ESCUELA POLITÉCNICA DEL EJÉRCITO

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

CARRERA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

IMPLEMENTACIÓN DE SERVICIOS SIP INTEGRADO AL SERVIDOR DE COMUNICACIONES 3COM NBX3000 DE LA ESCUELA POLITÉCNICA DEL EJÉRCITO

Previa a la obtención del título de:

INGENIERO DE SISTEMAS E INFORMÁTICA

POR: ILICH ALEXANDER ALMEIDA TORRES

PABLO JAVIER JAMI TAPIA

SANGOLQUÍ, Enero de 2009

CERTIFICACIÓN

Certifico que el presente trabajo fue realizado en su totalidad por los Sres. Alexander Almeida T. y Pablo Jami T., como requerimiento parcial a la obtención del titulo de INGENIERO DE SISTEMAS E INFORMÁTICA.

SANGOLQUÍ, 08 de Enero del 2009

ING CRISTÓBAL ESPINOSA

Profesor Director

DEDICATORIA

A Dios quien ha sido uno de mis pilares e inspiración en el desarrollo de este trabajo, a mis padres quienes con su cariño, guía y experiencia han sido la base fundamental de mi formación como profesional, pero principalmente como persona, a mis hermanos que con su ejemplo me alientan a seguir cada vez mas superándome en el plano profesional y en la vida misma, a mi sobrino que con su presencia me brinda esa vitalidad y ganas de ser un ejemplo para mi familia, a mi novia Amanda que con su amor y comprensión me ha sabido guiar y dar la fortaleza necesaria en cada momento para no rendirme, al resto de mi familia que nunca dejaron de creer en mi y me demuestran su cariño a cada instante, a mis amigos que me han dado muchos consejos, información, ayuda y aliento para la culminación de mi carrera, a mi compañero de tesis con quien hemos superado varios obstáculos pero nunca desistimos ni bajamos la cabeza y por ultimo pero no menos importante a mi director de tesis quien nunca me negó su apoyo y me guió no solo en lo técnico sino en todos los campos.

Pablo Javier Jami Tapia

DEDICATORIA

A mis Padres y Hermana por creer y confiar siempre en mí, apoyándome en todas las decisiones que he tomado en la vida, por darme la oportunidad de ser un profesional pero mas que todo por el amor, paciencia, cuidado y sabiduria que me brindaron en cada momento de mi vida.

A mis compañeros y compañeras, por el apoyo y motivación que de ellos he recibido, especialmente a mi companero de tesis que ha sido más que todo un gran amigo.

A mis maestros, por compartir sus amplios conocimientos y experiencia, ademas de brindarnos su amistad. Especialmente agradezco a mi director de tesis quien entrego su apoyo, tiempo y consejos de una manera desinteresada en todo momento.

A las personas que siempre estuvieron apoyandome para lograr éxito en mis objetivos y con quienes tuve la dicha de compartir muchos momentos que ahora forman parte de las situaciones y experiencias que han construido lo que hoy soy.

Ilich Alexander Almeida Torres

Índice de contenidos

LISTADO DE FIGURAS

CAPÍTULO I

1.- INTRODUCCIÓN

Voz sobre IP conocida como VoIP (Voice over Internet Protocol, voz sobre el Protocolo Internet) desde hace algunos años ha venido revolucionando la telefonía debido a que utiliza el Internet como medio lo que abarata los costos de las llamadas realizadas.

Además a nivel de organizaciones se está migrando desde las centrales telefónicas con tecnología analógica a centrales con tecnología digital y por último a centrales basadas en software que permiten aumentar la funcionalidad de la misma a la vez de reducir costos.

La oferta de VoIP para consumidores se está expandiendo rápidamente, y ahora incluye servicios tanto para clientes de telefonía fija como móvil.

Esta tecnología trabaja sobre varios protocolos que son los distintos lenguajes que utilizan los dispositivos VoIP entre los que tenemos:

H.323 - Protocolo definido por la ITU-T.

Megaco (También conocido como H.248) y MGCP - Protocolos de control.

SIP - Protocolo definido por la IETF.

Skinny Client Control Protocol - Protocolo propiedad de Cisco.

MiNet - Protocolo propiedad de Mitel.

CorNet-IP - Protocolo propiedad de Siemens.

IAX - Protocolo original para la comunicación entre PBXs Asterisk (obsoleto).

Skype - Protocolo propietario peer-to-peer utilizado en la aplicación Skype.

IAX2 - Protocolo para la comunicación entre PBXs Asterisk en reemplazo de IAX.

Jingle - Protocolo abierto utilizado en tecnología Jabber

Para el desarrollo de nuestro proyecto utilizaremos el protocolo SIP.

2.- JUSTIFICACIÓN

El desarrollo de este proyecto ayudara a la interconexión de oficinas dentro de la escuela a bajos costos, eliminando la compra de nuevas tarjetas para la escalabilidad dentro del servidor de comunicaciones 3COM modelo NBX3000 ya existente actualizándolo a Release 6.

Esta nueva posibilidad permitirá a los estudiantes contar con extensiones y sus propios casilleros de voz, así también como servicios personalizados como por ejemplo la posibilidad de saber cuando tengo un correo de voz, conservando todos los beneficios ofrecidos actualmente por el sistema ya implantado.

Será desarrollado de manera que se aproveche el servidor de comunicaciones 3COM NBX 3000 que posee la Escuela Superior Politécnica del Ejercito con sede en Sangolquí utilizando el protocolo SIP optimizándolo a Release 6, ampliando así la funcionalidad del mismo y haciendo un uso más eficiente del recurso que la institución ya posee actualmente.

Este servicio utilizara teléfonos abiertos, refiriéndose a que estos pueden trabajar con protocolos estandarizados en nuestro caso SIP y/o IAX eliminando la necesidad de depender de un proveedor especifico pala la adquisición de los equipos terminales.

Al ser el sistema basado en software libre, la escalabilidad no incurrirá en costos, ayudando así al crecimiento del sistema ya existente.

Se tendrá la oportunidad de utilizar softphones (Teléfonos virtuales) como una nueva alternativa de bajo costo.

3.- HIPOTESIS

Se busca escalabilidad del sistema menos costosa que la que se podría realizar con la tecnología existente. A través de extensiones SIP virtuales, sin tener que incurrir en nuevas compras de tarjetas físicas para la ampliación de la misma.

4.- OBJETIVOS

4.1- OBJETIVO GENERAL

Integrar un nuevo servidor Asterisk SIP de libre distribución emulando las funcionalidades del sistema cerrado 3COM NBX 3000 con el que actualmente cuenta la Escuela Politécnica del Ejército.

4.2- OBJETIVOS ESPECÍFICOS

- Agregar capacidad de escalabilidad a bajos costos para la red de VoIP.
- Implementar un sistema de VoIP que trabaje con protocolo SIP de manera que se autentifique teléfonos abiertos es decir que no manejen protocolos propietarios.
- Reconfigurar el sistema de extensiones telefónicas para integrar softphones y en general telefonía SIP a la red existente.
- Establecer un plan piloto de los dos sistemas.

5.- ALCANCE

Actualizar la central telefónica de la ESPE 3COM NBX 3000 a su Release 6.

Configurar un servidor de telefonía IP con protocolo SIP bajo el sistema operativo Linux Fedora core 6, para que este sea compatible con otros servidores de telefonía IP independientemente del protocolo que estos utilicen.

El proyecto lograra que el sistema cerrado actualmente se convierta en uno con características más flexibles y más escalables, planteara un sistema de marcaciones que pueda acoger a usuarios docentes, administrativos y estudiantes.

Acoplara nuevos servicios que permitan configurar cuartos de conferencia, consulta de buzón de correo vía Web y recepción de mensajes por Correo Electrónico.

6.- HERRAMIENTAS

- Un computador con sistema operativo Linux, que en nuestro caso será Linux Fedora core 6.
- Actualización 6ta de 3com.
- Archivos de instalación de Asterisk.
- Codecs para la reproducción del sonido.
- Codecs g723 y g729 para la posible incorporación de teléfonos SIP.

7.- METODOLOGÍA

Se utilizará una metodología que parte de la recopilación de información de ejemplos de servidores de telefonía IP.

8.- FACTIBILIDAD

8.1 FACTIBILIDAD TÉCNICA

Para el desarrollo de este trabajo de investigación existen varias fuentes de información, siendo la más importante el Internet que reúne soluciones a problemas ya detectados durante la creación de servidores y servicios similares.

8.2- FACTIBILIDAD OPERATIVA

Debido al previo funcionamiento de un servidor de telefonía IP con protocolo H323 (con variaciones propietarias por parte del fabricante que lo convierte en un protocolo cerrado), se cuenta con el apoyo y auspicio de la UTIC para el desarrollo del sistema planteado. (Ya que se conseguirá al final de este proyecto pasar de un sistema cerrado a un sistema abierto de comunicaciones).

CAPÍTULO II

MARCO TEÓRICO

2.1- Introducción

2.2- Historia de la Tecnología SIP y del servicio de VOIP

2.2.1- Términos y Definiciones

Definiciones sobre VOIP

VoIP

En español sus siglas significan Voz Sobre el Protocolo de Internet; llamado también como Telefonía IP o telefonía por Internet; es el enrutamiento de conversaciones de voz a través de Internet o cualquier otra red basada en el protocolo IP. El protocolo de internet (IP) fue diseñado originalmente como red para transmitir datos, y debido a su gran éxito fue adaptado como red de voz.

La transmisión de voz sobre IP (VoIP) puede facilitar muchos procesos y servicios que normalmente son muy difíciles y costosos de implementar usando la tradicional red de voz PSTN:

Se puede transmitir más de una llamada sobre la misma línea telefónica. De esta manera, la transmisión de voz sobre IP hace más fácil el proceso de aumentar líneas telefónicas cuando llegan nuevos usuarios a la red.

Funcionalidades que normalmente son facturadas con cargo extra por las compañías de teléfonos como: identificación de la persona que llama, transferencia de llamadas o remarcado automático, son fáciles de implementar con la tecnología de voz sobre IP y sin costo alguno.

Permite la integración de otros servicios disponibles en la red de Internet como son video, mensajes instantáneos, etc.

PSTN

En español sus siglas significan Red Pública de Telefonía Conmutada. Es la concentración de las redes públicas mundiales de circuitos conmutados, al igual que Internet es la concentración de redes públicas mundiales de paquetes conmutados basados en IP.

ISDN

En español sus siglas significan Red Digital de Servicios Integrados. Es un tipo de sistema de telefonía en red de circuitos conmutados diseñados para permitir la transmisión digital de voz y datos sobre los cables telefónicos de cobre comunes, lo que implica una mejor calidad y mayor velocidad que la disponible con los sistemas analógicos.

PBX

Centralita o también llamada Central Telefónica para Negocios Privados. Es una central telefónica propiedad de una empresa privada, a diferencia de la central que es propiedad de un operador de telecomunicaciones o de una empresa de telefonía.

ASTERISK

Es una aplicación de software libre (bajo licencia GPL) que proporciona funcionalidades de una central telefónica (PBX). Como cualquier PBX, se puede conectar un número determinado de teléfonos para hacer llamadas entre sí e incluso conectar a un proveedor de VoIP o bien a una RDSI.

IVR

En español sus siglas significan Respuesta Interactiva de Voz. Es un sistema informático que permite que una persona, típicamente quien llama por teléfono, seleccione una opción de un menú de voz y se interconecte con otro sistema computarizado.

DID

En español sus siglas significan Discado Directo Interno, en Europa es conocido como DDI. Es una función que ofrecen las empresas de telefonía para usar con la centralita telefónica de sus clientes mediante la cual la empresa de telefonía asigna un rango de números conectados a la centralita de su cliente.

RFC

En español sus siglas significan Petición de Comentarios. Es uno dentro de una serie de documentos informativos numerados de Internet y estándares que tanto el software comercial y el freeware y las comunidades Unix siguen ampliamente.

Teléfono VoIP

El teléfono VoIP, permite al usuario hacer llamadas a cualquier otro teléfono por medio de la tecnología de voz sobre IP (VoIP). De esta manera, la voz es transmitida sobre la red de internet, en lugar del sistema tradicional PSTN.

Un teléfono VoIP puede ser un sencillo teléfono basado en software, ó un aparato telefónico que se asemeja mucho a un teléfono común.

SIP - Session Initiation Protocol

En español sus siglas significan Protocolo de Inicio de Sesión. Es un protocolo estándar desarrollado por el Grupo de Trabajo IETF MMUSIC propuesto para iniciar, modificar y terminar una sesión de usuario interactiva que implica elementos multimedia, tal como video, voz, mensajería instantánea, juegos en línea y realidad virtual, entre otros.

Esta es una descripción general de lo que el protocolo es, más adelante se explica más a fondo el funcionamiento del mismo.

Historia del protocolo SIP

El 22 de febrero de 1996 Mark Handley y Eve Schooler presentaron al IETF un borrador del Session Invitation Protocol conocido ahora como SIPv1. El mismo estaba basado en trabajos anteriores de Thierry Turletti como INRIA Videoconferencing System o IVS y de Eve Schooler como Multimedia Conference Control o MMCC.

La principal fortaleza que SIPv1 tenía y la misma que fue heredada por la versión actual de SIP, era el concepto de registración, que consiste en que un usuario informa a la red dónde; en qué host de Internet; podía recibir invitaciones a conferencias.

El mismo día el Dr. Henning Schulzrinne presentó un borrador del Simple Conference Invitation Protocol (SCIP), el cual se encontraba estaba basado en el HTTP (Hypertext Transport Protocol) y usaba TCP (Transmission Control Protocol) como protocolo de transporte.

Para poder identificar a los usuarios utilizaba direcciones de correo electrónico, esto para permitir el uso de una misma dirección para recibir correos electrónicos e invitaciones a conferencias multimedia.

El IETF decidió combinar ambos en un único protocolo denominado Session Initiation Protocol, dando un nuevo significado a las siglas del acrónimo, y su número de versión fue el dos, 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 "proposed standard" en la [RFC 2543] publicado 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 para posteriormente en Junio de 2002 publicar 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.

Diseño del protocolo

El protocolo SIP fue diseñado por el IETF con el concepto de "caja de herramientas", es decir, el protocolo SIP se vale de las funciones aportadas por otros protocolos, las que da por hechas y no vuelve a desarrollarlas. Este es el motivo del porque SIP funciona en colaboración con otros muchos protocolos.

El protocolo SIP se concentra en el establecimiento, modificación y terminación de las sesiones, se complementa, entre otros, con el SDP, que describe el contenido multimedia de la sesión, por ejemplo qué direcciones IPs, puertos y codecs se usarán durante la comunicación. También se complementa con el RTP (Real-time Transport Protocol). RTP

es el verdadero portador para el contenido de voz y video que intercambian los participantes en una sesión establecida por SIP.

Otro concepto importante en su diseño es el de extensibilidad. Esto significa que las funciones básicas del protocolo, definidas en la RFC 3261, pueden ser extendidas mediante otras RFC (Requests for Comments) dotando al protocolo de funciones más potentes.

Las funciones básicas del protocolo incluyen:

- Determinar la ubicación de los usuarios, proveyendo nomadicidad.
- Establecer, modificar y terminar sesiones multipartitas entre usuarios.

El protocolo SIP adopta el modelo cliente-servidor y es transaccional. El cliente realiza peticiones (requests) que el servidor atiende y genera una o más respuestas, esto dependiendo de la naturaleza o método de la petición. Por ejemplo para iniciar una sesión el cliente realiza una petición con el método INVITE en donde indica con qué usuario (o recurso) quiere establecer la sesión. El servidor responde ya sea rechazando o aceptado esa petición en una serie de respuestas. Las respuestas llevan un código de estatus que brinda información acerca de si las peticiones fueron resueltas con éxito o si se produjo un error. La petición inicial y todas sus respuestas constituyen una transacción.

Los servidores, por defecto, utilizan el puerto 5060 en TCP (Transmission Control Protocol) y UDP (User Datagram Protocol) para recibir las peticiones de los clientes SIP.

Uno de los principales objetivos de SIP fue aportar un conjunto de funciones de procesamiento de llamadas y capacidades presentes en la red pública conmutada de telefonía. A partir de este objetivo, implementó funciones típicas de dicha red, como son:

llamar a un número, provocar que un teléfono suene al ser llamado, escuchar la señal de tono o de ocupado. La implementación y terminología en SIP son diferentes.

SIP también implementa muchas de las más avanzadas características del procesamiento de llamadas de SS7, aunque los dos protocolos son muy diferentes. SS7 es altamente centralizado, caracterizado por una compleja arquitectura central de red y unos terminales tontos (los tradicionales teléfonos de auricular). SIP es un protocolo peer to peer (p2p). Como tal requiere un núcleo de red sencillo y altamente escalable, con inteligencia distribuida en los extremos de la red, incluida en los terminales, ya sea mediante hardware o software. Muchas características de SIP son implementadas en los terminales a diferencia de las tradicionales características de SS7, que son implementadas en la red.

Aunque existen muchos otros protocolos de señalización para VoIP, SIP se caracteriza porque sus promotores tienen sus raíces en la comunidad IP y no en la industria de las telecomunicaciones. SIP ha sido estandarizado y dirigido principalmente por el IETF mientras que el protocolo de VoIP H.323 ha sido tradicionalmente más asociado con la Unión Internacional de Telecomunicaciones.

SIP es similar a HTTP y comparte algunas características en sus principios de diseño:

- Es legible por humanos y sigue una estructura de petición-respuesta. Los promotores de SIP afirman que es más simple que H.323. Aunque originalmente uno de los objetivos fue simplicidad ahora tiene una complejidad similar a H.323.
- SIP comparte muchos códigos de estado de HTTP, como el familiar '404 no encontrado' (404 not found).

- SIP y H.323 no se limitan a comunicaciones de voz y pueden mediar en cualquier tipo de sesión comunicativa desde voz hasta vídeo o futuras aplicaciones todavía sin realizar.

2.2.2- Delimitación

2.2.3- Origen de Asterisk

La aplicación Asterisk fue desarrollada por Mark Spencer, por entonces estudiante de ingeniería informática en la Universidad de Auburn, Alabama.

Mark había creado en 1999 la empresa "Linux Support Services" con el objetivo de dar soporte a usuarios de Linux. Para ello necesitaba una centralita telefónica, pero ante la imposibilidad de adquirirla dados sus elevados precios, decidió construir una con un PC bajo Linux, utilizando lenguaje C.

Figura 2.1: Mark Spencer en el 2006 O'Reilly Emerging Telephony Conference.

Este fue el principio del fenómeno mundialmente conocido como Asterisk®, la centralita telefónica construida por Mark después de su experiencia desarrollando GAIM (ahora llamado Pidgin) entre otros proyectos de software libre.

Mark explico sus necesidades de capital a sus amigos en Adtran ellos se ofrecieron a invertir en su compañía. Se dio cuenta que recibía más interés en el PBX Asterisk que por sus servicios generales de consultoría Linux.

Mark se reunió con Jim Dixon que estaba construyendo hardware open source. Su primer proyecto fue construir una tarjeta T1 open source. Estos ingresos les mantenían a flote pero no recibían contribuciones de nadie y el resto tan solo tomaban sus diseños y manufacturaban tarjetas que competían con las suyas.

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

El dinero era escaso en Digium hasta que un día un vendedor de DeltaCom (una competitiva compañía de comercio local) entró para venderles a Mark y a Jim una T1. Después de entender lo que Mark y Jim habían hecho el vendedor se ofreció a ayudarles. A partir de este punto empezaron a ver un incremento en las ventas, y acabaron el año con beneficios. Después de grandes ingresos durante largo tiempo Mark fue capaz de hacer crecer el negoció sin recabar mucho en los beneficios.

Cuando Mark empezó con Asterisk hizo una cosa muy inteligente. Se le requería firmar un acuerdo a cada desarrollador que contribuía en el código para que el copyright se asignara a Asterisk y el compromiso que no hay encumbramientos en el código contribuido. Esto le permitió sentirse confortable con su proyecto que era completamente open source y que su compañía podría re licenciar el código a vendedores como 3COM y NTT. Digium también ha hecho las cosas bien al mantener la versión de la comunidad con la funcionalidad completa y no crear una escisión entre ellos y los que los apoyan.

La primera Release fue Asterisk 0.1, y esta ocupaba tan sólo 124.3K que una vez descomprimido venían a ser unos 506 KB en 96 archivos. Para correr Asterisk necesitábamos básicamente Linux y libaudiofile.

Metodología de Desarrollo del proyecto

El modelo de desarrollo se basa en el uso del sistema de control de versiones Subversion y en un procedimiento de informe de errores denominado Asterisk Bug Tracker, el cual cuenta a su vez con un sistema "de méritos", denominado Karma, en el que aparecen los colaboradores en un ranking, de acuerdo con una puntuación (positiva o negativa) otorgada a los aportes que han realizado.

Además se utilizan las habituales herramientas de este tipo de proyectos, como listas de correo, IRC, o documentación on line.

Estructura organizativa actual del proyecto

Mark Spencer es el organizador y principal desarrollador, apoyado por un grupo de colaboradores que reciben el nombre de "administradores". Los administradores realizan principalmente labores de programación y control del software generado. Existe también un amplio grupo de programadores, llamados "managers" que pueden aportar soluciones a errores documentados o crear nuevas funcionalidades. Por último están los denominados "reporters", todos aquellos colaboradores que realizan informes sobre errores detectados.

Toda nueva funcionalidad es probada exhaustivamente antes de formar parte del repositorio del sistema de control de versiones y ha de contar finalmente con el visto bueno de los responsables de los repositorios, de acuerdo a criterios de oportunidad, prioridad o importancia de la nueva funcionalidad propuesta.

Industria relacionada

Existen multitud de empresas relacionadas con Asterisk. La mayor parte de ellas siguiendo uno de los modelos de negocio más habituales del software libre, como es el de aportar valor añadido al software, en este caso mediante el diseño, instalación, formación y mantenimiento de centralitas telefónicas basadas en Asterisk.

Digium, la empresa creada por Mark Spencer, amplia este modelo de negocio tanto con la venta de hardware específico, fundamentalmente tarjetas de comunicación, como con la venta de software propietario, entre el que destaca el "Asterisk Business Edition", aplicación basada en Asterisk a la que se le incorporan ciertas funcionalidades.

Estado actual del proyecto

La versión estable de Asterisk está compuesta por los módulos siguientes:

- Asterisk: Ficheros base del proyecto.
- Zaptel: Soporte para hardware. Drivers de tarjetas.
- Addons: Complementos y añadidos del paquete Asterisk. Opcional.
- Libpri: Soporte para conexiones digitales. Opcional.
- Sounds: Aporta sonidos y frases en diferentes idiomas.

Cada módulo cuenta con una versión estable y una versión de desarrollo. La forma de identificar las versiones se realiza mediante la utilización de tres números separados por un punto. Teniendo desde el inicio como primer número el uno, el segundo número indica la versión, mientras que el tercero muestra la revisión liberada. En las revisiones se llevan a cabo correcciones, pero no se incluyen nuevas funcionalidades.

En las versiones de desarrollo el tercer valor siempre es un cero, seguido de la palabra "beta" y un número, para indicar la revisión.

Versiones

Las versiones tanto estables como de desarrollo de cada módulo pueden descargarse en la zona de descargas de la página oficial de Asterisk.

A fecha de Abril de 2008 son las siguientes:

- Versión 1.6 (en pruebas)
    - Asterisk Versión 1.6.0-rc6

- Versión 1.4 Estable
    - Asterisk Version 1.4.21.2
    - Zaptel Version 1.4.12.1
    - Libpri Version 1.4.7
    - Addons Version 1.4.7
- Versión 1.2 y 1.0
    - Estas versiones se consideran paralizadas y no se continuarán manteniendo.

Nota: Actualmente la rama 1.4 es la aconsejada para sistemas en producción.

2.2.4- Protocolos y Codecs

Protocolos de Señalización

- Estos códigos se utilizan para Loguearse / desloguearse de un servidor VoIP remoto.
- Transmitir las claves de inicio, fin, corte, ok, invitación.
- Indicar que tipo de datos se están transmitiendo.
- Transmitir la dirección de origen y destino.

Los códigos de señal más importantes son H323 y SIP. Adicionalmente ha de mencionarse el código IAX desarrollado para troncales entre centrales telefónicas IP desarrolladas por Asterisk.

H.323

Fue diseñado con el objetivo principal de proveer a los usuarios con tele-conferencias las cuales poseen capacidades de voz, video y datos sobre redes de conmutación de paquetes.

El estándar fue diseñado específicamente con los siguientes objetivos:

Basarse en los estándares existentes, incluyendo H.320, RTP y Q.931.

Incorporar algunas de las ventajas que las redes de conmutación de paquetes ofrecen para transportar datos en tiempo real.

Solucionar la problemática que plantea el envío de datos en tiempo real sobre redes de conmutación de paquetes.

Los diseñadores de H.323 lo definieron de tal manera que las empresas que manufacturan los equipos pueden agregar sus propias especificaciones al protocolo y pueden definir otras estructuras de estándares que permiten a los dispositivos adquirir nuevas clases de características o capacidades.

SIP

Es el más difundido y dominante. SIP (Session Initiation Protocol) es un protocolo de señalización para conferencia, telefonía, presencia, notificación de eventos y mensajería instantánea a través de Internet. El propósito de SIP es la comunicación entre dispositivos multimedia. SIP hace posible esta comunicación gracias a dos protocolos que son RTP y SDP. El protocolo RTP se usa para transportar los datos de voz en tiempo real, de la misma forma que para el protocolo H323, mientras que el protocolo SDP se usa para la negociación de las capacidades de los participantes, tipo de codificación, etc.

CODECS

Un Códec convierte una señal analógica a una digital para transmitirla sobre una red de datos. Los siguientes Codecs están en uso hoy en día:

- GSM - 13 Kbps (full rate), tamaño de marco de 20ms
- iLBC - 15Kbps, tamaño de marco 20ms: 13.3 Kbps, tamaño de marco 30ms
- ITU G.711 - 64 Kbps, basado en muestras. También conocido como alaw/ulaw
- ITU G.722 - 48/56/64 Kbps
- ITU G.723.1 - 5.3/6.3 Kbps, tamaño de marco 30ms
- ITU G.726 - 16/24/32/40 Kbps
- ITU G.728 - 16 Kbps
- ITU G.729 - 8 Kbps, tamaño de marco 10ms
- Speex - 2.15 to 44.2 Kbps
- LPC10 - 2.5 Kbps
- DoD CELP - 4.8 Kbps

2.2.5- SoftPhones

En inglés la combinación de Software y de Telephone, es un software que hace una simulación de teléfono convencional por computadora. Es decir, permite usar la computadora para hacer llamadas a otros softphones o a otros teléfonos.

Normalmente, un Softphone es parte de un entorno Voz sobre IP y puede estar basado en el estándar SIP/H.323 o ser privativo.

Hay muchas implementaciones disponibles, como la ampliamente disponible Microsoft Windows Messenger o NetMeeting.

Funcionan bien con la mayoría de los ITSP - Proveedores de Servicios de Telefonía por Internet. Se puede conectar usando un teléfono USB o un enlace USB a un SoftPhone y obtener un servicio gratuito VoIP de teléfono a teléfono.

2.3- Funcionamiento

2.3.1- Instalación y configuración de software en Linux.

La instalación y configuración se lo detallara en el punto 3.2.1.1.

2.3.2- Características destacadas del Protocolo SIP

Características

SIP describe la comunicación necesaria para establecer una llamada telefónica.

El protocolo es parecido al protocolo HTTP, es basado en texto, y muy abierto y flexible. En muchos casos esta reemplazado el estándar H.323.

"SIP hace uso de elementos llamados servidores proxy para ayudar a enrutar las peticiones hacia la localización actual del usuario, autenticar y autorizar usuarios para darles servicio, posibilitar la implementación de políticas de enrutamiento de llamadas, y aportar capacidades añadidas al usuario." Tomado del RFC 3261.

"SIP también aporta funciones de registro que permiten al usuario informar de su localización actual a los servidores proxy." Tomado del RFC 3261.

"Es un concepto importante que la distinción entre los tipos de servidores SIP es lógica y no física." Tomado del RFC 3261.

Elementos de una Red SIP práctica

Los terminales físicos, dispositivos con el aspecto y forma de teléfonos tradicionales, pero que usan SIP y RTP para la comunicación, están disponibles comercialmente gracias a muchos fabricantes. Algunos de ellos usan numeración electrónica (ENUM) o DUNDi para traducir los números existentes de teléfono a direcciones SIP usando DNS (Domain Name Server), así llaman a otros usuarios SIP saltándose la red telefónica, con lo que un proveedor de servicio normalmente actúa de camino hacia la red pública conmutada de telefonía para los números de teléfono tradicionales (cobrando el uso de la red).

Hoy en día, ya son habituales los terminales con soporte SIP por software. Microsoft Windows Messenger usa SIP y en Junio de 2003 Apple Computer anunció y publicó en fase beta su iChat, una nueva versión compatible con el AOL Instant Messenger que soporta charlas de audio y vídeo a través de SIP.

SIP también requiere proxy y elementos de registro para dar un servicio práctico. Aunque dos terminales SIP puedan comunicarse sin intervención de infraestructuras SIP (razón por la que el protocolo se define como punto-a-punto), este enfoque es impracticable para un servicio público. Hay varias implementaciones de softswitch que pueden actuar como proxy y elementos de registro.

TELEFONOS SIP

Los teléfonos SIP son lo mismo que los teléfonos VoIP o los teléfonos basados en software (soft phones). Estos son teléfonos que permiten hacer llamadas utilizando tecnología VoIP (voice over internet protocol) ó de voz sobre internet.

Hay dos tipos de teléfonos SIP:

- **Teléfonos SIP basados en hardware,** el cual es similar a un teléfono tradicional pero el cual pude hacer y recibir llamadas utilizando Internet en vez del sistema PSTN tradicional.

- **Teléfonos SIP basados en software**, estos permiten que cualquier computador pueda ser utilizado como teléfono, usando un auricular con micrófono y una tarjeta de sonido. Se requiere también de una conexión de banda ancha a un proveedor VOIP, o a un Servidor SIP.

Clases de teléfono SIP/teléfonos VOIP

Un sistema telefónico VOIP requiere el uso de teléfonos SIP/teléfonos VOIP. Los teléfonos SIP se presentan en varias clases:

Teléfonos virtuales SIP/VOIP – teléfono SIP basado en software

Un teléfono SIP basado en software, es un programa que utiliza el micrófono y los altavoces de su ordenador, o auriculares conectados, para permitirle realizar o recibir llamadas. Ejemplos de teléfonos SIP son: Xlite, SJPhone de SJlabs (http://www.sjlabs.com), Xten (http://www.xten.net), o el Teléfono VOIP de 3CX para Windows. OBS - Aquí habría que poner información del Xlite que utilizamos.

Figura 2.2: Pantalla de plataforma del softphone X-Lite

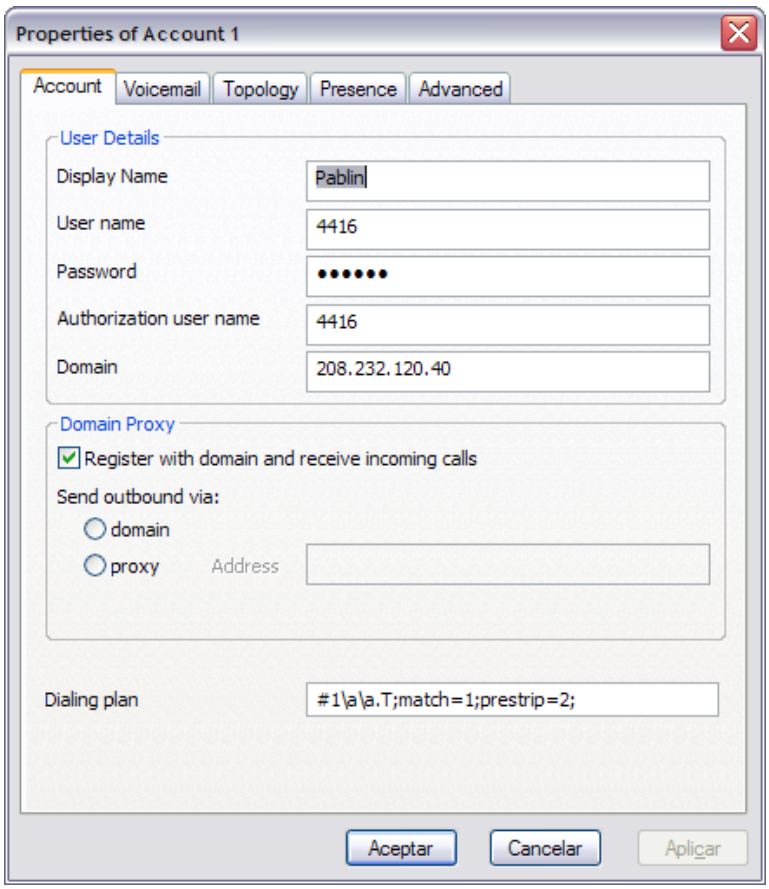


Figura 2.3: Pantalla de configuración de cuenta para el softphone X-Lite

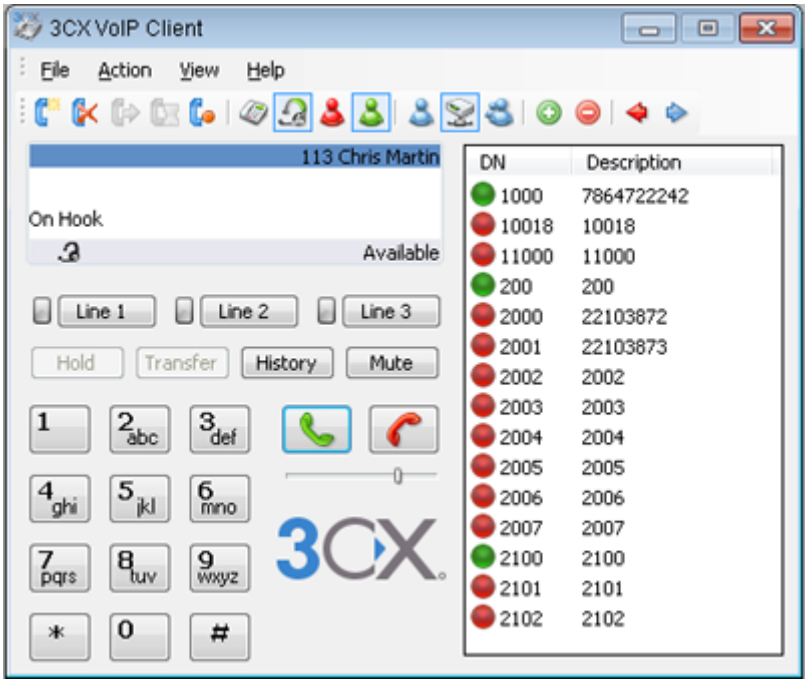


Figura 2.4: Teléfono Virtual de 3CX

Teléfonos VOIP con USB

Un teléfono con USB se enchufa al puerto USB de un ordenador y mediante el uso de un software para teléfono VOIP/SIP actúa igual que un teléfono. Básicamente no es más que un micrófono con un altavoz, sin embargo, como tiene la apariencia de teléfono normal, para el usuario es más intuitivo de utilizar.

