PONTIFICIA **UNIVERSIDAD CATÓLICA** DEL PERÚ

PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

DISEÑO E IMPLEMENTACIÓN DE UNA RED DE TELEFONÍA IP CON SOFTWARE LIBRE EN LA RAAP

TESIS PARA OPTAR EL TÍTULO DE

INGENIERO DE LAS TELECOMUNICACIONES

PRESENTADO POR

Diego Quintana Cruz

LIMA – PERÚ

2007

RESUMEN

El presente proyecto de tesis consiste en analizar, diseñar e implementar una red piloto de telefonía IP en la Red Académica Peruana (RAAP) usando software libre. Durante el desarrollo de este proyecto se realiza una comparación de los diversos protocolos de señalización: SIP, IAX2; del hardware a utilizar: Teléfonos IP, ATAs; así como también de las diversas clases de *codecs*. Luego del análisis, se implementará la red VoIP. Esta red consistirá en un servidor principal y otro de respaldo para poder brindar una alta disponibilidad en caso de fallas. Ambos servidores contarán con el software Asterisk y un Sistema Operativo GNU/Linux. Una vez implementada la red de VoIP, se harán pruebas de esfuerzo para determinar la capacidad máxima de llamadas simultáneas que pueda soportar el sistema. Por último, se elaborará una recomendación formal a la RAAP sobre el uso de estas tecnologías.

DEDICATORIA

A mis padres,

A mis abuelos,

A mi hermano,

A Diana,

Y a mis amigos de toda la vida.

AGRADECIMIENTOS

Primero quiero agradecer a Dios por haber puesto en mi camino a todas aquellas personas que me ayudaron de alguna manera al desarrollo de la tesis.

A mis padres y hermano, por darme siempre la mejor educación que haya podido tener y su apoyo incondicional siempre que los necesito.

Al ingeniero David Chávez, por confiar en mí y darme toda la logística necesaria para poder realizar este y otros trabajos de investigación.

Al ingeniero Enrique Larios (Kike), por darme la oportunidad de enseñar lo poco que sé y lo que he ido aprendiendo.

Al ingeniero Daniel Díaz, por ser mi asesor y apoyarme en todas las correcciones de la Tesis y en el planteamiento de la misma.

A la comunidad mundial de Asterisk, en especial a Tzafrir Cohen, quien en más de una ocasión me ayudó con algunos problemas en la instalación y de configuración, a pesar de la enorme diferencia horaria. *Thanks a lot Tzafrir!*

A mis compañeros de promoción, sin su ayuda y consejos no hubiera podido terminar a tiempo. Quiero agradecer especialmente a Arturo Díaz por su apoyo en lo concerniente a la gestión del sistema.

Por último, quisiera agradecer a todas aquellas personas que sin querer olvido, ¡muchas gracias de todo corazón!

ÍNDICE

LISTA DE FIGURAS

LISTA DE TABLAS

GLOSARIO

APAN	*Asia Pacific Advanced Network*
APT	*Advanced Packaging Tool*
ATA	*Analog Telephone Adapter*
CLARA	Cooperación Latino Americana de Redes Avanzadas
FXO	*Foreign eXchange Office*
FXS	*Foreign eXchange Subscriber*
GSM	*Global System for Mobile communications*
FWD	*Free World Dialup*
IETF	*Internet Engineering Task Force*
IAX2	*Inter-Asterisk eXchange*, versión 2
iLBC	*Internet Low Bit-Rate Codec*
INICTEL	Instituto Nacional de Investigación y Capacitación en Telecomunicaciones
IP	*Internet Protocol*
IVR	*Interactive Voice Response*
LTP	*Long Term Prediction*
MGCP	*Media Gateway Control Protocol*
MOS	*Mean Opinion Score*
NAT	*Network Address Translation*
PBX	*Private Branch Exchange*
PCM	*Pulse Code Modulation*
PSTN	*Public Switch Telephone Network*
PUCP	Pontificia Universidad Católica del Perú
QoS	*Quality of Service* (Calidad de Servicio)
RAAP	Red Académica Peruana
RAID	*Redundant Array of Independent/Inexpensive Disks*
RAS	*Registration, Admission and Status*
RDSI	Red digital de Servicios Integrados
RFC	*Request For Comments*

