de 20Hz a 20kHz.

Esto nos hace suponer que un micrófono debe ser capaz de capturar y transmitir todo este rango de frecuencias. Sin embargo, en la actualidad sabemos que para transmitir voz "entendible" no es necesario transmitir todas las frecuencias sino un rango mucho menor y transmitir un rango menor de frecuencias tiene sus ventajas pues facilita la transmisión como veremos más adelante. Por lo tanto los teléfonos comerciales solo transmiten un rango aproximado de 400Hz a 4kHz. Esto distorsiona un poquito la voz pero de todas maneras se puede entender. Es por eso que cuando oímos a alguien por teléfono su voz suena ligeramente diferente que en la vida real pero aun así podemos entender la conversación.

El micrófono

El micrófono fue un elemento clave en la invención del teléfono pues era el dispositivo que realizaba la conversión de las ondas mecánicas a ondas eléctricas.

Hay muchos tipos de micrófonos que operan sobre diferentes principios. Uno que se usó por mucho tiempo en teléfonos era el de carbón que consistía en una cápsula llena de granitos de carbón entre dos placas metálicas. Una de las placas era una membrana que vibraba con las ligeras presiones de las ondas de voz; de esta manera la resistencia eléctrica de la cápsula variaba con la voz y se generaba una señal eléctrica correspondiente.

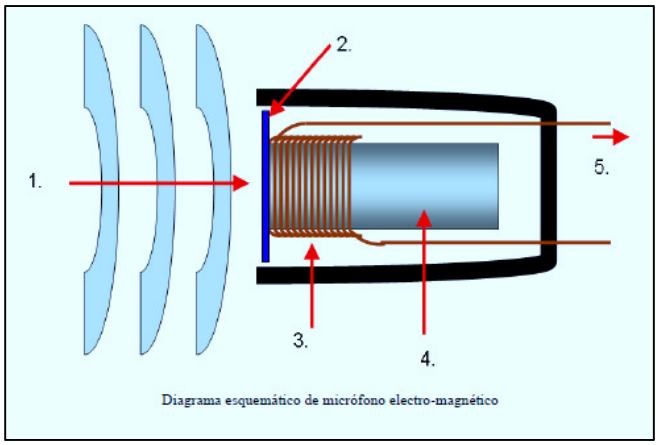


Diagrama esquemático de micrófono electro-magnético

Otro tipo de micrófono muy común en la actualidad es el dinámico o electro-magnético que consiste en una bobina de hilo de cobre enrollada sobre un núcleo de material ferromagnético. Este núcleo se encuentra sujetado a un diafragma que vibra con la presión de las ondas de voz. De esta manera se induce una ligera corriente eléctrica en la bobina que es amplificada luego al interior del teléfono.

En la figura anterior podemos observar algunos componentes del micrófono electromagnético reaccionando frente al estímulo de las ondas de voz.

1 Ondas de voz

2 Diafragma

3 Bobina

4 Núcleo ferromagnético

5 Corriente inducida

Ancho de banda y capacidad de información

Podemos decir que el ancho de banda es la medida de la cantidad de información que podemos transmitir por un medio por unidad de tiempo. Debido a que es una medida por unidad de tiempo muchas veces se hace una analogía con la velocidad.

El ancho de banda puede pensarse como el número de carriles de una avenida, es decir el número de carriles hará que los autos (la información) transiten y por lo tanto lleguen a destino más rápido o más lento.

Medidas comunes para expresar el ancho de banda son los bits por segundo. Esta medida también equivale a bits/s, bps o baudios.

Cuando se trata de telefonía el ancho de banda es un término muy importante, las comunicaciones en tiempo real necesitan un ancho de banda mínimo asegurado para entregar una comunicación de calidad.

Digitalización de la Voz

Digitalizar una señal de voz es tomar muestras (a intervalos de tiempos regulares) de la amplitud de la señal analógica y transformar esta información a binario. Este proceso es denominado muestreo.

Además de la etapa de muestreo en el proceso de digitalización de la voz intervienen otras etapas como son:

La cuantificación

El proceso de cuantificación convierte las muestras analógicas en muestras que pueden tomar un conjunto discreto de valores. De esta manera, los valores de las muestras se “cuantifican” en cantidades discretas.

Al pasar de infinitos valores (señal analógica) a un conjunto discreto de valores, se introduce naturalmente una distorsión a la señal original. Esta distorsión se conoce normalmente como “Ruido de Cuantificación”.

Codificación

Para poder procesar los “valores discretos” obtenidos en cada muestra, es necesario “codificarlos”, es decir, asignarles un valor numérico.

El proceso de cuantificación y codificación adoptado en telefonía implementa un algoritmo no lineal, de manera de obtener una calidad de voz aceptable, minimizando la cantidad de “niveles de cuantificación”. Este algoritmo se basa en tener distorsiones pequeñas para las amplitudes pequeñas de la señal, y distorsiones mayores para las amplitudes mayores de la señal.

Teorema de Nyquist

Harry Nyquist, un ingeniero Suizo que trabajaba para AT&T, resolvió en 1928, el dilema de cuanto es necesario muestrear una señal como mínimo para poder reconstruirla luego de forma exacta a la original.

El teorema propuesto decía que como mínimo se necesita el doble de ancho de banda como frecuencia de muestreo. Esto queda reflejado de mejor manera con la siguiente expresión.

$$f_m \geq 2\, BW_s$$

Hagamos un breve cálculo mental acerca de cuál sería la frecuencia de muestreo para poder convertir una señal de voz humana a digital y luego poder reconstruirla en destino.

Ya habíamos dicho que para que la voz humana sea entendible es suficiente transmitir un rango de frecuencias de entre 400Hz a 4,000Hz. Por lo tanto, según el teorema de Nyquist como mínimo deberíamos muestrear al doble de la frecuencia mayor, es decir a 8,000Hz.

Luego veremos que es precisamente esa frecuencia de muestreo de 8,000Hz la que se usa en la mayoría de codecs.

Redes orientadas a circuitos

Las redes orientadas a circuitos (circuit switched) son aquellas donde se estable un circuitos exclusivo o dedicado entre los nodos antes de que los usuarios se puedan comunicar.

Una vez que se establece un circuito entre dos puntos que quieren comunicarse, el resultado básicamente es el equivalente a conectar físicamente un par de cables de un extremo a otro. Una vez establecido el circuito, éste ya no puede ser usado por otros.

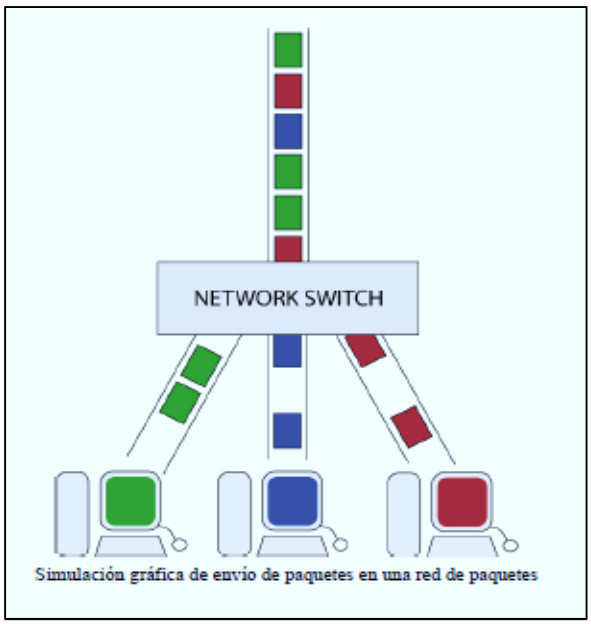
En cada circuito el retardo es constante, lo cual es una ventaja. Sin embargo, este tipo de redes es costoso debido al mismo hecho de que se necesita un circuito dedicado para cada abonado.

Este tipo de redes es el tradicionalmente usados por compañías telefónicas alrededor del mundo y es el mismo que usó Bell en sus inicios; guardando las distancias tecnológicas correspondientes.

Es común que ciertas personas confundan las redes de circuitos con las redes analógicas pero es necesario aclarar que las redes de circuitos bien pueden transportar datos digitalmente.

Redes orientadas a paquetes

Una red de paquete es aquella que por un mismo medio pueden pasar simultáneamente diferentes flujos de información. Para hacer esto divide el tráfico de cada flujo de información en fragmentos o paquetes que envía intercaladamente. Luego, en el destino los paquetes se reensamblan para reproducir el mensaje original.



Simulación gráfica de envío de paquetes en una red de paquetes

Un ejemplo de este tipo de redes son las redes IP como es el caso del Internet, donde por una misma conexión pueden llegarnos distintos flujos de información. De esta manera podemos estar haciendo video-conferencia al mismo tiempo que enviamos un correo electrónico o navegamos por el Web. Inclusive por este tipo de redes pueden circular simultáneamente flujos de información para diferentes destinos o direcciones IP.

A diferencia de las redes orientadas a circuitos, en este tipo de redes el ancho de banda no es fijo ya que depende del tráfico de la red en un momento dado. Adicionalmente cada paquete de un mismo flujo de información no está obligado a seguir el mismo camino por lo que los paquetes que originalmente fueron generados en secuencia pueden llegar desordenados a su destino. Este tipo de factores son muy importantes a tener en cuenta cuando se trafica voz sobre una red de paquetes ya que afectan la calidad de la llamada.

Las redes de paquetes se han vuelto populares, principalmente porque optimizan recursos debido al hecho de poder utilizar el mismo medio para enviar varios flujos de información.

Red Pública Telefónica (PSTN)

La Red Pública Telefónica o PSTN (por sus siglas en inglés) es esencialmente una red basada en circuitos. Esta red cubre tanto telefonía fija como móvil y es la red que hace posible que podamos comunicarnos con cualquier persona en nuestra ciudad o alrededor del mundo.

Originalmente fue una red analógica pero actualmente es una red en su mayoría digital; por tanto existen dos tipos de circuitos: analógicos y digitales.

Circuitos analógicos

Los circuitos analógicos son comúnmente pares de cobre que llegan a los abonados del servicio telefónico y por donde se transmite la señal eléctrica de la voz de manera analógica. El mismo circuito lleva adicionalmente la señalización necesaria para establecer, mantener y terminar una llamada. Estos circuitos analógicos se deben conectar a un switch telefónico encargado de direccionar la comunicación entre los abonados.

Los circuitos analógicos están en decadencia pues las compañías telefónicas encontraron muchas ventajas en las comunicaciones digitales y es por esa razón que pese a que en la actualidad aún vemos circuitos analógicos esto se trata tan solo de la "última milla". En cierto punto de la red telefónica esta comunicación es convertida a digital y transmitida a un switch telefónico digital.

La circuitería analógica comúnmente se asocia con el término de "telefonía tradicional".

Como en el pasado era más común que los teléfonos pudieran estar ubicados en áreas rurales donde no llegaba la electricidad se decidió que la red telefónica proveyera cierto voltaje de alimentación. Es por eso que algunos modelos de teléfonos analógicos no necesitan conectarse a la alimentación eléctrica.

En todo caso la OC (Oficina Central) genera 48 Voltios de corriente directa para alimentar a los teléfonos de los abonados. Usando léxico estricto deberíamos decir -48 Voltios debido a que este voltaje se mide con respecto a uno de los conductores. Sin embargo para ser prácticos en este libro usaremos indistintamente 48V o -48V para referirnos a lo mismo.

Señalización analógica

Para que las llamadas telefónicas funcionen correctamente es necesario contar con indicaciones o señales eléctricas que nos permitan intercambiar información entre el abonado y la OC. En breve veremos en qué consisten las señales más comunes.

Existen básicamente 3 métodos de señalización analógica que la industria ha desarrollado a través de los años. Estos se llaman ***loop start, ground start y kewlstart***. Es importante cuando se configura una central telefónica que va conectada a una línea analógica que escojamos el método de señalización adecuado pues en caso contrario podemos encontrarnos con problemas extraños como que la línea se cuelga inesperadamente o que no podemos colgar la línea correctamente, entre otras cosas.

La diferencia entre loop start y ground start radica en la manera en la que el teléfono requiere tono de marcado a la OC (proceso también llamado seizure). Ground start requiere tono de marcado aterrizando (de allí el término ground) uno de los conductores de la línea telefónica mientras que loop start lo hace realizando un corto circuito entre ambos conductores (es decir creando un lazo o loop).

Kewlstart es una evolución de loop start que le añade un poco más de inteligencia a la detección de desconexiones (colgado de la llamada) pero básicamente sigue siendo un loop start.

Debido a que ground start no es muy común en nuestros días, casi siempre nos veremos usando loop start.

A continuación explicaremos más al detalle la señalización analógica para los eventos más comunes. Para hacerlo nos basaremos en el progreso de una llamada típica usando señalización loop start. El progreso de una llamada lo podemos dividir en seis instancias: ***colgado (on-hook), descolgado, marcación, conmutación, ringado y conversación.***

Colgado

Mientras el teléfono está colgado la OC provee un voltaje DC de 48 Voltios. El teléfono mantiene un circuito abierto con la línea telefónica; es decir que actúa como si no estuviera conectado y por lo tanto no fluye corriente por la línea.

Este estado también es conocido como on-hook por su significado en inglés.

Descolgado

Cuando el usuario descuelga el auricular el teléfono envía una señal a la OC. Esta señal consiste en cerrar el circuito, es decir que internamente el teléfono conecta entre sí los dos cables de la línea telefónica a través de una resistencia eléctrica.

Apenas la OC se da cuenta de esto envía tono de marcado al teléfono. Este tono de marcado le indica al abonado que ya puede marcar el número.