Figura 2.5: Teléfono VoIP con USB

Teléfono SIP basado en hardware

Un teléfono SIP basado en hardware tiene la apariencia de un "teléfono" normal y actúa como tal. Sin embargo, se conecta directamente a la red de datos. Estos teléfonos tienen un mini concentrador integrado para que puedan compartir la conexión de red con el ordenador. De esa forma, no se necesita un punto de red adicional para el teléfono.

Figura 2.6: Teléfono SIP basado en hardware

Adaptador ATA

Si se desea usar su teléfono actual normal con el sistema telefónico VOIP, puede usar un adaptador ATA. Un adaptador ATA le permite enchufar la clavija de red Ethernet en el adaptador y luego enchufar el teléfono en el adaptador. De esa forma, su teléfono antiguo aparecerá en el software del sistema telefónico VOIP como un teléfono SIP normal.

Figura 2.7: Adaptador ATA que permite que un teléfono analógico se conecte a un sistema VOIP.

2.3.2- Proceso de Comunicación y Métodos SIP

Funcionamiento del protocolo

El protocolo SIP permite el establecimiento de sesiones multimedia entre dos o más usuarios. Para hacerlo se vale del intercambio de mensajes entre las partes que quieren comunicarse.

Agentes de Usuario

Los usuarios, que pueden ser seres humanos o aplicaciones de software, utilizan para establecer sesiones lo que el protocolo SIP denomina "Agentes de usuario".

Estos son los puntos extremos del protocolo, es decir son los que emiten y consumen los mensajes del protocolo SIP, los que pueden ser un videoteléfono, un teléfono, un cliente de software (SoftPhone) y cualquier otro dispositivo similar. El protocolo SIP no se ocupa de la interfaz de estos dispositivos con el usuario final, sólo se interesa en los mensajes que estos generan y cómo se comportan al recibir determinados mensajes.

Los agentes de usuario se comportan como clientes (UAC: User Agent Clients) y como servidores (UAS: User Agent Servers). Son UAC cuando realizan una petición y son UAS cuando la reciben. Por esto los agentes de usuario deben implementar un UAC y un UAS.

Además de los agentes de usuario existen otras entidades que intervienen en el protocolo, estos son los Servidores de Registro o Registrar, los Proxy y los Redirectores.

Servidores de Registro o Registrar

El protocolo SIP permite establecer la ubicación física de un usuario determinado, refiriéndose con esto a en qué punto de la red está conectado. Para ello se vale del mecanismo de registro. Este mecanismo funciona como se explica aquí:

Cada usuario tiene una dirección lógica que es invariable respecto de la ubicación física del usuario. Una dirección lógica del protocolo SIP es de la forma usuario@dominio es decir tiene la misma forma que una dirección de correo electrónico.

La dirección física, denominada "dirección de contacto", es dependiente del lugar en donde el usuario está conectado, es decir de su dirección IP.

Cuando un usuario inicializa su terminal (por ejemplo conectando su teléfono o abriendo su software de telefonía SIP) el agente de usuario SIP que reside en dicho terminal envía una petición con el método REGISTER a un Servidor de Registro, informando a qué dirección física debe asociarse la dirección lógica del usuario. El servidor de registro realiza entonces dicha asociación, denominada binding.

Esta asociación tiene un período de vigencia y si no es renovada, caduca. También puede terminarse mediante un desregistro. La forma en que dicha asociación es almacenada en la

red no es determinada por el protocolo SIP, pero es vital que los elementos de la red SIP accedan a dicha información.

Servidores Proxy y de Redirección

Para direccionar un mensaje entre un agente de usuario cliente y un agente de usuario servidor normalmente se recurre a los servidores. [ ] Estos servidores pueden actuar de dos maneras:

- Como Proxy, encaminando el mensaje hacia su destino.
- Como Redirector (Redirect) generando una respuesta que indica al origen la dirección del destino o de otro servidor que lo acerque al destino.

La principal diferencia es que el servidor proxy queda formando parte del camino entre el UAC y el/ los UAS, mientras que el servidor de redirección una vez que indica al UAC cómo encaminar el mensaje ya no interviene más.

Un mismo servidor puede actuar como Redirector o como Proxy dependiendo de la situación.

Formato de los mensajes

Los mensajes que se intercambian en el protocolo SIP pueden ser peticiones o respuestas.

Las peticiones tienen una línea de petición, una serie de encabezados y un cuerpo.

En la línea de petición se indica el propósito de la petición y el destinatario de la petición.

Las peticiones tienen distintas funciones. El propósito de una petición está determinado por lo que se denomina el Método (Method) de dicha petición, que no es más que un identificador del objetivo que la petición tiene. En la [RFC 3261] se definen los métodos básicos del protocolo. Existen otros métodos definidos en extensiones al protocolo SIP.

Las respuestas tienen una línea de respuesta, una serie de encabezados y un cuerpo.

En la línea de respuesta se indica el código de estado de la respuesta que es un número indica el resultado del procesamiento de la petición.

Los encabezados de peticiones y respuestas se utilizan para diversas funciones del protocolo relacionadas con el encaminamiento de los mensajes, autenticación de los usuarios, entre otras. La extensibilidad del protocolo permite crear nuevos encabezados para los mensajes agregando de esta manera funcionalidad.

El cuerpo de los mensajes es opcional y se utiliza entre otras cosas para transportar las descripciones de las sesiones que se quieren establecer, utilizando la sintaxis del protocolo SDP.

Flujo de establecimiento de una sesión

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

Un usuario ingresa la dirección lógica de la persona con la que quiere comunicarse, puede indicar al terminal también las características de las sesión que quiere establecer (voz, voz y video, etc.), o estas pueden estar implícitas por el tipo de terminal del que se trate. El agente de usuario SIP que reside en el terminal, actuando como UAC envía la petición (en este caso con el método INVITE) al servidor que tiene configurado. Este servidor se vale

del sistema DNS para determinar la dirección del servidor SIP del dominio del destinatario. El dominio lo conoce pues es parte de la dirección lógica del destinatario. Una vez obtenida la dirección del servidor del dominio destino, encamina hacia allí la petición. El servidor del dominio destino establece que la petición es para un usuario de su dominio y entonces se vale de la información de registración de dicho usuario para establecer su ubicación física. Si la encuentra, entonces encamina la petición hacia dicha dirección. El agente de usuario destino si se encuentra desocupado comenzará a alertar al usuario destino y envía una respuesta hacia el usuario originante con un código de estado que indica esta situación (180 en este caso). La respuesta sigue el camino inverso hacia el originante. Cuando el usuario destino finalmente acepta la invitación, se genera una respuesta con un código de estado (el 200) que indica que la petición fue aceptada. La recepción de la respuesta final es confirmada por el UAC originante mediante una petición con el método ACK (de Acknowledgement), esta petición no genera respuestas y completa la transacción de establecimiento de la sesión.

Normalmente la petición con el método INVITE lleva un cuerpo donde viaja una descripción de la sesión que quiere establecer, esta descripción es realizada con el protocolo SDP.[6] En ella se indica el tipo de contenido a intercambiar (voz, video, etc.) y sus características (códecs, direcciones, puertos donde se espera recibirlos, velocidades de transmisión, etc.). Esto se conoce como "oferta de sesión SDP". La respuesta a esta oferta viaja, en este caso, en el cuerpo de la respuesta definitiva a la petición con el método INVITE. La misma contiene la descripción de la sesión desde el punto de vista del destinatario. Si las descripciones fueran incompatibles, [ ] la sesión debe terminarse (mediante una petición con el método BYE).

Al terminar la sesión, lo que puede hacer cualquiera de las partes, el agente de usuario de la parte que terminó la sesión, actuando como UAC, envía hacia la otra una petición con el método BYE. Cuando lo recibe el UAS genera la respuesta con el código de estado correspondiente.

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

Ejemplo de sesión de llamada SIP entre 2 teléfonos

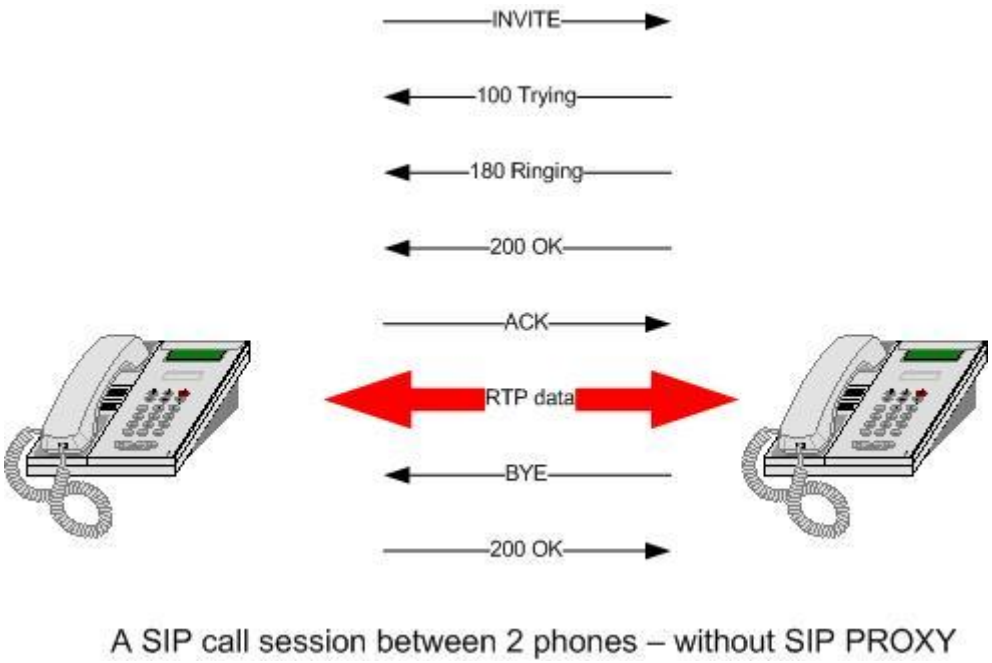


Figura 2.8: Ilustración de peticiones y respuestas generados en llamada SIP entre 2 Teléfonos

Una sesión de llamada SIP entre 2 teléfonos es establecida como sigue:

- El teléfono llamante envía un “invite”

- El teléfono al que se llama envía una respuesta informativa 100 – Tratando – retorna.
- Cuando el teléfono al que se llama empieza a sonar una respuesta 180 – sonando – es retornada.
- Cuando el receptor levanta el teléfono, el teléfono al que se llama envía una respuesta 200 – OK
- El teléfono llamante responde con un ACK – confirmado
- Ahora la conversación es transmitida como datos vía RTP
- Cuando la persona a la que se llama cuelga, una solicitud BYE es enviada al teléfono llamante
- El teléfono llamante responde con un 200 – OK.

Es tan simple como eso. El protocolo SIP es fácil de entender y es lógico.

METODOS, SOLICITUDES Y RESPUESTAS SIP

SIP utiliza Métodos / Solicitudes y correspondientes Respuestas para establecer una sesión de llamada.

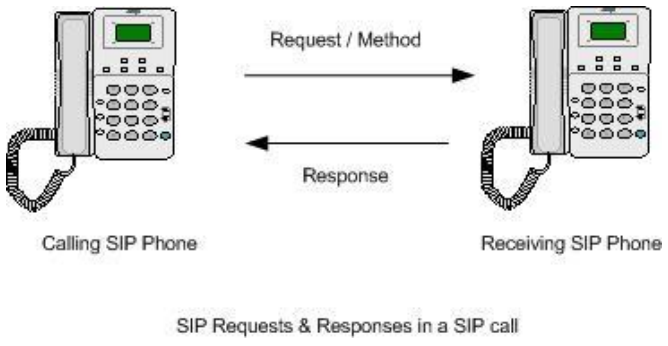


Figura 2.9: Ilustración de solicitud y respuesta entre teléfonos SIP

Solicitudes SIP:

Hay seis tipos de métodos / solicitudes:

- INVITE = Establece una sesión
- ACK = Confirma una solicitud INVITE
- BYE = Finaliza una sesión
- CANCEL = Cancela el establecimiento de una sesión
- REGISTER = Comunica la localización de usuario (nombre de equipo, IP)
- OPTIONS = Comunica la información acerca de las capacidades de envío y recepción de teléfonos SIP

Respuestas SIP:

Las solicitudes SIP son respondidas con respuestas SIP, de las cuales hay 6 clases:

- 1xx = respuestas informativas, tal como 180, la cual significa teléfono sonando
- 2xx = respuestas de éxito
- 3xx = respuestas de redirección
- 4xx = errores de solicitud
- 5xx = errores de servidor
- 6xx = errores globales

Note la similitud con HTTP. La belleza de SIP está en su claridad y simplicidad.

2.3.3- Aplicaciones con uso de Protocolo SIP

Figura 2.10: Logotipos de Aplicaciones con uso de Protocolo SIP

Clientes SIP (User Agents)

Multi-Platform

- SFLphone**:** Desarrollado en Linux pero fue exportado a otras plataformas.
- Linphone**:** Cliente VoIP con voz, video y mensajería de texto instantánea. Estable en Linux y ha sido reportado su funcionamiento con FreeBSD y OpenBSD.
- Minisip**:** Desarrollado por estudiantes de Doctorado y Masterado de Royal Institute of Technology (KTH, Stockholm, Sweden). Corre sobre multiples sistemas operativos como Linux PC, Linux familiar IPAQ PDA, Windows XP and son Windows Mobile 2003 SE.

- OpenWengo**:** Softphone integrable con contactos IM. Disponible para Linux, MacOSX, and Windows.

- PhoneGaim**:** Viene incluido con portátiles Linspire.

- sipXtapi**:** Librerías de cliente y software development kit (SDK) para SIP. Disponible para Linux, MacOSX, and Windows.

- OpenZoep**:** Desarrollado por Voipster, Softphone y mensajería instantánea. Soporta P2P.

Linux

- Cockatoo**:** Proyecto para la implementación de SIP/SIMPLE.

- YeaPhone**:** El objetivo de este es funcionar conjuntamente con Yealink USB handset (USB-P1K) y al mismo tiempo hacer innecesarios el teclado y monitor del computador.

- Twinkle**:** SIP softphone.

Windows

- 1videoConference**:** Web, Audio/Video teléfono. Usuarios Skype, Msn y Yahoo pueden participar en conferencias Web sin largas descargas ni complicadas instalaciones.

SIP Proxies

- Open Source SIP**:** Creado en Marzo 2006, auspiciado por Solegy, producto de cerca de 6 años de investigación y desarrollo.

- Partysip**:** Aplicación modular. Puede ser usada como SIP registrar, SIP servidor de redirección o servidor completo.

- MjSip**:** Implementación SIP completamente en Java que provee una API para el protocolo. MjSip incluye todas las clases y métodos para la creación de aplicaciones basadas en SIP.

- OpenSER**:** Proyecto de código abierto que permite desarrollar un robusto y escalable servidor SIP.

- SIP Express Router**:** De alto rendimiento, configurable y gratuito servidor SIP. Este actúa como registrar, proxy o servidor de redirección. Basado en Web.

- Siproxd**:** Proxy y enmascarador para protocolo SIP que maneja registraciones de clientes SIP en una red IP privada. Permite a los clientes SIP trabajar detrás de un firewall o router con IPs enmascaradas.

Clientes SIP en teléfonos móviles

Los celulares Nokia de última generación tales como los Nokia N80 Edición de Internet, E65, N95 entre otros tienen integrados clientes SIP, que pueden ser configurados para trabajar con cualquier servicio SIP nativamente desde el teléfono. Los demás celulares que no cuentan con este servicio pero que si le podemos instalar cuentan con aplicaciones como Fring el cual soporta comunicación con servicios SIP, como Gizmo, VoipStunt, VoipDiscount, VoipCheap, NetAppel, VoipBuster, etc.

Aplicaciones móviles más populares

- **Gimo:** Uno de las aplicaciones SIP más populares. Disponible en muchas plataformas, incluyendo Windows, Mac, Linux y Symbian Serie60. Hay una aplicación Java disponible en Gizmo5.com. Aunque recomiendan utilizar en su lugar la aplicación nativa de Gizmo que puedes descargar de la página de Nokia.

- **Truphone:** Truphone suele ser la aplicación de VoIP favorita, ya que entre otras ventajas, se integra fácilmente con la agenda personal del móvil. Otra de las ventajas es que permite cambiar entre diferentes redes automáticamente así como utilizar la conexión Wifi o 3G según la cobertura.

- **EQO:** Es capaz de utilizar la red GSM para realizar una llamada VoIP, por lo que puedes recibir llamadas VoIP y, en el caso de que no tengas cobertura Wifi o 3G, recibirás una llamada GSM. Además de esta ventaja, permite mensajería instantánea (IM) como MSN, Yahoo y GoogleTalk.

- **Skype:** El cliente de VoIP tan popular de PC ahora ya está disponible para móviles con Symbian.

- Fring**:** Tiene todas las ventajas que se han dicho en las anteriores aplicaciones, además de ser compatible con Skype, ICQ, permitir roaming entre 3G y Wifi y el envío de archivos P2P entre usuarios Fring.

2.3.4- Servidores Asterisk

A través del tiempo los servidores Asterisk han evolucionado a tal punto que compañías enteras se dedican a la comercialización de Asterisk listo con hardware de conexión para distintos protocolos como SS7, SIP o el mismo IAX que es el protocolo nativo de Asterisk.

2.3.5- Mensajería de Voz

Una pieza fundamental de las comunicaciones telefónicas es la mensajería de voz, ya que con esto se permite a los usuarios tener una comunicación permanente, con las personas que le llaman.

2.4- Topología

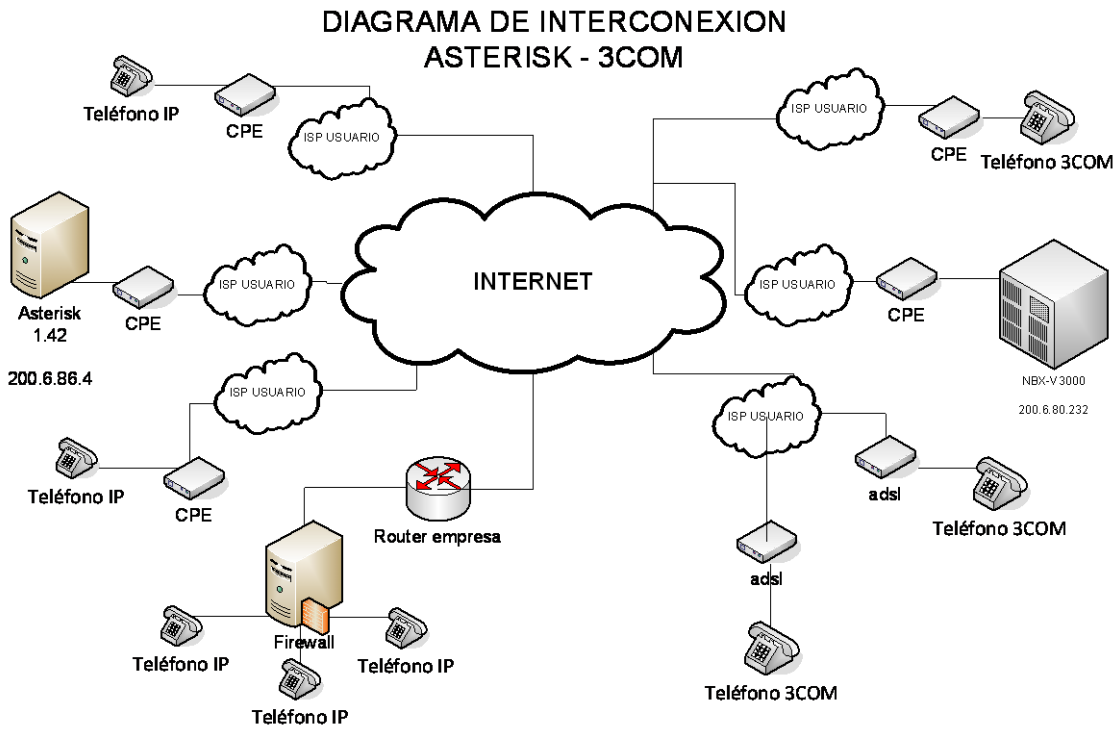


Figura 2.4: Topologia

2.4.1- Protocolos SIP, H323

Session Initiation Protocol es un protocolo de control y señalización usado en su mayoria para sistemas de Telefonía IP, que fue desarrollado por el IETF (RFC 3261). Dicho

protocolo permite crear, modificar y finalizar sesiones con uno o más participantes y sus principales ventajas son su simplicidad y consistencia.

Existen otros protocolos de señalización tales como el H.323 de la ITU, el SCCP de Cisco, o el MGCP, pero poco a poco SIP está ganando mucho espacio en el area. Cisco está progresivamente adoptando SIP como protocolo en sus sistemas de telefonía IP. Microsoft ha eligido SIP como protocolo para su nuevo OCS (Office Communication Server), y los operadores (de móvil y fijo) también están implantando SIP aprovechando la escalabilidad y interoperabilidad que nos proporciona el protocolo SIP.

Funciones SIP

Se enumeran las más importantes a continuación:

- Localización de usuarios (SIP proporciona soporte para la movilidad).
- Capacidades de usuario (SIP permite la negociación de parámetros).
- Disponibilidad del usuario
- Establecimiento y mantenimiento de una sesión.

El protocolo SIP permite la interacción entre dispositivos, cosa que se consigue con distintos tipos de mensajes propios del protocolo. Dichos mensajes proporcionan capacidades para registrar y/o invitar un usuario a una sesión, negociar los parámetros de una sesión, establecer una comunicación entre dos a más dispositivos y finalizar sesiones.

Beneficios del protocolo SIP frente otros protocolos

En la actualidad, los protocolos más usados en VoIP son: SIP, H.323 y IAX2.

H.323 es un estándar de la ITU que provee especificaciones para ordenadores, sistemas y servicios multimedia por redes que no proveen QoS (calidad de servicio). Como principales características de H.323 tenemos:

- Implementa QoS de forma interna.
- Control de conferencias

IAX2 (Inter Asterisk eXchange) es un protocolo creado y estandarizado por Asterisk. Unas de sus principales características son: Media y señalización viajan en el mismo flujo de datos.

- Trunking
- Cifrado de datos

SIP un protocolo cada día más sólido. Ventajas mas importantes son:

- El control de llamadas es stateless o sin estado, y proporciona escalabilidad entre los dispositivos telefónicos y los servidores.
- SIP necesita menos ciclos de CPU para generar mensajes de señalización de forma que un servidor podrá manejar más transacciones.
- Una llamada SIP es independiente de la existencia de una conexión en la capa de transporte.
- SIP soporta autentificación de llamante y llamado mediante mecanismos HTTP.
- Autenticación, criptográfica y encriptación son soportados salto a salto por SSL/TSL pero SIP puede usar cualquier capa de transporte o cualquier mecanismo de seguridad de HTTP, como SSH o S-HTTP.

- Un proxy SIP puede controlar la señalización de la llamada y puede bifurcar a cualquier número de dispositivos simultáneamente.

2.4.2- Codecs de audio en telefonía IP

La comunicación de voz es analógica, mientras que la red de datos es digital. El proceso de convertir ondas analógicas a información digital se hace con un codificador-decodificador (el CODEC). Hay muchas maneras de transformar una señal de voz analógica, todas ellas gobernadas por varios estándares. El proceso de la conversión es complejo. Es suficiente decir que la mayoría de las conversiones se basan en la modulación codificada mediante pulsos (PCM) o variaciones.

Además de la ejecución de la conversión de analógico a digital, el CODEC comprime la secuencia de datos, y proporciona la cancelación del eco. La compresión de la forma de onda representada puede permitir el ahorro del ancho de banda. Esto es especialmente interesante en los enlaces de poca capacidad y permite tener un mayor número de conexiones de VoIP simultáneamente. Otra manera de ahorrar ancho de banda es el uso de la supresión del silencio, que es el proceso de no enviar los paquetes de la voz entre silencios en conversaciones humanas.

En el"Anexo - Tabla de Codecs de Audio" se muestra una tabla resumen con los codecs más utilizados actualmente y sus características.

2.5- Interconexión

2.5.1- Configuración de archivos.

etc/asterisk/sip.config

En este archivo definimos las características SIP y otras para cada una de las extensiones, definiendo las características necesarias para la misma. A continuación se presenta un ejemplo de la definición de características de una extensión:

[4417]

type=friend

context=espevoip

secret=123456

host=dynamic

nat=yes

dtmfmode=rfc2833

username=4417

call-limit=3

disallow=all

allow=g723.1

allow=g729

allow=gsm

allow=alaw

allow=ulaw

allow=h263

allow=h263p

progressinband=yes

mailbox=4517

callerid="Pablin con video2"<4417>

callgroup=1

pickupgroup=1

etc/asterisk/extensions.conf

En este archivo definimos las extensiones y el número de Voice Mail. A continuación se presenta un ejemplo de la definición de una extensión:

exten => 4417,1,Dial(SIP/4417,15,Ttr)

exten => 4417,2,VoiceMail,4517

exten => 4417,3,Hangup

Igualmente en este archivo debemos definir el contexto de la central 3com, también se pueden definir otros contextos. Siempre debemos incluir al contexto principal (espevoip)

dentro de los otros a definirse (espeexternos) para que puedan comunicarse. Un ejemplo como sigue a continuación:

```
[espeexternos]

exten => _10XX,1,Dial(SIP/${EXTEN}@200.6.80.232,30,Ttr)

include => espevoip
```

También debemos definir los casilleros de voz a utilizarse. Un ejemplo del cómo definir los casilleros de voz:

```
[casillerosvoz]

exten => 4517,1,Ringing

exten => 4517,2,Wait(2)

exten => 4517,3,Authenticate(2222)

exten => 4517,4,VoicemailMain,s4517

include => espevoip
```

2.6- Factibilidad

2.6.1- Ventajas

Entre las principales ventajas que encontramos en esta implementación tenemos:

- El sistema anterior no brindaba una barata escalabilidad del mismo, la implementación que se presenta en esta tesis soluciona la posibilidad de poder crecer en número de usuarios sin una inversión demasiado grande.

- La posibilidad de usar teléfonos SIP no propietarios abaratara los costos en compra de nuevos teléfonos SIP,. Además nos brinda la posibilidad de no estar atados a lo que 3Com nos ofrece, pidiendo tomar lo mejor y más conveniente económicamente del mercado según los requerimientos y presupuesto que se mantenga.

- El tener un protocolo abierto como lo es SIP abre las puertas a que alumnos de la institución puedan empezar a desarrollar, investigar y experimentar sobre este de una manera gratuita y con una tecnología tan usada en el medio de VoIP.

- La oportunidad de unir varios servidores Asterisk es una gran ventaja para la escalabilidad de la implementación ya que esto nos permite implementar la misma solución en diferentes localidades de la institución, obteniendo las mismas ventajas y funcionalidades ya nombradas, logrando un sistema integral y no dependiente de la localidad.

2.6.2- Desventajas

- La implementación del protocolo SIP en la central telefónica demando la compra de un Release que permita la configuración de SIP dentro de la central NBX, además del upgrade de una tarjeta de memoria RAM.

- La implementación de NAT para la red interna para reemplazar el uso de direcciones IPs públicas por privadas para usuarios de la red interna también requiere de actualización de hardware en la central telefónica.

2.6.3- Situación actual del servidor NBX 3000

La central telefónica 3COM cuenta con la versión 6.059 con protocolo SIP abierto esto nos permite establecer una comunicación con otros equipos en el mismo protocolo como es nuestro caso con un servidor Asterisk en su distribución 1.41.

Capítulo III

3.1- Requerimientos

Se usaran los paquetes básicos en software y hardware a su vez también utilizaremos la central 3COM NBX V3000 en su Release 6_59 y sus paquetes de actualización.

3.1.1- Paquetes de software básicos

Fedora 9: Se utilizara a versión 9 de Fedora tomando en cuenta que se utiliza como base para la instalación de otros paquetes cabe destacar que por renovación constante utilizaremos esta distribución en lugar de Fedora 6 como estuvo pensada en un inicio.

Asterisk 1.4: como se explico en el capítulo 2 es un paquete de distribución libre que tiene las capacidades de un PBX

Zaptel: este modulo sirve de interconexión con tarjetas analógicas que en su mayoría son distribuidas por la compañía Digium para utilización de distintos protocolos como ss7, h323 en nuestro caso utilizaremos un emulador denominado ZDummy que se utiliza si no se cuenta con tarjetas adicionales para simular el sincronismo y a su vez el paso de tonos.

Libpri: Nos provee de las librerías necesarias para la configuración de varias características de Asterisk en específico soporte para conexiones digitales.

Mrtg: Nos permite monitorear el consumo del CPU, el ancho de banda del servidor y así controlar el flujo de las llamadas realizadas

Send Mail: Este servidor de correo no ayudara a notificar a los usuarios ciertos detalles de su extensión.

3.1.2- Hardware básico

Se utilizara un computador clon con las siguientes características.

- Procesador Intel core 2 quad
- Disco duro 250 Gb
- 2 Mb Memoria RAM
- 1 tarjeta de red

3.2- Instalación

Existen varias distribuciones del Asterisk, pero para nuestro desarrollo utilizaremos Asterisk 1.46 que a nuestro modo de ver es una versión estable.

3.2.1- Sistema Operativo

Tomando en cuenta que uno de los sistemas de Linux más estables es Fedora utilizaremos su versión 9 para nuestra implementación.

3.2.1.1- Instalación de Linux Fedora Core 9

1) Iniciamos el computador considerando que si el computador tiene datos, en este tipo de instalación se perderán, seleccionamos la primera opción mostrada en el grafico 1 y pulsamos enter.

Figura 3.1: Pantallas de Instalacion

2) Seguimos con la instalación pulsando el botón Next (Siguiente) mostrado en la Figura 3.2

Figura 3.2: Pantallas de Instalacion

3) Seleccionamos el idioma español que será el idioma utilizado durante el proceso de instalación.

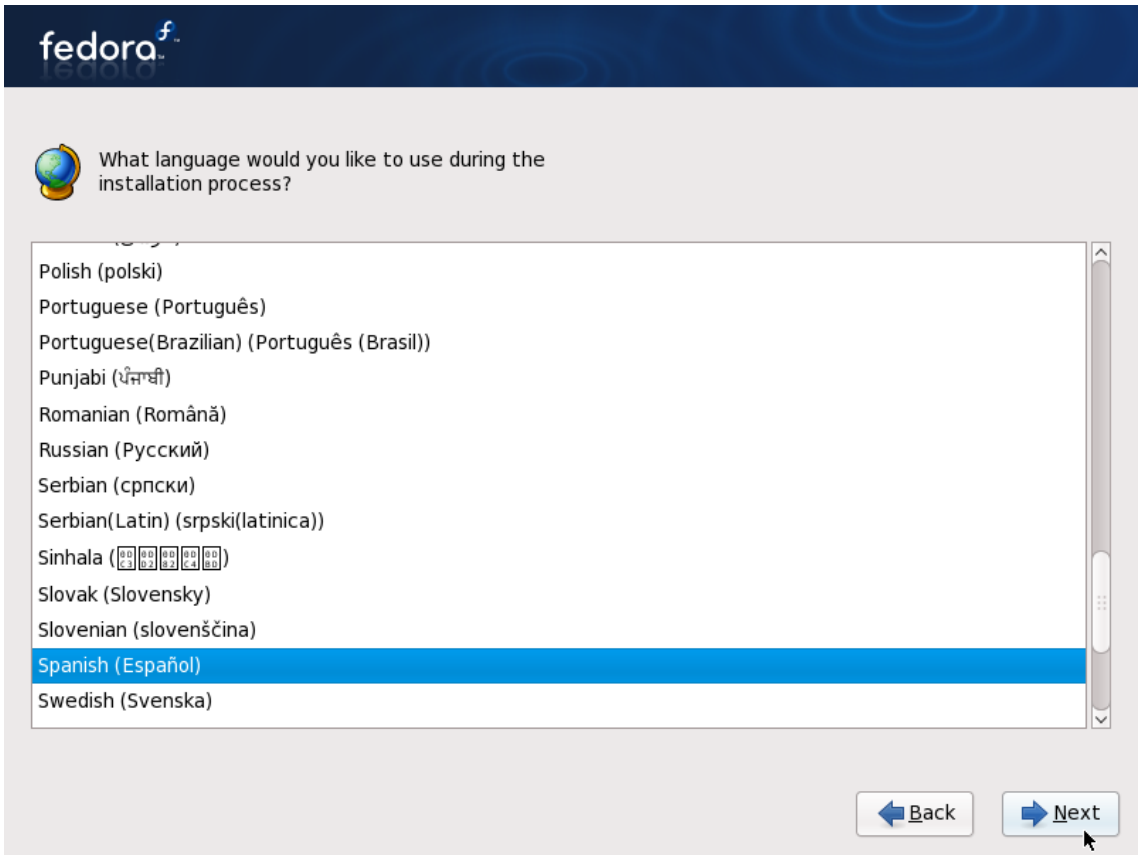


Figura 3.3: Pantallas de Instalacion

4) Seleccionamos idioma español para el uso del teclado.