RPE	*Regular Pulse Excitation*
RTB	Red de Telefonía Básica
RTCP	*Real-time Transport Control Protocol*
RTP	*Real-time Transport Protocol*
RTSP	*Real Time Streaming Protocol*
SCCP	*Skinny Client Control Protocol*
SDP	*Session Description Protocol*
SIP	*Session Initiation Protocol*
SNMP	*Simple Network Management Protocol*
TCP	*Transport Control Protocol*
TTS	*Text To Speech*
UAC	*User Agent Client*
UAS	*User Agent Server*
UDP	*User Datagram Protocol*
UIT	Unión Internacional de las Telecomunicaciones
VoIP	*Voice over IP*
YATE	*Yet Another Telephony Engine*

1. Fundamentos de la Tesis

1.1. Introducción

Se vive en una era en la cual se necesita estar comunicado. Internet, gracias al auge de la pila de protocolos TCP/IP, ha traído grandes avances y muchas posibilidades de servicios y aplicaciones que pueden usar esta red. Sin embargo, se presenta un problema: Internet en estos momentos funciona en modo *best-effort*, lo que no permite dar calidad de servicio a las aplicaciones de tiempo real, como VoIP.

La telefonía IP, por otro lado, es una tecnología emergente en el mundo de las Telecomunicaciones, y básicamente consiste en brindar los mismos servicios que la telefonía tradicional, pero usando como base la pila de protocolos TCP/IP. Esto proporciona una gran ventaja, al darle mayor uso a la infraestructura ya establecida de datos en un área local, pero también grandes retos cuando se quiera implementar este servicio en Internet, pues no se cuenta con calidad garantizada.

Frente al carácter comercial de Internet, una nueva red educativa y de investigación ha surgido, se trata de la unión de varias redes académicas del mundo. En el caso de Perú, esta red es la RAAP (Red Académica Peruana), que a su vez está conectada a otra red regional denominada CLARA (Cooperación Latino Americana de Redes Avanzadas). CLARA por su parte se conecta a las redes de otras regiones como Abeline (más conocida como Internet2) y GÉANT (red europea), y estas últimas a la red regional APAN (*Asia Pacific Advanced Network*) de Asia, formándose así la red académica mundial.

Esta red académica avanzada mundial tiene una tasa de bits máxima de datos muy superior (cientos de Mbps) a la actual Internet porque utiliza tecnologías emergentes como MPLS, lo que le permite ofrecer mejor calidad de servicio.

La meta principal del presente trabajo es la combinación de estas dos tecnologías, es decir, implementar servicios de telefonía IP en una red avanzada, concretamente en la red peruana RAAP.

1.2. Objetivos

La descripción mostrada líneas arriba permite plantear los siguientes objetivos para el desarrollo de la tesis:

- **Comparar los diversos protocolos de señalización y hardware necesarios para la implementación de una red con telefonía IP:** Para esto se tendrá una red de telefonía IP con clientes (*hard phones* o *soft phones*) que soporten IAX2 (*Inter-Asterisk Exchange* versión 2) y clientes que soporten SIP (*Session Initiation Protocol*). También se hará uso de teléfonos analógicos e IP. Luego de ambas comparaciones, se elaborará una recomendación a la RAAP sobre las tecnologías a usarse.

- **Implementar una red piloto de telefonía IP usando software libre en la RAAP**: La red piloto conectará por lo menos dos puntos de la topología de la RAAP. Estos puntos inicialmente serían la PUCP e INICTEL, pudiendo extenderse a las demás instituciones que forman parte de la RAAP. Se contará con un servidor de respaldo que tendrá la misma configuración que el servidor principal.

1.3. Estructura de la Tesis

Al inicio de la tesis se muestran las listas de figuras, tablas y acrónimos (Glosario) usados a lo largo del desarrollo de la presente tesis. La tesis contiene en total 6 capítulos, incluido el capítulo de Fundamentos. Cada capítulo se detalla a continuación:

El capítulo 1 explica los fundamentos de la tesis y se describen los objetivos.

El capítulo 2 es el marco teórico y presenta los diversos conceptos necesarios para el correcto entendimiento de la tesis.

El capítulo 3 analiza mediante implementaciones aisladas las tecnologías descritas en el capítulo anterior.

Los capítulos 4 y 5 muestran la implementación de los servidores; y las pruebas de desempeño a las que fueron sometidos y sus resultados; respectivamente.

Por último, el capítulo 6 da las conclusiones del trabajo y elabora una recomendación formal a la RAAP sobre las tecnologías a ser usadas. Adicionalmente se plantean trabajos futuros alrededor del tema desarrollado.

Al final se muestran las referencias usadas por el autor para la elaboración del presente trabajo de tesis, así como también la tabla de anexos usados.