En gran parte de América el tono de marcado consiste en dos ondas senoidales enviadas simultáneamente. Estas ondas son de 350 Hz y 440 Hz. En Europa el tono de marcado consiste en una sola onda de 425 Hz. Sin embargo hay países en los que estos valores podrían ser diferentes.

Marcación

La marcación puede ser por pulsos o por tonos. Los pulsos ya casi no se usan y fueron populares en los tiempos de los teléfonos de disco. Los tonos son pares de frecuencias asociadas con los dígitos telefónicos. Estas frecuencias se transmiten hasta la OC quien traduce estos tonos a números.

Conmutación

Una vez recibidos los dígitos la OC tratará de asociar este número marcado con el circuito de un abonado. En caso de que el destinatario no fuera un abonado local, enviará la llamada a otro switch telefónico para su terminación.

Timbrado o Ringado

Una vez que la OC encuentre al abonado destino tratará de timbrarlo (ringing). La señal de ring es una onda sinusoidal de 20 Hz y de 90 Voltios de amplitud.

Adicionalmente a la señal de ring que la OC envía al destinatario también envía una notificación a quien originó la llamada. Este tono audible recibe el nombre de ring-back y consiste en dos ondas sinusoidales superpuestas de 440 Hz y 480 Hz. Estas ondas van intercaladas con espacios de silencio.

En caso de que el destinatario se encuentre ya en una llamada activa entonces en lugar del ring-back se devuelve un tono de ocupado a quien originó la llamada. Este tono de ocupado consiste en dos ondas sinusoides superpuestas de 480 Hz y 620 Hz intercaladas con espacios de silencios de medio segundo.

Conversación

Si el destinatario decide contestar la llamada el teléfono cerrará el circuito telefónico (de la misma manera que ocurrió con el teléfono que originó la llamada en la etapa de descolgado). Esta señal le informará a la OC que el destinatario decidió contestar y completará la conexión. La llamada telefónica está finalmente en curso.

DTMFs

Frecuentemente es necesario enviar dígitos a través de la línea telefónica tanto para marcar como en medio de una conversación. Con esta finalidad se pensaron los DTMFs. DTMFs es un acrónimo de Dual-

Tone Multi Frequency. Es decir que cada DTMF es en realidad dos tonos mezclados enviados simultáneamente por la línea telefónica. Esto se hace así para disminuir los errores.

	1209 Hz	1336 Hz	1477 Hz	1633 Hz
697 Hz	1	2	3	A
770 Hz	4	5	6	B
852 Hz	1	8	9	C
941 Hz	*	0	#	D

Ilustración de los pares de frecuencias para cada digito.

Como se puede ver en la tabla también hay correspondencias para los signos * y # así como también para los caracteres A, B, C y D.

El teléfono análogo

Este es un componente importante de la red telefónica recordemos que su invención fue lo que marcó el desarrollo del negocio de la telefonía.

También es importante hablar del teléfono analógico porque todavía es el tipo de teléfono más común en el planeta y porque la comprensión de su funcionamiento nos permitirá entender en el futuro algunos conceptos clave como por ejemplo el eco.

En realidad el teléfono, en su forma más básica, es un dispositivo sencillo compuesto de pocos componentes.

- Auricular
- Micrófono
- Switch para colgado/descolgado
- Convertidor de dos a cuatro hilos (también llamado híbrido)
- Marcador (dialer)
- Campana o dispositivo de timbrado

La mayoría de los componentes se explican por si solos. Sin embargo algunos se preguntarán ¿de qué se trata el convertidor de 2 a 4 hilos?

Convertidor de 2 a 4 hilos.

Un componente importante de un teléfono es el convertidor de dos a cuatro hilos, conocido también como dispositivos 2H/4H, bobina hibrida o simplemente hibrido. Este dispositivo es necesario para separar la señal de audio de ida de la de venida ya que son dos participantes en una conversación y solo existe un par de cables para esto. Si existieran tres o cuatro cables (2 de ida y 2 de venida) el convertidor de 2 a 4 hilos no fuera necesario.

Circuitos digitales

La PSTN también sirve a sus abonados con circuitos digitales. Estos circuitos ofrecen la ventaja de poder multiplexar más de una línea en el mismo medio por lo que resulta atractivo para abonados con necesidades de un gran número de líneas telefónicas, por lo general empresas.

DS-0

DS-0 es un canal digital de 64Kbit/s. Un DS-0 es por tanto una medida de canal estándar o unidad que nos sirve para definir múltiplos mayores como los circuitos que veremos a continuación.

Circuitos T-carrier y E-carrier

Los circuitos T-carrie (o portadora-T) fueron diseñados como nomenclatura para circuitos digitales multiplexados y fueron desarrollados por Laboratorios Bell hace más de cincuenta años. Los circuitos E-carrier son la equivalente europea.

El más conocido de los circuitos T-carrier es el popular T1 (y su contraparte E1). Un T1 es un circuito digital compuesto de 24 DS-0's mientras que un E1 está compuesto por 32 DS-0's. En un circuito T1 se envían 1.544 Mbit/s mientras que un E1 2048 Mbit/s.

Además de los circuitos T1 también tenemos circuitos como T2, T3, T4 y T5.

SONET y Circuitos Ópticos

SONET (Synchronous Optical Networking) fue desarrollado con el objetivo de contar con una nomenclatura similar a las T-carrier pero usando la tecnología de fibra óptica. SONET utiliza múltiplos de T3 para sus anchos de banda y su circuito base es el llamado OC-1.

Luego del OC-1 tenemos los OC-3, OC-12, OC-24, OC-48, entre otros.

Protocolos de Señalización Digital

Los protocolos de señalización se utilizan para transmitir información de estado del canal de comunicaciones (como "desconectado", "timbrando", "respondió"), información de control y otra información como DTMFs, callerID, entre otros.

Los protocolos de señalización se pueden agrupar en dos tipos llamados CAS (Channel Associate Signaling) y CCS (Common Channel Signaling). La diferencia es que mientras CAS transmite la señalización en el mismo canal en que viaja la información, CCS la transmite en un canal separado. Por este hecho es que con CAS se reduce ligeramente el ancho de banda disponible o útil para la comunicación ya que una parte de él se está usando para señalización. Esa es una de las razones por las cuales las compañías telefónicas han adoptados en su mayoría CCS.

No confunda el lector **CAS y CCS** con protocolos de señalización. Tan solo son tipos de protocolos que se explican aquí para hacer más fácil la categorización o agrupación de los mismos.

Señalización Asociada al Canal (CAS)

El protocolo CAS más conocidos es robbed-bit y es usado en circuitos T1 y E1 alrededor del mundo.

Robbed-bit toma (o “roba”, de allí su nombre) el octavo bit de cada canal de comunicación cada seis frames y lo remplaza por información de señalización. El bit original robado simplemente se pierde.

Hay que aclarar de lo anterior que esto es posible debido a que la voz no es muy sensible que digamos a la pérdida de ese bit de información ya que es el bit menos significativo. Pero cuando transportamos data la pérdida de un bit no puede pasar desapercibida y la calidad de la transmisión se degrada de manera sensible.

Otro protocolo CAS que aún subsiste en nuestros días es R2. Se trata de un protocolo que fue popular en los años 60s. En realidad R2 es una familia de protocolos en donde cada implementación se denomina “variante”. Existen variantes dependiendo del país o inclusive de la compañía telefónica que lo ofrece.

Señalización de Canal Común (CCS)

ISDN

ISDN (Integrated Services Digital Network) nos permite transmitir voz y datos simultáneamente sobre pares telefónicos de cobre con calidad superior a las líneas telefónicas analógicas.

El objetivo de ISDN fue el de facilitar las conexiones digitales para poder ofrecer una amplia gama de servicios integrados a los usuarios. ISDN establece dos tipos de interfaces para cumplir su fin.

- **BRI:** Basic Rate Interface
- **PRI:** Primary Rate Interface.

BRI estuvo orientada a hogares. Un BRI supone 2 canales útiles (llamados canales B) de 64Kbit/s cada uno más un canal de señalización de 16Kbit/s (llamado canal D) que en total suman 144Kbit/s.

PRI es la opción para usuarios de mayor envergadura como negocios o empresas pues puede aglutinar más canales B. Actualmente es muy popular y se transmite sobre circuitos T-carrier y E-carrier.

Capítulo 2

Introducción a la Telefonía VoIP

La voz sobre IP o VoIP consiste en transmitir voz sobre protocolo IP.

La industria de las telecomunicaciones se ha extendido por más de 100 años, y Asterisk integra la mayoría --si no todas-- de las principales tecnologías que se haya hecho uso en el último siglo.

Para sacar el máximo provecho de Asterisk, no es necesario ser un profesional en todos los ámbitos, pero comprender las diferencias entre los distintos CODECS y protocolos le dará una mayor apreciación y comprensión del sistema en su conjunto.

Este capítulo explica los conceptos de Voz sobre IP y lo que hace que las redes de VoIP sean diferentes a las redes de voz con conmutación de circuitos tradicionales.

En este tema vamos a ver porque son necesarios los protocolos VoIP, haremos un breve resumen de la historia y el futuro potencial de cada uno. También veremos las consideraciones de seguridad y las habilidades de estos protocolos para trabajar dentro de topologías NAT (traducción de direcciones de red). Se tratarán los siguientes protocolos de VoIP (algunos más breve que otros):

- IAX
- SIP
- H.323
- MGCP
- Skinny/SCCP
- UNISTIM

Los CODEC son los medios por los cuales la voz análoga puede ser convertida a una señal digital y llevada a través del Internet. El ancho de banda en cualquier localización es limitado, y el número de conversaciones simultáneas que cualquier conexión puede llevar está directamente relacionado al tipo de códec que se implemente. En este módulos también hablaremos de las diferencias entre los siguientes codecs independientemente de los requerimiento de ancho de banda (nivel de comprensión) y calidad.

- G711
- G726

- G729A
- GSM
- ILBC
- Speex
- MP3

Para finalizar el modulo, discutiremos como el tráfico de voz puede ser enrutado fiablemente, que causas el ECO y como tratar con este, y como Asterisk controlas la autenticación de las llamadas entrantes y salientes.

¿Por qué son necesarios los Protocolos VoIP?

La necesidad básica de los protocolos VoIP es la paquetización del audio para ser transportado sobre las redes basadas en los protocolos de Internet.

Los retos para lograr esto se refieren a la manera en que los humanos se comunican. No sólo es necesario que la señal llegue esencialmente de la misma forma que se transmite, pero tiene que hacerlo en menos de 150 milisegundos. Si los paquetes se pierden o se retrasan, habrá degradación de la calidad de las comunicaciones, lo que significa que dos personas tienen dificultades para mantener una conversación telefónica.

Los protocolos de transporte que en su conjunto son llamados **"Internet**" no fueron originalmente diseñados con transmisión de streaming de audio en tiempo real en mente. Se espera que los nodos receptores resuelvan el problema de las pérdidas de paquetes por esperar un largo tiempo para que el paquete arribe, solicitando la retransmisión, o en algunos casos, considerando la información perdida como buena y simplemente seguir adelante. Nuestras conversaciones no se adaptan bien a la pérdida de letras o palabras, ni de ningún retraso apreciable entre transmisión y recepción.

La PSTN tradicional fue diseñada especialmente para el propósito de transmisión de voz, y es perfectamente adecuada para esta tarea desde el punto de vista técnico. Desde un punto de vista más flexible, sin embargo, sus defectos son evidentes incluso para las personas con un conocimiento muy limitado de la tecnología. VoIP mantiene la promesa de incorporar las comunicaciones de voz en todos los otros protocolos que llevamos en nuestras redes, pero debido a las exigencias especiales de una conversación de voz, se necesitan habilidades especiales para diseñar, construir y mantener estas redes.

El problema con la transmisión de voz basada en paquetes se deriva del hecho de que la forma en la que hablamos es totalmente incompatible con la forma en que IP transporta datos. Hablar y escuchar consiste en la retransmisión de una secuencia de audio, mientras que los protocolos de Internet están diseñados para dividir todo, encapsular los bits de información en miles de paquetes, y luego entregar cada paquete de cualquier manera posible para el extremo receptor. Es evidente que se requiere alguna forma de lidiar con esto.

Protocolos VoIP

Es por lo expuesto anteriormente que surgen los protocolos para VoIP, cuyo mecanismo para conexión implica una serie de transacciones de señalización entre los puntos finales (y entre gateway), que cargan dos flujos de audio para cada dirección de la conversación. Hay varios protocolos en existencias para manejar esto. En esta sección, vamos a hablar de los protocolos que en general son importantes para VoIP y especialmente para Asterisk.

El Protocolos IAX (Inter-Asterisk Exchange)

IAX es un protocolo abierto, lo que significa que cualquiera puede descargarlo, modificarlo o mejorarlo a su gusto.

En Asterisk, IAX es implementado a través módulo de **chan_iax2.so**. Este protocolo fue desarrollado por Digium con el propósito de comunicarse con otros servidores Asterisk, y entre servidores y clientes que también utilizan el protocolo IAX. Es muy importante señalar que IAX no está en lo absoluto limitado a Asterisk. Es un estándar abierto para que cualquiera lo use, y es apoyado por muchos otros proyectos de telecomunicaciones de código abierto, así como por varios proveedores de hardware. IAX es un protocolo de transporte (como SIP) que usa un solo puerto UDP (4569) para ambos canales de señalización y datos. Permite manejar una gran cantidad de códecs y un gran número streams, lo que significa que puede ser utilizado para transportar cualquier tipo de dato.