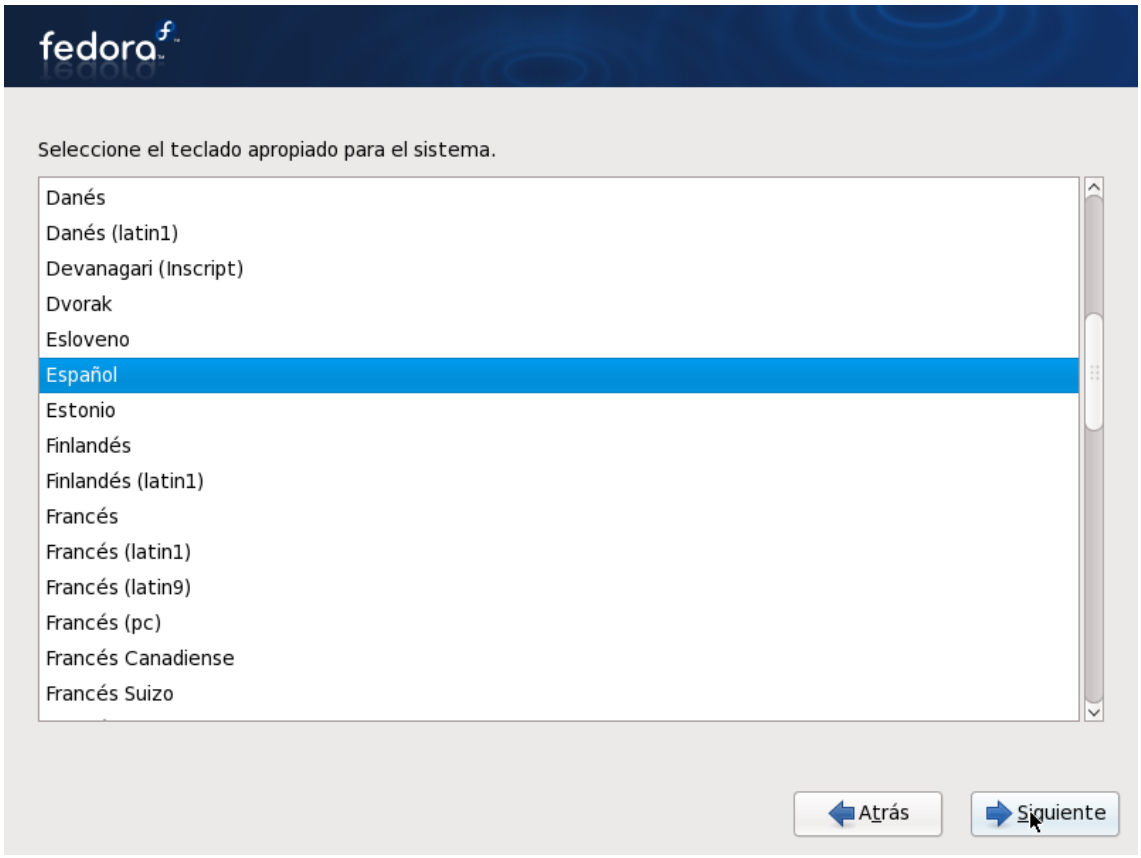


Figura 3.4: Pantallas de Instalacion

5) Configuramos la conexión de red, debido a que nuestro proveedor tiene disponibilidad de asignarnos direcciones públicas y con esto evitamos realizar NAT en la red de borde.

En ningún caso podemos trabajar con un cliente DHCP porque esto nos causaría muchas molestias y tendríamos que reconfigurar los parámetros del servidor Asterisk, la interconexión en la central 3com y la configuración de los teléfonos.

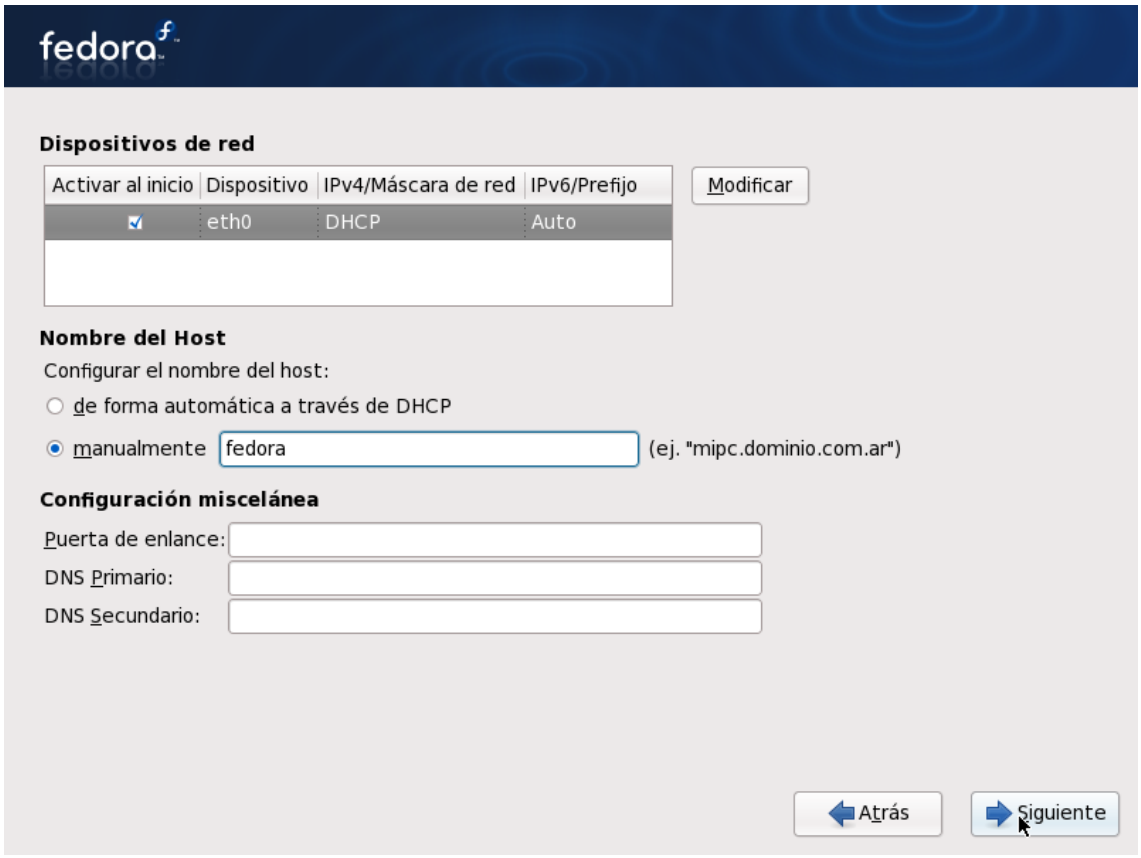


Figura 3.5: Pantallas de Instalacion

6) Seleccionamos el uso horario e Introducimos la contraseña para el usuario root.

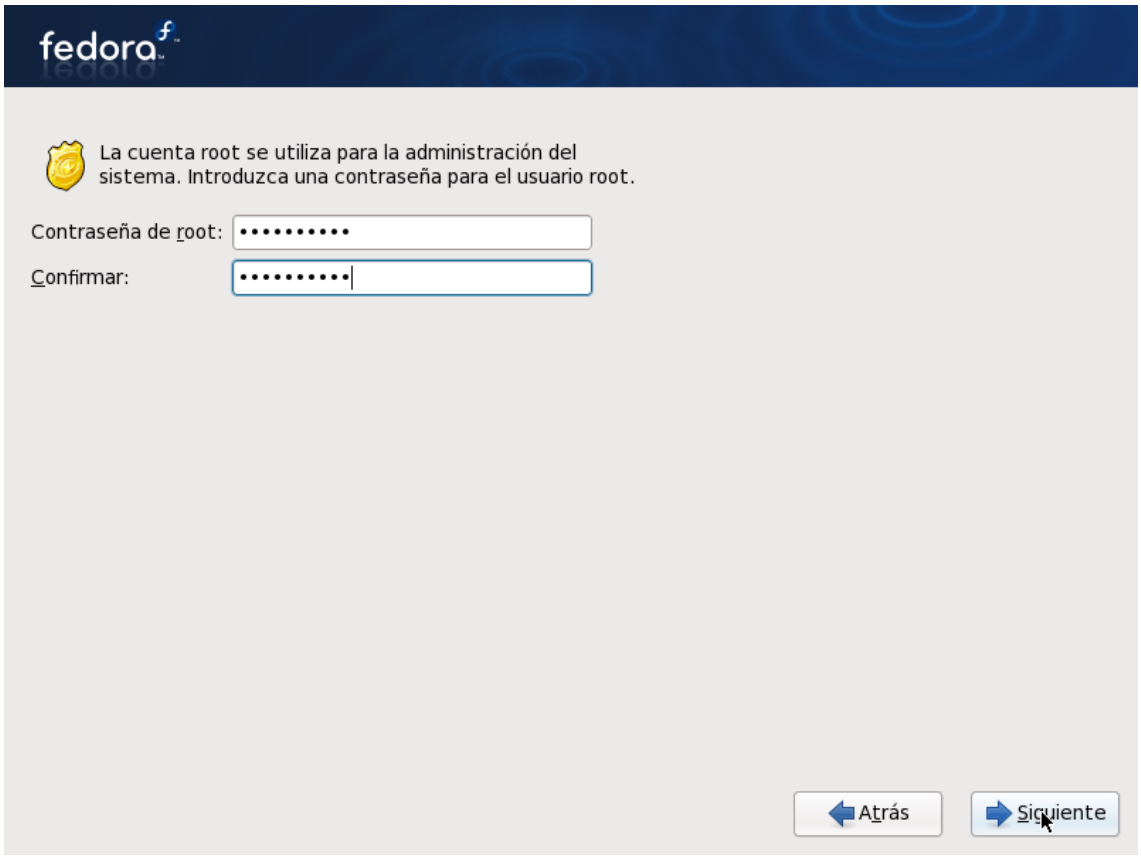


Figura 3.6: Pantallas de Instalacion

8) Distribuimos el espacio en disco duro con la siguiente configuración:

Nombre	**Asignación (Mb)**	**CARACTERISTICAS**
SWAP	4000	Hace las funciones memoria RAM pero de más rápido acceso.
/	76059264	El directorio raíz es en donde se van a almacenar
/VAR	80632188	En el directorio VAR se almacenan los mensajes de errores, las librerías, los archivos

		de páginas web a ser publicados.
/HOME	80632188	Se almacena las configuraciones y archivos de los usuarios de Linux.
/BOOT	194442	Guarda los archivos de inicio y un menú que nos permite iniciar distintos sistemas operativos instalados en el computador.

También con en este paso borramos cualquier posible partición existente en el sistema debido a que estamos instalando la última versión estable de Linux así evitaremos cualquier conflicto si tuviéramos instalado alguna versión de Linux.

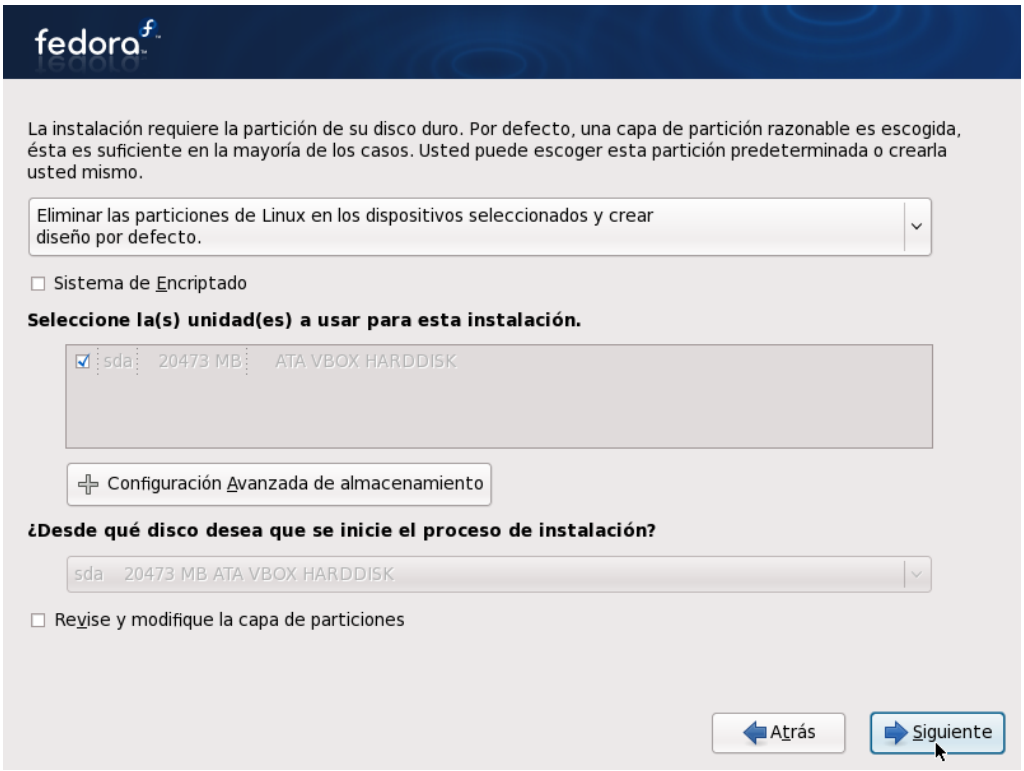


Figura 3.7: Pantallas de Instalacion

9) Verificamos que los datos asignados sean los correctos.

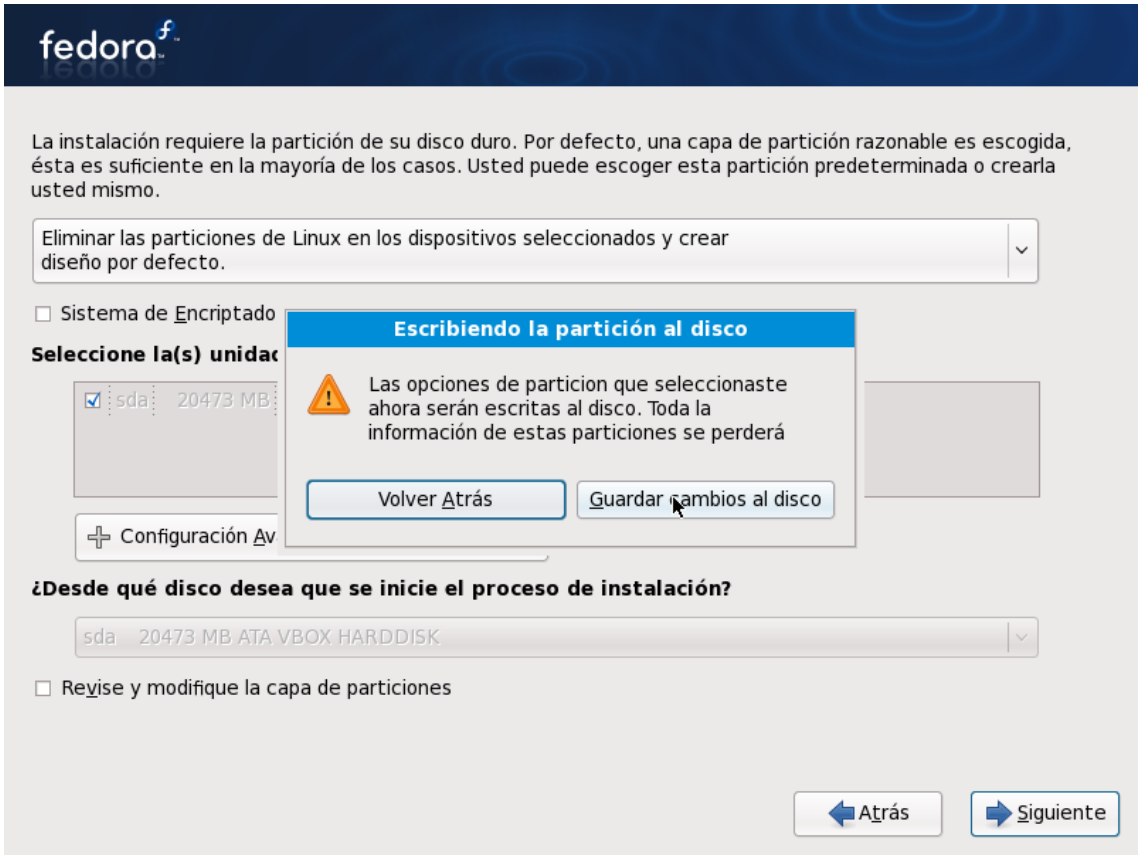


Figura 3.8: Pantallas de Instalacion

10) Procedemos con el formateo del disco duro y sus nuevas particiones.

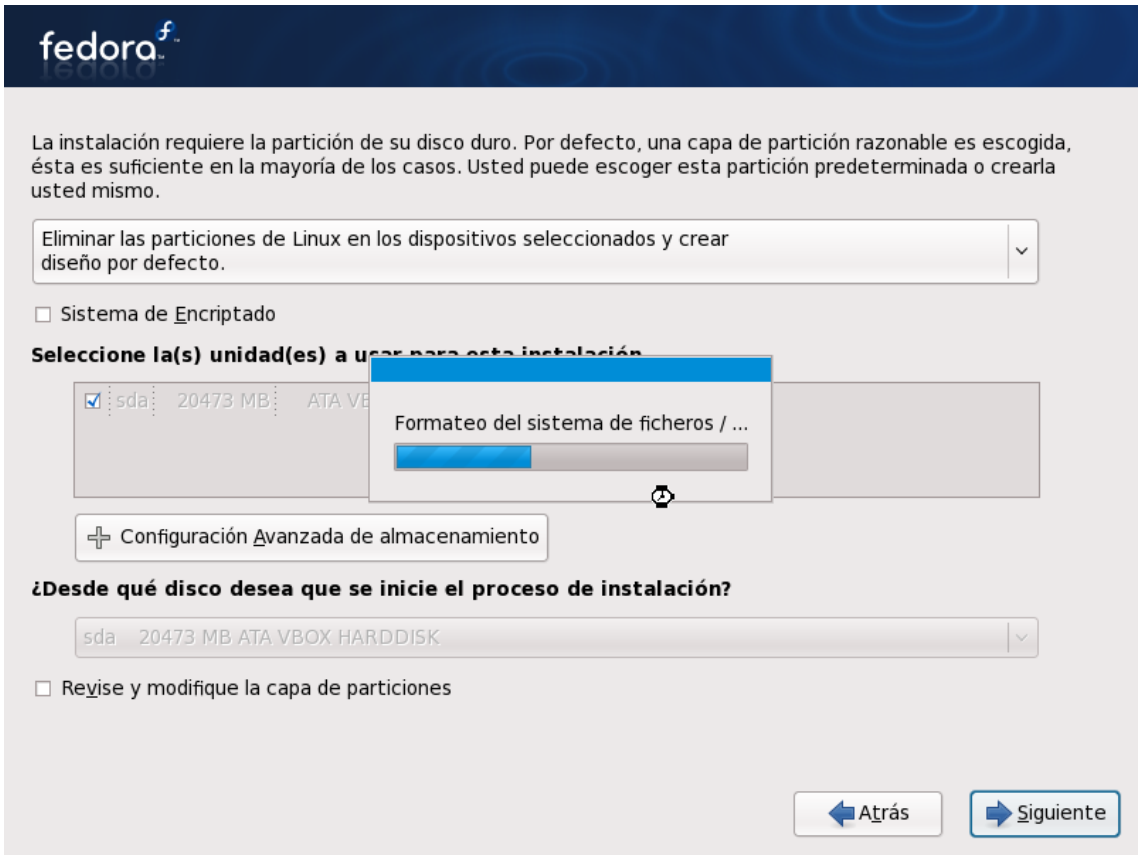


Figura 3.9: Pantallas de Instalacion

11) Las particiones son creadas y formateadas.

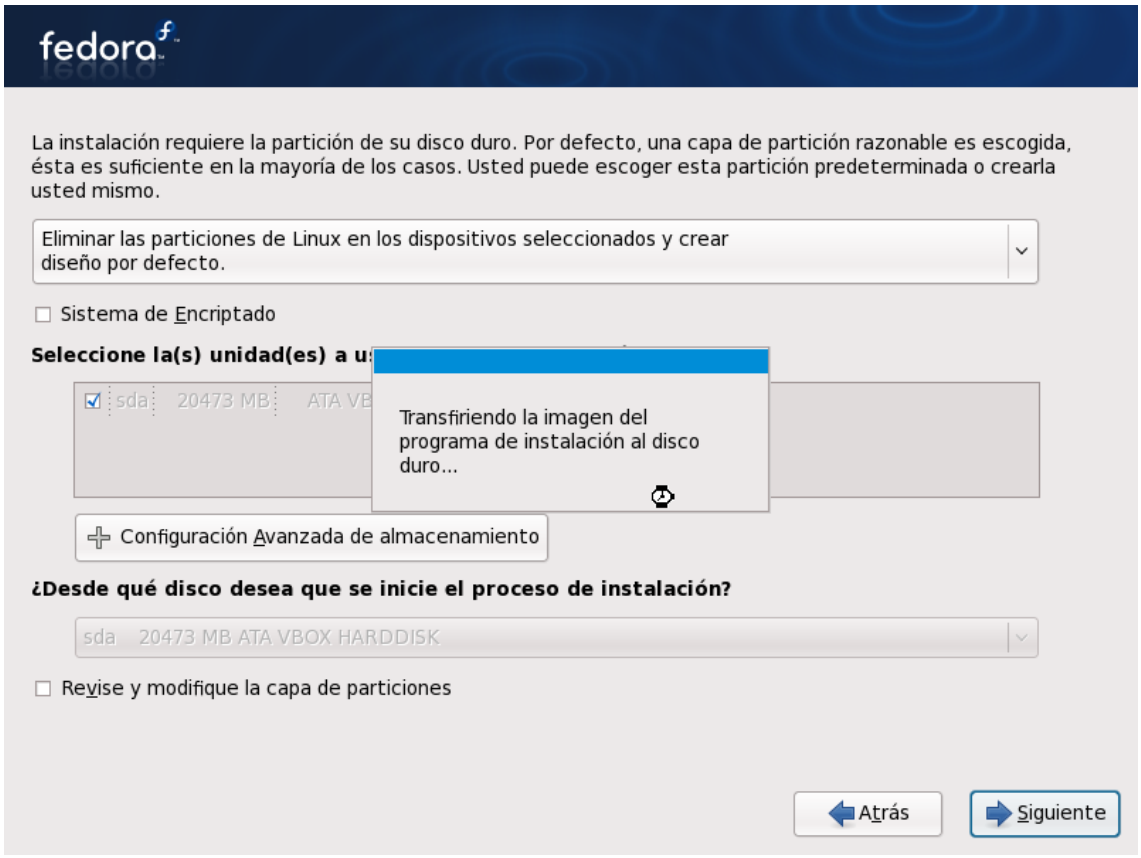


Figura 3.10: Pantallas de Instalacion

12) Se copian los archivos de instalación en el disco duro.

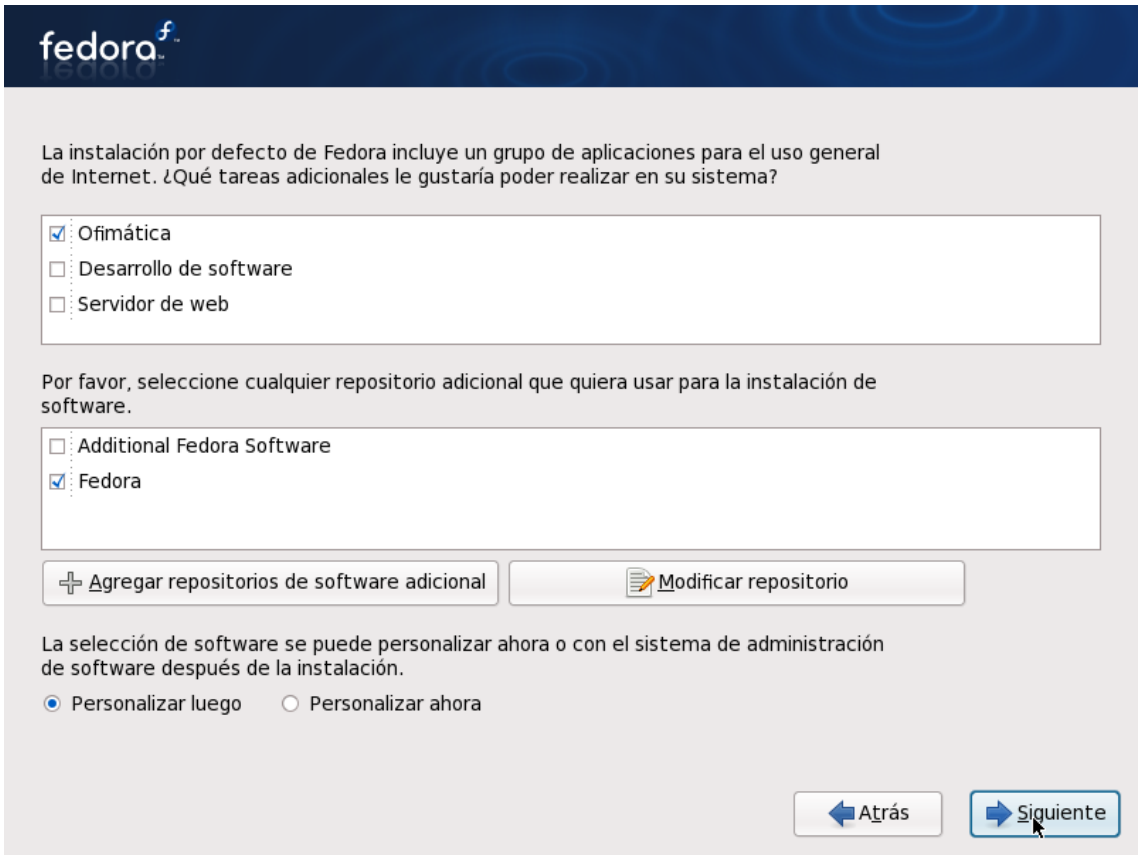


Figura 3.11: Pantallas de Instalacion

13) Se seleccionan los paquetes a ser instalados, en este caso seleccionamos paquetes básicos que nos permitan descargar programas del internet (wget), acceso remoto (ssh), transferencia de archivos (ftp), monitoreo de tráfico (iptraf) publicación de páginas web (httpd) e instalamos también la distribución open office que cuenta con editor de palabras, hojas de cálculo, editor de diapositivas entre los más importantes.

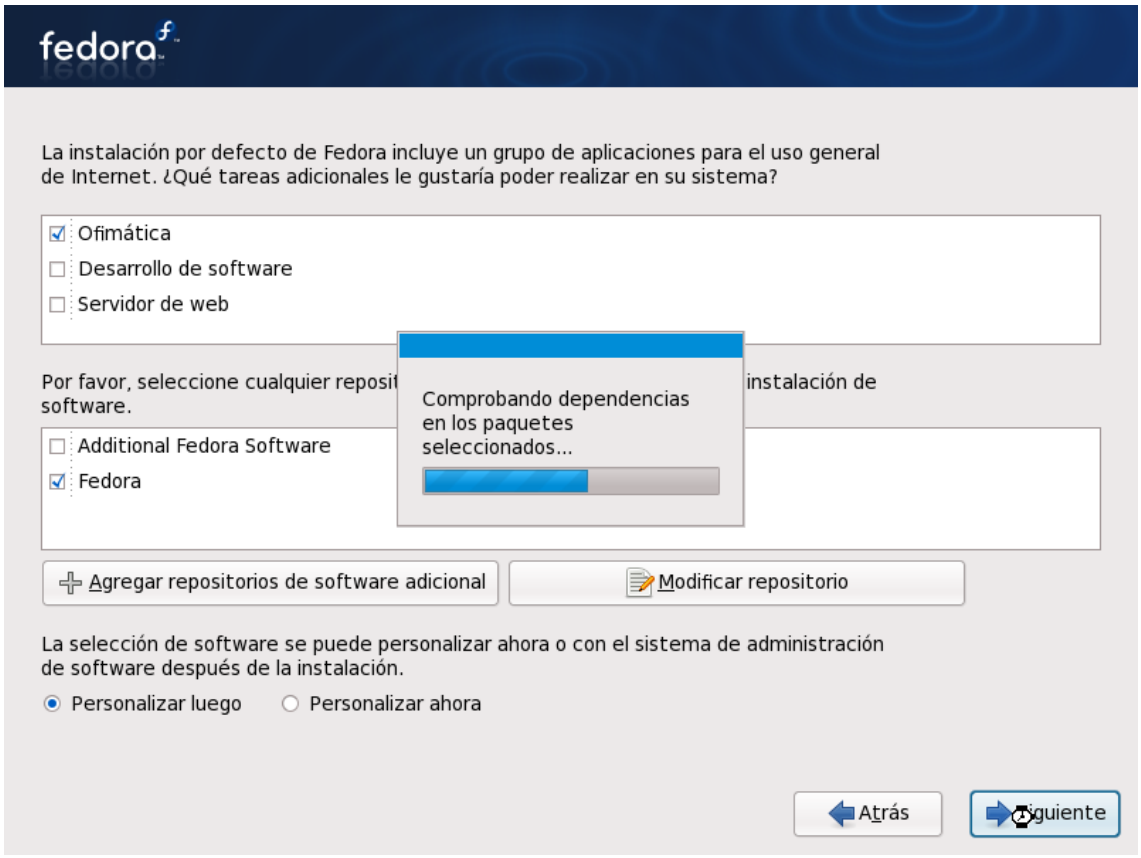


Figura 3.12: Pantallas de Instalacion

14) Se instalan las dependencias de los paquetes seleccionados.

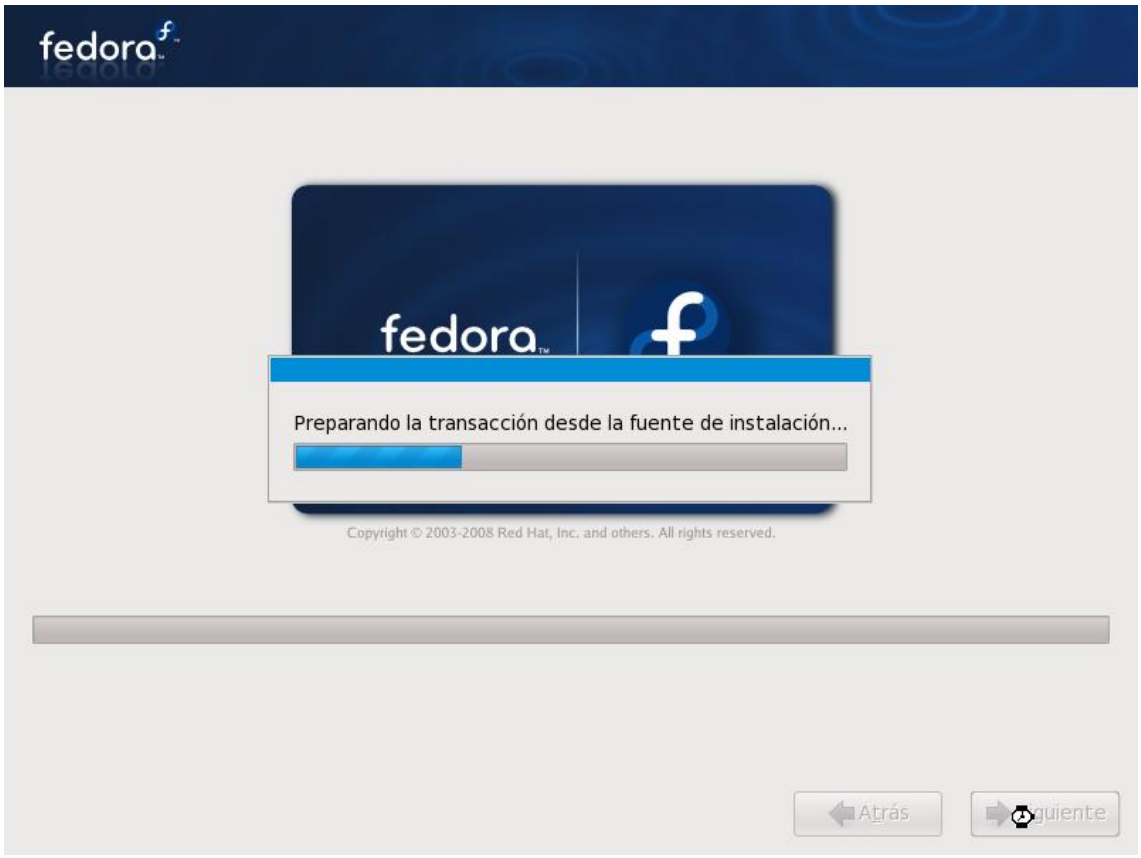


Figura 3.13: Pantallas de Instalacion

15) Se preparan las fuentes de instalación.

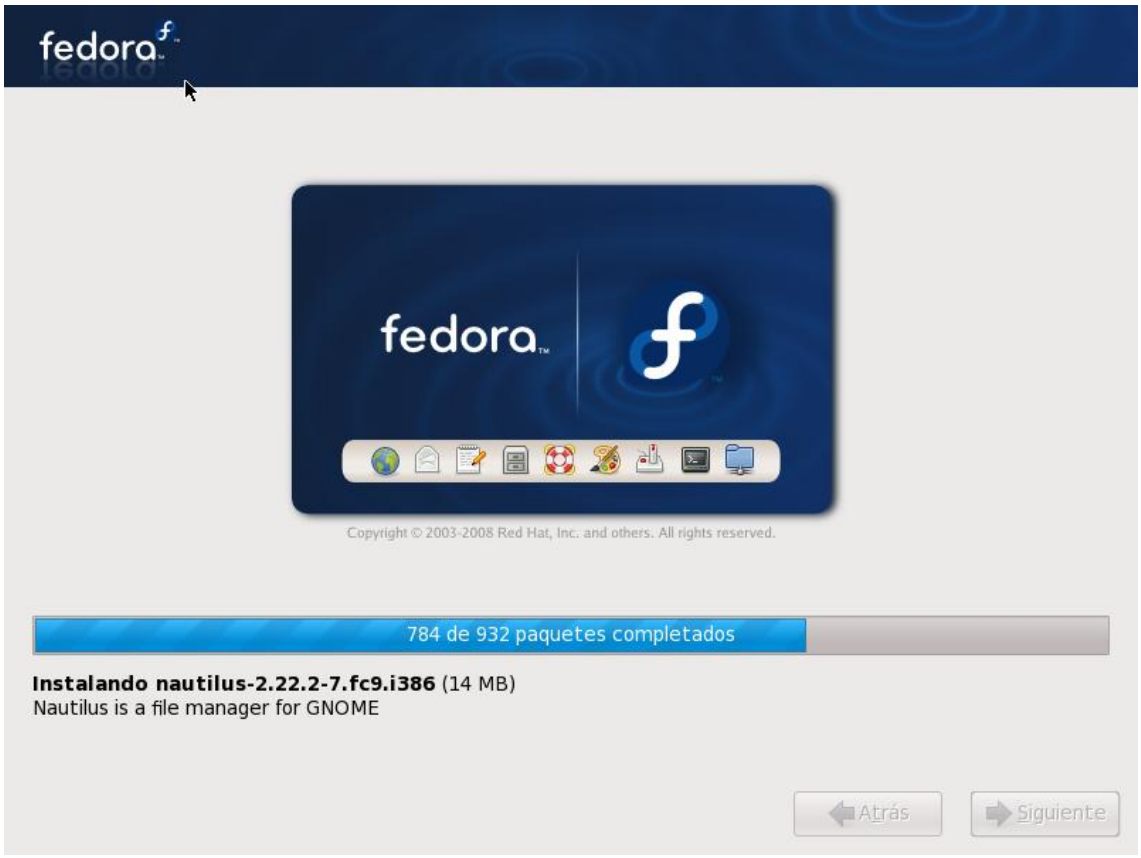


Figura 3.14: Pantallas de Instalacion

16) Se instalan los paquetes

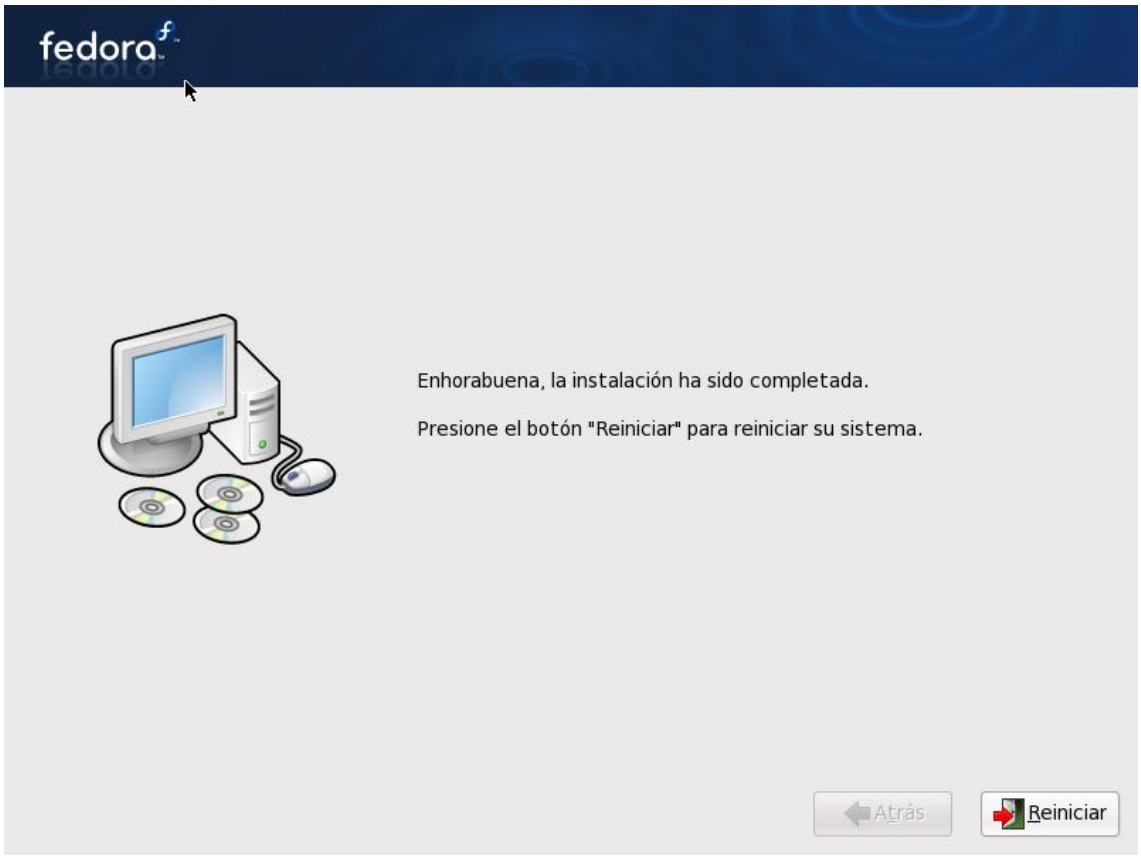


Figura 3.15: Pantallas de Instalacion

17) Se reinicia el sistema tomando en cuenta que el dvd de instalación sea retirado

Figura 3.16: Pantallas de Instalacion

18) Se configura un usuario y se inicia e sistema.

3.2.1.2- Configuración de hardware.

La central 3com NBX V3000 tiene como su principal propósito la comunicación mediante el protocolo H323 en su versión original, con su Release 6_1 se actualiza al protocolo SIP abierto, protocolo que usa Asterisk, con esta actualización la central telefónica pierde el servicio de mensajería y la capacidad de realizar conferencias.

Lo primero que realizaremos es ingresar a la página de 3COM para buscar las actualizaciones.

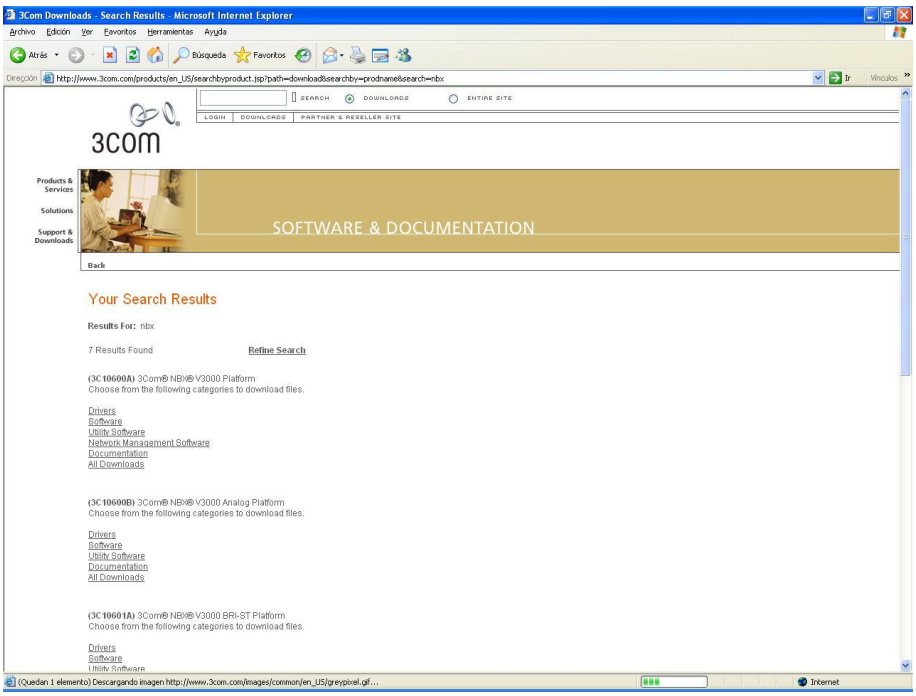


Figura 3.17: Configuracion de Central Telefonica

Elegimos la opción de download & drivers y luego elegimos la central NBX V3000.

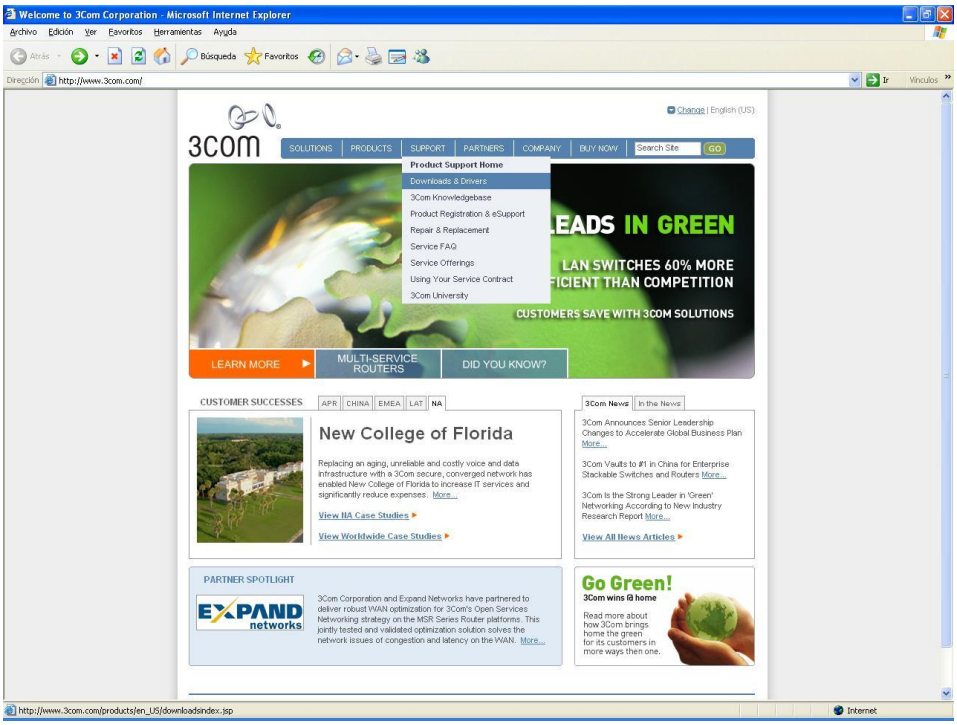


Figura 3.18: Configuracion de Central Telefonica

Buscamos los paquetes a descargar

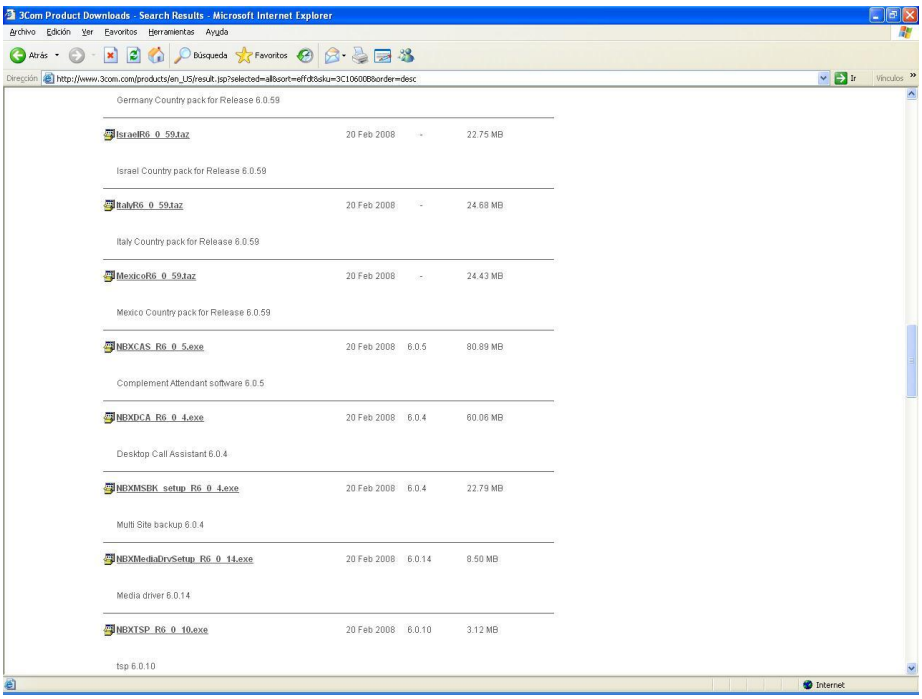


Figura 3.19: Configuracion de Central Telefonica

Seleccionamos el paquete regional de México en español debido a que es el paquete en español más cercano al ecuatoriano.

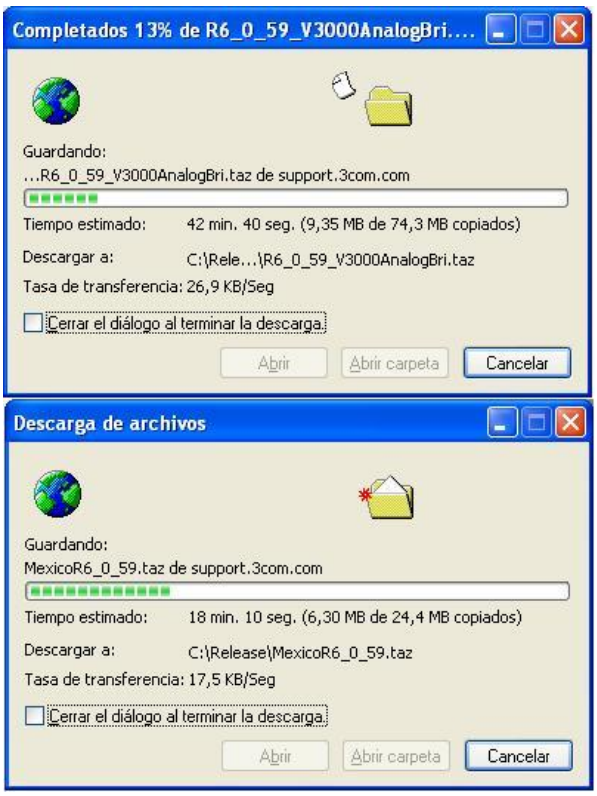


Figura 3.20: Configuracion de Central Telefonica

Una vez descargados los paquetes accedemos a la central telefónica vía http, verificamos la versión actual.

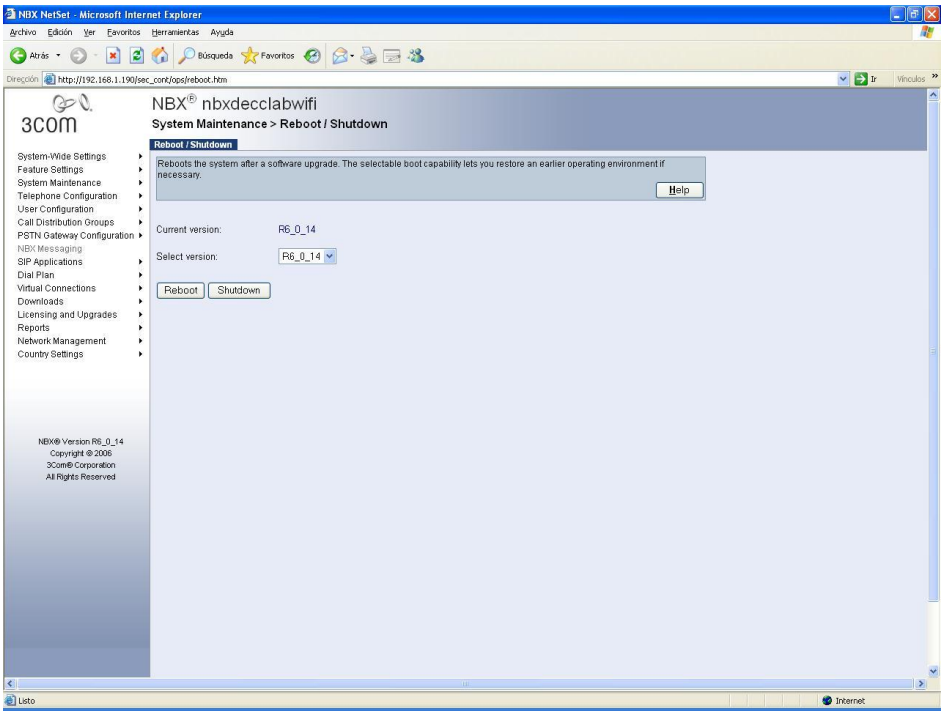


Figura 3.21: Configuracion de Central Telefonica

Y nos dirigimos a LICENSING AND UPGRADES.

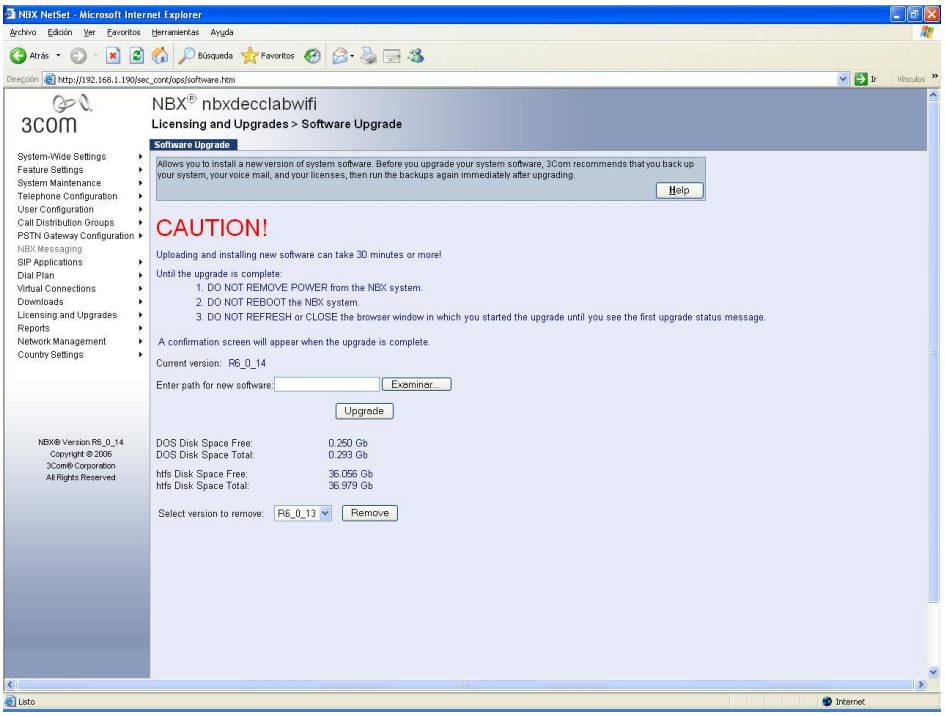


Figura 3.22: Configuracion de Central Telefonica

Y cargamos el paquete para la actualización.

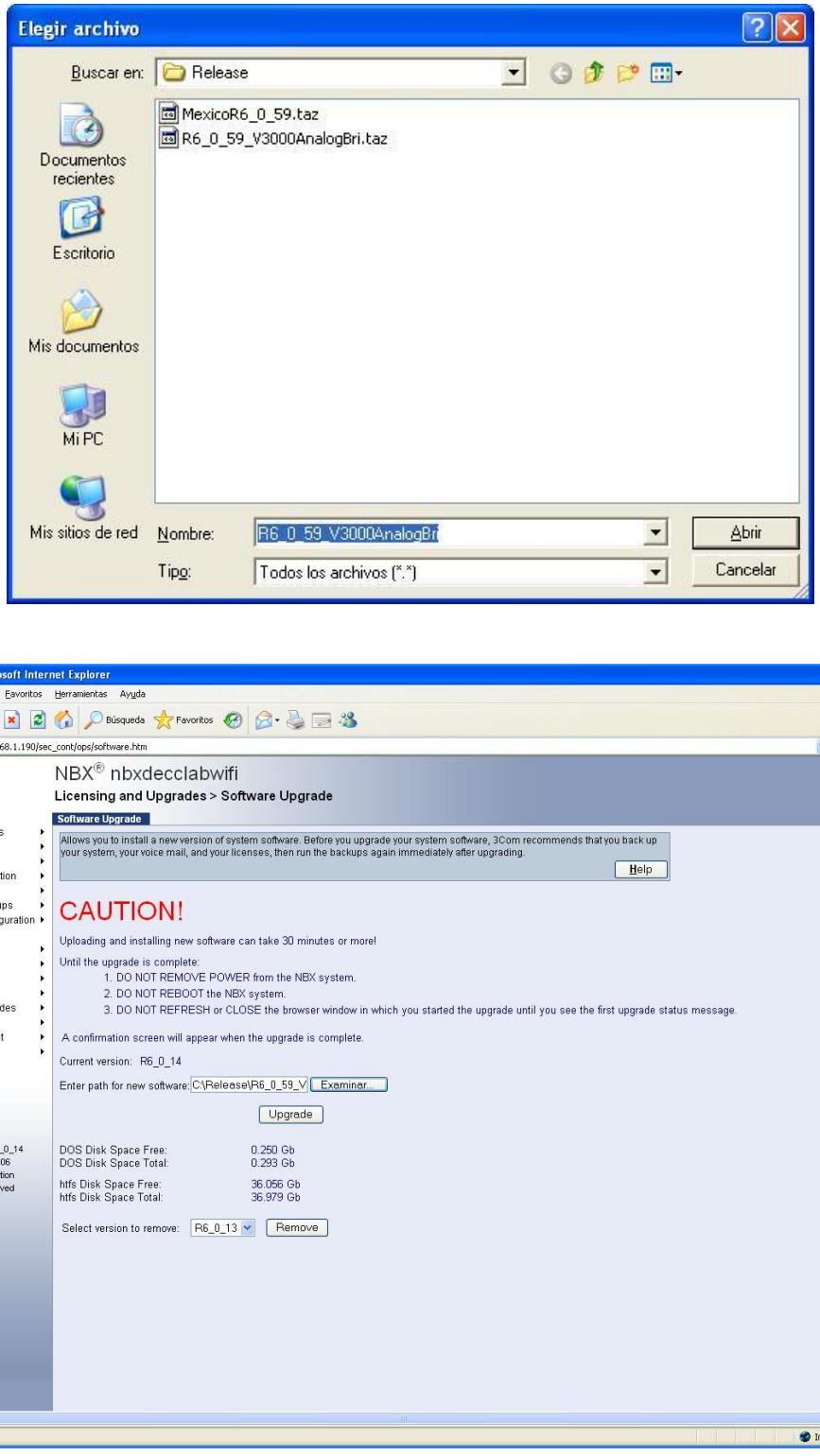


Figura 3.23: Configuracion de Central Telefonica

Presionamos UPGRADE

Figura 3.24: Configuracion de Central Telefonica

Y procedemos a la actualización, tomando en cuenta que debemos tener una conexión estable de internet por aproximadamente 2 horas, de igual manera procedemos con la actualización del paquete de región.

Luego de colocar la dirección IP externa tendremos acceso a la central NBX desde cualquier parte del mundo a través de internet, lo que nos facilitara la interconexión con otros dispositivos en nuestro caso un servidor Asterisk

3.2.2- Instalación de Asterisk

Existen distribuciones de Linux que viene integrado Asterisk como es el caso de TRIXBOX pero para nuestro desarrollo lo instalaremos sobre un sistema operativo ya instalado para explicar su proceso.

3.2.2.1- Instalación de paquetes básicos

Como primer paso instalamos un firewall respetando los derechos de autor.

[root@voip /]# vi etc/rc.d/init.d/espevoip

Lo editamos y asignamos permisos

[root@voip /]# chmod 711 etc/rc.d/init.d/espevoip

[root@voip /]# ls -l etc/rc.d/init.d/espevoip

-rwx--x--x 1 root root 32959 2008-09-08 15:13 etc/rc.d/init.d/espevoip

Ejecutamos el firewall

[root@voip /]# /etc/rc.d/init.d/espevoip

Inicializando Proxy Firewall

Definición de variables

Instalación de Módulos

Definición de Políticas Básicas

Definición de Políticas Generales

paquetes icmp

ssh

Auth

DNS

Acceso snmp

Acceso web local

Web remoto

correo saliente

proceso finalizado

Creamos la carpeta en la que descargaremos todos los paquetes requeridos.

mkdir downloads

Descargamos la distribución 1.4 de Asterisk de la página oficial.

wget http://www.digium.com/elqNow/elqRedir.htm?ref=http://downloads.digium.com/pub/asterisk/releases/asterisk-1.4.0-rc4.tar.g

Se requieren los siguientes paquetes para la operación de Asterisk:

- Ncurses.

- Openssl.
- Zlib.
- Bison.

Verificamos los paquetes requeridos de la siguiente manera.

rpm -qa |grep ncurses

ncurses-5.6-16.20080301.fc9.x86_64

ncurses-libs-5.6-16.20080301.fc9.i386

ncurses-devel-5.6-16.20080301.fc9.x86_64

ncurses-base-5.6-16.20080301.fc9.x86_64

ncurses-libs-5.6-16.20080301.fc9.x86_64

rpm -qa |grep openssl

openssl-devel-0.9.8g-6.fc9.x86_64

openssl-0.9.8g-6.fc9.x86_64

rpm -qa |grep zlib

zlib-1.2.3-18.fc9.x86_64

zlib-devel-1.2.3-18.fc9.x86_64

jzlib-1.0.7-5jpp.1.x86_64

rpm -qa |grep bison

bison-2.3-5.fc9.x86_64

Para iniciar la instalación debemos ejecutar lo siguiente

[root@voip]#cp downloads/asterisk-1.4.0-rc4.tar.gz [root@voip]#usr/local/src/

[root@voip]#tar zxvf asterisk-1.4.0-rc4.tar.gz

[root@voip]#cd asterisk-1.4.0-rc4

[root@voip]#./configure

[root@voip]#make

[root@voip]#make install

[root@voip]#make samples

[root@voip]#make config

Tomando en cuenta que con los paquetes instalados anteriormente no nos despliegue ningún error.

3.2.2.2- Instalación de Zaptel.

Descomprimimos el archivo de instalación de ZAPTEL tomando en cuenta que tenga la misma versión de Asterisk.

[root@voip src]# tar zxvf zaptel-1.4.12.1.tar.gz

[root@voip src]# cd zaptel-1.4.12.1

[root@voip zaptel-1.4.12.1]#

[root@voip zaptel-1.4.12.1]# make clean

[root@voip zaptel-1.4.12.1]# make

[root@voip zaptel-1.4.12.1]# make install

Instalamos libpri y addons que nos ayuda a interconexiones virtuales y características especiales como pruebas de eco, reloj, etc.

[root@voip src]# tar zxvf libpri-1.4.7.tar.gz

[root@voip src]# cd libpri-1.4.7

[root@voip libpri-1.4.7]# make clean

[root@voip libpri-1.4.7]# make

[root@voip libpri-1.4.7]# make install

[root@voip libpri-1.4.7]# make

[root@voip src]# tar zxvf asterisk-addons-1.6.0-rc1.tar.gz

[root@voip asterisk-addons-1.6.0-rc1]# make clean

[root@voip asterisk-addons-1.6.0-rc1]# ./configure

[root@mail asterisk-addons-1.6.0-rc1]# make

[root@voip asterisk-addons-1.6.0-rc1]# make install

[root@voip asterisk-addons-1.6.0-rc1]# make samples

3.2.2.3- Instalación de codecs de audio

Para su instalación necesitamos descargar los codecs en la carpeta.

usr/lib/asterisk/modules/

Los codecs tiene la extensión de archivos .so

Los descargamos utilizando la función wget, tomando en cuenta la versión de nuestro sistema operativo, la arquitectura y la versión de asterisk.

wget http://asterisk.hosting.lv/bin/codec_g723-ast14-icc-glibc-x86_64-core2.so

wget http://asterisk.hosting.lv/bin/codec_g729-ast14-icc-glibc-x86_64-core2.so

Los renombramos

cp codec_g723-ast14-icc-glibc-x86_64-core2.so codec_g723.so

cp codec_g729-ast14-icc-glibc-x86_64-core2.so codec_g729.so

Y concedemos todos los permisos.

chmod 777 codec_g723.so

chmod 777 codec_g729.so

3.2.3- Configuración

Para poder operar el sistema es necesario contar con un contexto principal, que abarcara a todas las extensiones y servicios alojados en el servidor.

3.2.3.1- Configuración SIP

Las configuraciones son realizadas en el archivo SIP.CONF, en este archivo hay que destacar los siguientes puntos.

disallow=all

codecs

allow=g723.1

allow=g729

allow=ulaw

allow=alaw

Hay que tomar en cuenta que estos parámetros son generales, pero se pueden configurar para cada uno de los usuarios SIP.

Definimos el idioma

language=es

Definimos el modo en el que se transmite los tonos.

dtmfmode = rfc2833

Habilitamos la opción de video para realizar pruebas del mismo.

videosupport=yes

Con esta opción activamos los mensajes de asterisk así nos damos cuenta que paquetes envía y hacia a donde.

sipdebug = yes

Guardamos los mensajes con las sentencias.

recordhistory=yes

dumphistory=yes

Uso de NAT (Network address translation, traducción de direcciones de red) en modo señalización simétrica

nat=no

En SIP los **invites** se utilizan para establecer llamadas y redirigir el audio o video. Cualquier invite después del invite inicial en la misma conversación se considera un reinvite.

Cuando dos usuarios han establecido la comunicación con canreinvite= yes (por defecto) los paquetes RTP de audio podrían ser enviados de extremo a extremo sin pasar por el servidor Asterisk. Esto, normalmente, no suele ser conveniente en casos en los que haya NAT en alguno de los clientes. (NAT=yes).

Usando canreinvite=no se forza a Asterisk a estar en medio no permitiendo que los puntos finales intercambien mensajes RTP directamente.

canreinvite=yes

Definimos un contexto en este caso la central 3com que es motivo de nuestro estudio tomando en cuenta que nuestro contexto principal es espevoip (Servidor Asterisk) lo definiremos más adelante en el archivo extensions.conf.

[espeexternos]

type=peer

context=espevoip

host=200.6.80.232

canreinvite=no

allow=g723.1

allow=g729

allow=ulaw

allow=alaw

Establecemos las extensiones a usar, nuestro plan de marcación debe tener concordancia con la central 3com, esto quiere decir que no tenga conflictos con las extensiones o servicios de dicha central.

Para esto se define las extensiones 44XX como vemos a continuación.

[4410]

type=friend

context=espevoip

secret=123456

host=dynamic

nat=yes

dtmfmode=rfc2833

username=4410

call-limit=3

disallow=all

allow=g723.1

allow=g729

allow=ulaw

allow=alaw

allow=gsm

progressinband=yes

mailbox=4510

callerid="Pablo"<4410>

callgroup=1

pickupgroup=1

A continuación detallamos algunas opciones.

User	Peer	Explicación y opciones
Context	**Context**	Indica el contexto asociado en el dialplan para un usuario o peer
Permit	**Permit**	Permitir una IP
Deny	**Deny**	No permitir una IP
Secret	**Secret**	Contraseña para el registro
md5secret	**md5secret**	Contraseña encriptada con md5
Dtmfmode	**Dtmfmode**	El modo en el que se transmiten los tonos. Pueden

		ser "RFC2833" o "INFO"
Canreinvite	**Canreinvite**	Con "no" se fuerza a Asterisk a no permitir que los puntos finales intercambien mensajes RTP directamente.
Nat	**Nat**	Indica si el dispositivo está detrás de un NAT con "yes"
Callgroup	**Callgroup**	Define un grupo de llamadas
Pickupgroup	**Pickupgroup**	Define el grupo de llamadas validas para una aplicacion pickup()
Language	**Language**	Define las señales para un pais. Debe estar presente en el archivo indications.conf
Allow	**Allow**	permite habilitar un codec. Pueden ponerse varios en un mismo usuario Posibles Valores: "allow=all" ,"allow=alaw", "allow=ulaw", "allow=g723.1" ; allow="g729" , "allow=ilbc" , "allow=gsm".
Disallow	**Disallow**	permite deshabilitar un codec. Puede tomar los mismos valores que allow
Insecure	**Insecure**	Define como manejar las conexiones con peers Tiene los siguientes valores very|yes|no|invite|port Por defecto es "no" que quiere decir que hay que autenticarse siempre.
Trustpid	**Trustpid**	Si la cabecera **Remote-Party-ID** es de confianza. Por defecto "no"

progressinband	**Progressinband**	Si se deben generar señales en banda siempre. Por defecto **never**
Promiscredir	**Promiscredir**	Permite soportar redirecciones 302. Por defecto "no"
Callerid		Define el identificador cuando no hay ninguna otra informacion disponible
Accountcode		Los usuarios pueden estar asociados con un accountcode . Se usa para facturacion.
Amaflags		Se usa para guardar en los CDR y temas de facturación . Puede ser "default", "omit", "billing", o "documentation"
Incominglimit		Limite de llamadas simultaneas para un cliente
Restrictcid		Se usa para esconder el ID del llamante. Anticuada y en desuso
	Mailbox	Extensión del contestador
	Username	Si Asterisk actua como cliente SIP este es el nombre de usuario que presenta en el servidor SIP al que llama
	Fromdomain	Pone el campo From: de los mensajes SIP
	Regexten	
	Fromuser	Pone el nombre de usuario en el from por encima de lo que diga el callerID
	Host	dirección o host donde se encuentra el dispositivo remoto. Puede tomar valores: - Una IP o un host concreto

		- "dynamic" con lo que valdría cualquier IP pero necesita contraseña - "static" vale cualquier IP pero no es necesario contraseña
	Mask	
	Port	Puerto UDP en el que responderá el Asterisk
	Qualify	Para determinar cuando el dispositivo puede ser alcanzado
	Defaultip	IP por defecto del cliente **host=** cuando es especificado como "dynamic"
	Rtptimeout	Termina la llamada cuando llega a ese timeout si no ha habido tráfico rtp
	Rtpholdtimeout	Termina la llamada cuando llega a ese timeout si no ha habido tráfico rtp "on hold"

3.2.3.2- Configuración de extensiones

Para poder configurar nuestras extensiones nos dirigimos al archivo:

etc/asterisk/extensions.conf

Los campos en su mayoría se encuentran comentados, no debemos olvidar el contexto principal nombrado en el archivo sip.conf, incluidas sus extensiones, sin olvidarnos de incluir los contextos con los que se comunicara, y lo definimos de la siguiente manera.