El protocolo IAX2 fue deliberadamente diseñado para trabajar detrás de dispositivos NAT. El uso de un único puerto UDP tanto para la señalización y la transmisión de datos, el tráfico de voz es transmitido *in-band*, hacen de IAX un protocolo casi transparente a los cortafuegos y realmente eficaz para trabajar dentro de redes internas. Está diseñado para darle prioridad a los paquetes de voz sobre redes IP. En esto se diferencia de SIP, que utiliza cadena RTP out-of-band (fuera-de-banda) para entregar la información.

IAX2 soporta Trunking, donde un simple enlace permite enviar datos y señalización por múltiples canales. Cuando se realiza Trunking, los datos de múltiples llamadas son manejados en un único conjunto de paquetes, lo que significa que un datagrama IP puede entregar información para más llamadas sin crear latencia adicional. Esto es una gran ventaja para los usuarios de VoIP, donde las cabeceras IP son un gran porcentaje del ancho de banda utilizado.

SIP

El protocolo de Inicialización de Sección (SIP) es un protocolo desarrollado por el grupo de trabajo MMUSIC del IETF 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 en línea y realidad virtual.

El protocolo SIP trata cada extremo de una conexión como un par (un peer) y se encarga de negociar las capacidades entre ambos extremo. Lo que hace a SIP atractivo es que es un protocolo relativamente simple, con una sintaxis similar a la de otros protocolos conocidos, como HTTP y SMTP. SIP es soportado en Asterisk a través del módulo **chan_sip.so**.

Historia

El 22 de febrero de 1996, Mark Handley y Eve Schooler presentaron al IETF un borrador del Protocolo de Invitación de Sección conocido ahora como SIPv1. El mismo estaba basado en trabajos anteriores de Thierry Turletti (INRIA Video conferencing System o IVS) y de Eve Schooler (Multimedia Conference Control o MMCC). Su principal fortaleza, heredada por la versión actual de SIP, era el concepto de registro, por el cual un usuario informaba a la red dónde (en qué host de Internet) podía recibir invitaciones a conferencias. Esta característica permitía la movilidad del usuario.

Ese mismo día el Dr. Henning Schulzrinne presentó un borrador del Simple Conference Invitation Protocol (SCIP), que estaba basado en el HTTP (Hypertext Transport Protocol). Usaba TCP (Transmission Control Protocol) como protocolo de transporte. Como identificadores de los usuarios utilizaba direcciones de correo electrónico para permitir el uso de una misma dirección para recibir correos

electrónicos e invitaciones a conferencias multimedia. No utilizaba al SDP para la descripción de los contenidos sino que creaba un mecanismo propio.

El IETF decidió combinar ambos en un único protocolo denominado Session Initiation Protocol, (es decir cambiando el significado de la inicial I en el acrónimo "SIP") y su número de versión fue la 2, dando origen al SIPv2. En diciembre de 1996 los tres autores (Schulzrinne, Handley y Schooler), presentaron el borrador del SIPv2. El mismo luego de ser discutido en el grupo de trabajo MMUSIC (Multiparty Multimedia Session Control) del IETF alcanzó el grado de "estándar propuesto" en la RFC 2543 publicada en febrero de 1999.

En septiembre de 1999 se creó el grupo de trabajo SIP en el IETF que continuó con el desarrollo del protocolo y en junio de 2002 se publicó la RFC 3261 que reemplazó a la anterior introduciendo modificaciones propuestas durante el trabajo del grupo SIP. Los autores de esta última RFC, hoy vigente son: Jonnathan Rosenberg, Henning Schulzrinne, Gonzalo Camarillo, Allan Johnston, Jon Peterson, Robert Sparks, Mark Handley y Eve Schooler.

SIP es un protocolo de señalización de capa de aplicación que usa el puerto bien conocido 5060 para comunicación. SIP puede ser transportado con los protocolos de capa de transporte ya sea UDP o TCP.

SIP no transporta datos entre dispositivos finales. Al contrario es un protocolo de transporte en tiempo real (RTP) y es usado para este propósito. RTP utiliza una numeración de puertos altos, o puertos no privilegiados en Asterisk (10.000 – 20.000, por defecto).

SIP no fue el primer protocolo, y no es el único usado hoy día (existen otros como: H.323, MGCP, IAX, entre otros). La ventaja del protocolo SIP es que es ampliamente aceptado y su flexibilidad arquitectónica.

SIP y NAT

Probablemente el mayor obstáculo técnico de SIP es que tiene que conquistar el reto de llevar a cabo transacciones eficientemente a través de NAT. Porque SIP encapsulas la información de direccionamiento en frames de datos (o tramas), y NAT se lleva a cabo en la capa de red, la información de direccionamiento no es modificada automáticamente, y por lo tanto los flujos de datos de comunicación no tendrán la información de direccionamiento correcta necesaria para completar la conexión cuando existe un dispositivo NAT de por medio. Adicional a esto, los firewalls normalmente integrado con NAT no consideran los media stream entrantes como parte de una transacción SIP, y bloquean la conexión. Los nuevos firewalls y los controladores de sección fronteriza (SBCs) son conscientes de SIP pero todavía esto es considerado como una limitante en este protocolo.

Seguridad en SIP

SIP usa un sistema de demanda/repuesta para autenticar los usuarios. Un INVITE inicial es enviado al proxy con el cual el dispositivo desea comunicarse. El proxy entonces responde con un mensaje de solicitud de autorización (407), el cual contiene un conjunto aleatorios de caracteres que es referido como un nonce. Este nonce se utiliza junto con la contraseña para generar un hash MD5, que es enviada de vuelta en el INVITE posterior. Asumiendo que el has MD5 coincida con el que el proxy ha generado, el cliente es entonces autenticado.

Los Ataque de denegación de servicios (DoS) son probablemente los tipos de ataques más comunes en las comunicaciones VoIP. Un ataque DoS puede ocurrir cuando un gran volumen de solicitud de INVITE invalido son enviados para un servidor proxy en un intento de saturar el sistema. Esto ataques son relativamente simple para implementar, y sus efectos sobres los usuarios del sistema son inmediatos. SIP cuentas con varios métodos de minimizar los efectos de los ataques DoS, pero últimamente son

imposibles de prevenir. SIP implementa un sistema para garantizar la seguridad, un mecanismo de transporte encriptado (llamado Transport Layer Security o TLS) es usado para establecer la comunicación entre la persona que llama y el dominio del destinatario. Más allá de eso, la solicitud es enviada de forma segura para el dispositivo final, basado sobre la política de seguridad local de la red.

H.323

Este protocolo fue originalmente diseñado para proveer un mecanismo de transporte IP para video conferencias. H.323 es el estándar de los equipos de video conferencias basado en IP, y disfrutó de una breve fama como un protocolo de VoIP también. Mientras muchos debates sobre si SIP o H.323 (o IAX) será el protocolo VoIP que dominara el mundo, en Asterisk, H.323 ha quedado en desuso en gran medida a favor de IAX y SIP. H.323 no ha tenido muchos éxitos entre los consumidores y las empresas, a pesar de que todavía podría ser el protocolo de VoIP más usado entre las empresas telefónicas.

Asterisk soporta tres versiones de H.323 y son manejadas por los módulos **chan_h323.so** (suministrado con Asterisk), **chan_oh323.so** (disponible como un complemento gratuito), y **chan_ooh323.so** (suministrado con Asterisk-addons)

Historia

H.323 fue desarrollado por la Unión de Telecomunicaciones Internacional (ITU) en mayo de 1996 como un medio para transmitir comunicación de voz, video, data y fax a través de las redes IP mientras mantenían la conectividad con el PSTN. Desde entonces, H.323 ha pasado por varias versiones y anexos (las cuales han agregado funcionalidad para el protocolo), permitiéndole operar en redes VoIP pura y más ampliamente en redes distribuida.

Consideraciones de Seguridad.

H.323 es un protocolo relativamente seguro y no requiere muchas consideraciones de seguridad más allá de la que son comunes para cualquier comunicación de redes con el Internet. Puesto que H.323 usa el protocolo RTP para medio de comunicación, este no soporta de forma nativa las rutas de comunicación encriptadas. El uso de VPN u otros túneles encriptados entre puntos finales es la manera más común de encapsular la comunicación de forma segura.

H.323 y NAT

El estándar H.323 usa el protocolo RTP para transportar los datos entre los puntos finales. Debido a esto, H.323 tiene los mismos problemas que tiene SIP cuando se trata de topologías de red que implica un NAT. El método más fácil es simplemente redirigir los puertos apropiados a través de los dispositivos NAT para el cliente interno.

Para recibir llamadas, usted siempre necesita redirigir el puerto TCP 1720 para el cliente. Adicional a esto, usted necesita redirigir el puerto UDP para el medio RTP y el RTCP control streams. Los clientes más viejos, como Microsoft NetMeeting, también requieren redireccionar el puerto TCP para el tunneling H.245.

Si usted tiene un número de clientes detrás de un dispositivo NAT, necesitará utilizar un gatekeeper (controlador de acceso) corriendo en modo proxy. El controlador de acceso requerirá una interfaz conectada a la subred IP privada y la Internet pública. Su cliente H.323 sobre la subred IP privada tendrá que registrarse con el controlador de acceso, el cual llamará al proxy en nombre del cliente. Tenga en cuenta que cualquier cliente que desee llamar se le pedirá que se registre con el servidor proxy.

MGCP

El Media Gateway Control Protocol (MGCP) también nos llega del IETF. Mientras que la implementación de MGCP está mucho más extendida de lo que se podría pensar, está perdiendo terreno rápidamente antes protocolos como SIP y IAX. Pero Asterisk aún tiene soporte para él, aunque rudimentario.

Este fue diseñado para hacer los dispositivos finales (tales como teléfonos) tan simple como sea posible, y tener todas la lógica y procesamiento de llamadas manejada por el media gateways y los agentes de llamadas. Al contrario de SIP, MGCP usa un modelo centralizado. Los teléfonos MGCP no pueden llamar directamente a otros teléfonos MGCP; siempre deben de ir a través de algún tipo de controlador.

Asterisk soporta MGCP a través del módulo **chan_mgcp.so**, y los dispositivos finales son definidos en el archivo de configuración **mgcp.conf**.

Protocolos Propietarios

Por último, vamos a echar un vistazo a dos protocolos propietarios que son compatibles con Asterisk.

Skinny/SCCP

El Protocolo de Control de Cliente Skinny es propietario para los equipos VoIP Cisco. Este es el protocolo por defecto para los dispositivos finales sobre una PBX Cisco Call Manager. Skinny es soportado por Asterisk, pero sí usted están conectando teléfono Cisco para Asterisk, es generalmente recomendado que usted obtenga una imagen SIP para cualquier teléfono y conecte este vía SIP.

UNISTIM

El soporte de Asterisk para el protocolo VoIP propietario de Nortel, UNISTIM, lo convierte en la primera PBX en la historia para soportar de forma nativa terminales IP propietaria de dos grandes jugadores en VoIP. Nortel y Cisco. El soporte de UNISTIM es totalmente experimental y todavía no trabaja bien pero está en producción, pero el caso es que alguien se ha tomado la libertad para implementar este y demostrar el poder de la plataforma Asterisk.

Codificación de la Voz

Debemos tener claro que para transportar la voz se utilizan algunos protocolos como SIP, IAX y otros como RTP o RTCP. Pero la voz es una onda analógica que necesita transformarse a digital en algún formato antes de ser transmitida sobre el medio. Como nos encontramos en una red de paquetes debemos paquetizar esta información. Esa búsqueda de un formato óptimo generó algunas alternativas de formatos de transmisión llamados CODECS.

CODECs

La palabra codec proviene de abreviar las palabras Codificación y Decodificación. Su función principal es la de adaptar la información digital de la voz para obtener algún beneficio. Este beneficio en muchos casos es la compresión de la voz de tal manera que podamos utilizar menos ancho de banda del necesario.

Generalmente se entiende por Codecs que son varios modelos matemáticos usados para codificar (y comprimir) la información de audio análoga. Muchos de estos modelos toman en cuenta la capacidad del cerebro humano para formar una impresión a partir de información incompleta.

Todos hemos visto las ilusiones ópticas, del mismo modo, los algoritmos de compresión de voz se aprovechan de nuestra tendencia a interpretar lo que creemos que debemos escuchar, en lugar de lo que realmente escuchamos. El propósito de los diferentes algoritmos de codificación es estrictamente mantener el balance entre la eficiencia y la calidad.

Antes de profundizar más en cada uno de los Codecs individuales, vamos a echar un vistazo en la siguiente tabla.

CODEC	DATA BITRATE (KBPS)	¿REQUIERE LICENCIA?
G.711	64 Kbps	No
G.726	16, 24, 32 o 40 Kbps	No
G.729A	8 Kbps	Si
GSM	13 Kbps	No
ILBC	13.3 Kbps(30-ms frames) o 15.2 Kbps (20-ms frames)	No
Speex	Variables (entre 2.15 y 22.4 Kbps)	No
G.722	64 Kbps	No

G.711

G.711 es el códec fundamental de la PSTN. De hecho, si alguien se refiere a PCM con respecto a una red telefónica, se le permite pensar en G.711. Dos métodos de comprensión son usados: ulaw en Norte América y alaw en el resto del mundo. Cualquieras de los 2 métodos proporciona una palabra de 8 bit transmitida 8,000 veces por segundo. Si saca cuentas, usted veras que se requieren 64,000 bits para ser transmitidos por segundo.

Mucha gente le dirá que G.711 es un códec sin compresión. Esto no es exactamente cierto, como el companding (compresión-expansión) es considerado una forma de compresión. Lo que es cierto es que G.711 es el códec base desde el cual todos los demás se derivan.

G.711 impone una carga mínima (casi cero) en la CPU.