[espevoip]

```
exten => 4410,1,Dial(SIP/4410,15,Ttr)

exten => 4410,2,VoiceMail,4510

exten => 4410,3,Hangup


exten => 4411,1,Dial(SIP/4411,15,Ttr)

exten => 4411,2,VoiceMail,4511

exten => 4411,3,Hangup

.

.

.

exten => 4499,1,Dial(SIP/4499,15,Ttr)

exten => 4499,2,VoiceMail,4599

exten => 4499,3,Hangup


include => casillerosvoz

include => conferencia

include => espeexternos
```

Definimos también su plan de marcación, es decir que dirección tomara la llamada dependiendo del marcado, también en cada uno de los contextos incluimos nuestro contexto principal

[espeexternos]

exten => _10XX,1,Dial(SIP/${EXTEN}@200.6.80.232,30,Ttr)

ver punto 2.5.1.xxx

include => espevoip

Definimos los casilleros de voz, que luego serán configurados en el archivo voicemail.conf.

[casillerosvoz]

exten => 4510,1,Ringing

exten => 4510,2,Wait(2)

exten => 4510,3,Authenticate(2222)

exten => 4510,4,VoicemailMain,s4510

exten => 4511,1,Ringing

exten => 4511,2,Wait(2)

exten => 4511,3,Authenticate(2222)

exten => 4511,4,VoicemailMain,s4511

.

.

.

exten => 4599,1,Ringing

exten => 4599,2,Wait(2)

exten => 4599,3,Authenticate(2222)

exten => 4599,4,VoicemailMain,s4512

include => espevoip

[conferencia]

exten => 4400,1,MeetMe(4001,ri)

exten => 4401,1,MeetMe(4002)

exten => 4402,1,MeetMe(4003,r)

exten => 4403,1,MeetMe(4004,ri)

exten => 4404,1,MeetMe(4004,mq)

include => espe3com

include => espevoip

3.2.3.3- Configuración de buzón de mensajes

Asterisk puede enviar correos con la información de su mensaje en forma de archivo, esta opción es muy útil y se la configura en el archivo.

etc/asterisk/ voicemail.conf

Los parámetros que debemos configurar son los siguientes.

fromstring=Asterisk-3COM ESPE

emailsubject=[ESPE]: Nuevo mensaje desde ${VM_MSGNUM} al numero ${VM_MAILBOX}

emailbody=Mensaje #

${VM_MSGNUM}.\n\nEstimado ${VM_NAME}:\n\n\tQueremos informarle que tiene un nuevo mensaje en su buzon de voz de ${VM_DUR}seg. de duracion\nen el numero telefonico ${VM_MAILBOX} desde el numero telefonico ${VM_CALLERID}, el ${VM_DATE}, \n\nAtt.\n\n\t\t\t\tESPE 3COM\n\n\n\n\nFavor no responder este mensaje.

3.3 Implementación del proyecto

3.3.1- Equipos

Se utilizara.

- Computador que hará las funciones de servidor Asterisk.
- Teléfonos IP Yuxin autentificados en el servidor asterisk.
- Central telefónica 3COM NBXv3000 con licencia para 15 usuarios.
- Teléfonos 3COM 3101 Basic.
- 3COM PCXSET (Softphone 3COM)
- Softphone Xlite (Asterisk)

3.3.1.1- Ubicación

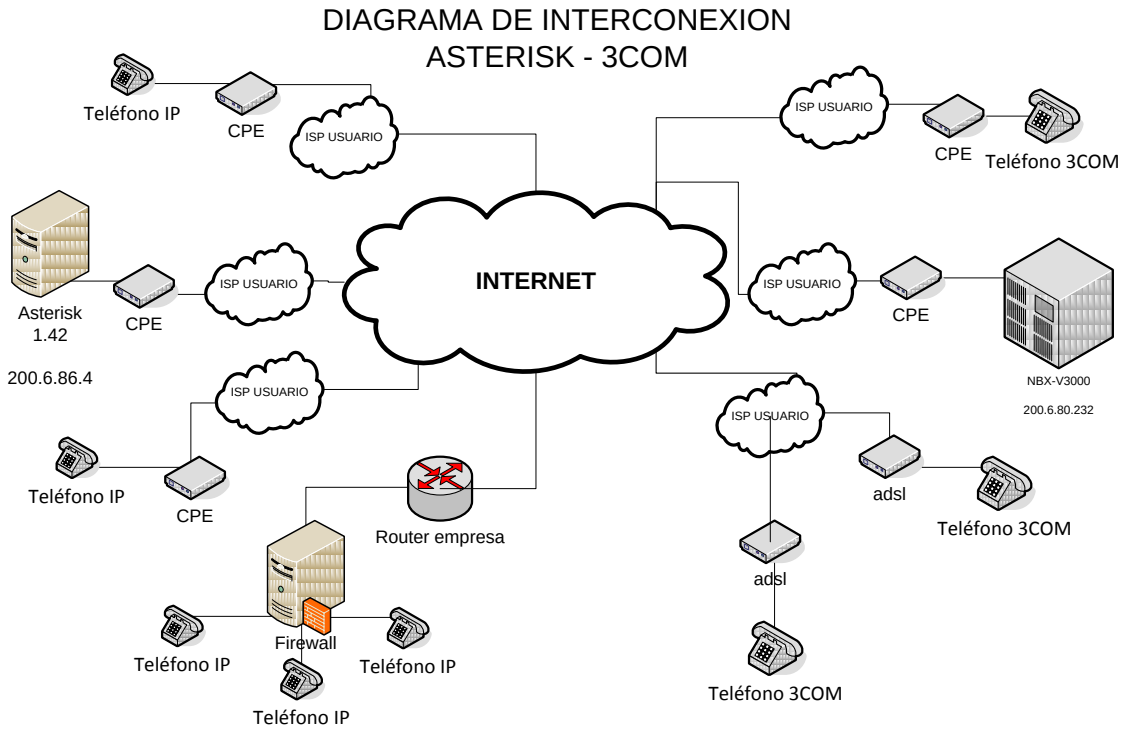


Figura 3.25: Topologia Proyecto

3.3.1.2- Definición de extensiones

Se ha definido las extensiones 44XX para el servidor Asterisk, se ha pensado en tener solamente 4 digitos para mantener una simetría con la central 3COM que tiene las extensiones 10XX.

3.3.1.3- Instalación de un sistema de monitoreo de trafico.

Conjuntamente con los archivos de instalación como lo vimos en el punto 3.1.1 instalamos el paquete iptraf que nos muestra los puertos utilizados y paquetes bytes enviados, que se muestra en una manera grafica como lo indica la figura.

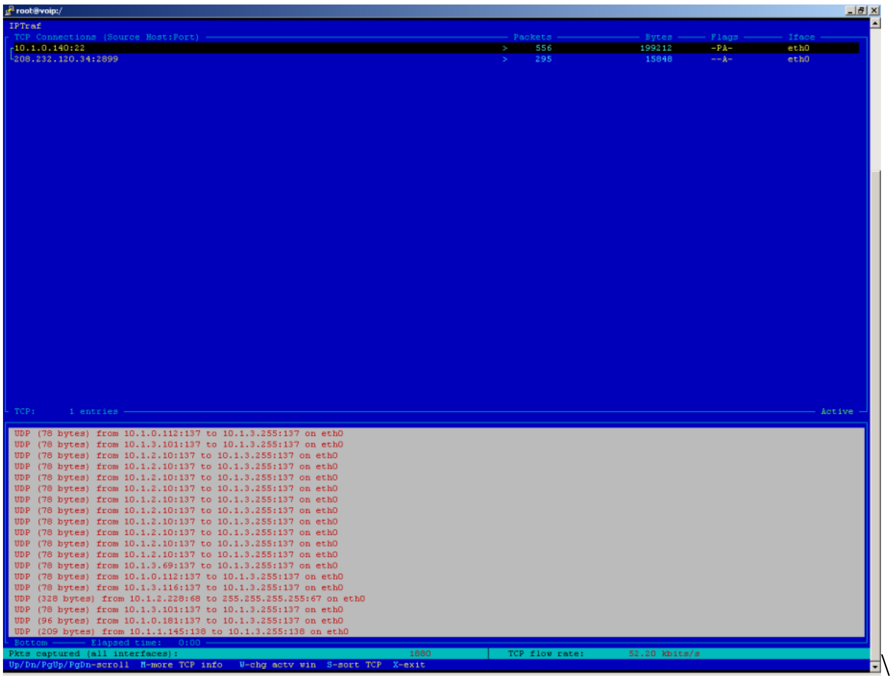


Figura 3.26: Sistema de Monitoreo.- Analisis de Paquetes 1

http://127.0.0.1/reporte

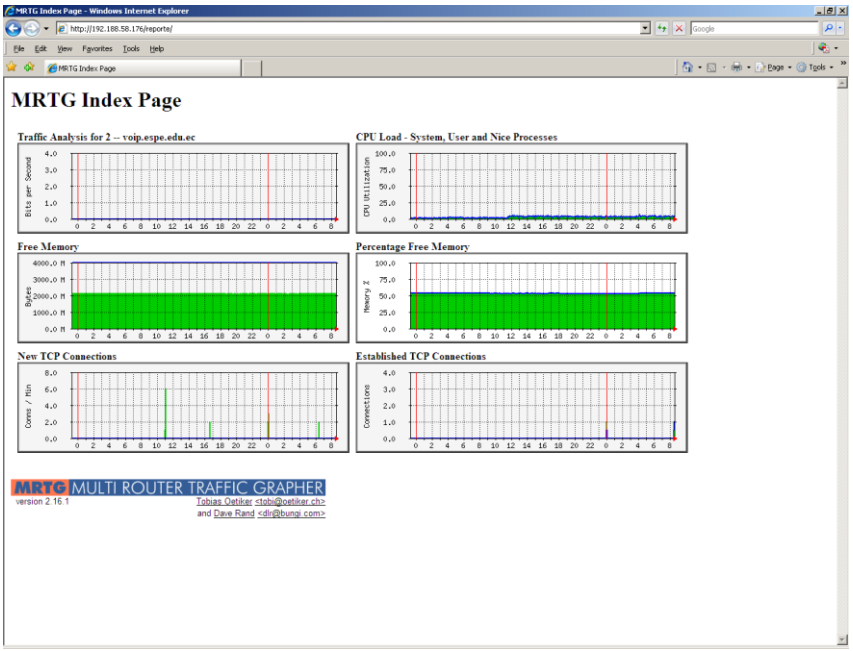


Figura 3.27: Sistema de Monitoreo.- Analisis de Paquetes 2

3.3.2- Pruebas

Para iniciar el servicio se digita el siguiente comando:

[root@voip /]# asterisk –rvvvv

Y se obtiene el siguiente mensaje en el momento que esta funcionando.

Asterisk 1.4.21.2, Copyright (C) 1999 - 2008 Digium, Inc. and others.

Created by Mark Spencer <markster@digium.com>

Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.

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.

```
============================================================
=========

  == Parsing '/etc/asterisk/asterisk.conf': Found

  == Parsing '/etc/asterisk/extconfig.conf': Found

Connected to Asterisk 1.4.21.2 currently running on voip (pid = 18120)

Verbosity is at least 4

voip*CLI>
```

Una vez dentro de la consola podemos revisar las llamadas, entre que contextos se hace la interconexión y el resultado de la interconexión, esto quiere decir el estado de la llamada.

3.3.2.1- Control del funcionamiento del servicio

Registramos un teléfono en el servidor asterisk en este caso el teléfono será un nokia E65 con capacidad de SIP.

```
-- Registered SIP '4410' at 190.154.89.139 port 1209 expires 3600

   -- Saved useragent "Nokia RM-208 3.0633.69.00" for peer 4410
```

Dentro de la consola de asterisk no cercioramos que las conexiones esten establecida con el comando

```
voip*CLI> sip show peers

espeexternos                200.6.80.232            5060     Unmonitored
```

4411/4411 (Unspecified) D 0 Unmonitored

4410/4410 190.154.89.139 D N 1209 Unmonitored

Una vez registrado probamos que este funcionando el buzón de voz, esto lo hacemos llamando a una extensión que no este asignado un teléfono y los resultados se verán de la siguiente manera.

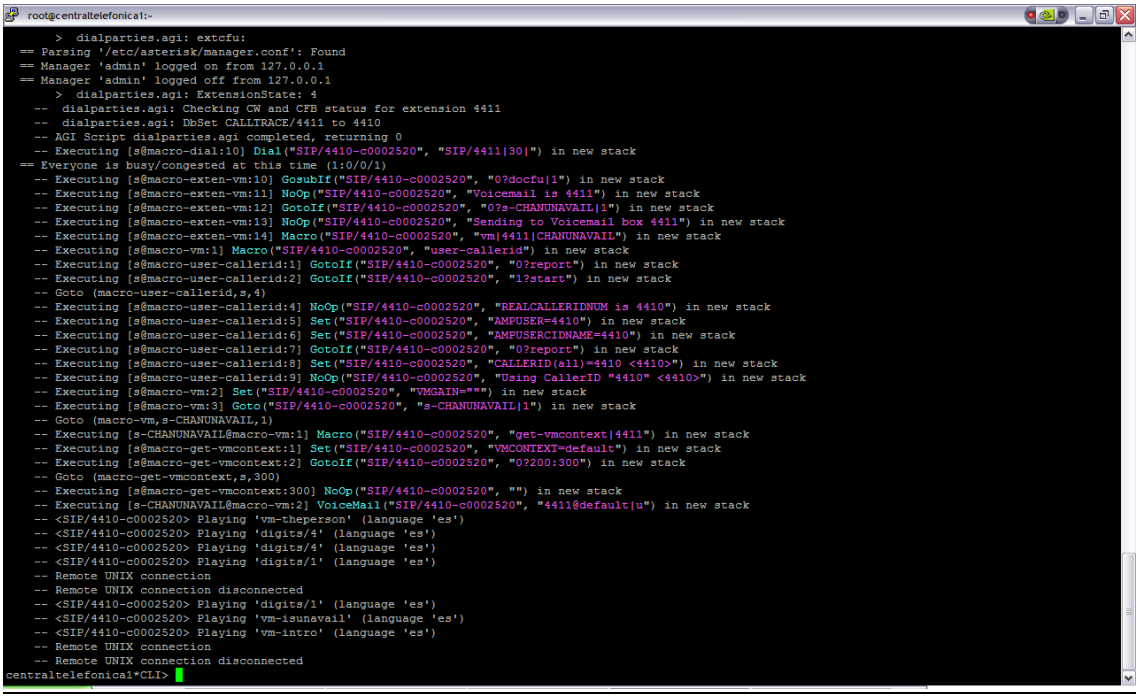


Figura 3.28: Consola Asterisk.- Llamado a una extensión 1

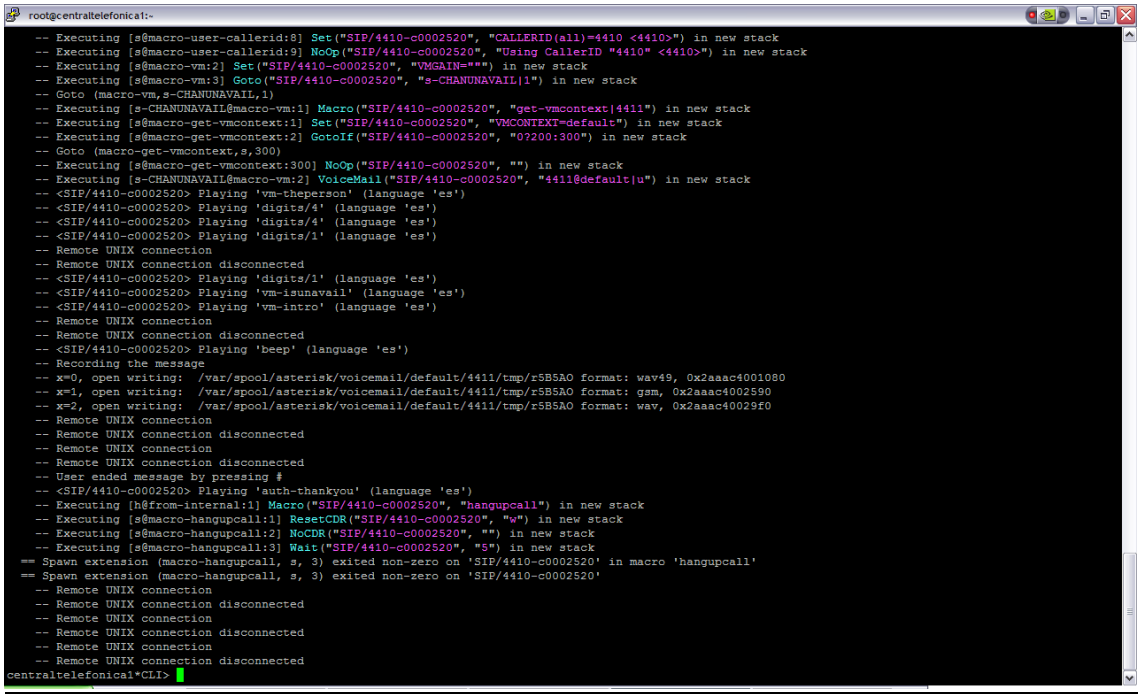


Figura 3.29: Consola Asterisk.- Llamado a una extensión 2

Realizamos una llamada a un teléfono autentificado en la central 3COM

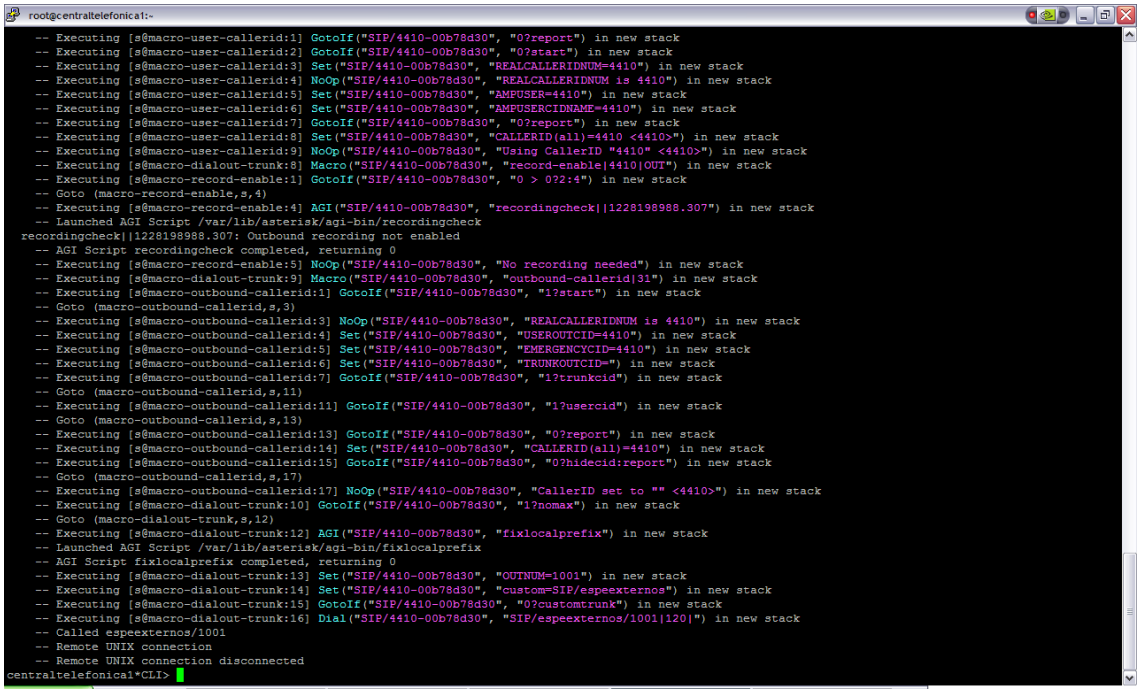


Figura 3.30: Consola Asterisk.- Llamado a telefono autentificado 3COM

3.3.2.3- Interpretación de los resultados

Nos damos cuenta que la llamada se enruta hacia el canal 3COM por el mensaje

-- Executing [s@macro-dialout-trunk:16] Dial("SIP/4410-c000beb0", "SIP/espeexternos/1001|120|") in new stack

Como ya habíamos descrito se enrutan de acuerdo a los contextos designados en este caso se enruta por el contexto espeexternos.

Las llamadas dentro del servidor se las toma como internas al igual que las llamadas hacia el buzon de correo

3.3.3- Manual de Administrador

El administrador tiene que cerciorarse que estén bien configuradas las interconexiones en la central 3COM y en el servidor asterisk.

Para comprobar que exista conexión entre 3COM y Asterisk nos dirigimos a Sip Applications -> Trusted SIP Interfaces

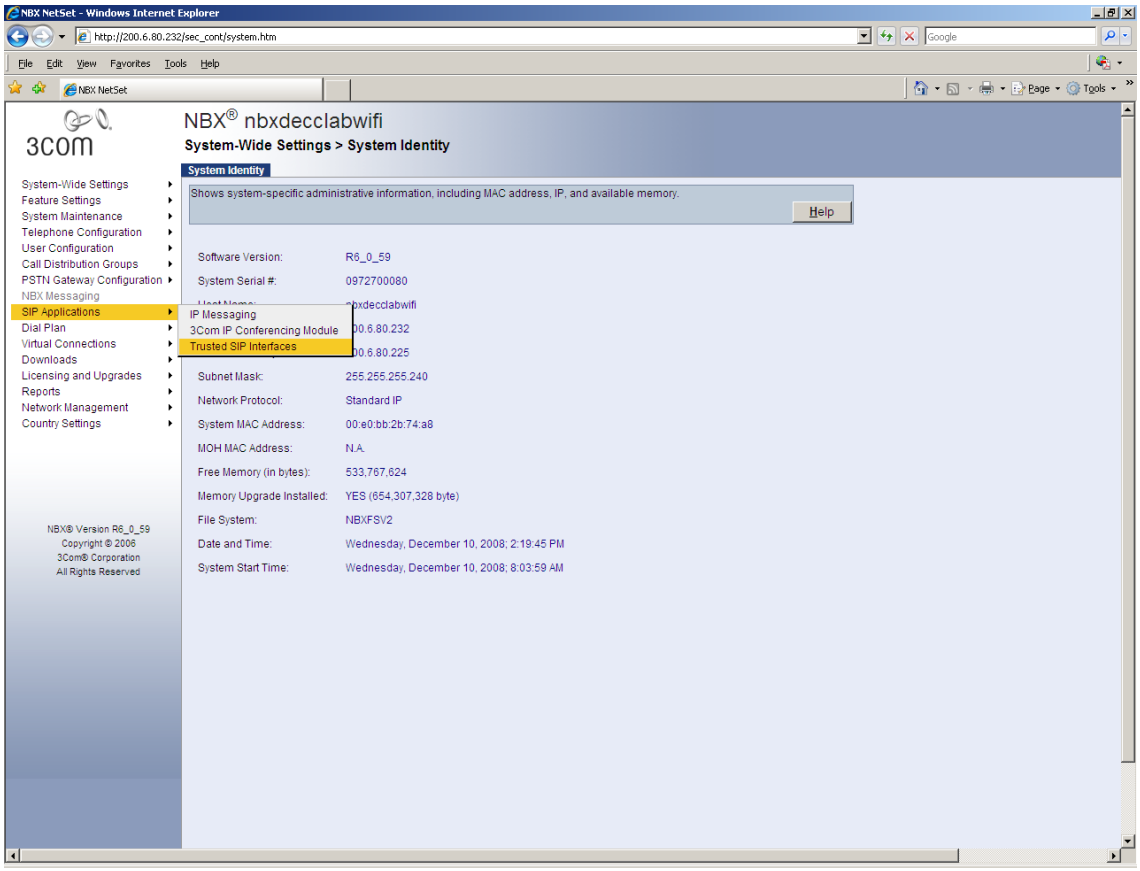


Figura 3.31: Central Telefonica NBX.- trusted SIP Interfaces

Nos dirigimos a la extensión asignada que en este caso es la 7001

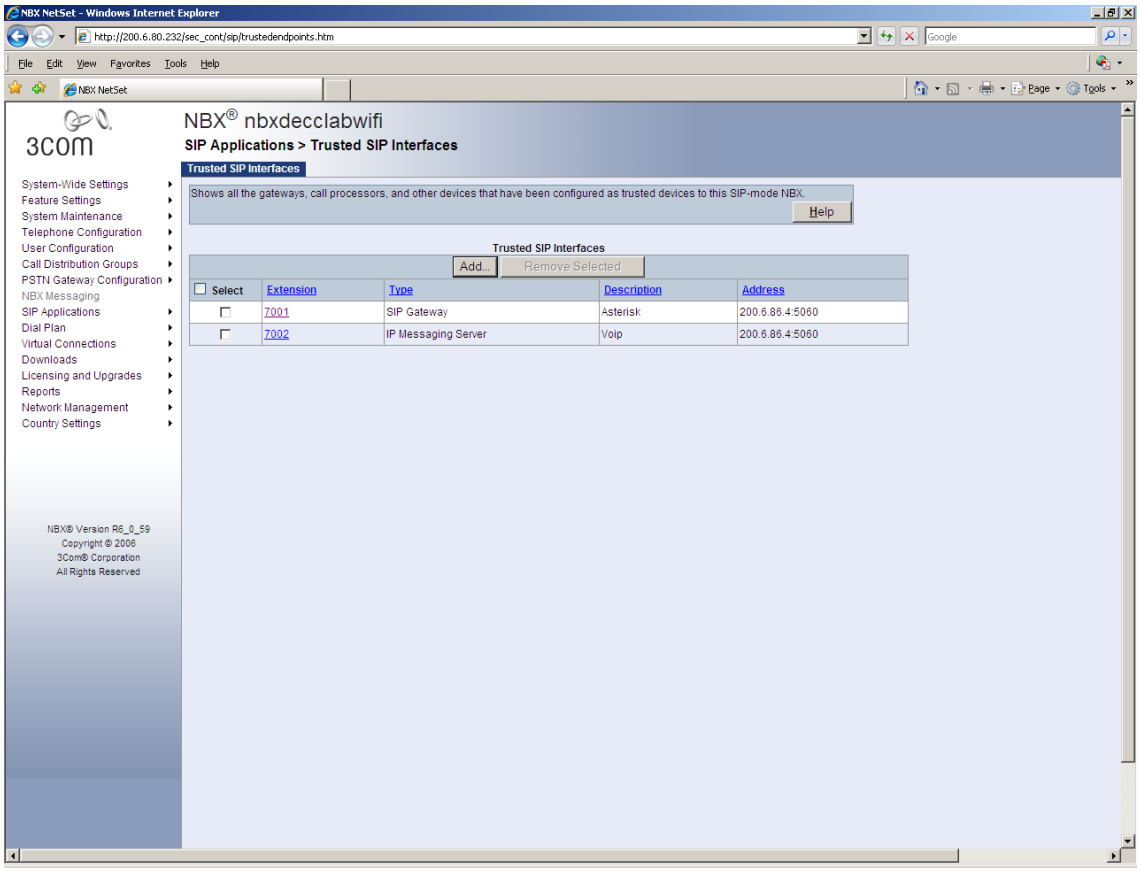


Figura 3.32: Central Telefonica NBX.- Asignacion de Extensiones

Y verificamos que se encuentre configurada la dirección ip de nuestro servidor que es 200.6.86.4

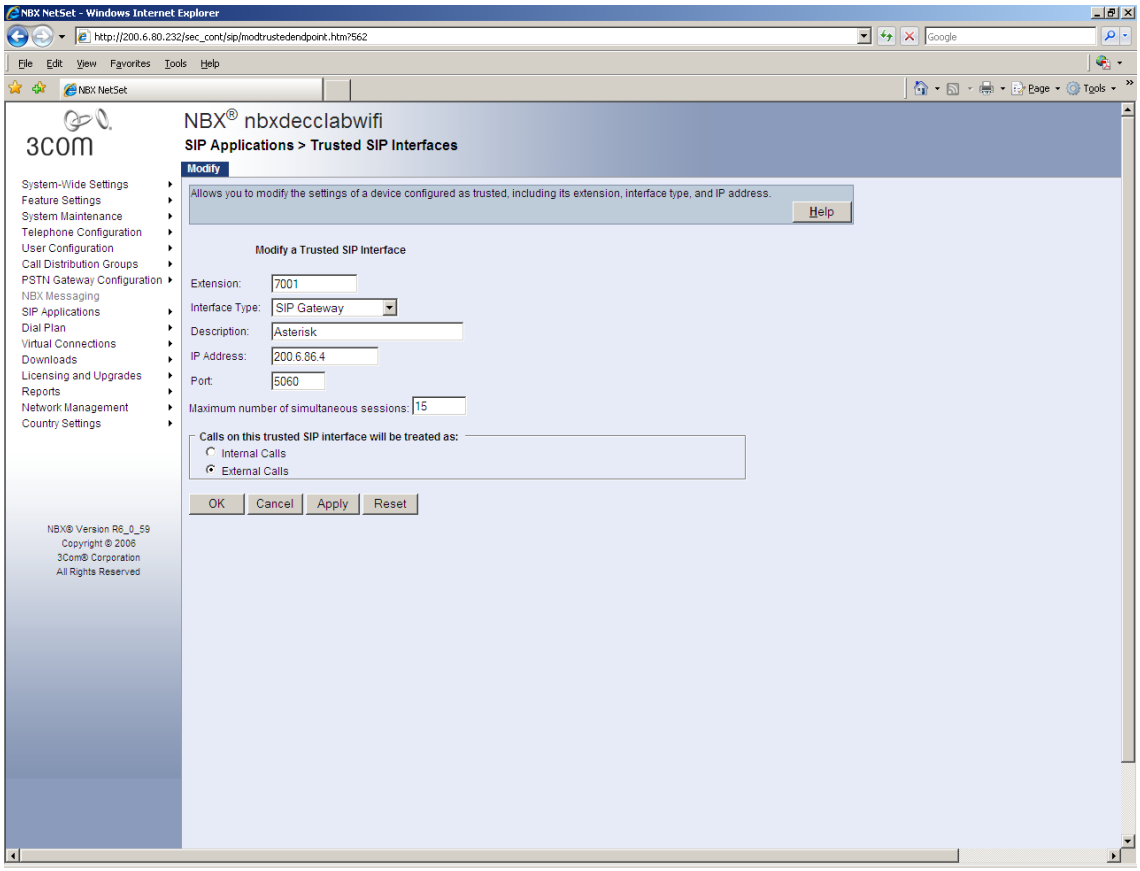


Figura 3.33: Central Telefonica NBX.- Direccion IP Servidor

Con esto comprobamos que la interconexión este realizada.

Definición de usuarios.

Para añadir un usuario en Asterisk nos dirigimos a la ubicación.

cd etc/asterisk/

Editamos el archivo sip.conf

vi sip.conf

Al final de este archivo añadimos una extensión nueva.

[4417]

type=friend

context=espevoip

secret=123456

host=dynamic

nat=yes

dtmfmode=rfc2833

username=4417

call-limit=3

disallow=all

allow=g723.1

allow=g729

allow=gsm

allow=alaw

allow=ulaw

allow=h263

allow=h263p

progressinband=yes

mailbox=4517

callerid="Tesis prueba"<4417>

callgroup=1

pickupgroup=1

Salimos de este archivo guardando los cambios.

Definición de extensiones.

Una vez definido el usuario definimos la extensión editamos el archivo extensions.conf

vi etc/asterisk/extensions.conf

exten => 4417,1,Dial(SIP/4417,15,Ttr)

exten => 4417,2,VoiceMail,4517

exten => 4417,3,Hangup

El contexto para la central 3com es espeexternos, el contexto de los casilleros de voz es casillerosvoz y los definimos así.

[espeexternos]

exten => _10XX,1,Dial(SIP/${EXTEN}@200.6.80.232,30,Ttr)

include => espevoip

Siempre hay q incluir el contexto principal.

[casillerosvoz]

exten => 4517,1,Ringing

exten => 4517,2,Wait(2)

exten => 4517,3,Authenticate(2222)

exten => 4517,4,VoicemailMain,s4517

include => espevoip

Salimos de este archivo guardando los cambios.

Buzón de Voz

Modificamos el archivo voicemail.conf

vi etc/asterisk/voicemail.conf

Y añadimos al final del archivo la siguiente línea por cada usuario

4511 => 2222,Pablo Jami T,pjami@eolnet.net

En donde definimos el correo electrónico y el nombre de la persona a cargo de la extensión.

Salimos de este archivo guardando los cambios.

3.3.4- Manual de Usuario

Cuando se configura una extensión del servidor ASTERISK

Se debe tomar en cuenta:

Usuario

Clave

Servidor de dominio

Para nuestro ejemplo utilizaremos el usuario 4411

Y nuestro servidor será 200.6.86.4

Ejemplo de configuración con un Soft phone.

Figura 3.34: X-lite Softphone

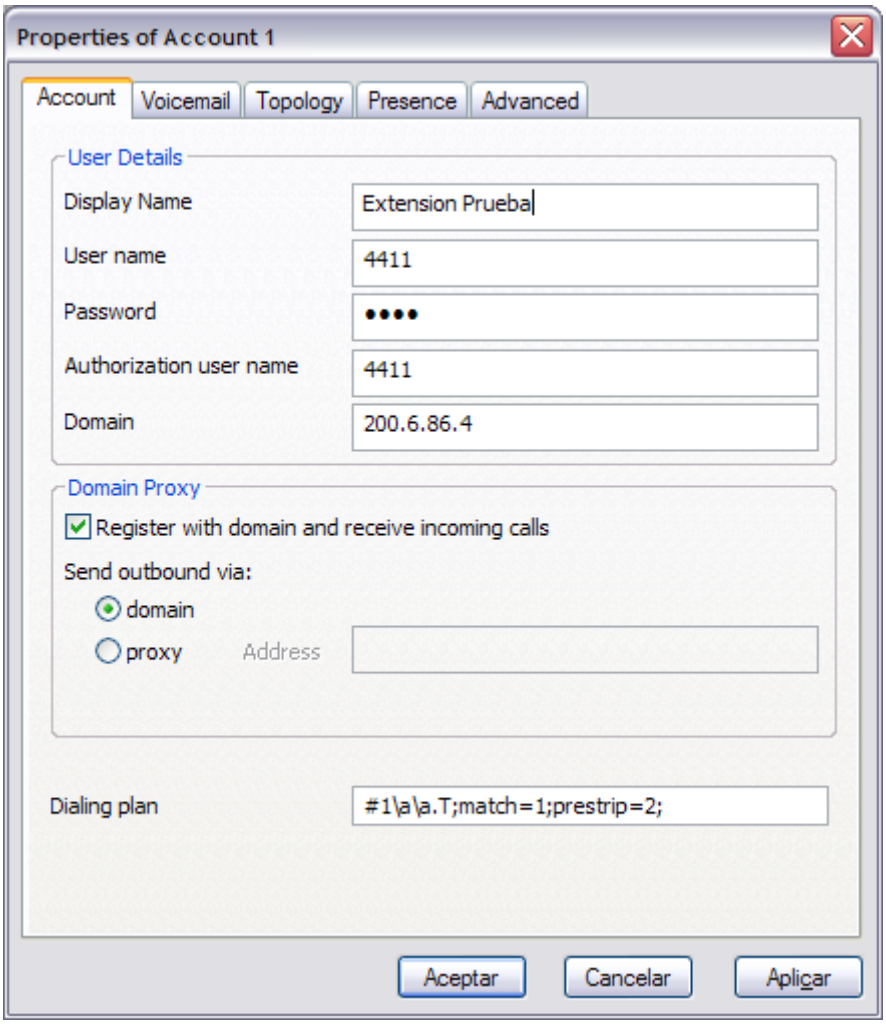
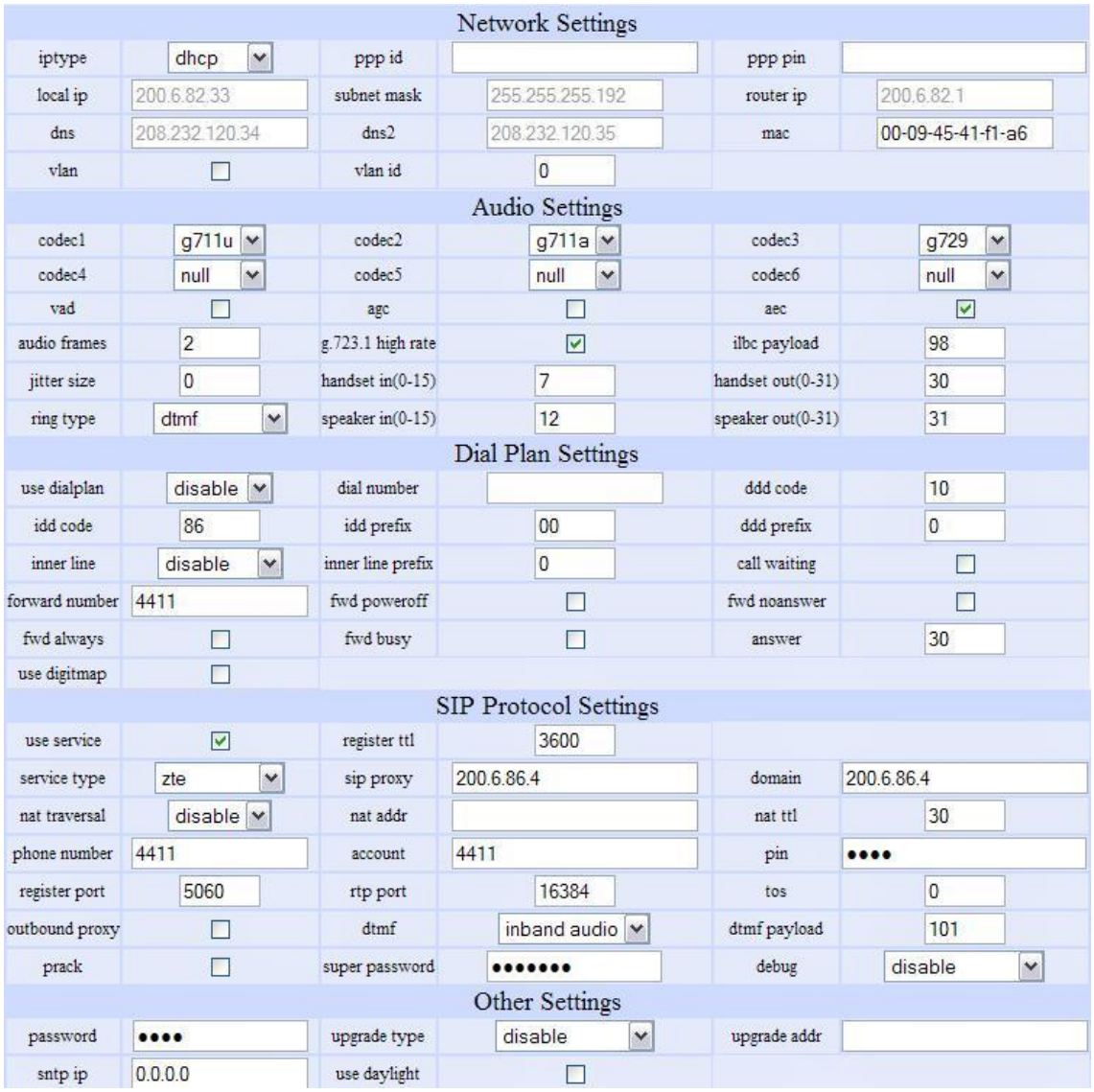


Figura 3.35: X-lite Softphone. – Configuracion de Softphone

Ejemplo de configuración con un teléfono IP marca Yuxin.



Network Settings					
iptype	dhcp	ppp id		ppp pin	
local ip	200.6.82.33	subnet mask	255.255.255.192	router ip	200.6.82.1
dns	208.232.120.34	dns2	208.232.120.35	mac	00-09-45-41-f1-a6
vlan	☐	vlan id	0		
Audio Settings					
codec1	g711u	codec2	g711a	codec3	g729
codec4	null	codec5	null	codec6	null
vad	☐	agc	☐	aec	☑
audio frames	2	g.723.1 high rate	☑	ilbc payload	98
jitter size	0	handset in(0-15)	7	handset out(0-31)	30
ring type	dtmf	speaker in(0-15)	12	speaker out(0-31)	31
Dial Plan Settings					
use dialplan	disable	dial number		ddd code	10
idd code	86	idd prefix	00	ddd prefix	0
inner line	disable	inner line prefix	0	call waiting	☐
forward number	4411	fwd poweroff	☐	fwd noanswer	☐
fwd always	☐	fwd busy	☐	answer	30
use digitmap	☐				
SIP Protocol Settings					
use service	☑	register ttl	3600		
service type	zte	sip proxy	200.6.86.4	domain	200.6.86.4
nat traversal	disable	nat addr		nat ttl	30
phone number	4411	account	4411	pin	••••
register port	5060	rtp port	16384	tos	0
outbound proxy	☐	dtmf	inband audio	dtmf payload	101
prack	☐	super password	•••••••	debug	disable
Other Settings					
password	••••	upgrade type	disable	upgrade addr	
sntp ip	0.0.0.0	use daylight	☐		

Figura 3.36: Configuracion Telefono Yuxin.

Como observamos en los ejemplos las configuraciones SIP principales son sencillas.

Capítulo IV
CONCLUSIONES Y RECOMENDACIONES

4.1- Conclusiones

Asterisk desde su creación ha venido evolucionando de una manera muy fuerte que ha hecho que fabricantes como 3COM de arquitecturas muy cerradas integren asterisk en sus equipos.

El protocolo SIP abierto se puede unir varios sistemas como centrales telefónicas como es nuestro caso, servidores asterisk, softswitchs, etc.

La escalabilidad (crecimiento en numero de usuarios a bajos costos, posibilidad de video conferencia, conferencias, etc) de los sistemas de telefonía IP a través de SIP se beneficia ampliamente, compensando servicios que carezcan ciertos sistemas como es el caso de la central 3COM NBX V3000.

Los sistemas open source (Código abierto) cuentan con soporte a través de foros, preguntas en línea o de sus mismos creadores, que hace mas fácil la consecución de los mismos.

4.2- Recomendaciones

Se recomienda usar direcciones IPs públicas para de esta manera poder acceder a la interconexión con varios servidores desde cualquier parte, sin necesidad de equipos adicionales, como routers, VPNs y demás.

Para futuras implementaciones en la central 3COM se recomienda utilizar algún equipo que permita hacer NAT con los teléfonos IP COM, para optimizar las direcciones IPs públicas.

Las actualizaciones de software en las centrales 3COM, vienen acompañadas de upgrades (mejoras, actualizaciones) de hardware, como en el caso de nuestra implementación, la actualización al Release 6.0.59 para la central NBX; que nos permitiría utilizar el protocolo SIP no propietario; fue necesario la compra de una memoria RAM de 512 Kb.

Se recomienda la instalación y uso de codecs de audio como g723 que optimizan el ancho de banda y mejorar la calidad de una conversación (evitar la robotización en la voz y los cortes en la misma debido a saturación de ancho de banda).

4.3- Bibliografia

- http://www.ietf.org/html.charters/sip-charter.html
- http://www.monografias.com/trabajos16/telefonia-senalizacion/telefonia-senalizacion.shtml
- http://www.sipcenter.com/sip.nsf/html/What+Is+SIP+Introduction
- http://voipex.blogspot.com/2006/03/protocolos-de-sealizacin.html
- http://es.wikipedia.org/wiki/Softphone
- http://www.microtronix.ca/sip_vs_h323.htm
- http://www.quarea.com/tutorial/SIP_session_initiation_protocol
- www.osmosislatina.com/**fedora**/**instalacion**.htm
- www.kmesystems.com/products/pdf/**3com_V3000_NBX**.pdf
- http://www.3com.com/products/en_US/detail.jsp?tab=features&pathtype=purchase&sku=3C10600A
- www.**asterisk**.org
- http://www.freesoftwaremagazine.com/articles/asterisk_the_easy_way?page=0%2C3

ANEXOS

Sip.conf

```
                                                ;
; SIP Configuration example for Asterisk
;
; Syntax for specifying a SIP device in extensions.conf is
; SIP/devicename where devicename is defined in a section below.
;
; You may also use
; SIP/username@domain to call any SIP user on the Internet
; (Don't forget to enable DNS SRV records if you want to use this)
;
; If you define a SIP proxy as a peer below, you may call
; SIP/proxyhostname/user or SIP/user@proxyhostname
; where the proxyhostname is defined in a section below
;
; Useful CLI commands to check peers/users:
;   sip show peers          Show all SIP peers (including friends)
;   sip show users          Show all SIP users (including friends)
;   sip show registry               Show status of hosts we register with
;
;   sip debug                       Show all SIP messages
;
;   reload chan_sip.so              Reload configuration file
;                                   Active SIP peers will not be reconfigured
;

[general]
context=default                 ; Default context for incoming calls
;allowguest=no                  ; Allow or reject guest calls (default is yes)
allowoverlap=no                 ; Disable overlap dialing support. (Default is yes)
;allowtransfer=no               ; Disable all transfers (unless enabled in peers or users)
                                ; Default is enabled
;realm=mydomain.tld             ; Realm for digest authentication
                                ; defaults to "asterisk". If you set a system name in
                                ; asterisk.conf, it defaults to that system name
                                ; Realms MUST be globally unique according to RFC 3261
                                ; Set this to your host name or domain name
bindport=5060                   ; UDP Port to bind to (SIP standard port is 5060)
                                ; bindport is the local UDP port that Asterisk will listen on
bindaddr=0.0.0.0        ; IP address to bind to (0.0.0.0 binds to all)
srvlookup=yes                   ; Enable DNS SRV lookups on outbound calls
                                ; Note: Asterisk only uses the first host
                                ; in SRV records
                                ; Disabling DNS SRV lookups disables the
                                ; ability to place SIP calls based on domain
                                ; names to some other SIP users on the Internet

;domain=mydomain.tld            ; Set default domain for this host
                                ; If configured, Asterisk will only allow
                                ; INVITE and REFER to non-local domains
                                ; Use "sip show domains" to list local domains
;pedantic=yes                   ; Enable checking of tags in headers,
                                ; international character conversions in URIs
                                ; and multiline formatted headers for strict
                                ; SIP compatibility (defaults to "no")

; See doc/ip-tos.txt for a description of these parameters.
;tos_sip=cs3            ; Sets TOS for SIP packets.
```

```
;tos_audio=ef              ; Sets TOS for RTP audio packets.
;tos_video=af41            ; Sets TOS for RTP video packets.

;maxexpiry=3600                    ; Maximum allowed time of incoming registrations
                                   ; and subscriptions (seconds)
;minexpiry=60                      ; Minimum length of registrations/subscriptions (default 60)
;defaultexpiry=120                 ; Default length of incoming/outgoing registration
;t1min=100                         ; Minimum roundtrip time for messages to monitored hosts
                                   ; Defaults to 100 ms
;notifymimetype=text/plain         ; Allow overriding of mime type in MWI NOTIFY
;checkmwi=10                       ; Default time between mailbox checks for peers
;buggymwi=no                       ; Cisco SIP firmware doesn't support the MWI RFC
                                   ; fully. Enable this option to not get error messages
                                   ; when sending MWI to phones with this bug.
;vmexten=voicemail                 ; dialplan extension to reach mailbox sets the
                                   ; Message-Account in the MWI notify message
                                   ; defaults to "asterisk"
disallow=all                       ; First disallow all codecs
allow=g723.1
allow=g729
allow=ulaw
allow=alaw

;
; This option specifies a preference for which music on hold class this channel
; should listen to when put on hold if the music class has not been set on the
; channel with Set(CHANNEL(musicclass)=whatever) in the dialplan, and the peer
; channel putting this one on hold did not suggest a music class.
;
; This option may be specified globally, or on a per-user or per-peer basis.
;
;mohinterpret=default
;
; This option specifies which music on hold class to suggest to the peer channel
; when this channel places the peer on hold. It may be specified globally or on
; a per-user or per-peer basis.
;
;mohsuggest=default
;
language=en                        ; Default language setting for all users/peers
                                   ; This may also be set for individual users/peers
;relaxdtmf=yes                     ; Relax dtmf handling
;trustrpid = no                    ; If Remote-Party-ID should be trusted
;sendrpid = yes                    ; If Remote-Party-ID should be sent
;progressinband=never              ; If we should generate in-band ringing always
                                   ; use 'never' to never use in-band signalling, even in cases
                                   ; where some buggy devices might not render it
                                   ; Valid values: yes, no, never Default: never
;useragent=Asterisk PBX            ; Allows you to change the user agent string
;promiscredir = no          ; If yes, allows 302 or REDIR to non-local SIP address
                            ; Note that promiscredir when redirects are made to the
                            ; local system will cause loops since Asterisk is incapable
                            ; of performing a "hairpin" call.
;usereqphone = no                  ; If yes, ";user=phone" is added to uri that contains
                                   ; a valid phone number
dtmfmode = rfc2833                 ; Set default dtmfmode for sending DTMF. Default: rfc2833
                                   ; Other options:
                                   ; info : SIP INFO messages
                                   ; inband : Inband audio (requires 64 kbit codec -alaw, ulaw)
```

```
                                ; auto : Use rfc2833 if offered, inband otherwise

;compactheaders = yes           ; send compact sip headers.
;
videosupport=yes                ; Turn on support for SIP video. You need to turn this on
                                ; in the this section to get any video support at all.
                                ; You can turn it off on a per peer basis if the general
                                ; video support is enabled, but you can't enable it for
                                ; one peer only without enabling in the general section.
maxcallbitrate=384              ; Maximum bitrate for video calls (default 384 kb/s)
                                ; Videosupport and maxcallbitrate is settable
                                ; for peers and users as well
;callevents=no                  ; generate manager events when sip ua
                                ; performs events (e.g. hold)
;alwaysauthreject = yes         ; When an incoming INVITE or REGISTER is to be rejected,
                                ; for any reason, always reject with '401 Unauthorized'
                                ; instead of letting the requester know whether there was
                                ; a matching user or peer for their request

;g726nonstandard = yes          ; If the peer negotiates G726-32 audio, use AAL2 packing
                                ; order instead of RFC3551 packing order (this is required
                                ; for Sipura and Grandstream ATAs, among others). This is
                                ; contrary to the RFC3551 specification, the peer _should_
                                ; be negotiating AAL2-G726-32 instead :-(

;matchexterniplocally = yes     ; Only substitute the externip or externhost setting if it matches
                        ; your localnet setting. Unless you have some sort of strange network
                        ; setup you will not need to enable this.

;
; If regcontext is specified, Asterisk will dynamically create and destroy a
; NoOp priority 1 extension for a given peer who registers or unregisters with
; us and have a "regexten=" configuration item.
; Multiple contexts may be specified by separating them with '&'. The
; actual extension is the 'regexten' parameter of the registering peer or its
; name if 'regexten' is not provided.  If more than one context is provided,
; the context must be specified within regexten by appending the desired
; context after '@'.  More than one regexten may be supplied if they are
; separated by '&'.  Patterns may be used in regexten.
;
;regcontext=sipregistrations
;
;--------------------------- RTP timers ----------------------------------------------
; These timers are currently used for both audio and video streams. The RTP timeouts
; are only applied to the audio channel.
; The settings are settable in the global section as well as per device
;
;rtptimeout=60                  ; Terminate call if 60 seconds of no RTP or RTCP activity
                                ; on the audio channel
                                ; when we're not on hold. This is to be able to hangup
                                ; a call in the case of a phone disappearing from the net,
                                ; like a powerloss or grandma tripping over a cable.
;rtpholdtimeout=300             ; Terminate call if 300 seconds of no RTP or RTCP activity
                                ; on the audio channel
                                ; when we're on hold (must be > rtptimeout)
;rtpkeepalive=<secs>            ; Send keepalives in the RTP stream to keep NAT open
                                ; (default is off - zero)
;--------------------------- SIP DEBUGGING -------------------------------------------
sipdebug = yes                  ; Turn on SIP debugging by default, from
```

```
                                        ; the moment the channel loads this configuration
recordhistory=yes                       ; Record SIP history by default
                                        ; (see sip history / sip no history)
dumphistory=yes             ; Dump SIP history at end of SIP dialogue
                                        ; SIP history is output to the DEBUG logging channel


;--------------------------- STATUS NOTIFICATIONS (SUBSCRIPTIONS) ----------------------------
; You can subscribe to the status of extensions with a "hint" priority
; (See extensions.conf.sample for examples)
; chan_sip support two major formats for notifications: dialog-info and SIMPLE
;
; You will get more detailed reports (busy etc) if you have a call limit set
; for a device. When the call limit is filled, we will indicate busy. Note that
; you need at least 2 in order to be able to do attended transfers.
;
; For queues, you will need this level of detail in status reporting, regardless
; if you use SIP subscriptions. Queues and manager use the same internal interface
; for reading status information.
;
; Note: Subscriptions does not work if you have a realtime dialplan and use the
; realtime switch.
;
;allowsubscribe=no                      ; Disable support for subscriptions. (Default is yes)
;subscribecontext = default             ; Set a specific context for SUBSCRIBE requests
                                        ; Useful to limit subscriptions to local extensions
                                        ; Settable per peer/user also
;notifyringing = yes                    ; Notify subscriptions on RINGING state (default: no)
;notifyhold = yes            ; Notify subscriptions on HOLD state (default: no)
                                        ; Turning on notifyringing and notifyhold will add a lot
                                        ; more database transactions if you are using realtime.
;limitonpeers = yes                     ; Apply call limits on peers only. This will improve
                                        ; status notification when you are using type=friend
                                        ; Inbound calls, that really apply to the user part
                                        ; of a friend will now be added to and compared with
                                        ; the peer limit instead of applying two call limits,
                                        ; one for the peer and one for the user.
                                        ; "sip show inuse" will only show active calls on
                                        ; the peer side of a "type=friend" object if this
                                        ; setting is turned on.

;---------------------------------------- T.38 FAX PASSTHROUGH SUPPORT -----------------------
;
; This setting is available in the [general] section as well as in device configurations.
; Setting this to yes, enables T.38 fax (UDPTL) passthrough on SIP to SIP calls, provided
; both parties have T38 support enabled in their Asterisk configuration
; This has to be enabled in the general section for all devices to work. You can then
; disable it on a per device basis.
;
; T.38 faxing only works in SIP to SIP calls, with no local or agent channel being used.
;
; t38pt_udptl = yes           ; Default false
;
;---------------------------------------- OUTBOUND SIP REGISTRATIONS  -----------------------
; Asterisk can register as a SIP user agent to a SIP proxy (provider)
; Format for the register statement is:
;       register => user[:secret[:authuser]]@host[:port][/extension]
;
; If no extension is given, the 's' extension is used. The extension needs to
```

```
; be defined in extensions.conf to be able to accept calls from this SIP proxy
; (provider).
;
; host is either a host name defined in DNS or the name of a section defined
; below.
;
; Examples:
;
;register => 1234:password@mysipprovider.com
;
;    This will pass incoming calls to the 's' extension
;
;
;register => 2345:password@sip_proxy/1234
;
;   Register 2345 at sip provider 'sip_proxy'.  Calls from this provider
;   connect to local extension 1234 in extensions.conf, default context,
;   unless you configure a [sip_proxy] section below, and configure a
;   context.
;   Tip 1: Avoid assigning hostname to a sip.conf section like [provider.com]
;   Tip 2: Use separate type=peer and type=user sections for SIP providers
;          (instead of type=friend) if you have calls in both directions

;registertimeout=20                ; retry registration calls every 20 seconds (default)
;registerattempts=10               ; Number of registration attempts before we give up
                                   ; 0 = continue forever, hammering the other server
                                   ; until it accepts the registration
                                   ; Default is 0 tries, continue forever

;----------------------------------------- NAT SUPPORT ------------------------
; The externip, externhost and localnet settings are used if you use Asterisk
; behind a NAT device to communicate with services on the outside.

externip = 192.188.58.176; Address that we're going to put in outbound SIP
                                   ; messages if we're behind a NAT

                                   ; The externip and localnet is used
                                   ; when registering and communicating with other proxies
                                   ; that we're registered with
;externhost=foo.dyndns.net         ; Alternatively you can specify an
                                   ; external host, and Asterisk will
                                   ; perform DNS queries periodically.  Not
                                   ; recommended for production
                                   ; environments!  Use externip instead
;externrefresh=10                  ; How often to refresh externhost if
                                   ; used
                                   ; You may add multiple local networks.  A reasonable
                                   ; set of defaults are:
localnet=10.1.0.0/255.255.252.0; All RFC 1918 addresses are local networks
;localnet=10.0.0.0/255.0.0.0       ; Also RFC1918
;localnet=172.16.0.0/12            ; Another RFC1918 with CIDR notation
;localnet=169.254.0.0/255.255.0.0 ;Zero conf local network

; The nat= setting is used when Asterisk is on a public IP, communicating with
; devices hidden behind a NAT device (broadband router).  If you have one-way
; audio problems, you usually have problems with your NAT configuration or your
; firewall's support of SIP+RTP ports.  You configure Asterisk choice of RTP
; ports for incoming audio in rtp.conf
;
```

```
nat=no                          ; Global NAT settings  (Affects all peers and users)
                        ; yes = Always ignore info and assume NAT
                        ; no = Use NAT mode only according to RFC3581 (;rport)
                        ; never = Never attempt NAT mode or RFC3581 support
                                ; route = Assume NAT, don't send rport
                                ; (work around more UNIDEN bugs)

;----------------------------------- MEDIA HANDLING ------------------------------
; By default, Asterisk tries to re-invite the audio to an optimal path. If there's
; no reason for Asterisk to stay in the media path, the media will be redirected.
; This does not really work with in the case where Asterisk is outside and have
; clients on the inside of a NAT. In that case, you want to set canreinvite=nonat
;
canreinvite=yes         ; Asterisk by default tries to redirect the
                        ; RTP media stream (audio) to go directly from
                        ; the caller to the callee.  Some devices do not
                        ; support this (especially if one of them is behind a NAT).
                        ; The default setting is YES. If you have all clients
                        ; behind a NAT, or for some other reason wants Asterisk to
                        ; stay in the audio path, you may want to turn this off.

                        ; In Asterisk 1.4 this setting also affect direct RTP
                        ; at call setup (a new feature in 1.4 - setting up the
                        ; call directly between the endpoints instead of sending
                        ; a re-INVITE).

;directrtpsetup=yes             ; Enable the new experimental direct RTP setup. This sets up
                        ; the call directly with media peer-2-peer without re-invites.
                        ; Will not work for video and cases where the callee sends
                        ; RTP payloads and fmtp headers in the 200 OK that does not match the
                        ; callers INVITE. This will also fail if canreinvite is enabled when
                        ; the device is actually behind NAT.

;canreinvite=nonat              ; An additional option is to allow media path redirection
                        ; (reinvite) but only when the peer where the media is being
                        ; sent is known to not be behind a NAT (as the RTP core can
                        ; determine it based on the apparent IP address the media
                        ; arrives from).

;canreinvite=update             ; Yet a third option... use UPDATE for media path redirection,
                        ; instead of INVITE. This can be combined with 'nonat', as
                        ; 'canreinvite=update,nonat'. It implies 'yes'.

;---------------------------------------- REALTIME SUPPORT -----------------------
; For additional information on ARA, the Asterisk Realtime Architecture,
; please read realtime.txt and extconfig.txt in the /doc directory of the
; source code.
;
;rtcachefriends=yes             ; Cache realtime friends by adding them to the internal list
                        ; just like friends added from the config file only on a
                        ; as-needed basis? (yes|no)

;rtsavesysname=yes              ; Save systemname in realtime database at registration
                        ; Default= no

;rtupdate=yes                   ; Send registry updates to database using realtime? (yes|no)
                        ; If set to yes, when a SIP UA registers successfully, the ip address,
                        ; the origination port, the registration period, and the username of
                        ; the UA will be set to database via realtime.
```

```
                                ; If not present, defaults to 'yes'.
;rtautoclear=yes         ; Auto-Expire friends created on the fly on the same schedule
                                ; as if it had just registered? (yes|no|<seconds>)
                                ; If set to yes, when the registration expires, the friend will
                                ; vanish from the configuration until requested again. If set
                                ; to an integer, friends expire within this number of seconds
                                ; instead of the registration interval.

;ignoreregexpire=yes             ; Enabling this setting has two functions:
                                ;
                                ; For non-realtime peers, when their registration expires, the
                                ; information will _not_ be removed from memory or the Asterisk
database
                                ; if you attempt to place a call to the peer, the existing information
                                ; will be used in spite of it having expired
                                ;
                                ; For realtime peers, when the peer is retrieved from realtime storage,
                                ; the registration information will be used regardless of whether
                                ; it has expired or not; if it expires while the realtime peer
                                ; is still in memory (due to caching or other reasons), the
                                ; information will not be removed from realtime storage

;---------------------------------------- SIP DOMAIN SUPPORT -----------------------
; Incoming INVITE and REFER messages can be matched against a list of 'allowed'
; domains, each of which can direct the call to a specific context if desired.
; By default, all domains are accepted and sent to the default context or the
; context associated with the user/peer placing the call.
; Domains can be specified using:
; domain=<domain>[,<context>]
; Examples:
; domain=myasterisk.dom
; domain=customer.com,customer-context
;
; In addition, all the 'default' domains associated with a server should be
; added if incoming request filtering is desired.
; autodomain=yes
;
; To disallow requests for domains not serviced by this server:
; allowexternaldomains=no

;domain=mydomain.tld,mydomain-incoming
                                ; Add domain and configure incoming context
                                ; for external calls to this domain
;domain=1.2.3.4                 ; Add IP address as local domain
                                ; You can have several "domain" settings
;allowexternaldomains=no        ; Disable INVITE and REFER to non-local domains
                                ; Default is yes
;autodomain=yes                 ; Turn this on to have Asterisk add local host
                                ; name and local IP to domain list.

; fromdomain=mydomain.tld       ; When making outbound SIP INVITEs to
                    ; non-peers, use your primary domain "identity"
                    ; for From: headers instead of just your IP
                    ; address. This is to be polite and
                    ; it may be a mandatory requirement for some
                    ; destinations which do not have a prior
                    ; account relationship with your server.

;------------------------------ JITTER BUFFER CONFIGURATION --------------------------
```

```
; jbenable = yes          ; Enables the use of a jitterbuffer on the receiving side of a
                         ; SIP channel. Defaults to "no". An enabled jitterbuffer will
                         ; be used only if the sending side can create and the receiving
                         ; side can not accept jitter. The SIP channel can accept jitter,
                         ; thus a jitterbuffer on the receive SIP side will be used only
                         ; if it is forced and enabled.

; jbforce = no            ; Forces the use of a jitterbuffer on the receive side of a SIP
                         ; channel. Defaults to "no".

; jbmaxsize = 200         ; Max length of the jitterbuffer in milliseconds.

; jbresyncthreshold = 1000    ; Jump in the frame timestamps over which the jitterbuffer is
                         ; resynchronized. Useful to improve the quality of the voice, with
                         ; big jumps in/broken timestamps, usually sent from exotic devices
                         ; and programs. Defaults to 1000.

; jbimpl = fixed          ; Jitterbuffer implementation, used on the receiving side of a SIP
                         ; channel. Two implementations are currently available - "fixed"
                         ; (with size always equals to jbmaxsize) and "adaptive" (with
                         ; variable size, actually the new jb of IAX2). Defaults to fixed.

; jblog = no              ; Enables jitterbuffer frame logging. Defaults to "no".
;-----------------------------------------------------------------------------

[authentication]
; Global credentials for outbound calls, i.e. when a proxy challenges your
; Asterisk server for authentication. These credentials override
; any credentials in peer/register definition if realm is matched.
;
; This way, Asterisk can authenticate for outbound calls to other
; realms. We match realm on the proxy challenge and pick an set of
; credentials from this list
; Syntax:
;       auth = <user>:<secret>@<realm>
;       auth = <user>#<md5secret>@<realm>
; Example:
;auth=mark:topsecret@digium.com
;
; You may also add auth= statements to [peer] definitions
; Peer auth= override all other authentication settings if we match on realm

;-----------------------------------------------------------------------------
; Users and peers have different settings available. Friends have all settings,
; since a friend is both a peer and a user
;
; User config options:      Peer configuration:
; --------------------      -------------------
; context                   context
; callingpres                       callingpres
; permit                    permit
; deny                      deny
; secret                    secret
; md5secret                 md5secret
; dtmfmode                  dtmfmode
; canreinvite               canreinvite
; nat                       nat
; callgroup                 callgroup
; pickupgroup               pickupgroup
```

```
; language                language
; allow                   allow
; disallow                disallow
; insecure                insecure
; trustrpid               trustrpid
; progressinband          progressinband
; promiscredir            promiscredir
; useclientcode           useclientcode
; accountcode             accountcode
; setvar                  setvar
; callerid                callerid
; amaflags                        amaflags
; call-limit                      call-limit
; allowoverlap                    allowoverlap
; allowsubscribe          allowsubscribe
; allowtransfer                   allowtransfer
; subscribecontext                subscribecontext
; videosupport                    videosupport
; maxcallbitrate          maxcallbitrate
; rfc2833compensate       mailbox
; t38pt_usertpsource      username
;                         template
;                         fromdomain
;                         regexten
;                         fromuser
;                         host
;                         port
;                         qualify
;                         defaultip
;                         rtptimeout
;                         rtpholdtimeout
;                         sendrpid
;                         outboundproxy
;                         rfc2833compensate
;                         t38pt_usertpsource

;[sip_proxy]
; For incoming calls only. Example: FWD (Free World Dialup)
; We match on IP address of the proxy for incoming calls
; since we can not match on username (caller id)
;type=peer
;context=from-fwd
;host=fwd.pulver.com

[coriparstun]
type=peer
context=espevoip
host=200.6.86.3
allow=g723.1
allow=g729
allow=ulaw
allow=alaw

[grupocoripar]
type=peer
context=espevoip
host=200.6.86.4
allow=g723.1
allow=g729
```

```
allow=ulaw
allow=alaw

[coripar]
type=peer
context=espevoip
host=200.6.81.68
allow=g723.1
allow=g729
allow=ulaw
allow=alaw

[zte]
type=peer
context=espevoip
host=66.165.169.100
allow=g723.1
allow=g729
allow=ulaw
allow=alaw

[espeexternos]
type=peer
context=espevoip
host=200.6.80.232
canreinvite=no
allow=g723.1
allow=g729
allow=ulaw
allow=alaw

;[sip_proxy-out]
;type=peer                          ; we only want to call out, not be called
;secret=guessit
;username=yourusername                      ; Authentication user for outbound proxies
;fromuser=yourusername                      ; Many SIP providers require this!
;fromdomain=provider.sip.domain
;host=box.provider.com
;usereqphone=yes                            ; This provider requires ";user=phone" on URI
;call-limit=5                               ; permit only 5 simultaneous outgoing calls to this peer
;outboundproxy=proxy.provider.domain        ; send outbound signaling to this proxy, not directly to the peer
                                            ; Call-limits will not be enforced on real-time peers,
                                            ; since they are not stored in-memory
;port=80                            ; The port number we want to connect to on the remote side
                                            ; Also used as "defaultport" in combination with "defaultip"
settings

;------------------------------------------------------------------------------
; Definitions of locally connected SIP devices
;
; type = user       a device that authenticates to us by "from" field to place calls
; type = peer       a device we place calls to or that calls us and we match by host
; type = friend two configurations (peer+user) in one
;
; For device names, we recommend using only a-z, numerics (0-9) and underscore
;
; For local phones, type=friend works most of the time
;
; If you have one-way audio, you probably have NAT problems.
```

```
; If Asterisk is on a public IP, and the phone is inside of a NAT device
; you will need to configure nat option for those phones.
; Also, turn on qualify=yes to keep the nat session open

;[grandstream1]
;type=friend
;context=from-sip                ; Where to start in the dialplan when this phone calls
;callerid=John Doe <1234>        ; Full caller ID, to override the phones config
                                 ; on incoming calls to Asterisk
;host=192.168.0.23               ; we have a static but private IP address
                                 ; No registration allowed
;nat=no                          ; there is not NAT between phone and Asterisk
;canreinvite=yes          ; allow RTP voice traffic to bypass Asterisk
;dtmfmode=info                   ; either RFC2833 or INFO for the BudgeTone
;call-limit=1                    ; permit only 1 outgoing call and 1 incoming call at a time
                                 ; from the phone to asterisk
                                 ; 1 for the explicit peer, 1 for the explicit user,
                                 ; remember that a friend equals 1 peer and 1 user in
                                 ; memory
                                 ; This will affect your subscriptions as well.
                                 ; There is no combined call counter for a "friend"
                                 ; so there's currently no way in sip.conf to limit
                                 ; to one inbound or outbound call per phone. Use
                                 ; the group counters in the dial plan for that.
                                 ;
;mailbox=1234@default            ; mailbox 1234 in voicemail context "default"
;disallow=all                    ; need to disallow=all before we can use allow=
;allow=ulaw                      ; Note: In user sections the order of codecs
                                 ; listed with allow= does NOT matter!
;allow=alaw
;allow=g723.1                    ; Asterisk only supports g723.1 pass-thru!
;allow=g729                      ; Pass-thru only unless g729 license obtained
;callingpres=allowed_passed_screen          ; Set caller ID presentation
                                 ; See doc/callingpres.txt for more information


;[xlite1]
; Turn off silence suppression in X-Lite ("Transmit Silence"=YES)!
; Note that Xlite sends NAT keep-alive packets, so qualify=yes is not needed
;type=friend
;regexten=1234                   ; When they register, create extension 1234
;callerid="Jane Smith" <5678>
;host=dynamic                    ; This device needs to register
;nat=yes                  ; X-Lite is behind a NAT router
;canreinvite=no                  ; Typically set to NO if behind NAT
;disallow=all
;allow=gsm                       ; GSM consumes far less bandwidth than ulaw
;allow=ulaw
;allow=alaw
;mailbox=1234@default,1233@default       ; Subscribe to status of multiple mailboxes


;[snom]
;type=friend                     ; Friends place calls and receive calls
;context=from-sip                ; Context for incoming calls from this user
;secret=blah
;subscribecontext=localextensions ; Only allow SUBSCRIBE for local extensions
;language=de                     ; Use German prompts for this user
;host=dynamic                    ; This peer register with us
```

```
;dtmfmode=inband                ; Choices are inband, rfc2833, or info
;defaultip=192.168.0.59         ; IP used until peer registers
;mailbox=1234@context,2345      ; Mailbox(-es) for message waiting indicator
;subscribemwi=yes               ; Only send notifications if this phone
                                ; subscribes for mailbox notification
;vmexten=voicemail              ; dialplan extension to reach mailbox
                                ; sets the Message-Account in the MWI notify message
                                ; defaults to global vmexten which defaults to "asterisk"
;disallow=all
;allow=ulaw                     ; dtmfmode=inband only works with ulaw or alaw!


;[polycom]
;type=friend                    ; Friends place calls and receive calls
;context=from-sip               ; Context for incoming calls from this user
;secret=blahpoly
;host=dynamic                   ; This peer register with us
;dtmfmode=rfc2833               ; Choices are inband, rfc2833, or info
;username=polly                 ; Username to use in INVITE until peer registers
                                ; Normally you do NOT need to set this parameter
;disallow=all
;allow=ulaw             ; dtmfmode=inband only works with ulaw or alaw!
;progressinband=no              ; Polycom phones don't work properly with "never"


;[pingtel]
;type=friend
;secret=blah
;host=dynamic
;insecure=port                  ; Allow matching of peer by IP address without
                                ; matching port number
;insecure=invite         ; Do not require authentication of incoming INVITEs
;insecure=port,invite           ; (both)
;qualify=1000                   ; Consider it down if it's 1 second to reply
                                ; Helps with NAT session
                                ; qualify=yes uses default value
;
; Call group and Pickup group should be in the range from 0 to 63
;
;callgroup=1,3-4         ; We are in caller groups 1,3,4
;pickupgroup=1,3-5              ; We can do call pick-p for call group 1,3,4,5
;defaultip=192.168.0.60         ; IP address to use if peer has not registered
;deny=0.0.0.0/0.0.0.0           ; ACL: Control access to this account based on IP address
;permit=192.168.0.60/255.255.255.0

;[cisco1]
;type=friend
;secret=blah
;qualify=200                    ; Qualify peer is no more than 200ms away
;nat=yes                ; This phone may be natted
                                ; Send SIP and RTP to the IP address that packet is
                                ; received from instead of trusting SIP headers
;host=dynamic                   ; This device registers with us
;canreinvite=no                 ; Asterisk by default tries to redirect the
                                ; RTP media stream (audio) to go directly from
                                ; the caller to the callee.  Some devices do not
                                ; support this (especially if one of them is
                                ; behind a NAT).
;defaultip=192.168.0.4          ; IP address to use until registration
```

```
;username=goran                 ; Username to use when calling this device before registration
                                ; Normally you do NOT need to set this parameter
;setvar=CUSTID=5678             ; Channel variable to be set for all calls from this device

;[pre14-asterisk]
;type=friend
;secret=digium
;host=dynamic
;rfc2833compensate=yes          ; Compensate for pre-1.4 DTMF transmission from another Asterisk
machine.
                                ; You must have this turned on or DTMF reception will work improperly.
;t38pt_usertpsource=yes         ; Use the source IP address of RTP as the destination IP address for UDPTL
packets
                        ; if the nat option is enabled. If a single RTP packet is received Asterisk will know the
                        ; external IP address of the remote device. If port forwarding is done at the client side
                        ; then UDPTL will flow to the remote device.


[9910]
type=friend
context=espevoip
secret=123456
host=dynamic
nat=no
canreinvite=yes
dtmfmode=rfc2833
username=9910
call-limit=3
disallow=all
allow=g723.1
allow=g729
allow=ulaw
allow=alaw
allow=gsm
progressinband=yes
mailbox=9810
callerid="Cristobal Espinosa"<9910>
callgroup=1
pickupgroup=1

[9911]
type=friend
context=espevoip
secret=123456
host=dynamic
nat=no
dtmfmode=rfc2833
username=9911
call-limit=3
disallow=all
allow=g723.1
allow=g729
allow=ulaw
allow=alaw
allow=gsm
progressinband=yes
mailbox=9811
callerid="Pablo Jami"<9911>
callgroup=1
```

```
pickupgroup=1

[9912]
type=friend
context=espevoip
secret=123456
host=dynamic
nat=yes
dtmfmode=rfc2833
username=9912
call-limit=3
disallow=all
allow=g723.1
allow=g729
allow=ulaw
allow=alaw
allow=gsm
progressinband=yes
mailbox=9812
callerid="Test"<9912>
callgroup=1
pickupgroup=1


[9913]
type=friend
context=espevoip
secret=123456
host=dynamic
nat=yes
dtmfmode=rfc2833
username=9913
call-limit=3
disallow=all
allow=g723.1
allow=g729
allow=ulaw
allow=alaw
allow=gsm
progressinband=yes
mailbox=9813
callerid="Test yo"<9913>
callgroup=1
pickupgroup=1

[4410]
type=friend
context=espevoip
secret=123456
host=dynamic
nat=yes
dtmfmode=rfc2833
username=4410
call-limit=3
disallow=all
allow=g723.1
allow=g729
allow=ulaw
allow=alaw
```

```
allow=gsm
progressinband=yes
mailbox=4510
callerid="Pablo"<4410>
callgroup=1
pickupgroup=1

[4411]
type=friend
context=espevoip
secret=123456
host=dynamic
nat=no
dtmfmode=rfc2833
username=4411
call-limit=3
disallow=all
allow=g723.1
allow=g729
allow=ulaw
allow=alaw
allow=gsm
progressinband=yes
mailbox=4511
callerid="Pablin"<4411>
callgroup=1
pickupgroup=1

[4412]
type=friend
context=espevoip
secret=123456
host=dynamic
nat=yes
dtmfmode=rfc2833
username=4412
call-limit=3
disallow=all
allow=g723.1
allow=g729
allow=ulaw
allow=alaw
allow=gsm
allow=h263
allow=h263p
progressinband=yes
mailbox=4512
callerid="Pablin Celu"<4412>
callgroup=1
pickupgroup=1


[4413]
type=friend
context=espevoip
secret=123456
host=dynamic
nat=yes
dtmfmode=rfc2833
```