G.726

Este codec ha existido desde hace un tiempo (solía ser G.721, que ahora es obsoleto), y es uno de los códecs comprimidos originales. Es también conocido como Adaptive Differential Pulse-Code Modulation (ADPCM), y este puede correr en varias velocidades de bits (bitrates). Las velocidades más comunes son 16 Kbps, 24 Kbps y 32 Kbps. Hasta el momento de escribir estas líneas, Asterisk únicamente soportaba el rate de ADPCM-32 que es el tipo más difundido y popular de este códec.

G.726 ofreces una calidad casi idéntica a G.711, pero este usa la mita de ancho de banda. Esto es posible porque en lugar de enviar el resultado de la medición de cuantificación, se envía sólo información suficiente para describir la diferencia entre la muestra actual y la anterior. G.726 cayó en desgracia en la década de 1990 debido a su incapacidad para llevar las señales de fax y módem, pero debido a su relación calidad-ancho de banda/CPU ahora está haciendo una reaparición. G.726 es especialmente atractivo, ya que no requiere una gran cantidad de trabajo computacional del sistema.

G.729A

Teniendo en cuenta el poco ancho de banda que utiliza, G.729A ofrece una impresionante calidad de sonido. Esto lo hace mediante el uso de Estructura Conjugada Predicción Lineal Algebraica con Excitación por Código (CS-ACELP.). A causa de las patentes, no se puede utilizar G.729A sin tener que pagar una cuota de licencia, sin embargo, es muy popular y está bien apoyado en muchos teléfonos y sistemas diferentes.

Para lograr su impresionante relación de compresión, este codec requiere una igualmente impresionante cantidad de esfuerzo por parte del CPU. En un sistema Asterisk, el uso de codecs muy comprimidos agotará rápidamente los recursos del CPU.

G.729A utiliza 8 Kbps de ancho de banda.

GSM

El códec Sistema Global para Comunicaciones Móviles (GSM) es el favorito de Asterisk. Este codec no viene gravado con un requisito de licencia de la manera que lo hace G.729A, y ofrece un excelente rendimiento con respecto a la demanda de CPU.

La calidad de sonido se considera generalmente que es de un grado menor que la producida por G.729A, pero gran parte de esto se reduce a opinión personal. GSM opera a 13 Kbps.

ILBC

El Internet Códec Low Bitrate (iLBC) ofrece una atractiva mezcla de bajo ancho de banda y calidad, y es especialmente adecuado para mantener una calidad razonable en los enlaces de red con pérdidas.

Naturalmente, Asterisk soporta iLBC, pero no es tan popular como el codecs ITU, y este puede ser compatible con los teléfonos IP comunes y sistemas de VoIP comerciales. IETF RFC 3951 y 3952 se han publicado en apoyo de iLBC y iLBC está en la pista de las normas IETF.

Debido a que iLBC utiliza algoritmos complejos para lograr sus altos niveles de compresión, tiene un costo bastante elevado en consumo de CPU en Asterisk.

Mientras que usted está autorizado para utilizarla iLBC sin pagar licencia, el titular de la patente iLBC, Global IP Sound (GIPS), quiere saber cada vez que lo utilice en una aplicación comercial. La forma de hacerlo es mediante la descarga y la impresión de una copia de la iLBC licencia, firmarlo y devolverlo a GIPS.

iLBC opera a 13.3 Kbps (frames de 30 ms) y 15,2 Kbps (frames de 20 ms).

Speex

Speex es un códec de tasa de bits variable (VBR), lo que significa que es capaz de modificar dinámicamente su tasa de bits para responder a las condiciones cambiantes de la red. Se ofrece en versiones tanto de banda estrecha y de banda ancha, en función de si quieres calidad telefónica o mejor.

Speex es un codec totalmente gratis, licenciado bajo la variante Xiph.org de la licencia BSD.

Un proyecto de Internet para Speex está disponible, y más información sobre Speex se puede encontrar en su página web (http://www.speex.org).

Speex puede funcionar a cualquier tasa de bits desde 2,15 hasta 22,4 Kbps, debido a su velocidad de bits variable.

G.722

G.722 es un códec estándar ITU-T que fue aprobado en 1998. El códec G.722 produce una voz de mucho más calidad en el mismo espacio que G.711 (64 Kbps) y está empezando a ser popular entre los fabricantes de dispositivos de VoIP. Las patentes de G.722 han caducado, por lo que es de libre acceso. Si tiene acceso a dispositivos que admiten G.722, usted quedará impresionado por la mejora de la calidad.

MP3

Seguro está pensando, MP3 es un codec. En concreto, es el Moving Picture Experts Group Audio Layer 3 estándar de codificación. Con un nombre así, no es de extrañar que lo llamemos MP3! En Asterisk, el codec MP3 se utiliza normalmente para la música en espera (MoH). MP3 no es un códec de telefonía, ya que está optimizado para la música, no la voz, sin embargo, es muy popular entre los sistemas de telefonía VoIP como un método de entrega de MoH.

Calidad de Servicios

QoS es un conjunto de tecnologías para administrar el tráfico de red de forma rentable a fin de optimizar la experiencia del usuario tanto en entornos empresariales como en hogares y oficinas pequeñas. Las tecnologías de QoS permiten medir el ancho de banda, detectar cambios en las condiciones de la red (por ejemplo, congestión o disponibilidad del ancho de banda) y clasificar el tráfico por orden de prioridad o limitarlo. Por ejemplo, se puede usar QoS para clasificar el tráfico por orden de prioridad en aplicaciones dependientes de la latencia (como las aplicaciones de voz o vídeo) y para controlar el impacto del tráfico dependiente de la latencia (como las transferencias masivas de datos).

Aunque no hay una regla fija, en general se acepta que si usted puede entregar el sonido producido por el altavoz para el oído del oyente a menos de 150 milisegundos, es posible un flujo normal de conversación. Si el retardo excede de 300 milisegundos, se hace difícil evitar la interrupción de uno al otro. Más allá de 500 milisegundos, una conversación normal se vuelve cada vez más difícil y frustrante.

Además de conseguir que la información llegue a tiempo, también es esencial garantizar que la información transmitida llegue intacta. Demasiados paquetes perdidos impedirán que en el otro extremo se reproduzca completamente el audio muestreado, en casos graves, palabras o frases enteras perdidas. Incluso la pérdida del 5 por ciento de paquetes puede obstaculizar gravemente una red VoIP.

Protocolo IP

El protocolo IP (Internet Protocol) es un protocolo que trabaja a nivel de red donde la información se envía en paquetes llamados paquetes IP. Este protocolo ofrece un servicio “sin garantías” también llamado del “mejor esfuerzo”. Es decir que nada garantiza que los paquetes lleguen a su destino, sin embargo se hará lo posible por hacerlos llegar.

Protocolos de Transporte

TCP, UDP y SCTP

Si usted va a enviar datos en una red basada en IP, serán transportados mediante una de los tres protocolos de transporte discutido aquí.

TCP (Transmission Control Protocol)

El protocolo de Control de Transmisión (TCP) es casi nunca usado para VoIP, como el protocolo IP no garantiza que los datos lleguen a su destino. Solo hace su mejor esfuerzo para que lleguen. Era necesario un protocolo que se encargue de controlar la transmisión de datos y por esta razón se crea el Protocolo de Control de Transmisión o TCP. TCP es un protocolo de transporte que se transmite sobre IP.

TCP ayuda controlando que los datos transmitidos se encuentre libre de errores y sean recibidos por las aplicaciones en el mismo orden en que fueron enviados. Si se pierden datos en el camino introduce mecanismo para que estos datos sean reenviados.

Esto implica carga extra de información en el flujo de datos ya que hay que enviar información de control adicional. Es por esto que TCP es un buen protocolo para control de sesiones pero no es bueno para transmisión de datos en tiempo real. Por este motivo la voz no se envía usando este protocolo.

UDP (User Datagram Protocol)

UDP es otro protocolo de transporte. La principal diferencia con TCP es que a UDP no le importa si los datos llegan con errores o no y tampoco le importa si llegan en secuencia. UDP divide la información en paquetes, también llamados datagramas, los paquetes se colocan en el medio los más rápidamente posible y se liberan para encontrar su camino al destino final, sin ninguna confirmación de si llegaron o no.

Al no ser necesario incluir mucha información de control, el protocolo UDP reduce la cantidad de información extra en los paquetes por lo que es un protocolo más rápido que TCP y adecuado para transmisión de información que debe ser transmitida en tiempo real como la voz.

Es por esta razón que la voz en aplicaciones de VoIP es transmitida sobre el protocolo UDP.

Stream Control Transmission Protocol (SCTP)

Fue Aprobado por el IETF como un estándar propuesto en el RFC 2960, SCTP es un protocolo de transporte relativamente nuevo. Desde el principio, fue diseñado para hacer frente a las deficiencias de los protocolos TCP y UDP, especialmente en relación con los tipos de servicios que solían ser entregados a través de redes de telefonía por conmutación de circuitos (circuit-switched).

Algunos de los objetivos de SCTP fueron:

- Mejorar las técnica para evitar la congestión (especialmente, evitar los ataques de denegación de servicios)
- Estricta secuencia de los datos enviados.
- Mínima latencia para mejorar la transmisión en tiempo real.

Capítulo 3

Introducción a Asterisk PBX

¿Qué es Asterisk?

Asterisk es un programa de software libre (bajo licencia GPL) que proporciona funcionalidades de una central telefónica con capacidad para voz sobre IP.

Asterisk incluye muchas características que anteriormente sólo estaban disponibles en costosos sistemas propietarios PBX, como buzón de voz, conferencias, IVR, distribución automática de llamadas, y otras muchas. Los usuarios pueden crear nuevas funcionalidades escribiendo un dialplan en el lenguaje de script de Asterisk o añadiendo módulos escritos en lenguaje C o en cualquier otro lenguaje de programación soportado en GNU/Linux.

Esto ha hecho que muchas empresas consideren a Asterisk como una opción seria al momento de planificar su proyecto telefónico y por esta razón Asterisk ha tenido gran acogida a nivel mundial.

Al ver la oportunidad de negocio muchos fabricantes se han sumado a ofrecer hardware telefónico compatible con Asterisk, principalmente tarjetas PCI para conexión con la PSTN y esto ha hecho que la oferta de centrales telefónicas basadas en Asterisk crezca en los últimos tiempos.

Breve historia de Asterisk

Asterisk fue desarrollado por Mark Spencer, para entonces estudiante de ingeniería de informática de la Universidad de Aubum, Alabama. Mark había creado en 1999 la empresa “Linux Support Services” con el objetivo de dar soporte a usuarios de GNU/Linux. Para ello necesitaba una central telefónica, pero ante la imposibilidad de adquirirla debido a sus elevados precios, decidió construir una con un PC bajo Linux, utilizando lenguaje C.

En 1999, cuando tuvo un código digno de mostrar al mundo decidió liberarlo bajo licencia GPL.

Mark se dio cuenta de que su software necesitaba interactuar con hardware telefónico y se topó con el proyecto Zaptel, un proyecto de código abierto creado por Jim Dixon, que tenía el objetivo de crear drivers abiertos para tarjetas telefónicas de computadoras. A partir de entonces Asterisk y Zaptel caminarían de la mano; tanto así que en la actualidad los dos proyectos son mantenidos por la misma compañía.

Para el año 2002 “Linux Support Services” se convertiría en “Digium”, redirigiendo sus objetivos al desarrollo y soporte de Asterisk.

El proyecto Zaptel

El proyecto fue concebido por Jim Dixom en el año 2000 cuyo nombre era una abreviatura de Zapata Telephony Project. Se le da este nombre al proyecto en honor a Emiliano Zapata, héroe de la revolución mexicana.

La Tecnología Zapata se basa en el concepto de que el hardware de computadora es ahora lo suficientemente rápido para manejar todo el procesamiento para aplicaciones de telefonía, incluyendo lo que se había limitado tradicionalmente a DSP y controladores embebidos.

DSP está optimizado para aplicaciones de procesamiento de señales y funciones, y lo hacen mucho más eficientemente que los procesadores tradicionales de propósito general, como los que se encuentran en las computadoras y sistemas informáticos estándar. A pesar de su seria ventaja en este tipo de aplicaciones, tienen muchas deficiencias logísticas que a menudo hacen que su uso se alejes del alcance de los mortales comunes. Estas deficiencias incluyen arquitecturas totalmente propietaria, tanto hardware como software, que requieren conocimientos y experiencia especializada, por no mencionar muy especializada (y sobre manera muy costosos) en el desarrollo de hardware y software.

La tecnología de Zapata intenta direccionar estas limitaciones.

Los drivers fueron liberados bajo licencia GPL de modo que cualquiera puede tener acceso al código.

En un principio los drivers contenido en zaptel eran para tarjetas Tormenta fabricada por Zapata Telecom, pero pronto Digium comenzó a mejorar los drivers y a extender el soporte para nuevos modelos de hardware y se convierte en el principal desarrollador de Zaptel.

Por otro lado a pesar de que Asterisk es un software poderoso no puede hacer todo el trabajo. Zaptel es uno de los paquetes de software que los complementa. Zaptel es un conjunto de drivers para controlar hardware telefónico con las tarjetas PCI que nos permiten conectarnos a la PSTN.

Estos drivers se comunican con Asterisk a través de un módulo de Asterisk llamado **chan_zap.so** que se configura a través del archivo **zapata.conf**.

Zaptel tiene su propio archivo de configuración, independiente de Asterisk, llamado **zaptel.conf** y está ubicado en la carpeta **/etc/.**

Luego de varios años de Digium mantener Zaptel, se da cuenta de que Zaptel era una marca registrada de Zapata Telecom y para evitar cualquier posible confusión decide cambiar de nombre a sus drivers, los cuales son ahora llamados DAHDI.