username=4413
call-limit=3
disallow=all
allow=g723.1
allow=g729
allow=ulaw
allow=alaw
allow=gsm
allow=h263
allow=h263p
progressinband=yes
mailbox=4513
callerid="Pablin"<4413>
callgroup=1
pickupgroup=1

[4414]
type=friend
context=espevoip
secret=123456
host=dynamic
nat=yes
dtmfmode=rfc2833
username=4414
call-limit=3
disallow=all
allow=g723.1
allow=g729
allow=ulaw
allow=alaw
allow=gsm
progressinband=yes
mailbox=4514
callerid="Pablin"<4414>
callgroup=1
pickupgroup=1

[4415]
type=friend
context=espevoip
secret=123456
host=dynamic
nat=yes
dtmfmode=rfc2833
username=4415
call-limit=3
disallow=all
allow=g723.1
allow=g729
allow=ulaw
allow=alaw
allow=gsm
progressinband=yes
mailbox=4515
callerid="Pablin"<4415>
callgroup=1
pickupgroup=1

```
[4416]
type=friend
context=espevoip
secret=123456
host=dynamic
nat=yes
dtmfmode=rfc2833
username=4416
call-limit=3
disallow=all
allow=g723.1
allow=g729
allow=gsm
allow=alaw
allow=ulaw
allow=h263
allow=h263p
progressinband=yes
mailbox=4516
callerid="Pablin con video"<4416>
callgroup=1
pickupgroup=1


[4417]
type=friend
context=espevoip
secret=123456
host=dynamic
nat=yes
dtmfmode=rfc2833
username=4417
call-limit=3
disallow=all
allow=g723.1
allow=g729
allow=gsm
allow=alaw
allow=ulaw
allow=h263
allow=h263p
progressinband=yes
mailbox=4517
callerid="Pablin con video2"<4417>
callgroup=1
pickupgroup=1
```

extensions.conf

```
; extensions.conf - the Asterisk dial plan
;
; Static extension configuration file, used by
; the pbx_config module. This is where you configure all your
; inbound and outbound calls in Asterisk.
;
; This configuration file is reloaded
; - With the "dialplan reload" command in the CLI
; - With the "reload" command (that reloads everything) in the CLI

;
; The "General" category is for certain variables.
;
[general]
;
; If static is set to no, or omitted, then the pbx_config will rewrite
; this file when extensions are modified.  Remember that all comments
; made in the file will be lost when that happens.
;
; XXX Not yet implemented XXX
;
static=yes
;
; if static=yes and writeprotect=no, you can save dialplan by
; CLI command "dialplan save" too
;
writeprotect=no
;
; If autofallthrough is set, then if an extension runs out of
; things to do, it will terminate the call with BUSY, CONGESTION
; or HANGUP depending on Asterisk's best guess. This is the default.
;
; If autofallthrough is not set, then if an extension runs out of
; things to do, Asterisk will wait for a new extension to be dialed
; (this is the original behavior of Asterisk 1.0 and earlier).
;
;autofallthrough=no
;
; If clearglobalvars is set, global variables will be cleared
; and reparsed on an extensions reload, or Asterisk reload.
;
; If clearglobalvars is not set, then global variables will persist
; through reloads, and even if deleted from the extensions.conf or
; one of its included files, will remain set to the previous value.
;
; NOTE: A complication sets in, if you put your global variables into
; the AEL file, instead of the extensions.conf file. With clearglobalvars
; set, a "reload" will often leave the globals vars cleared, because it
; is not unusual to have extensions.conf (which will have no globals)
; load after the extensions.ael file (where the global vars are stored).
; So, with "reload" in this particular situation, first the AEL file will
; clear and then set all the global vars, then, later, when the extensions.conf
; file is loaded, the global vars are all cleared, and then not set, because
; they are not stored in the extensions.conf file.
;
```

```
clearglobalvars=no
;
; If priorityjumping is set to 'yes', then applications that support
; 'jumping' to a different priority based on the result of their operations
; will do so (this is backwards compatible behavior with pre-1.2 releases
; of Asterisk). Individual applications can also be requested to do this
; by passing a 'j' option in their arguments.
;
;priorityjumping=yes
;
; User context is where entries from users.conf are registered.  The
; default value is 'default'
;
;userscontext=default
;
; You can include other config files, use the #include command
; (without the ';'). Note that this is different from the "include" command
; that includes contexts within other contexts. The #include command works
; in all asterisk configuration files.
;#include "filename.conf"

; The "Globals" category contains global variables that can be referenced
; in the dialplan with the GLOBAL dialplan function:
; ${GLOBAL(VARIABLE)}
; ${${GLOBAL(VARIABLE)}} or ${text${GLOBAL(VARIABLE)}} or any hybrid
; Unix/Linux environmental variables can be reached with the ENV dialplan
; function: ${ENV(VARIABLE)}
;
[globals]
CONSOLE=Console/dsp                     ; Console interface for demo
;CONSOLE=Zap/1
;CONSOLE=Phone/phone0
IAXINFO=guest                           ; IAXtel username/password
;IAXINFO=myuser:mypass
TRUNK=Zap/G2                            ; Trunk interface
;
; Note the 'G2' in the TRUNK variable above. It specifies which group (defined
; in zapata.conf) to dial, i.e. group 2, and how to choose a channel to use in
; the specified group. The four possible options are:
;
; g: select the lowest-numbered non-busy Zap channel
;    (aka. ascending sequential hunt group).
; G: select the highest-numbered non-busy Zap channel
;    (aka. descending sequential hunt group).
; r: use a round-robin search, starting at the next highest channel than last
;    time (aka. ascending rotary hunt group).
; R: use a round-robin search, starting at the next lowest channel than last
;    time (aka. descending rotary hunt group).
;
TRUNKMSD=1                              ; MSD digits to strip (usually 1 or 0)
;TRUNK=IAX2/user:pass@provider

;
; Any category other than "General" and "Globals" represent
; extension contexts, which are collections of extensions.
;
; Extension names may be numbers, letters, or combinations
; thereof. If an extension name is prefixed by a '_'
; character, it is interpreted as a pattern rather than a
```

```
; literal.  In patterns, some characters have special meanings:
;
;   X - any digit from 0-9
;   Z - any digit from 1-9
;   N - any digit from 2-9
;   [1235-9] - any digit in the brackets (in this example, 1,2,3,5,6,7,8,9)
;   . - wildcard, matches anything remaining (e.g. _9011. matches
;           anything starting with 9011 excluding 9011 itself)
;   ! - wildcard, causes the matching process to complete as soon as
;       it can unambiguously determine that no other matches are possible
;
; For example the extension _NXXXXXX would match normal 7 digit dialings,
; while _1NXXNXXXXXX would represent an area code plus phone number
; preceded by a one.
;
; Each step of an extension is ordered by priority, which must
; always start with 1 to be considered a valid extension.  The priority
; "next" or "n" means the previous priority plus one, regardless of whether
; the previous priority was associated with the current extension or not.
; The priority "same" or "s" means the same as the previously specified
; priority, again regardless of whether the previous entry was for the
; same extension.  Priorities may be immediately followed by a plus sign
; and another integer to add that amount (most useful with 's' or 'n').
; Priorities may then also have an alias, or label, in
; parenthesis after their name which can be used in goto situations
;
; Contexts contain several lines, one for each step of each
; extension, which can take one of two forms as listed below,
; with the first form being preferred.
;
;[context]
;exten => someexten,{priority|label{+|-}offset}[(alias)],application(arg1,arg2,...)
;exten => someexten,{priority|label{+|-}offset}[(alias)],application,arg1|arg2...
;
; Included Contexts
;
; One may include another context in the current one as well, optionally with a
; date and time.  Included contexts are included in the order
; they are listed.
; The reason a context would include other contexts is for their
; extensions.
; The algorithm to find an extension is recursive, and works in this
; fashion:
;           first, given a stack on which to store context references,
;         push the context to find the extension onto the stack...
;    a) Try to find a matching extension in the context at the top of
;       the stack, and, if found, begin executing the priorities
;       there in sequence.
;    b) If not found, Search the switches, if any declared, in
;       sequence.
;    c) If still not found, for each include, push that context onto
;       the top of the context stack, and recurse to a).
;    d) If still not found, pop the entry from the top of the stack;
;       if the stack is empty, the search has failed. If it's not,
;       continue with the next context in c).
; This is a depth-first traversal, and stops with the first context
; that provides a matching extension. As usual, if more than one
; pattern in a context will match, the 'best' match will win.
; Please note that that extensions found in an included context are
```

```
; treated as if they were in the context from which the search began.
; The PBX's notion of the "current context" is not changed.
; Please note that in a context, it does not matter where an include
; directive occurs. Whether at the top, or near the bottom, the effect
; will be the same. The only thing that matters is that if there is
; more than one include directive, they will be searched for extensions
; in order, first to last.
; Also please note that pattern matches (like _9XX) are not treated
; any differently than exact matches (like 987). Also note that the
; order of extensions in a context have no affect on the outcome.
;
; Timing list for includes is
;
;   <time range>|<days of week>|<days of month>|<months>
;
; Note that ranges may be specified to wrap around the ends.  Also, minutes are
; fine-grained only down to the closest even minute.
;
;include => daytime|9:00-17:00|mon-fri|*|*
;include => weekend|*|sat-sun|*|*
;include => weeknights|17:02-8:58|mon-fri|*|*
;
; ignorepat can be used to instruct drivers to not cancel dialtone upon
; receipt of a particular pattern.  The most commonly used example is
; of course '9' like this:
;
;ignorepat => 9
;
; so that dialtone remains even after dialing a 9.
;

;
; Sample entries for extensions.conf
;
;
[dundi-e164-canonical]
;
; List canonical entries here
;
;exten => 12564286000,1,Macro(stdexten,6000,IAX2/foo)
;exten => _125642860XX,1,Dial(IAX2/otherbox/${EXTEN:7})

[dundi-e164-customers]
;
; If you are an ITSP or Reseller, list your customers here.
;
;exten => _12564286000,1,Dial(SIP/customer1)
;exten => _12564286001,1,Dial(IAX2/customer2)

[dundi-e164-via-pstn]
;
; If you are freely delivering calls to the PSTN, list them here
;
;exten => _1256428XXXX,1,Dial(Zap/G2/${EXTEN:7}) ; Expose all of 256-428
;exten => _1256325XXXX,1,Dial(Zap/G2/${EXTEN:7}) ; Ditto for 256-325

[dundi-e164-local]
;
; Context to put your dundi IAX2 or SIP user in for
```

```
; full access
;
include => dundi-e164-canonical
include => dundi-e164-customers
include => dundi-e164-via-pstn

[dundi-e164-switch]
;
; Just a wrapper for the switch
;
switch => DUNDi/e164

[dundi-e164-lookup]
;
; Locally to lookup, try looking for a local E.164 solution
; then try DUNDi if we don't have one.
;
include => dundi-e164-local
include => dundi-e164-switch
;
; DUNDi can also be implemented as a Macro instead of using
; the Local channel driver.
;
[macro-dundi-e164]
;
; ARG1 is the extension to Dial
;
; Extension "s" is not a wildcard extension that matches "anything".
; In macros, it is the start extension. In most other cases,
; you have to goto "s" to execute that extension.
;
; For wildcard matches, see above - all pattern matches start with
; an underscore.
exten => s,1,Goto(${ARG1},1)
include => dundi-e164-lookup

;
; Here are the entries you need to participate in the IAXTEL
; call routing system.  Most IAXTEL numbers begin with 1-700, but
; there are exceptions.  For more information, and to sign
; up, please go to www.gnophone.com or www.iaxtel.com
;
[iaxtel700]
exten => _91700XXXXXXX,1,Dial(IAX2/${GLOBAL(IAXINFO)}@iaxtel.com/${EXTEN:1}@iaxtel)

;
; The SWITCH statement permits a server to share the dialplan with
; another server. Use with care: Reciprocal switch statements are not
; allowed (e.g. both A -> B and B -> A), and the switched server needs
; to be on-line or else dialing can be severly delayed.
;
[iaxprovider]
;switch => IAX2/user:[key]@myserver/mycontext

[trunkint]
;
; International long distance through trunk
;
exten => _9011.,1,Macro(dundi-e164,${EXTEN:4})
```

```
exten => _9011.,n,Dial(${GLOBAL(TRUNK)}/${EXTEN:${GLOBAL(TRUNKMSD)}})

[trunkld]
;
; Long distance context accessed through trunk
;
exten => _91NXXNXXXXXX,1,Macro(dundi-e164,${EXTEN:1})
exten => _91NXXNXXXXXX,n,Dial(${GLOBAL(TRUNK)}/${EXTEN:${GLOBAL(TRUNKMSD)}})

[trunklocal]
;
; Local seven-digit dialing accessed through trunk interface
;
exten => _9NXXXXXX,1,Dial(${GLOBAL(TRUNK)}/${EXTEN:${GLOBAL(TRUNKMSD)}})

[trunktollfree]
;
; Long distance context accessed through trunk interface
;
exten => _91800NXXXXXX,1,Dial(${GLOBAL(TRUNK)}/${EXTEN:${GLOBAL(TRUNKMSD)}})
exten => _91888NXXXXXX,1,Dial(${GLOBAL(TRUNK)}/${EXTEN:${GLOBAL(TRUNKMSD)}})
exten => _91877NXXXXXX,1,Dial(${GLOBAL(TRUNK)}/${EXTEN:${GLOBAL(TRUNKMSD)}})
exten => _91866NXXXXXX,1,Dial(${GLOBAL(TRUNK)}/${EXTEN:${GLOBAL(TRUNKMSD)}})

[international]
;
; Master context for international long distance
;
ignorepat => 9
include => longdistance
include => trunkint

[longdistance]
;
; Master context for long distance
;
ignorepat => 9
include => local
include => trunkld

[local]
;
; Master context for local, toll-free, and iaxtel calls only
;
ignorepat => 9
include => default
include => trunklocal
include => iaxtel700
include => trunktollfree
include => iaxprovider

;Include parkedcalls (or the context you define in features conf)
;to enable call parking.
include => parkedcalls
;
; You can use an alternative switch type as well, to resolve
; extensions that are not known here, for example with remote
; IAX switching you transparently get access to the remote
; Asterisk PBX
```

```
;
; switch => IAX2/user:password@bigserver/local
;
; An "lswitch" is like a switch but is literal, in that
; variable substitution is not performed at load time
; but is passed to the switch directly (presumably to
; be substituted in the switch routine itself)
;
; lswitch => Loopback/12${EXTEN}@othercontext
;
; An "eswitch" is like a switch but the evaluation of
; variable substitution is performed at runtime before
; being passed to the switch routine.
;
; eswitch => IAX2/context@${CURSERVER}

[macro-trunkdial]
;
; Standard trunk dial macro (hangs up on a dialstatus that should
; terminate call)
;   ${ARG1} - What to dial
;
exten => s,1,Dial(${ARG1})
exten => s,n,Goto(s-${DIALSTATUS},1)
exten => s-NOANSWER,1,Hangup
exten => s-BUSY,1,Hangup
exten => _s-.,1,NoOp

[macro-stdexten];
;
; Standard extension macro:
;   ${ARG1} - Extension  (we could have used ${MACRO_EXTEN} here as well
;   ${ARG2} - Device(s) to ring
;
exten => s,1,Dial(${ARG2},20)                   ; Ring the interface, 20 seconds maximum
exten => s,2,Goto(s-${DIALSTATUS},1)            ; Jump based on status
(NOANSWER,BUSY,CHANUNAVAIL,CONGESTION,ANSWER)

exten => s-NOANSWER,1,Voicemail(${ARG1},u)  ; If unavailable, send to voicemail w/ unavail announce
exten => s-NOANSWER,2,Goto(default,s,1)             ; If they press #, return to start

exten => s-BUSY,1,Voicemail(${ARG1},b)              ; If busy, send to voicemail w/ busy announce
exten => s-BUSY,2,Goto(default,s,1)             ; If they press #, return to start

exten => _s-.,1,Goto(s-NOANSWER,1)              ; Treat anything else as no answer

exten => a,1,VoicemailMain(${ARG1})             ; If they press *, send the user into VoicemailMain

[macro-stdPrivacyexten];
;
; Standard extension macro:
;   ${ARG1} - Extension  (we could have used ${MACRO_EXTEN} here as well
;   ${ARG2} - Device(s) to ring
;   ${ARG3} - Optional DONTCALL context name to jump to (assumes the s,1 extension-priority)
;   ${ARG4} - Optional TORTURE context name to jump to (assumes the s,1 extension-priority)`
;
exten => s,1,Dial(${ARG2},20|p)                 ; Ring the interface, 20 seconds maximum, call
screening
                                                ; option (or use P for databased call screening)
```

```
exten => s,2,Goto(s-${DIALSTATUS},1)             ; Jump based on status
(NOANSWER,BUSY,CHANUNAVAIL,CONGESTION,ANSWER)

exten => s-NOANSWER,1,Voicemail(${ARG1},u) ; If unavailable, send to voicemail w/ unavail announce
exten => s-NOANSWER,2,Goto(default,s,1)                ; If they press #, return to start

exten => s-BUSY,1,Voicemail(${ARG1},b)                 ; If busy, send to voicemail w/ busy announce
exten => s-BUSY,2,Goto(default,s,1)              ; If they press #, return to start

exten => s-DONTCALL,1,Goto(${ARG3},s,1)                ; Callee chose to send this call to a polite
"Don't call again" script.

exten => s-TORTURE,1,Goto(${ARG4},s,1)                 ; Callee chose to send this call to a telemarketer
torture script.

exten => _s-.,1,Goto(s-NOANSWER,1)               ; Treat anything else as no answer

exten => a,1,VoicemailMain(${ARG1})              ; If they press *, send the user into VoicemailMain

[macro-page];
;
; Paging macro:
;
;      Check to see if SIP device is in use and DO NOT PAGE if they are
;
;   ${ARG1} - Device to page

exten => s,1,ChanIsAvail(${ARG1}|js)                   ; j is for Jump and s is for ANY call
exten => s,n,GoToIf([${AVAILSTATUS} = "1"]?autoanswer:fail)
exten => s,n(autoanswer),Set(_ALERT_INFO="RA")                         ; This is for the PolyComs
exten => s,n,SIPAddHeader(Call-Info: Answer-After=0)       ; This is for the Grandstream, Snoms, and
Others
exten => s,n,NoOp()                                    ; Add others here and Post on the Wiki!!!!
exten => s,n,Dial(${ARG1}||)
exten => s,n(fail),Hangup


[demo]
;
; We start with what to do when a call first comes in.
;
exten => s,1,Wait(1)                     ; Wait a second, just for fun
exten => s,n,Answer                      ; Answer the line
exten => s,n,Set(TIMEOUT(digit)=5)       ; Set Digit Timeout to 5 seconds
exten => s,n,Set(TIMEOUT(response)=10) ; Set Response Timeout to 10 seconds
exten => s,n(restart),BackGround(demo-congrats)    ; Play a congratulatory message
exten => s,n(instruct),BackGround(demo-instruct)   ; Play some instructions
exten => s,n,WaitExten                   ; Wait for an extension to be dialed.

exten => 2,1,BackGround(demo-moreinfo) ; Give some more information.
exten => 2,n,Goto(s,instruct)

exten => 3,1,Set(LANGUAGE()=fr)                  ; Set language to french
exten => 3,n,Goto(s,restart)                     ; Start with the congratulations

;exten => 1000,1,Goto(default,s,1)
;
; We also create an example user, 1234, who is on the console and has
; voicemail, etc.
```

```
;
exten => 1234,1,Playback(transfer,skip)                 ; "Please hold while..."
                                                        ; (but skip if channel is not up)
exten => 1234,n,Macro(stdexten,1234,${GLOBAL(CONSOLE)})

exten => 1235,1,Voicemail(1234,u)                       ; Right to voicemail

exten => 1236,1,Dial(Console/dsp)               ; Ring forever
exten => 1236,n,Voicemail(1234,b)                       ; Unless busy

;
; # for when they're done with the demo
;
exten => #,1,Playback(demo-thanks)              ; "Thanks for trying the demo"
exten => #,n,Hangup                             ; Hang them up.

;
; A timeout and "invalid extension rule"
;
exten => t,1,Goto(#,1)                          ; If they take too long, give up
exten => i,1,Playback(invalid)                  ; "That's not valid, try again"

;
; Create an extension, 500, for dialing the
; Asterisk demo.
;
exten => 500,1,Playback(demo-abouttotry); Let them know what's going on
exten => 500,n,Dial(IAX2/guest@pbx.digium.com/s@default)            ; Call the Asterisk demo
exten => 500,n,Playback(demo-nogo)          ; Couldn't connect to the demo site
exten => 500,n,Goto(s,6)                ; Return to the start over message.

;
; Create an extension, 600, for evaluating echo latency.
;
exten => 600,1,Playback(demo-echotest)      ; Let them know what's going on
exten => 600,n,Echo                             ; Do the echo test
exten => 600,n,Playback(demo-echodone)    ; Let them know it's over
exten => 600,n,Goto(s,6)                ; Start over

;
;           You can use the Macro Page to intercom a individual user
exten => 76245,1,Macro(page,SIP/Grandstream1)
; or if your peernames are the same as extensions
exten => _7XXX,1,Macro(page,SIP/${EXTEN})
;
;
; System Wide Page at extension 7999
;
exten => 7999,1,Set(TIMEOUT(absolute)=60)
exten => 7999,2,Page(Local/Grandstream1@page&Local/Xlite1@page&Local/1234@page/n|d)

; Give voicemail at extension 8500
;
exten => 8500,1,VoicemailMain
exten => 8500,n,Goto(s,6)
;
; Here's what a phone entry would look like (IXJ for example)
;
;exten => 1265,1,Dial(Phone/phone0,15)
```

```
;exten => 1265,n,Goto(s,5)

;
;       The page context calls up the page macro that sets variables needed for auto-answer
;       It is in is own context to make calling it from the Page() application as simple as
;       Local/{peername}@page
;
[page]
exten => _X.,1,Macro(page,SIP/${EXTEN})

;[mainmenu]
;
; Example "main menu" context with submenu
;
;exten => s,1,Answer
;exten => s,n,Background(thanks)            ; "Thanks for calling press 1 for sales, 2 for support, ..."
;exten => s,n,WaitExten
;exten => 1,1,Goto(submenu,s,1)
;exten => 2,1,Hangup
;include => default
;
;[submenu]
;exten => s,1,Ringing                                    ; Make them comfortable with 2 seconds of
ringback
;exten => s,n,Wait,2
;exten => s,n,Background(submenuopts)    ; "Thanks for calling the sales department.  Press 1 for steve, 2
for..."
;exten => s,n,WaitExten
;exten => 1,1,Goto(default,steve,1)
;exten => 2,1,Goto(default,mark,2)

[default]
;
; By default we include the demo.  In a production system, you
; probably don't want to have the demo there.
;
include => demo

;
; An extension like the one below can be used for FWD, Nikotel, sipgate etc.
; Note that you must have a [sipprovider] section in sip.conf
;
;exten => _41X.,1,Dial(SIP/${EXTEN:2}@sipprovider,,r)

; Real extensions would go here. Generally you want real extensions to be
; 4 or 5 digits long (although there is no such requirement) and start with a
; single digit that is fairly large (like 6 or 7) so that you have plenty of
; room to overlap extensions and menu options without conflict.  You can alias
; them with names, too, and use global variables

;exten => 6245,hint,SIP/Grandstream1&SIP/Xlite1,Joe Schmoe ; Channel hints for presence
;exten => 6245,1,Dial(SIP/Grandstream1,20,rt)       ; permit transfer
;exten => 6245,n(dial),Dial(${HINT},20,rtT)         ; Use hint as listed
;exten => 6245,n,Voicemail(6245,u)                  ; Voicemail (unavailable)
;exten => 6245,s+1,Hangup                           ; s+1, same as n
;exten => 6245,dial+101,Voicemail(6245,b)           ; Voicemail (busy)
;exten => 6361,1,Dial(IAX2/JaneDoe,,rm)             ; ring without time limit
;exten => 6389,1,Dial(MGCP/aaln/1@192.168.0.14)
;exten => 6390,1,Dial(JINGLE/caller/callee) ; Dial via jingle using labels
```

```
;exten => 6391,1,Dial(JINGLE/asterisk@digium.com/mogorman@astjab.org) ;Dial via jingle using asterisk as the transport and calling mogorman.
;exten => 6394,1,Dial(Local/6275/n)                    ; this will dial ${MARK}

;exten => 6275,1,Macro(stdexten,6275,${MARK})  ; assuming ${MARK} is something like Zap/2
;exten => mark,1,Goto(6275|1)                  ; alias mark to 6275
;exten => 6536,1,Macro(stdexten,6236,${WIL})   ; Ditto for wil
;exten => wil,1,Goto(6236|1)

;If you want to subscribe to the status of a parking space, this is
;how you do it. Subscribe to extension 6600 in sip, and you will see
;the status of the first parking lot with this extensions' help
;exten => 6600,hint,park:701@parkedcalls
;exten => 6600,1,noop
;
; Some other handy things are an extension for checking voicemail via
; voicemailmain
;
;exten => 8500,1,VoicemailMain
;exten => 8500,n,Hangup
;
; Or a conference room (you'll need to edit meetme.conf to enable this room)
;
;exten => 8600,1,Meetme(1234)
;
; Or playing an announcement to the called party, as soon it answers
;
;exten = 8700,1,Dial(${MARK},30,A(/path/to/my/announcemsg))
;
; For more information on applications, just type "core show applications" at your
; friendly Asterisk CLI prompt.
;
; "core show application <command>" will show details of how you
; use that particular application in this file, the dial plan.
; "core show functions" will list all dialplan functions
; "core show function <COMMAND>" will show you more information about
; one function. Remember that function names are UPPER CASE.

[espevoip]

exten => 9910,1,Dial(SIP/9910,15,Ttr)
exten => 9910,2,VoiceMail,9810
exten => 9910,3,Hangup

exten => 9911,1,Dial(SIP/9911,15,Ttr)
exten => 9911,2,VoiceMail,9811
exten => 9911,3,Hangup

exten => 9912,1,Dial(SIP/9912,15,Ttr)
exten => 9912,2,VoiceMail,9812
exten => 9912,3,Hangup

exten => 9913,1,Dial(SIP/9913,15,Ttr)
exten => 9913,2,VoiceMail,9813
exten => 9913,3,Hangup

exten => 4410,1,Dial(SIP/4410,15,Ttr)
exten => 4410,2,VoiceMail,4510
exten => 4410,3,Hangup
```

```
exten => 4411,1,Dial(SIP/4411,15,Ttr)
exten => 4411,2,VoiceMail,4511
exten => 4411,3,Hangup

exten => 4412,1,Dial(SIP/4412,15,Ttr)
exten => 4412,2,VoiceMail,4512
exten => 4412,3,Hangup

exten => 4413,1,Dial(SIP/4413,15,Ttr)
exten => 4413,2,VoiceMail,4513
exten => 4413,3,Hangup

exten => 4414,1,Dial(SIP/4414,15,Ttr)
exten => 4414,2,VoiceMail,4514
exten => 4414,3,Hangup

exten => 4415,1,Dial(SIP/4415,15,Ttr)
exten => 4415,2,VoiceMail,4515
exten => 4415,3,Hangup

exten => 4416,1,Dial(SIP/4416,15,Ttr)
exten => 4416,2,VoiceMail,4516
exten => 4416,3,Hangup

exten => 4417,1,Dial(SIP/4417,15,Ttr)
exten => 4417,2,VoiceMail,4517
exten => 4417,3,Hangup


include => casillerosvoz
include => espe3com
include => conferencia
include => coriparstun
include => coripar
include => grupocoripar
include => zte
include => espeexternos

[coriparstun]

exten => _96011XX,1,Dial(SIP/${EXTEN}@200.6.86.3,30,Ttr)
include => espevoip

[coripar]

exten => _4901XXX,1,Dial(SIP/${EXTEN}@200.6.81.68,30,Ttr)
include => espevoip

[grupocoripar]

exten => _4902XXX,1,Dial(SIP/${EXTEN}@200.6.86.4,30,Ttr)
include => espevoip

[zte]

exten => _26011XX,1,Dial(SIP/${EXTEN}@66.165.169.100,30,Ttr)
include => espevoip
```

```
[espeexternos]

exten => _10XX,1,Dial(SIP/${EXTEN}@200.6.80.232,30,Ttr)
include => espevoip

[casillerosvoz]

exten => 9810,1,Ringing
exten => 9810,2,Wait(2)
exten => 9810,3,Authenticate(2222)
exten => 9810,4,VoicemailMain,s9810

exten => 9811,1,Ringing
exten => 9811,2,Wait(2)
exten => 9811,3,Authenticate(2222)
exten => 9811,4,VoicemailMain,s9811

exten => 9812,1,Ringing
exten => 9812,2,Wait(2)
exten => 9812,3,Authenticate(2222)
exten => 9812,4,VoicemailMain,s9812

exten => 9813,1,Ringing
exten => 9813,2,Wait(2)
exten => 9813,3,Authenticate(2222)
exten => 9813,4,VoicemailMain,s9813

exten => 4510,1,Ringing
exten => 4510,2,Wait(2)
exten => 4510,3,Authenticate(2222)
exten => 4510,4,VoicemailMain,s4510

exten => 4511,1,Ringing
exten => 4511,2,Wait(2)
exten => 4511,3,Authenticate(2222)
exten => 4511,4,VoicemailMain,s4511

exten => 4512,1,Ringing
exten => 4512,2,Wait(2)
exten => 4512,3,Authenticate(2222)
exten => 4512,4,VoicemailMain,s4512

exten => 4513,1,Ringing
exten => 4513,2,Wait(2)
exten => 4513,3,Authenticate(2222)
exten => 4513,4,VoicemailMain,s4513

exten => 4514,1,Ringing
exten => 4514,2,Wait(2)
exten => 4514,3,Authenticate(2222)
exten => 4514,4,VoicemailMain,s4514

exten => 4515,1,Ringing
exten => 4515,2,Wait(2)
exten => 4515,3,Authenticate(2222)
exten => 4515,4,VoicemailMain,s4515

exten => 4516,1,Ringing
exten => 4516,2,Wait(2)
```

```
exten => 4516,3,Authenticate(2222)
exten => 4516,4,VoicemailMain,s4516

exten => 4517,1,Ringing
exten => 4517,2,Wait(2)
exten => 4517,3,Authenticate(2222)
exten => 4517,4,VoicemailMain,s4517


include => espevoip

[conferencia]

exten => 4400,1,MeetMe(4001,ri)
exten => 4401,1,MeetMe(4002)
exten => 4402,1,MeetMe(4003,r)
exten => 4403,1,MeetMe(4004,ri)
exten => 4404,1,MeetMe(4004,mq)

include => espe3com
include => espevoip

[mm-announce]
;Used by cbEnd to announce that a conference is about to end
;This sample assumes a five minute warning matching the value
;on line 138 of cbEnd.php
exten => 9999,1,Set(CALLERID(name)="MMGETOUT")
exten => 9999,n,Answer
exten => 9999,n,Playback(conf-will-end-in)
exten => 9999,n,Playback(digits/5)
exten => 9999,n,Playback(minutes)
exten => 9999,n,Hangup

[default]
;Use a DiD or internal extension in place of 1995
exten => 1995,1,Answer
exten => 1995,n,Wait(3)
exten => 1995,n,CBMysql()
exten => 1995,n,Hangup
```

Voicemail.conf

```
;
; Voicemail Configuration
;

;
; NOTE: Asterisk has to edit this file to change a user's password.  This does
; not currently work with the "#include <file>" directive for Asterisk
; configuration files, nor when using realtime static configuration.
; Do not use them with this configuration file.
;

[general]
; Formats for writing Voicemail.  Note that when using IMAP storage for
; voicemail, only the first format specified will be used.
;format=g723sf|wav49|wav
format=wav49|gsm|wav
;
; WARNING:
; If you change the list of formats that you record voicemail in
; when you have mailboxes that contain messages, you _MUST_ absolutely
; manually go through those mailboxes and convert/delete/add the
; the message files so that they appear to have been stored using
; your new format list. If you don't do this, very unpleasant
; things may happen to your users while they are retrieving and
; manipulating their voicemail.
;
; In other words: don't change the format list on a production system
; unless you are _VERY_  sure that you know what you are doing and are
; prepared for the consequences.
;
; Who the e-mail notification should appear to come from
serveremail=asterisk
;serveremail=asterisk@linux-support.net
; Should the email contain the voicemail as an attachment
attach=yes
; Maximum number of messages per folder.  If not specified, a default value
; (100) is used.  Maximum value for this option is 9999.
;maxmsg=100
; Maximum length of a voicemail message in seconds
;maxmessage=180
; Minimum length of a voicemail message in seconds for the message to be kept
; The default is no minimum.
;minmessage=3
; Maximum length of greetings in seconds
;maxgreet=60
; How many milliseconds to skip forward/back when rew/ff in message playback
skipms=3000
; How many seconds of silence before we end the recording
maxsilence=10
; Silence threshold (what we consider silence: the lower, the more sensitive)
silencethreshold=128
; Max number of failed login attempts
maxlogins=3
;
; User context is where entries from users.conf are registered.  The
; default value is 'default'
```

```
;
;userscontext=default
;
; If you need to have an external program, i.e. /usr/bin/myapp
; called when a voicemail is left, delivered, or your voicemailbox
; is checked, uncomment this.  It can also be set to 'smdi' to use
; smdi for external notification.  If it is 'smdi', smdiport should
; be set to a valid port as specified in smdi.conf.

;externnotify=/usr/bin/myapp
;smdiport=/dev/ttyS0

; If you need to have an external program, i.e. /usr/bin/myapp
; called when a voicemail password is changed, uncomment this:
;externpass=/usr/bin/myapp
; For the directory, you can override the intro file if you want
;directoryintro=dir-intro
; The character set for voicemail messages can be specified here
;charset=ISO-8859-1
; The ADSI feature descriptor number to download to
;adsifdn=0000000F
; The ADSI security lock code
;adsisec=9BDBF7AC
; The ADSI voicemail application version number.
;adsiver=1
; Skip the "[PBX]:" string from the message title
;pbxskip=yes
; Change the From: string
fromstring=Asterisk-3COM ESPE
; Permit finding entries for forward/compose from the directory
;usedirectory=yes
; Voicemail can be stored in a database using the ODBC driver.
; The value of odbcstorage is the database connection configured
; in res_odbc.conf.
;odbcstorage=asterisk
; The default table for ODBC voicemail storage is voicemessages.
;odbctable=voicemessages
;
; Change the from, body and/or subject, variables:
;    VM_NAME, VM_DUR, VM_MSGNUM, VM_MAILBOX, VM_CALLERID, VM_CIDNUM,
;    VM_CIDNAME, VM_DATE
;
; Note: The emailbody config row can only be up to 512 characters due to a
;       limitation in the Asterisk configuration subsystem.
emailsubject=[ESPE]: Nuevo mensaje desde ${VM_MSGNUM} al numero ${VM_MAILBOX}
; The following definition is very close to the default, but the default shows
; just the CIDNAME, if it is not null, otherwise just the CIDNUM, or "an unknown
; caller", if they are both null.
emailbody=Mensaje # ${VM_MSGNUM}.\n\nEstimado ${VM_NAME}:\n\n\tQueremos informarle que
tiene un nuevo mensaje en su buzon de voz de ${VM_DUR}seg. de duracion\nen el numero telefonico
${VM_MAILBOX} desde el numero telefonico ${VM_CALLERID}, el ${VM_DATE},
\n\nAtt.\n\n\t\t\t\tESPE 3COM\n\n\n\n\nFavor no responder este mensaje.
;
; You can also change the Pager From: string, the pager body and/or subject.
; The above defined variables also can be used here
;pagerfromstring=The Asterisk PBX
;pagersubject=New VM
;pagerbody=New ${VM_DUR} long msg in box ${VM_MAILBOX}\nfrom ${VM_CALLERID}, on
${VM_DATE}
```

```
;
; Set the date format on outgoing mails. Valid arguments can be found on the
; strftime(3) man page
;
; Default
emaildateformat=%A, %B %d, %Y at %r
; 24h date format
;emaildateformat=%A, %d %B %Y at %H:%M:%S
;
; You can override the default program to send e-mail if you wish, too
;
;mailcmd=/usr/sbin/sendmail -t
;
; Users may be located in different timezones, or may have different
; message announcements for their introductory message when they enter
; the voicemail system. Set the message and the timezone each user
; hears here. Set the user into one of these zones with the tz= attribute
; in the options field of the mailbox. Of course, language substitution
; still applies here so you may have several directory trees that have
; alternate language choices.
;
; Look in /usr/share/zoneinfo/ for names of timezones.
; Look at the manual page for strftime for a quick tutorial on how the
; variable substitution is done on the values below.
;
; Supported values:
; 'filename'    filename of a soundfile (single ticks around the filename
;               required)
; ${VAR}        variable substitution
; A or a        Day of week (Saturday, Sunday, ...)
; B or b or h   Month name (January, February, ...)
; d or e        numeric day of month (first, second, ..., thirty-first)
; Y             Year
; I or l        Hour, 12 hour clock
; H             Hour, 24 hour clock (single digit hours preceded by "oh")
; k             Hour, 24 hour clock (single digit hours NOT preceded by "oh")
; M             Minute, with 00 pronounced as "o'clock"
; N             Minute, with 00 pronounced as "hundred" (US military time)
; P or p        AM or PM
; Q             "today", "yesterday" or ABdY
;               (*note: not standard strftime value)
; q             "" (for today), "yesterday", weekday, or ABdY
;               (*note: not standard strftime value)
; R             24 hour time, including minute
;
;
;
; Each mailbox is listed in the form <mailbox>=<password>,<name>,<email>,<pager_email>,<options>
; if the e-mail is specified, a message will be sent when a message is
; received, to the given mailbox. If pager is specified, a message will be
; sent there as well. If the password is prefixed by '-', then it is
; considered to be unchangeable.
;
; Advanced options example is extension 4069
; NOTE: All options can be expressed globally in the general section, and
; overridden in the per-mailbox settings, unless listed otherwise.
;
; tz=central                    ; Timezone from zonemessages below. Irrelevant if envelope=no.
; attach=yes                    ; Attach the voicemail to the notification email *NOT* the pager email
```

```
; attachfmt=wav49          ; Which format to attach to the email.  Normally this is the
                            ; first format specified in the format parameter above, but this
                            ; option lets you customize the format sent to particular mailboxes.
                            ; Useful if Windows users want wav49, but Linux users want gsm.
                            ; [per-mailbox only]
; saycid=yes                ; Say the caller id information before the message. If not described,
                            ;    or set to no, it will be in the envelope
; cidinternalcontexts=intern          ; Internal Context for Name Playback instead of
                            ; extension digits when saying caller id.
; sayduration=no ; Turn on/off the duration information before the message. [ON by default]
; saydurationm=2       ; Specify the minimum duration to say. Default is 2 minutes
; dialout=fromvm ; Context to dial out from [option 4 from mailbox's advanced menu].
            ; If not specified, option 4 will not be listed and dialing out
            ; from within VoiceMailMain() will not be permitted.
sendvoicemail=yes ; Allow the user to compose and send a voicemail while inside
             ; VoiceMailMain() [option 5 from mailbox's advanced menu].
             ; If set to 'no', option 5 will not be listed.
; searchcontexts=yes        ; Current default behavior is to search only the default context
                            ; if one is not specified.  The older behavior was to search all contexts.
                            ; This option restores the old behavior [DEFAULT=no]
; callback=fromvm           ; Context to call back from
                            ;    if not listed, calling the sender back will not be permitted
; exitcontext=fromvm    ; Context to go to on user exit such as * or 0
                  ;    The default is the current context.
; review=yes                ; Allow sender to review/rerecord their message before saving it [OFF by default
; operator=yes              ; Allow sender to hit 0 before/after/during  leaving a voicemail to
                            ;    reach an operator  [OFF by default]
; envelope=no               ; Turn on/off envelope playback before message playback. [ON by default]
                            ;    This does NOT affect option 3,3 from the advanced options menu
; delete=yes                ; After notification, the voicemail is deleted from the server. [per-mailbox only]
                            ;    This is intended for use with users who wish to receive their
                            ;    voicemail ONLY by email. Note:  "deletevoicemail" is provided as an
                            ;    equivalent option for Realtime configuration.
; volgain=0.0               ; Emails bearing the voicemail may arrive in a volume too
                            ;    quiet to be heard.  This parameter allows you to specify how
                            ;    much gain to add to the message when sending a voicemail.
                            ;    NOTE: sox must be installed for this option to work.
; nextaftercmd=yes          ; Skips to the next message after hitting 7 or 9 to delete/save current message.
                            ;    [global option only at this time]
; forcename=yes             ; Forces a new user to record their name.  A new user is
                            ;    determined by the password being the same as
                            ;    the mailbox number.  The default is "no".
; forcegreetings=no         ; This is the same as forcename, except for recording
                            ;    greetings.  The default is "no".
; hidefromdir=yes           ; Hide this mailbox from the directory produced by app_directory
                            ;    The default is "no".
;tempgreetwarn=yes          ; Remind the user that their temporary greeting is set

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



[default]
; Define maximum number of messages per folder for a particular context.
```

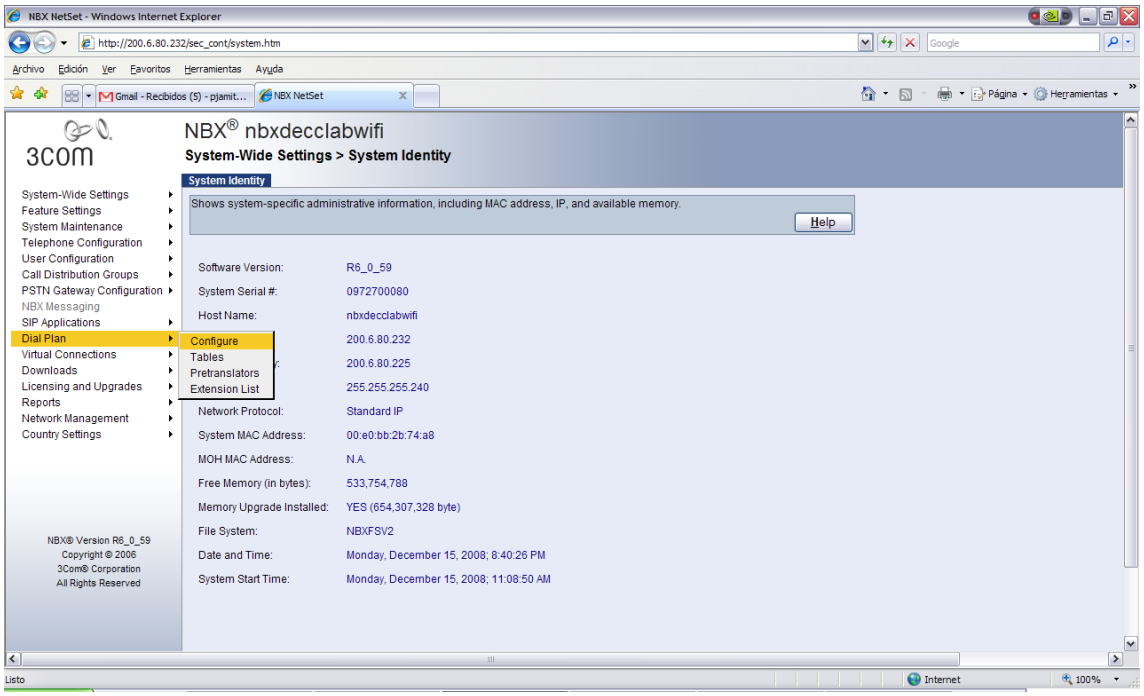
```
;maxmsg=50

1234 => 4242,Example Mailbox,root@localhost
;4200 => 9855,Mark Spencer,markster@linux-
support.net,mypager@digium.com,attach=no|serveremail=myaddy@digium.com|tz=central|maxmsg=10
;4300 => 3456,Ben Rigas,ben@american-computer.net
;4310 => -5432,Sales,sales@marko.net
;4069 => 6522,Matt
Brooks,matt@marko.net,,|tz=central|attach=yes|saycid=yes|dialout=fromvm|callback=fromvm|review=yes|op
erator=yes|envelope=yes|sayduration=yes|saydurationm=1
;4073 => 1099,Bianca Paige,bianca@biancapaige.com,,delete=1
;4110 => 3443,Rob Flynn,rflynn@blueridge.net
;4235 => 1234,Jim Holmes,jim@astricon.ips,,Tz=european

9810 => 2222,Cristobal Espinosa,pjjttb@hotmail.com
9811 => 2222,Cristobal Espinosa,cespinosa@eolnet.net
9812 => 2222,Pablo Jami T,pjami@eolnet.net
9813 => 2222,Alex ALmeida,pjami@eolnet.net

4511 => 2222,Pablo Jami T,pjami@eolnet.net
4512 => 2222,Pablo Jami Celu,pjjttb@hotmail.com
4513 => 2222,Pablin Nano,pjamitapia@gmail.com
4514 => 2222,Mauricio,cespinosa@eolnet.net
4515 => 2222,Pablin Xlite,pjami@eolnet.net
4516 => 2222,Pruebas,pjami@eolnet.net


;
; Mailboxes may be organized into multiple contexts for
; voicemail virtualhosting
;

[other]
;The intro can be customized on a per-context basis
;directoryintro=dir-company2
1234 => 5678,Company2 User,root@localhost
```

Meetme.conf

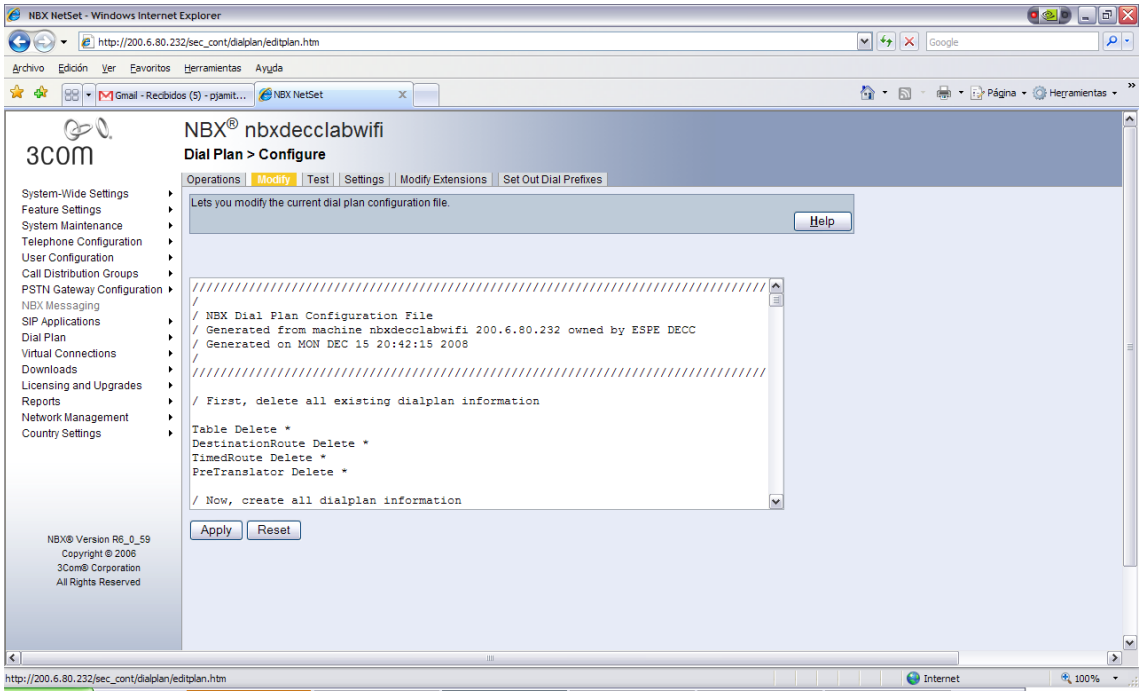
```
;
; Configuration file for MeetMe simple conference rooms for Asterisk of course.
;
; This configuration file is read every time you call app meetme()

[general]
;audiobuffers=32 ; The number of 20ms audio buffers to be used
                                ; when feeding audio frames from non-Zap channels
                                ; into the conference; larger numbers will allow
                                ; for the conference to 'de-jitter' audio that arrives
                                ; at different timing than the conference's timing
                                ; source, but can also allow for latency in hearing
                                ; the audio from the speaker. Minimum value is 2,
                                ; maximum value is 32.
;
[rooms]
;
; Usage is conf => confno[,pin][,adminpin]
;
; Note that once a participant has called the conference, a change to the pin
; number done in this file will not take effect until there are no more users
; in the conference and it goes away.  When it is created again, it will have
; the new pin number.
;
;conf => 1234
;conf => 2345,9938

conf => 4001,1111
conf => 4002
conf => 4004,1234
conf => 4003
conf => 4005,1234
```

Plan de marcación

Para configurar el plan de marcación nos dirigimos a Dial Plan y seleccionamos configurar



Seleccionamos modify



Y el plan de marcación debe estar configurado de la siguiente forma.

```
////////////////////////////////////////////////////////////////////////////
/
/ NBX Dial Plan Configuration File
/ Generated from machine nbxdecclabwifi 200.6.80.232 owned by ESPE DECC
/ Generated on MON DEC 15 20:42:15 2008
/
////////////////////////////////////////////////////////////////////////////

/ First, delete all existing dialplan information

Table Delete *
DestinationRoute Delete *
TimedRoute Delete *
PreTranslator Delete *

/ Now, create all dialplan information

////////////////////////////////////////////////////////////////////////////
/       Settings.   Note: ACD ranges included in HuntGroup category.
////////////////////////////////////////////////////////////////////////////

ExtensionLength 4
ExtensionRange Telephone     1000 3999
ExtensionRange Park          6000 6099
ExtensionRange AutoAttendant 5500 5599
ExtensionRange HuntGroup     4000 4099
ExtensionRange External      6000 7999
```

```
////////////////////////////////////////////////////////////////////////////
/ The ExtensionRange External Setting MUST include the Park range.
/ If the Call Park range is outside of the ExtensionRange External,
/ the Call Park feature will not work.
////////////////////////////////////////////////////////////////////////////
ExternalSettings 9 7250 500

////////////////////////////////////////////////////////////////////////////
/       Dial Plan Tables
////////////////////////////////////////////////////////////////////////////

Table Create 1 Internal 4 Digit Extensions
/               Id Entry  Digits      Min Max Class          Prio Route
/               -- -----  ------------ --- --- -------------   ---- -----
TableEntry Create   1     1  0           1   1 Internal         0     4
TableEntry Create   1     2  1           4   4 Internal         0     0
TableEntry Create   1     3  2           4   4 Internal         0     0
TableEntry Create   1     4  3           4   4 Internal         0     0
TableEntry Create   1     5  44          4   4 Internal         0     9
TableEntry Create   1     6  500         3   3 Internal         0     3
TableEntry Create   1     7  55          4   4 Internal         0     3
TableEntry Create   1     8  6           4   4 Internal         0     0
TableEntry Create   1     9  7           4   4 Diagnostics      0     0
TableEntry Create   1    10  9           8   8 Local            0     1
TableEntry Create   1    11  90          2  64 Operator         0     1
TableEntry Create   1    12  901         4  64 International    0     1
TableEntry Create   1    13  91          9  12 LongDistance     0     1
TableEntry Create   1    20  9101        9  64 AlternateLong    0     1
```

```
TableEntry Create   1    21  911          3   3 Emergency         0     2
TableEntry Create   1    22  91800        12  12 TollFree         0     1
TableEntry Create   1    23  91888        12  12 TollFree         0     1
TableEntry Create   1    24  91877        12  12 TollFree         0     1
TableEntry Create   1    25  91866        12  12 TollFree         0     1
TableEntry Create   1    26  91855        12  12 TollFree         0     1
TableEntry Create   1    27  91900        12  12 Toll             0     1
TableEntry Create   1    28  91976        12  12 Toll             0     1
TableEntry Create   1    30  9411         4   4 Operator          0     1
TableEntry Create   1    32  8            8   8 Local             0     8
TableEntry Create   1    33  80           2  64 Operator          0     8
TableEntry Create   1    34  801          4  64 International     0     8
TableEntry Create   1    35  81           9  12 LongDistance      0     8
TableEntry Create   1    36  8101         9  64 AlternateLong     0     8
TableEntry Create   1    37  81800        12  12 TollFree         0     8
TableEntry Create   1    38  81888        12  12 TollFree         0     8
TableEntry Create   1    39  81877        12  12 TollFree         0     8
TableEntry Create   1    40  81866        12  12 TollFree         0     8
TableEntry Create   1    41  81855        12  12 TollFree         0     8
TableEntry Create   1    42  81900        12  12 Toll             0     8
TableEntry Create   1    43  81976        12  12 Toll             0     8
TableEntry Create   1    44  8911         4   4 Emergency         0     8
TableEntry Create   1    45  8411         4   4 Operator          0     8
TableEntry Create   1    46  8*           4   4 COCode            0     8
TableEntry Create   1    47  7001         4   4 Internal          0     9
```

Table Create 2 Incoming 4 Digit DID and Auto At

```
/                Id Entry  Digits       Min Max Class          Prio Route
```

```
/               -- -----  ------------ --- --- -------------   ---- -----
TableEntry Create   2     1  0            1   1 Internal          0     4
TableEntry Create   2     2  1            4   4 Internal          0     0
TableEntry Create   2     3  2            4   4 Internal          0     0
TableEntry Create   2     4  3            4   4 Internal          0     0
TableEntry Create   2     5  4            4   4 Internal          0     0
TableEntry Create   2     6  500          3   3 Internal          0     3
TableEntry Create   2     7  55           4   4 Internal          0     3

Table Create 3 Least Cost Routing

////////////////////////////////////////////////////////////////////////////
/       Routes
////////////////////////////////////////////////////////////////////////////

/                   Route Description
/                   ----- -----------
DestinationRoute Create     1 LocalCO
DestinationRoute Create     2 LocalCONoStrip
DestinationRoute Create     3 Voice Application
DestinationRoute Create     4 Attendant
DestinationRoute Create     5 H323 ConneXtions Ports
DestinationRoute Create     8 8 Pool
DestinationRoute Create     9 SIP

/                       Route Entry DestinationExtension
/                       ----- ----- --------------------
DestinationRouteEntry Create     1     1 *0001
```

```
DestinationRouteEntry Create    1    2 *0002
DestinationRouteEntry Create    2    1 *0001
DestinationRouteEntry Create    2    2 *0002
DestinationRouteEntry Create    3    1 *0003
DestinationRouteEntry Create    4    1 *0004
DestinationRouteEntry Create    5    1 *0005
DestinationRouteEntry Create    8    1 *0008
DestinationRouteEntry Create    9    1 *1002

/                       Route Entry OperId Operation  Value
/                       ----- ----- ------ ---------  -----
DestinationRouteOperation Create    1    1     1 stripLead  1
DestinationRouteOperation Create    1    2     1 stripLead  1
DestinationRouteOperation Create    8    1     1 stripLead  1

////////////////////////////////////////////////////////////////////////////
/       Pretranslators
////////////////////////////////////////////////////////////////////////////

PreTranslator Create 1 5Digit DDI 4Digit Internal
/                   PreTransId Entry Digits
/                   ---------- ----- ------
PreTranslatorEntry Create        1    1 1
PreTranslatorEntry Create        1    2 2
PreTranslatorEntry Create        1    3 3
PreTranslatorEntry Create        1    4 4
PreTranslatorEntry Create        1    5 5
PreTranslatorEntry Create        1    6 6
```

```
PreTranslatorEntry Create        1     7 7

PreTranslatorEntry Create        1     8 8

PreTranslatorEntry Create        1     9 9

PreTranslatorEntry Create        1    10 0



/                        PreTransId Entry OperId Operation  Value

/                        ---------- ----- ------ ---------  -----

PreTranslatorOperation Create        1     1      1 stripLead  1

PreTranslatorOperation Create        1     2      1 stripLead  1

PreTranslatorOperation Create        1     3      1 stripLead  1

PreTranslatorOperation Create        1     4      1 stripLead  1

PreTranslatorOperation Create        1     5      1 stripLead  1

PreTranslatorOperation Create        1     6      1 stripLead  1

PreTranslatorOperation Create        1     7      1 stripLead  1

PreTranslatorOperation Create        1     8      1 stripLead  1

PreTranslatorOperation Create        1     9      1 stripLead  1

PreTranslatorOperation Create        1    10      1 stripLead  1


/ End of configuration



////////////////////////////////////////////////////////////////////////////

/ Configuration file command syntax guide:

/   Table Create {nTableId} {szDescription}

/   Table Delete {nTableId}

/   TableEntry Create {nTableId} {nEntryId} {szDigits}

/                    {nMinDigits} {nMaxDigits} {szCallClass}
```

/ {nPriority} {nRouteId}

/ TableEntry Delete {nTableId} {nEntryId}

/ DestinationRoute Create {nRouteId} {szDescription}

/ DestinationRoute Delete {nRouteId}

/ DestinationRouteEntry Create {nRouteId} {nEntryId} {szExtension}

/ DestinationRouteEntry Delete {nRouteId} {nEntryId}

/ DestinationRouteOperation Create {nRouteId} {nEntryId} {nOperId}

/ {szOperation} {szValue}

/ DestinationRouteOperation Delete {nRouteId} {nEntryId} {nOperId}

/ TimedRoute Create {nRouteId} {nDefaultDestinationRouteId} {szDescription}

/ TimedRoute Delete {nRouteId}

/ TimedRouteEntry Create {nRouteId} {nEntryId} {szStartTime} {szEndTime}

/ {szDaysOfWeek} {nDestinationRouteId}

/ TimedRouteEntry Delete {nRouteId} {nEntryId}

/ TimedRouteOperation Create {nRouteId} {nEntryId} {nOperId}

/ {szOperation} {szValue}

/ TimedRouteOperation Delete {nRouteId} {nEntryId} {nOperId}

/ PreTranslator Create {nPreTranslatorId} {szDescription}

/ PreTranslator Delete {nPreTranslatorId}

/ PreTranslatorEntry Create {nPreTranslatorId} {nEntryId} {szDigits}

/ PreTranslatorEntry Delete {nPreTranslatorId} {nEntryId}

/ PreTranslatorOperation Create {nPreTranslatorId} {nEntryId} {nOperId}

/ {szOperation} {szValue}

/ PreTranslatorOperation Delete {nPreTranslatorId} {nEntryId} {nOperId}

/ PreTranslatorISDNNumberType {nPreTranslatorId} {nISDNNumberType}

/ ExtensionLength {nExtensionLength}

/ ExtensionRange {szExtensionType} {szLowestExtension} {szHighestExtension}

/ ExternalSettings {szExternalKeysetPrefix} {szFirstAutoDiscoverExtension}

```
/                {szDefaultAutoExtension}
/ Notes:  1. Each command must be entered on one line.
/         2. Commands are case insensitive.
/         3. Tabs and spaces are ignored except in szDescription arguments.
/         4. The {} shown enclosing command argument names should not
/            be included in commands.
/         5. Command arguments beginning with n must be numbers.
/         6. Command arguments beginning with sz are strings.
/         7. nTableId 1 is the default Internal 3 digit dial plan table.
/         8. nTableId 2 is the default Incoming 3 digit dial plan table.
/         9. nTableId 3 is the default LCR dial plan table. (If used the
/            LCR table is checked first, if no entry exists, string is then
/            run through the associated internal dial plan.
/        10. szExtension *0001 is the default Line Card Port extension list
/        11. szExtension *0002 is the default T1 extension list
/        12. szExtension *0003 is the default Voicemail extension list
/        13. szExtension *0004 is the default Attendant extension list
/            (The lowest telephone extension that is Auto-discovered will
/             populate)
/        14. szExtension *0005 is the default H323 extension list
/        15. szExtension *0008 is the default 8 Pool extension list
/            (for backward compatibility, 8 Pool from R1.x upgrades)
/        16. szOperation can be: stripLead stripTrail replace prepend append
/        17. szCallClass can be: Internal Local LongDistance International WAN
/                                TollFree Emergency COCode Other Wireless Toll
/                                AlternateLong Operator TrunkToTrunk Diagnostics
/                                NotAllowed
/        18. route 0 always means look up internal device by extension
```

```
/       19. szStartTime and szEndTime are military time 00:00 through 23:59
/       20. szStartTime and szEndTime can be: open closed lunch other
/          (if specifying a system mode, both must be the same mode)
/       21. szExtensionType can be: telephone, park, autoAttendant, huntGroup,
/                              external, page
/       22. nISDNNumberType types for ETSI are as follows: (0, default) unknown;
/          (1) international; (2) national; (3) network; (4) subscriber
////////////////////////////////////////////////////////////////////////////////
```