A partir de la versión 1.6 de Asterisk, Digium ha anunciado oficialmente que Zaptel no será más soportado.

Versiones de Asterisk

Existen múltiples versiones de Asterisk, incluyendo las versiones congeladas. Una vez que una serie se pone a disposición, obtiene soporte por algún periodo de tiempo. Durante el periodo inicial, el soporte incluye cambios para correcciones de errores que se han reportado. En algún punto, la serie liberada será obsoleta y únicamente se mantiene con correcciones para solucionar problema de seguridad. Finalmente, la versión llegara al final de su vida, donde no recibirá más soporte ni cambio de ningún tipo.

El tipo de versión define cuánto tiempo va a ser apoyada. Una versión LTS (Long Term Support) tendrá soporte total por 4 años, con un año adicional de mantenimiento para corregir problema de seguridad.

Las versiones estándar solo tienen soporte por un corto periodo de tiempo, que será por lo menos un año de apoyo y un año adicional de mantenimiento de parches de seguridad.

En la siguiente tabla se muestran las líneas de tiempo de liberación para todas las versiones de Asterisk, incluyendo aquellos que han alcanzado el final de la vida.

SERIES	TIPO	FECHA DE PUBLICACIÓN	SOLO REVISIÓN DE SEGURIDAD	EOL
1.2.X		2005-11-21	2007-08-07	2010-11-21
1.4.X	LTS	2006-12-23	2011-04-21	2012-04-21
1.6.0.X	Estándar	2008-10-01	2010-05-01	2010-10-01
1.6.1.X	Estándar	2009-04-27	2010-05-01	2011-04-27
1.6.2.X	Estándar	2009-12-18	2011-04-21	2012-04-21
1.8.X	LTS	2010-10-21	2014-10-21	2015-10-21
10.X	Standard	2011-12-15	2012-12-15	2013-12-15
11.x	LTS	2012-10-25	2016-10-25	2017-10-25
12.x	Standard	2013-10 (tentativo)	2014-10 (tentativo)	2015-10 (tentativo)
13.x	LTS	2014-10 (tentativo)	2018-10 (tentativo)	2019-10 (tentativo)

Nuevas versiones de Asterisk se harán más o menos una vez al año, alternando entre versiones estándar y LTS. Dentro de una serie de lanzamientos con soporte total, actualizaciones de corrección de errores se proporcionan aproximadamente cada 4 semanas.

Si no estás seguro de que versión utilizar, elija la última versión, la más actualizadas o la última versión LTS para una plataforma que puede tener menos funciones, pero por lo general será por más tiempo.

Preparando un Sistema Asterisk

El tamaño de un sistema Asterisk en realidad no está dictado por el número de usuarios o extensiones, sino más bien por el número de llamadas simultáneas que se espera que el sistema soporte. Tenga en cuenta que ninguna guía puede decir exactamente cuántas llamadas un servidor Asterisk puede manejar. Hay un número increíblemente alto de variables que pueden afectar al momento de darle una respuesta a la pregunta de cuántas llamadas Asterisk puede manejar. La única forma de averiguar cuántos llama un servidor puede manejar es probarlo usted mismo en su propio entorno. Así que siéntase libre para experimentar y ver lo que funciona para usted.

La tabla a continuación enumera algunas pautas muy básicas que usted querrá tener en cuenta al planificar su sistema.

PROPÓSITO	NO. DE CANALES	MÍNIMO RECOMENDADO
Sistema de Hobby o Prueba	No más de 5	400-Mhz x86,256 MB RAM
Sistema SOHO (pequeña oficina/Hogar menos de 3 líneas y 5 extensiones)	5 a 10	1-GHZ x86 512 MB RAM
Sistema negocio pequeños	Hasta 25	3-GHZ x86, 1GB RAM
Sistema Mediano a grande	Más de 25	CPUs Dual o posiblemente múltiples servidores distribuidos.

Con instalaciones grandes de Asterisk, es común implementar funcionalidad a través de varios servidores. Una o más unidades centrales estarán dedicadas a procesamiento de llamadas, las cuales se complementan con uno o más servidores auxiliares que manejan los periféricos (tales como un sistema de base de datos, un sistema de correo de voz, un sistema de conferencia, un sistema de gestión, una interfaz web, un servidor de seguridad, y así sucesivamente).

Selección del hardware del servidor

La selección de un servidor es a la vez simple y complicado: simple porque, en realidad, cualquier plataforma basada en x86 será suficiente, pero complicado porque el rendimiento fiable de su sistema dependerá del cuidado que se ponga en el diseño de la plataforma. Cuando este seleccionando su hardware, usted debe considerar cuidadosamente el diseño global del sistema y la funcionalidad que necesita que este soporte. Esto le ayudará a determinar los requisitos para CPU, motherboard y fuente de alimentación. Si simplemente está configurando su primer sistema Asterisk con el propósito de aprender, puede ignorar la información de esta sección. Sin embargo, si usted está construyendo un sistema de misión crítica adecuado para la implementación, se trata de cuestiones que requieren una reflexión.

Problemas de rendimiento

Entre otras consideraciones, a la hora de seleccionar el hardware para una instalación de Asterisk debe tener en cuenta una pregunta que es fundamental: ¿Que tan poderoso debe ser mi sistema? Esta no es una pregunta fácil de responder, porque la manera en que el sistema se valla a utilizar va a jugar un papel importante en los recursos que este consuma. No hay tal cosa como una matriz de desempeño de ingeniería de Asterisk, por lo que tendrá que entender cómo Asterisk utiliza el sistema con el fin de tomar decisiones inteligentes acerca de que tipos de recursos serán necesarios. Usted tendrá que tener en cuenta varios factores, entre ellos:

- El número máximo de conexiones concurrente que se espera que el sistema soporte. Cada conexión aumenta la carga de trabajo sobre el sistema.

- El porcentaje de tráfico que requieren uso intensivo del procesador DSP de codecs comprimido (tales como G.729 y GSM).
  El procesamiento de señal digital (DSP) que se realiza en el software Asterisk puede tener un impacto impresionante sobre el número de llamadas simultáneas que serán soportadas. Un sistema que podría felizmente manejar 50 llamadas simultáneas usando el códec G.711 se puede poner de rodillas por una solicitud de conferencia con 10 canales comprimidos usando el códec G.729.

- Si se va a proporcionar conferencia y que nivel de actividad de conferencia se espera. Si el sistema será usado fuertemente? Las conferencias requieren que el sistema codifique (transcodificación) cada stream de audio entrante en múltiples stream de audio saliente. Mesclar múltiples streams de audio en tiempo casi real puede poner una carga de trabajo significante al CPU.

- Cancelación de Eco. La cancelación de eco puede ser necesaria sobre cualquier llamada donde una interface de Red Telefónica Pública Conmutada (PSTN) esté involucrada. Ya que la cancelación de eco es una función matemática, el sistema tendrá que realizarla, mayor será la carga sobre el CPU.

- Lógica scripting en el Dialplan. Si Asterisk tiene que pasar el control de la llamada para programa externo, habrá una ligera pérdida de rendimiento. Si se usaran scripts externo, ellos deben ser diseñados con consideraciones críticas de rendimiento y eficiencia.

Como el impacto de rendimiento de estos factores, es difícil de saber con exactitud. El efecto de cada uno se conoce en términos generales, pero una calculadora de rendimiento preciso aún no ha sido definida con éxito. Esto es en parte debido a que el efecto de cada componente del sistema depende de numerosas variables, tales como la potencia del CPU, chipset del motherboard y sobre todo la calidad, la carga de tráfico total sobre el sistema, la optimización del Kernel de Linux, el tráfico de redes, cantidad y tipo de las interfaces PSTN, y el trafico PSTN, sin mencionar cualquier servicios de Asterisk que el sistema corra concurrentemente. Echemos un vistazo a los efectos de varios factores clave:

Codecs y Transcodificación

En poca palabra, un códec (es la forma corta de codificar/decodificar, o comprimir/descomprimir) es un conjunto de reglas matemáticas que define como una forma de onda analógica será digitalizada. La diferencia entre los diferentes codecs se debe en gran parte al nivel de compresión y calidad que ellos ofrecen. Generalmente hablando, a mayor compresión requerida, mayor será el trabajo que el DSP debe realizar para codificar y decodificar la señal. Un códec sin compresión, pone mucho menos tensión sobre el CPU, pero requiere mucho más ancho de banda. La selección de un códec debe ser estrictamente balanceada entre el ancho de bando y el uso de procesador.

Unidad Central de Proceso (y unidad de punto flotante)

Un CPU está compuesto por varios componentes, uno de los cuales es la unidad de punto flotante (FPU). La velocidad del CPU, junto con la eficiencia de su FPU, juega un papel significante en la cantidad de conexiones concurrentes que un sistema puede efectivamente soportar.

Otros procesos corriendo concurrentemente sobre el sistema

Similar a Unix, Linux está diseñado para ser capaz de realizar múltiples tareas en diferentes procesos. Un problema sucede cuando uno de estos procesos (tal como Asterisk) exige un nivel muy alto de la capacidad de respuesta del sistema. Por defecto, Linux distribuye los recursos equitativamente entre todas las aplicaciones que los soliciten. Si usted instala un sistema con muchas aplicaciones diferentes, cada una de estas aplicaciones se le permitirá un uso razonable del CPU. Pero como Asterisk requiere frecuentemente acceso de alta prioridad para el CPU, este no se llevara bien con otras aplicaciones, y si Asterisk debe coexistir con otras aplicaciones, el sistema puede requerir optimización especial. Esto primeramente involucra la asignación de prioridades para las varias aplicaciones en el sistema y, durante la instalación, especial atención para cuales aplicaciones serán instaladas como servicios.

Optimización del Kernel

Un Kernel optimizado para el rendimiento de una aplicación específica es algo que muy poca distribuciones de Linux ofrecen por defecto, y esto requiere pensar un poco. Como mínimo, --cualquiera que sea la distribución que usted selecciones—usted debe descargar o compilar sobre su plataforma una copia fresca del Kernel de Linux (disponible desde http://www.kernel.org). También puede ser capaz de adquirir parches que mejoran el rendimiento, pero esto es considerado hackeo a los kernels oficialmente soportado.

IRQ latencia

IRQ latencia es básicamente el retraso entre el momento que una tarjeta periférica (tal como una tarjeta de interface telefónica) solicita el CPU para que pare lo que está haciendo y el momento cuando el CPU responde y está listo para manejar la tarea. Periféricos de Asterisk (especialmente las tarjetas DAHDI) han sido históricamente intolerante a la latencia de la IRQ, aunque han habidos mejoras considerables de DAHDI para ayudar a mejorar estos problemas. Esto no se debe a ningún problema con las tarjetas, sino que más bien es parte de la naturaleza de cómo un motor de TDM basado en software tiene que trabajar.

Versión del Kernel

Asterisk es oficialmente soportado con Linux versión 2.6. Casi la totalidad de Asterisk en realidad no se preocupa por la versión del kernel, pero DAHDI si requiere la versión 2.6.

Distribución Linux

Las distribuciones de Linux son muchas y variadas. Asterisk debería funcionar en todas ellas. Elija la que usted se sienta más cómodo.

Directorio de Asterisk

Para la mayoría de las instalaciones de Asterisk, el cambio de los directorios no es necesario. Sin embargo, esto puede ser útil para ejecutar más de una instancia de Asterisk, al mismo tiempo, o si le gustaría almacenar los archivos en una ubicación diferente.

Asterisk organiza sus archivos en algunos directorios. En la tabla siguiente tenemos los más importantes.

DIRECTORIO	DESCRIPCIÓN
/etc/asterisk/	Contiene los archivos de configuración de Asterisk
/usr/lib/asterisk/modules/	Ubicación donde se almacenan los módulos cargables.
/var/lib/asterisk	La ubicación base para la información de estado de las variable utilizada por varias partes de Asterisk.
/usr/sbin/	Reside el binario de Asterisk
/var/log/asterisk/	Directorio donde Asterisk guarda sus archivos de log.
/var/lib/asterisk/agi-bin/	Residen los scripts AGI
/var/lib/asterisk/mohmp3	Contiene archivos de sonidos para la música en espera.
/var/lib/asterisk/sounds	Sonidos que Asterisk utiliza como prompts de voz.

/var/spool/asterisk/	Directorio donde Asterisk guarda archivos que genera producto de su funcionamiento como voicemail y grabaciones de llamadas.
/var/run/asterisk	Archivos con información de los ID de procesos (PID).
/var/log/asterisk/	Aquí residen los archivos de log de Asterisk como el /var/log/asterisk/full o el log de texto de CDRs

Asterisk CLI

Asterisk CLI es el nombre dado a la consola de Asterisk. Es decir es la línea de comando que nos permite controlar Asterisk directamente. El CLI es la mejor manera para ver que está pasando con su sistema, esta interface provee varios niveles de salida y ofrece una gran cantidad de utilidades para permitirle afectar el funcionamiento de su sistema.

Para ingresar CLI de Asterisk debemos ingresar el siguiente comando **asterisk –r** desde la consola de Linux. Como podemos notar obtenemos un prompt desde donde podemos ejecutar una seria de comandos que veremos en breve.

```
[root@localhost ~]# asterisk -r
Asterisk SVN-branch-11-r397157, Copyright (C) 1999 - 2013 Digium, Inc. and other
s.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for detail
s.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Running as user 'asteriskpbx'
Running under group 'asteriskpbx'
Connected to Asterisk SVN-branch-11-r397157 currently running on localhost (pid
= 2398)
localhost*CLI> _
```

El CLI también nos proporciona información en tiempo real de la actividad de Asterisk. Podemos controlar el grado de detalles con el que queremos ver dicha información con algunos comandos. A continuación algunos de los más usados.

```
localhost*CLI> core set verbose 8
Set remote console verbosity to 8
localhost*CLI> core set debug 7
Core debug is at least 7
localhost*CLI> _
```

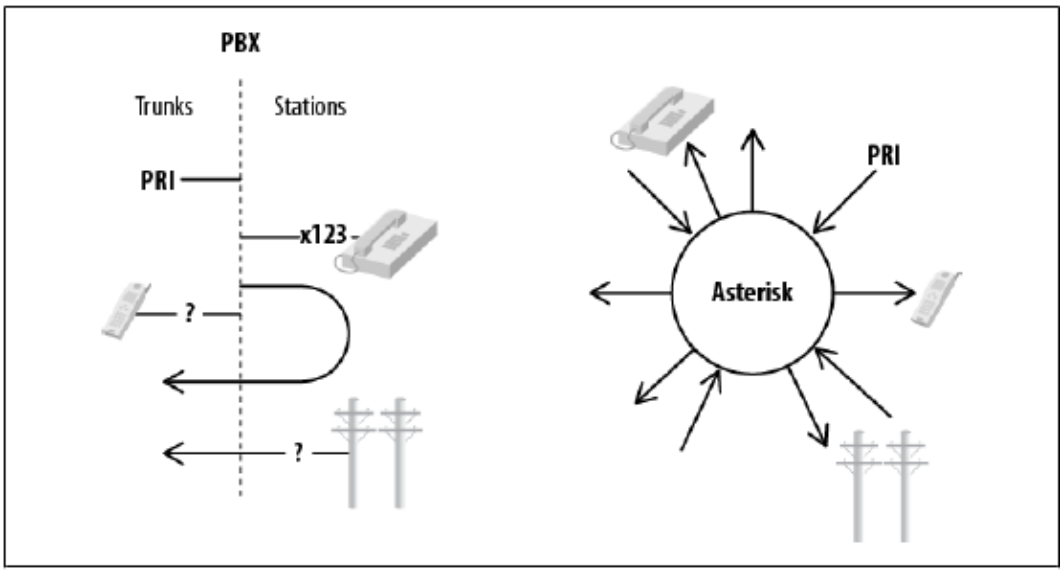
Capítulo 4

Arquitectura de Asterisk

Asterisk se diferencia de las mayorías de la PBX tradicionales, en que el dialplan de Asterisk trata todos los canales entrantes esencialmente de la misma manera.

En una PBX tradicional, hay una diferencia lógica entre estaciones (extensiones telefónicas) y trunks (recursos que conectan con el mundo exterior). Esto significa, por ejemplo, que no se puede instalar un gateway externo sobre un puerto de una estación y enrutar las llamadas externas sin necesidad de que los usuarios marquen el número de extensión primero. Además, el concepto de un recurso fuera del sitio (por ejemplo, un área de recepción) es mucho más difícil de implementar en una PBX tradicional, ya que el sistema no permitirá que los recursos externos tengan acceso a las funciones internas.

Asterisk, por otra parte, no tiene un concepto interno de troncos o estaciones. En Asterisk, todo lo que entra o sale del sistema pasa a través de un canal de algún tipo. Hay muchos tipos diferentes de canales, sin embargo, el dialplan de Asterisk maneja todos los canales de una manera similar, lo que significa que, por ejemplo, un usuario interno puede existir en el extremo de un tronco externo (por ejemplo, un teléfono celular) y ser tratado por el dialplan exactamente de la misma manera que ese usuario sería tratado si estuviera en una extensión interna. La imagen siguiente ilustra la diferencia entre estas dos arquitecturas.



Asterisk vs. Arquitectura PBX

Los Módulos

Asterisk es construido sobre módulos. Un módulo es un componente cargable que proporciona una funcionalidad específica, tal como un controlador de canal (por ejemplo, **chan_sip.so**), o un recurso que permite una conexión a una tecnología externa (tales como **func_odbc.so**). Los módulos de Asterisk son cargado basado en el archivo **/etc/asterisk/modules.conf.** Vamos a discutir el uso de muchos módulos en este libro. En este momento sólo queremos introducir el concepto de módulos, y darle una idea de los tipos de módulos que están disponibles.

En realidad, es posible iniciar Asterisk sin ningún módulo en absoluto, aunque en este estado no será capaz de hacer cualquier cosa. Es útil comprender la naturaleza modular de Asterisk con el fin de apreciar la arquitectura.

Los tipos de módulos en Asterisk incluyen los siguientes:

- De Aplicaciones
- Módulos Puentes
- Módulos de grabación de detalles de llamadas (CDR)
- Módulos de registro de eventos de Canales (CEL)
- Drivers de canales
- Traductores Codec
- Interprete de formato
- Funciones de Dialplan
- Módulos PBX
- Módulos de Recursos
- Módulos Addons
- Módulos de prueba

Vamos a echar un vistazo a los módulos, agrupados por tipo de módulo.

Módulos de Aplicaciones

Las aplicaciones de dialplan son usadas en el **extensions.conf** para definir las varias acciones que pueden ser aplicadas a las llamadas. La aplicación **Dial(),** por ejemplo, es responsable de realizar las conexiones salientes a los recursos externos y que es posiblemente la aplicación del dialplan más importante.

En la siguiente tabla veremos algunos módulos a modo de ejemplo, como el listado de módulos de aplicaciones es amplio solo listaremos algunos de los más populares.

NOMBRE	PROPÓSITO	ESTATUS
app_stack	Provee Gosub(),GoSubIf(),Return(), Stack, Pop(), LOCAL(), y LOCAL_PEEK()	Esencial
app_voicemail	Proporciona el correo de voz	Esencial
app_record	Graba el audio recibido para un archivo	Útil
app_read	Solicita la entrada de dígitos y asigna la entradas a una variables	Útil
app_sayunixtime	Reproduce el tiempo en un formato especifico	Útil
app_senddtmf	Transmite DTMF al que llama	Útil

app_softhangup	Solicita colgar el canal	Útil
app_transfer	Realiza una transferencia sobre el canal actual	Útil
app_cdr	Guarda los registros de CDR	Útil
app_chanspy	Permite escuchar desde un canal el audio en otro canal	Útil
app_directory	Presenta la lista de nombres de voicemail.conf	Útil
app_disa	Proporciona tono de marcado y acepta la entrada DTMF	Útil
app_meetme	Proporciona conferencia	Útil
app_queue	Provee distribución de llamadas automáticas (ACD)	Útil

Existen machismos más módulos de aplicaciones y que son bastante útiles y que a medida que sigamos avanzando veremos en más detalles.

Módulos Puentes

Los módulos puentes realizan el puente de canales en las nuevas API puente. Cada uno provee diferente funciones, los cuales son usados en diferentes situaciones dependiendo sobre que sea necesario un bridge o puente. Estos módulos, listados en la tabla siguiente, son actualmente únicamente usados por **app_confbridge**.

NOMBRE	PROPÓSITO
bridge_builtin_features	Realiza puenteo al utilizar las características incorporadas de usuario (como las que se encuentran en los features.conf).
bridge_multiplexed	Realiza multiplexación compleja, como la requerida en un gran sala de conferencias (múltiples participantes). Actualmente, sólo utilizado por app_confbridge.
bridge_simple	Realiza un puenteo simple de canal a canal.
bridge_softmix	Realiza multiplexación simple, como la requerida en un gran sala de conferencias (múltiples participantes). Actualmente, sólo utilizado por app_confbridge.

Módulos de Grabación de Detalles de Llamadas, CDR.

Los módulos CDR, están diseñados para facilitar todos los métodos de almacenamiento de registros de llamadas como sea posible. Usted puede almacenar CDRs para un archivo (método por defecto), una base de datos, RADIUS, o syslog. Sin embargo, estos no fueron pensados para ser usado en aplicaciones de facturación de llamadas complejas. Si usted requiere más control sobre la facturación y los reportes de llamadas, usted tendrá que buscar en el registro de eventos de canal. La ventaja del CDR es que son simples para trabajar. La tabla siguiente lista los módulos de grabación de detalles de llamadas.

NOMBRE	PROPÓSITO
cdr_adaptive_odbc	Permite escribir CDRs a través de framework ODBC con posibilidad de añadir campos personalizados.
cdr_csv	Escribe los CDR en el disco como un archivo de valores separados por comas.
cdr_custom	Como el anterior, pero permite la adición de campos personalizados
cdr_manager	Salidas CDRs a Asterisk Manager Interface (AMI)
cdr_odbc	Escribe los CDRs a través de un framework ODBC.
cdr_pgsql	Escribe los CDRs para PostgreSQL.
cdr_radius	Escribe los CDRs para RADIUS, no soporta campos personalizados.
cdr_sqlite	Escribe los CDRs para la base de datos SQLite2, este es obsoleto, use sqlite3_custom.
cdr_sqlite3_custom	Escribe los CDRs para la base de datos SQLite3, con campos personalizados.
cdr_syslog	Escribe los CDRs para syslog.
cdr_tds	Escribe los CDRs para Microsoft SQL o Sybase, requiere una versión vieja libtds.

Módulos de Registro de Eventos de Canales, CEL

Los registros de eventos de canales proveen un control mucho más poderoso sobre los reportes de actividades de llamadas. De la misma manera, se requiere planificar su dialplan con mucho más cuidado. Los modules CEL de Asterisk son listados en la tabla siguientes.

NOMBRE	PROPÓSITO
cel_custom	Escribe CEL para el disco en un archivo
cel_manager	CEL para Asterisk Manager Interface (AMI)
cel_odbc	Escribe los CELs a través de un framework ODBC
cel_pgsql	Escribe los CELs para PostgreSQL
cel_radius	Escribe los CELs para RADIUS, no soporta campos personalizados.
cel_sqlite3_custom	Escribe los CELs para la base de datos SQLite3
cel_tds	Escribe los CELs para Microsoft SQL o Sybase, requiere una versión vieja libtds.

Controladores de Canales

Sin drivers de canal, Asterisk no tendría manera de hacer llamadas. Cada controlador de canal específica el tipo de protocolo o tecnología que este soporta (SIP, ISDN, etc.). Los módulos de canales actúan como un gateway para el core o núcleo de Asterisk. Estos son listados en la tabla siguiente.

NOMBRE	PROPÓSITO
chan_agent	Proporciona un canal agente de Queue() (colas de llamadas)
chan_alsa	Proporciona conexión con Advanced Linux Sound Architecture
chan_bridge	Usado Internamente por la aplicación ConfBridge(); no se debe usar directamente.
chan_console	Ofrece conexión para un puerto de audio.
chan_dahdi	Proporciona conexión a tarjetas PSTN que utilizan controladores de canales DAHDI.
chan_gtalk	Conexión para Google Talk.
chan_h323	Proporciona conexión para terminales H.323; es ya obsoleto, usar chan_ooh323.
chan_iax2	Proporciona conexión para terminales IAX2.
chan_jingle	Ofrece conexión a los puntos finales de Jingle.
chan_local	Provee un mecanismo para tratar una porción del dialplan como un canal.
chan_mgcp	Controlador de canales Media Gateway Control Protocol.
chan_misdn	Proporciona conexión las tarjetas ISDN modular.
chan_multicast_rtp	Conexión para streams RTP multicast.
chan_nbs	Controlador de Canal Network Broadcast Sound; su uso es insignificantes
chan_oss	Controlador Open Sound System
chan_phone	Controlador de interfaz de telefonía Linux (muy viejo, es insignificante)
chan_sip	Controlador de Canales de SIP (Protocolo de Inicialización de sección)
chan_skinny	Controlador de canales Cisco Skinny Control Protocol (SCCP)
chan_unistim	Proporciona conexión para el protocolo controlador de canal UNISTIM de Nortel
chan_usbradio	Controlador de canal para tarjetas USB CM108 con interfaz de radio
chan_vpb	Controlador de canal Voicetronix; su uso es insignificante

Traductores Codec

Los traductores de códec les permiten a Asterisk convertir los formatos de stream de audio entre llamadas. Así que si se recibe una llamada en un circuito PRI (usando G.711) y debe ser pasada a un canal SIP comprimido (por ejemplo, usando G.729, uno de los muchos códecs que SIP puede manejar), el traductor códec pertinente realizará la conversión.

En la tabla siguiente se listan los diferentes traductores de códec y su propósito.

NOMBRE	PROPÓSITO
codec_adpcm	Adaptive Differential Pulse Coded Modulation códec; es insignificante.
codec_alaw	A-law PCM codec utilizado en todo el mundo (excepto Canadá/EE.UU.) en la PSTN; es esencial.
codec_a_mu	Convierte directamente A-law para mu-law.
codec_dahdi	Utilizado por las tarjetas de transcodificación propietaria de Digium.
codec_g722	Códec de audio de banda ancha.
codec_g726	Otro sabor de ADPCM; es insignificante.
codec_gsm	Codec Sistema Global para las Comunicaciones Móviles (GSM)
codec_ilbc	Internet Low Bitrate Codec; es insignificante.
codec_lpc10	Linear Predictive Coding vocoder (muy bajo ancho de banda); es insignificante.
codec_resample	Muestreo linear entre 8bit y 16 bit.
codec_speex	Speex códec.
codec_ulaw	Mu-law PCM codec usado en Canadá/USA en la PSTN; es esencial.

Intérpretes de Formatos

Los Intérpretes de formatos realizan la función de traductores codec, pero hacen su trabajo en los archivos en lugar de canales. Si usted tiene una grabación de un menú que se ha almacenado en formato GSM, un intérprete de formato tendría que ser utilizado para reproducir la grabación a cualquier canal que no utilice el codec GSM.

Si guarda una grabación en varios formatos (como WAV, GSM, etc.), asterisk determinar el formato de menor costo para usar cuando un canal requiere que se reproduzca la grabación.

NOMBRE	REPRODUCE ARCHIVOS ALMACENADOS EN:
format_g723	G.723 .g723
format_g726	G.726 .g726
format_g729	G.729 .g729
format_gsm	RPE-LTP (original codec GSM) .gsm
format_h263	H.263—video .h263
format_h264	H.264—video .h264
format_ilbc	Internet Low Bitrate Codec .ilbc
format_jpeg	Archivos gráfico .jpeg .jpg
format_ogg_vorbis	Ogg contenedor. Ogg
format_pcm	Varios PCM, formato: .alaw, .al, .alw, .pcm, .ulaw, .ul, .mu, .ulw, .g722, .au
format_siren14	G.722.1 Anexo C (14 kHz) .siren14

format_siren7	G.722.1 (7 kHz) .siren7
format_sln16	16-bit firmado lineal. Sln16
format_sln	8-bit firmado lineal .sln .raw
format_vox	.vox
format_wav	.wav
format_wav_gsm	Audio GSM en un contenedor WAV .WAV, .wav49

Funciones del Dialplan

Las funciones del dialplan, complementan las aplicaciones del dialplan. Proporcionan muchas mejoras útiles para cosas como manejo de cadenas, hora y fechas, y conectividad ODBC.

NOMBRE	PROPÓSITO
func_aes	Encripta/desencripta una cadena AES
func_audiohookinherit	Permite que las llamadas se graben después de ser transferida
func_base64	Codifica/decodifica una cadena base-64
func_blacklist	Escribe/lee la lista negra en AstDB
func_callcompletion	Obtiene/establece los parámetros de configuración de terminación de llamadas para el canal
func_callerid	Obtiene/establece el CallerID
func_cdr	Obtiene/establece variables CDR
func_channel	Establece/Obtiene información de canal
func_config	Incluye AST_CONFIG(); lee las variables del archivo config
func_connectedline	Cambios relacionados a la información de la línea conectada en los teléfonos compatibles
func_curl	Utiliza cURL para obtener datos desde un URI
func_cut	Divide y corta cadenas
func_db	Proporciona funciones Astdb
func_devstate	Obtiene el estado de un dispositivo
func_dialgroup	Crea un grupo de marcado simultaneo
func_dialplan	Valida que existe objetivo designado en el dialplan
func_enum	Realiza búsqueda ENUM
func_env	Incluye FILE( ), STAT(), y ENV( ); realiza acciones en el sistema operativo
func_extstate	Retorna el estado de una extensión dada.
func_global	Obtiene o establece las variables globales
func_groupcount	Obtiene o establece el número de canales para los miembros de un grupo

func_iconv	Convierte entre conjuntos de caracteres
func_lock	Incluyes LOCK(), UNLOCK(), and TRYLOCK(); establece un bloqueo para evitar condiciones anticipada en el dialplan
func_logic	Incluye ISNULL(), SET(), EXISTS(), IF(), IFTIME(), y IMPORT(); realizas varias funciones lógicas
func_math	Incluye MATH(), INC(), and DEC(); realiza funciones matemáticas
func_md5	Convierte una cada dada para un hash MD5
func_module	Chequea para ver si los módulos suministrados están cargado en memoria
func_odbc	Permite la integración del dialplan con recursos ODBC.
func_pitchshift	Cambia el tono de un stream de audio
func_rand	Devuelve un número aleatorio dentro de un rango dado
func_realtime	Realiza búsquedas en la Arquitectura Realtime Asterisk (ARA)
func_redirecting	Proporciona acceso a la información sobre, desde donde esta llamada fue redirigida
func_sha1	Convierte cadena suministrada a un hash SHA1
func_shell	Realiza operaciones de shell de Linux y devuelve los resultados
func_speex	Reduce el ruido y realiza la ganancia/pérdida de dB sobre un stream de audio
func_sprintf	Realiza funciones de formato de cadena similar a la función C del mismo nombre
func_srv	Realiza búsqueda SRV en el dialplan
func_strings	Incluye más de una docena de funciones de manipulación de cadenas
func_sysinfo	Obtiene información del sistema, como la memoria RAM, swap, medio carga, etc.
func_timeout	Obtiene/establece los tiempos de espera en el canal
func_uri	Convierte cadenas para codificación URI segura
func_version	Devuelve información sobre la versión de Asterisk
func_vmcount	Devuelve el número de mensaje en un folder voicemail para un usuario particular.
func_volume	Establece el volumen sobre un canal

Módulos PBX

Los módulos PBX son módulos periféricos que proporciona un mecanismo de mayor control y configuración. Por ejemplo, pbx_config es el módulo que cargar el dialplan tradicional. Los módulos PBX actualmente disponible son listados en la tabla siguiente.

NOMBRE	PROPÓSITO
pbx_ael	Asterisk Extensión Logic (AEL) ofrece un lenguaje de scripting para dialplan similar a un lenguaje de programación moderno
pbx_config	Este es el tradicional, y más popular, lenguaje de dialplan para Asterisk. Sin este módulo Asterisk no puede leer el archivo extensions.conf
pbx_dundi	Realiza búsqueda de datos sobre un sistema Asterisk remoto.
pbx_loopback	Realiza algo similar al include del dialplan, pero de una manera obsoleta
pbx_lua	Permite la creación de un dialplan usando el lenguaje de scripting Lua
pbx_realtime	Proporciona funcionalidad relacionada con la Arquitectura Asterisk Realtime.
pbx_spool	Proporciona soporte de spool saliente en relación con los archivos de llamadas de Asterisk.

Módulos de Recursos

Los módulos de recursos integran a Asterisk con los recursos externo. Por ejemplo, res_odbc permite a Asterisk interoperar con conexiones de base de datos ODBC. Los modules de recursos disponibles actualmente son listado en la siguiente tabla.

Nombre	Propósito
res_adsi	Proporciona ADSI
res_ael_share	Proporciona rutinas compartidas para su uso con pbx_ael
res_agi	Proporciona Asterisk Gateway Interface
res_ais	Proporciona indicación de mensaje en espera distribuido (MWI) y notificaciones del estado de dispositivo vía una implementación del estándar AIS, tal como OpenAIS.
res_calendar	Habilita la integración básica para sistemas de calendarios
res_calendar_caldav	Proporciona funciones de CalDAV específicos
res_calendar_exchange	Proporciona funciones de MS Exchange
res_calendar_icalendar	Proporciona funciones iCalendar Apple/Google
res_clialiases	Crea alias CLI
res_clioriginate	Origina una llamada desde CLI
res_config_curl	Extrae información de configuración usando Curl
res_config_ldap	Extrae información de configuración usando LDAP
res_config_odbc	Extrae información de configuración usando ODBC
res_config_pgsql	Extrae información de configuración usando PostgreSQL
res_config_sqlite	Extrae información de configuración usando SQLite

res_convert	Utiliza el CLI para realizar conversiones de archivos
res_crypto	Proporciona capacidades criptográficas
res_curl	Proporciona subrutinas comunes para otros módulos CURL
res_fax	Proporciona subrutinas comunes para otros módulos fax
res_fax_spandsp	Plug-in para fax usando los paquetes spandsp
res_http_post	Proporciona soporte de carga POST para servidores Asterisk HTTP
res_jabber	Proporciona recursos Jabber/XMPP
res_limit	Permite el ajuste de los límites del sistema sobre los proceso de Asterisk
res_monitor	Proporciona recursos de grabación de llamadas
res_musiconhold	Proporcionas recurso para música en espera (MOH)
res_mutestream	Permite silenciar/desactivar el silencio de los streams de audio
res_odbc	Proporciona subrutina comunes para otros módulos ODBC
res_phoneprov	Aprovisionamiento de teléfonos desde un servidor HTTP Asterisk
res_pktccops	Proporciona recursos PacketCable COPS
res_realtime	Proporciona comandos desde el CLI para la Arquitectura Realtime Asterisk (ARA)
res_rtp_asterisk	Provee RTP
res_rtp_multicast	Provee RTP multicast
res_security_log	Habilita el registro de seguridad
res_smdi	Proporciona notificaciones de correo de voz mediante el protocolo SMDI
res_snmp	Proporcionas información del estado del sistema para redes monitoreada con SNMP.
res_speech	API genérico de reconocimiento de voz
res_timing_dahdi	Proporciona el tiempo usando la interfaz del kernel DAHDI
res_timing_kqueue	Proporciona el tiempo utilizando una característica del kernel en algunos BSD, incluyendo Mac OS X
res_timing_pthread	Proporciona el tiempo utilizando sólo partes de la API pthread estándar, menos eficiente pero más portable que otros módulos el timing.
res_timing_timerfd	Proporciona el tiempo utilizando la API timerfd proporcionada por las nuevas versiones del kernel Linux

---

Módulos Addon

Los Módulos adicionales son módulos desarrolladas por comunidades con uso diferente o derechos de distribución diferente a los del código principal. Se mantienen en un directorio aparte y no se compilan e instalan de forma predeterminada. Para habilitar estos módulos, utilice el menuselect integrado con el utilitario de configuración. Los módulos Addon disponible son listados en la tabla siguiente.

NOMBRE	PROPÓSITO
app_mysql	Ejecuta consulta MySQL con una aplicación del dialplan; es obsoleto, use func_odbc.
app_saycountpl	Dice cuenta palabras polacas; es obsoleto – ahora es integrado con say.conf
cdr_mysql	Graba los registros de detalles de llamadas para una base de datos MySQL; se recomienda usar cdr_adaptive_odbc.
chan_mobile	Permite realizar y recibir llamadas usando teléfonos móviles a través de Bluetooth
chan_ooh323	Permite la realización y recepción de llamadas VoIP mediante el protocolo H.323
format_mp3	Permite a Asterisk reproducir archivos MP3
res_config_mysql	Utiliza una base de datos MySQL como backend de configuración en tiempo real

---

Módulos Test (o prueba)

Los módulos de prueba son utilizados por el equipo de desarrollo de Asterisk para validar código nuevo. Ellos son cambiados y agregados constantemente, y no son útiles a menos que usted esté desarrollando software para Asterisk.

Estructura de Archivo

Asterisk es un sistema complejo, compuesto de muchos recursos. Estos recursos hacen uso del sistema de archivos de varias maneras. Dado que Linux es muy flexible en este sentido, es útil comprender qué datos están siendo almacenados, de manera que puedas entender donde es posible que se encuentre, algunos de los datos particulares almacenados (por ejemplo, mensajes de correo de voz o archivos de registro).

Archivos de configuración

Los archivos de configuración de Asterisk incluyen extensions.conf, sip.conf, modules.conf, y docenas de otros archivos que definen los parámetros para los varios canales, recursos, módulos, y funciones que usted estará usando.

Estos archivos se encuentran en **/etc/Asterisk**. Usted va a trabajar mucho en esta carpeta, ya que permite configurar y administrar el sistema Asterisk.

Módulos

Los módulos de Asterisk son usualmente instalado en el directorio **/usr/lib/asterisk/modules**. Normalmente usted no tendrá que interactuar con este folder; sin embargo, ocasionalmente será útil saber dónde los módulos están ubicados. Por ejemplo, si usted está haciendo un upgrade a Asterisk y selecciona diferente módulos durante la fase de instalación de menuselect, los módulos viejos de la versión previa de Asterisk no serán borrados, y usted obtendrá una alerta desde el script de instalación. Los archivos viejos necesitan ser borrados del directorio modules. Esto puedo ser hecho manualmente o con “uninstall” haciendo (make uninstall) desde los fuentes de Asterisk.

Librería de Recursos

Hay varios recursos que requieren fuentes de datos externas. Por ejemplo, música en espera (MOH) no puede reproducirse a menos que tenga algo de música guardada. Los prompts del sistema también necesitan ser almacenado en algún lugar del disco duro. El directorio **/var/lib/asterisk** es donde los prompts del sistema, los scripts AGI, la música en espera, y otros archivos de recursos son guardados.

Spool

Es donde Linux almacenas los archivos que son cambiados con frecuencia, o serán procesados por otros procesos en un momento posterior. Por ejemplo, los trabajos de impresión de Linux y los correos pendientes son normalmente escritos para el spool hasta que ellos sean procesados.

En Asterisk, el spool es usado para almacenar objetos transitorios tales como mensajes de voz, grabaciones de llamadas, archivos de llamadas, y así sucesivamente.

El spool de Asterisk se encuentra bajo el directorio **/var/spool/Asterisk**.

Logging

Asterisk es capaz de generar varios tipos diferentes de archivos de log. El **/var/log/asterisk** es donde cosas tales como las grabaciones de detalles de llamadas (CDR), eventos de canales CEL, registros de depuración, registros de cola, mensajes, errores y otras salidas son escritas.

Este folder es extremadamente importante para cualquier esfuerzo de solución de problemas que usted se comprometa.

El Dialplan

El dialplan es el corazón de Asterisk. Todos los canales que llegan al sistema serán pasados a través del plan de marcado, que contiene la secuencia de comandos de flujo de llamadas que determina cómo se manejan las llamadas entrantes.

Un dialplan puede ser escrito en una de tres formas:

- Usando la sintaxis del dialplan tradicional en **/etc/asterisk/extensions.conf**
- Usando el Asterisk Extensión Logic (AEL) en **/etc/asterisk/extensions.ael**
- Usando LUA en **/etc/asterisk/extensions.lua**

Una vez que aprendes este lenguaje, es bastante fácil para la transición a AEL o LUA, si usted desea.

Hardware

Asterisk es capaz de comunicarse con un gran número de tecnologías diferentes. En general, estas conexiones son hechas a través de una conexión de redes; sin embargo, las conexiones para la mayoría de la tecnología de telecomunicaciones tradicionales, tal como la PSTN, requieren hardware específico.

Muchas compañías producen esto hardware, tales como Digium (patrocinador, propietario y desarrollador principal de Asterisk), Sangoma, Rhino, OpenVoX, Pika, Voicentronix, Junghanns, Dialogic,

Xorcom, beroNET, y muchos otros. Los hardwares más populares para Asterisk son generalmente diseñados para trabajar a través de Digium Asterisk Hardware Device Interface (mejor conocido como DAHDI). Estas tarjetas, todas tienen diferentes requisitos de instalación y diferentes ubicaciones de archivos.

Capítulo 5

Instalación de Asterisk

En este capítulo vamos a caminar a través de la instalación de Asterisk desde el código fuente. Muchas personas se apartan de este método, alegando que es demasiado difícil y consume tiempo. Nuestro objetivo es demostrar que la instalación de Asterisk desde el código fuente no es realmente tan difícil de hacer. Más importante aún, queremos ofrecerle la mejor plataforma de Asterisk en la cual aprender.

En este libro, le ayudaremos a construir un sistema Asterisk funcional desde cero. Este capítulo le proporciona las herramientas para construir una plataforma base para su sistema Asterisk. Teniendo en cuenta que vamos a hacer la instalación desde el código fuente, hay potencialmente una gran cantidad de variación en la forma en que usted puede hacer esto. El proceso que se discute aquí es el que hemos utilizado durante muchos años, siguiendo este le proporcionara los fundamentos adecuados para aprender y trabajar con Asterisk.

Como parte de este proceso, también vamos a explicar la instalación de algunas de las dependencias de software en la plataforma Linux que serán necesarios para temas que trataremos más adelante en este libro.

Vamos a mostrar las instrucciones para instalar Asterisk tanto sobre CentOS (que es una distribución basada Red Hat) y Ubuntu (una distribución basada en Debian), que creemos que cubre la gran mayoría de distribuciones de Linux que están instalada en la actualidad. Trataremos de mantener las instrucciones lo más general posible para que puedan ser útiles en cualquier distribución de Linux que escojas.

Hemos optado por instalar Asterisk en CentOS y Ubuntu, ya que son las opciones más populares, pero Asterisk generalmente se instalada en cualquier distribución. Incluso se instala en Solaris, BSD, Mac OS X, si lo deseas.

Instalación de Distribución

Porque Asterisk depende en gran medida de tener acceso prioritario al CPU, es esencial que instale Asterisk en un servidor sin interfaz gráfica, como el sistema de ventanas X (Gnome, KDE, etc.). Tanto CentOS y Ubuntu ofrecen una distribución GUI gratuita diseñada para el uso del servidor. Cubriremos las instrucciones para ambas distribuciones.

Servidor CentOS

CentOS significa “Sistema Operativo de la Comunidad Empresarial” (Community Enterprise Operating System), y está basado en Red Hat Enterprise Linux (RHEL). Para más información sobre CentOS, vea http://www.centos.org.

Usted necesita descargar un ISO desde la página web de CentOS, localizada en http://mirror.centos.org/centos/5/isos/. Seleccione el directorio i386 y x86_64 para hardware de 32 bits o 64 bits, respectivamente. Elija una de las imágenes, y se le presentará una lista de archivos para descargar. Comúnmente usted quera la primera selección disponible, la cual es el primer archivo ISO de un conjunto. Usted únicamente necesita el primer archivo ISO de la serie porque lo software adicional los instalaremos con **yum.**

Una vez que haya descargado el archivo ISO, grábelo en un CD o DVD e inicie el proceso de instalación. Si va a instalar en una máquina virtual (no se recomienda para uso en producción, pero puede ser una buena idea para probar Asterisk), usted debería ser capaz de montar el archivo ISO directamente e instalar desde allí.

Instalación básica del sistema

Inserte el disco (DVD) de instalación de CentOS 6, reinicie su computador para arrancar desde el CD/DVD y presiones **ENTER**.

En este punto la interfaz de instalación se iniciara. Se le preguntará si desee probar la integridad del medio de instalación. Estas instrucciones asumen que usted ya ha hecho esto, y por lo tanto puede saltarse este paso.

CentOS entonces le dará la bienvenida a la instalación. Presiones **ENTER** para continuar.

Selecciones su idioma y el idioma del teclado. Si está en Norte América, o simplemente se siente más cómodo con el idioma inglés. Es probable que sólo tenga que seleccionar los valores predeterminados. En mi caso he seleccionado el inglés como predeterminado.

Se le preguntara para inicializar la unidad, lo cual borrara todos los datos en su disco. Selecciones **YES (Si).**

El instalador le preguntara si desea remover el esquema de partición existente y crear uno nuevo. Selecciones **Remover todas las particiones** (Remove all partición) sobre la unidad seleccionada y cree un esquema por defecto. Si existe una opción más conveniente, podrá elegirla en su lugar. En la ventana de la unidad, compruebe que se ha seleccionado la unidad de disco correcto. (Presionando la tecla **TAB** se avanza por las diferentes opciones en la pantalla.) Una vez seleccionada la ventana de las unidad, puede desplazarse hacia arriba y hacia abajo (suponiendo que tiene varias unidades) y seleccionar el disco duro en el que desea instalar. Cambie las selecciones pulsando la barra espaciadora. Verifique que se ha seleccionado la unidad correcta, pulse **Tab** hasta que se resalte el botón **OK**, y pulse **Enter**.

Se presentará un mensaje para confirmar que desea eliminar todas las particiones Linux y crear el nuevo esquema de partición. Seleccione **YES (Si).**

Se le pedirá que revise el nuevo esquema de partición. Siéntase libre de modificar el esquema de partición si prefiere algo diferente, sin embargo, la respuesta **“NO”** por defecto no es una muy buena opción para su sistema en un ambiente de producción, use el esquema por defecto cuando los requisitos de almacenamiento serán mínimos.

Aparecerá un mensaje preguntando si desea configurar la interfaz de red **eth0** en su sistema. Selecciones **YES (Si).** Asegúrese de que la opción **Activate on boot** y activar las opciones de soporte IPv4 están habilitadas, a continuación, seleccione **Aceptar**.

Si su red proporciona aprovisionamiento de IP automática a través de DHCP, usted puede seleccionar **Aceptar.** De lo contrario, seleccione Configuración de la dirección manual, introduzca la información adecuada y, a continuación, seleccione **Aceptar**.

A continuación, se le pedirá que proporcione un nombre de host. Puede permitir que el servidor DHCP proporcione uno por usted (si su red asigna automáticamente nombres de host) o introduzca uno manualmente, seleccione **Aceptar**.

Se le presentará una lista de zonas horarias. Seleccione su zona horaria y seleccione **Aceptar**.

En este punto, se le pedirá la contraseña del usuario **root**. Escriba una contraseña segura y vuelva a escribirla para confirmar. Después de introducir su contraseña segura, seleccione **Aceptar**.

El Siguiente paso será la selección de paquetes. Varios paquetes que no necesitan instalarse (y que requieren archivos ISO adicionales que usted probablemente no ha descargado) se seleccionan de forma predeterminada. Quite la selección a todas las opciones en la lista con la barra espaciadora, a continuación, seleccione la opción de selección de software personalizada (**Customize**). Una vez que hayas hecho esto, seleccione **Aceptar**.

A continuación, se presenta una pantalla de selección de grupo de paquetes. Desplácese por la lista entera, anulando la selección de cada elemento. Si se seleccionan los paquetes, se le pedirá CDs adicionales que no se ha descargado. Instalaremos los paquetes adicionales con la aplicación **yum** después de instalar el sistema operativo. Una vez que haya anulado la selección de todos los paquetes, seleccione **Aceptar**.

Una comprobación de las dependencias se será realizada, y una confirmación de que la instalación está lista para empezar será presentada. Selecciones **Aceptar** para iniciar la instalación. El sistema de archivos entonces será formateado, la imagen de instalación transferida al disco duro, y la instalación de los paquetes del sistema se llevará a cabo. Tras la instalación, se le pedirá que reinicie su computador. Quite todos los medios de instalación de las unidades y seleccione la opción de **reinicio** (Reboot).

Actualización Básica del Sistema

Una vez que haya reiniciado el sistema, es necesario ejecutar el comando **yum update** para asegurarse de tener los últimos paquetes básicos. Para hacer esto, inicie sesión utilizando el nombre de usuario **root** y la contraseña que creó durante la instalación. Una vez iniciada la sesión, ejecute lo siguiente:

 # yum update

Is this ok [y/N]: y (Está de acuerdo [s / N]: s)

Cuando se le indique para instalar los paquetes más recientes, presiones **“y”** y espere a que los paquetes se actualicen. Si se le pide que acepte una clave GPG, presione **“y”.** Cuando se haya completado, reinicie el sistema, ya que es probable que el kernel se haya actualizado.

Si estás ejecutando CentOS a 64 bits, tendrá que eliminar todas las librerías de 32 bits de forma manual. Una vez que haya reiniciado, o justo antes de reiniciar el sistema, ejecute el comando siguiente:

 # yum remove *.i386 -y

Esto eliminará todas las librerías de 32 bits en su sistema de 64 bits, que de otro modo puede causar conflictos y problemas al compilar Asterisk y otro software.

¡Enhorabuena! Ha instalado y actualizado el sistema CentOS básico con éxito.

Habilitar NTP para sincronizar la hora del sistema

Mantener la hora exacta es esencial en el sistema Asterisk, tanto para el mantenimiento de los registros de detalles de llamadas precisos y para la sincronización con sus otros programas. Usted no querrá que las horas de las notificaciones del correo de voz presenten retraso por 10 ó 20 minutos, ya que esto puede llevar a confusión y pánico de aquellos que pudieran pensar que sus notificaciones de correo de voz se están tardando demasiado en ser entregada.

El comando **ntpd** se puede utilizar para asegurarse de que el tiempo en su servidor Asterisk se mantiene sincronizado con el resto del mundo:

```
# yum install ntp
...
Is this ok [y/N]: y        Está de acuerdo [s / N]: s
...
# ntpdate pool.ntp.org
# chkconfig ntpd on
# service ntpd start
```

Los valores por defecto que se envían con CentOS son suficientes para sincronizar la hora y mantener el tiempo de la máquina en sincronía con el resto del mundo.

Agregando Usuario al Sistema

El proceso de instalación del servidor de Ubuntu le pedirá que añada un usuario del sistema que no sea root, pero CentOS no hace esto. Para mantener la coherencia en el libro y que sea más seguro, vamos a agregar otro usuario del sistema y le proporcionaremos acceso **sudo**. Para agregar un nuevo usuario, ejecute el comando **adduser:**

```
# adduser pbxsystem
# passwd pbxsystem
Changing password for user pbxsystem.
New UNIX password:
Retype new UNIX password:
```

Ahora tenemos que proporcionar acceso sudo al usuario **pbxsystem**. Hacemos esto mediante la modificación del archivo **sudoers** con el comando **visudo**. Tendrá que instalar **visudo** la primera vez que lo utilice:

```
# yum install sudo
```

Con las aplicaciones y los archivos relacionados a sudo instalados, podemos modificar el archivo sudoers. Ejecute el comando visudo y busque las líneas que se muestran a continuación:

```
# visudo
## Allows people in group wheel to run all commands
%wheel    ALL=(ALL)       ALL
```

Con la línea **%wheel** sin comentario como se muestra en nuestro ejemplo, grabe el archivo presionando la tecla **ESC,** después escriba **:wq** y presione **Enter**. Ahora abra el archivo **/etc/group** en su editor favorito (nano es bien fácil de usar) y busque la línea que inicia con la palabra Wheel. Modifique esta línea como sigue:

wheel:x:10:root,pbxsystem

Grabe el archivo, cierre la sección del usuario root escribiendo **exit,** y habrá una nueva sección con el usuario **pbxsystem** que usted ha creado. Pruebe su acceso sudo ejecutando el siguiente comando:

$ sudo ls /root/
[sudo] password for pbxsystem:

Después de introducir su contraseña, usted debe conseguir la salida del directorio /root/. Si no lo hace, vuelva atrás y compruebe los pasos para asegurarse de que no salto o escribió nada mal. El resto de las instrucciones en este capítulo asume que el usuario **pbxsystem** tiene acceso sudo.

Una última cosa que se necesita hacer, lo que le permitirá introducir comandos sin tener que introducir la ruta completa. Por defecto sólo root tiene **/sbin/ y /usr/sbin/** en el PATH del sistema por defecto, pero nosotros podemos agregar nuestro usuario **pbxsystem** también ya que vamos a correr muchas aplicaciones localizada en estos directorios.

Comience abriendo el archivo oculto **.Bash_profile** ubicado dentro del directorio home **pbxsystem** con un editor. A continuación, vamos a añadir **:/usr/sbin :/sbin** en el final de la línea que inicia con PATH:

$ vim ~/.bash_profile
PATH=$PATH:$HOME/bin:/usr/sbin:/sbin

Como hicimos anteriormente, guarde el archivo con la tecla **ESC** y luego escriba **:wq** y presionar **ENTER**.

Con el sistema operativo instalado, está listo para instalar las dependencias necesarias para Asterisk. En la siguiente sección vamos a ver la instalación y preparación de Ubuntu server, por lo que pueden pasar directamente a la sección.

Versión GRATIS

Varios capítulos han sido omitidos

Para obtener la versión completa visita

https://payhip.com/b/2Qld | http://sncstore.blogspot.com/

El libro esta compuesto de 13 capítulo, 247 Pag.