2. Marco Teórico

2.1. Antecedentes

Actualmente existen diversas empresas que ofrecen soluciones propietarias de servicios de telefonía IP, entre las cuales se encuentran Cisco con su *Call Manager*, Avaya con *MultiVantage*, Alcatel, Mitel, etc. Estas compañías normalmente trabajan con estándares y protocolos propietarios, lo que dificulta su interacción con soluciones de otros fabricantes.

En cuanto a soluciones usando protocolos abiertos, existen varias implementaciones, entre las cuales destacan OpenPBX, PBX4Linux, YATE, FreeSwitch y Asterisk, siendo la predominante esta última.

Desarrollada por Mark Spencer de la empresa Digium, Asterisk implementa todas las funcionalidades de una centralita telefónica (PBX) usando estándares abiertos, por lo que su interoperabilidad está asegurada.

Amplios proyectos de Telefonía IP con Asterisk se han desarrollado, y varios de ellos involucran la implementación de teléfonos IP remotos.

Igualmente, otros tantos involucran alta disponibilidad en el servicio, de manera que la probabilidad de falla del sistema sea cercana al 0%. El presente trabajo de tesis propone una nueva arquitectura de Redes de Telefonía IP, combinando la alta disponibilidad con la implementación de extensiones remotas.

Es de particular atención el caso de las extensiones distantes, pues se requiere una conexión a Internet de banda ancha con QoS (calidad de servicio) para que haya una calidad de voz aceptable, lo que no se garantiza con la tecnología actual de Internet.

2.2. Arquitectura de protocolos de VoIP

La figura 2.1 muestra la estructura de los protocolos usados en VoIP. Se puede diferenciar entre los protocolos de señalización (H.323, SIP) y los protocolos de transporte (RTCP, RTP, RTSP) [MOR2002].

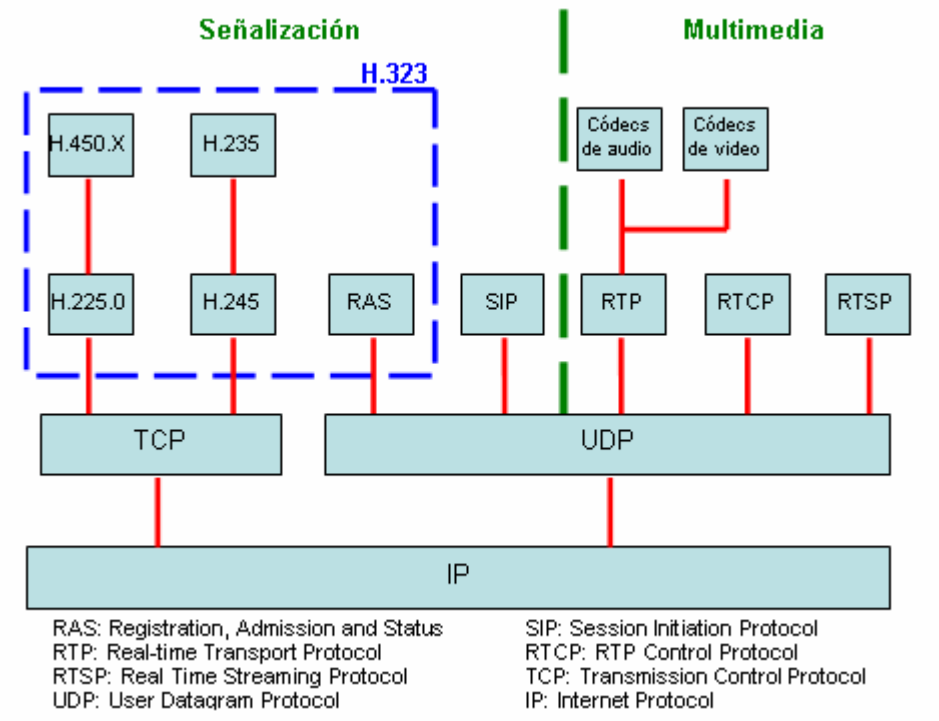

FIGURA 2.1. ESTRUCTURA DE PROTOCOLOS DE VOIP

Más adelante se explicará estas dos clases de protocolos poniendo énfasis en sus diferencias y sus características más saltantes. Adicionalmente, se compararán los diversos *codecs* usados en los protocolos de transporte. En la presente tesis se usará el término *codec* como abreviatura de COdificador/DECodificador de señales de voz, es decir, convierte la señal de voz en un flujo de datos para que pueda viajar por algún medio de transporte. En esta tesis, el medio de transporte es la red IP de la RAAP.

2.2.1. Protocolos de señalización

De acuerdo a la UIT en su recomendación H.323 [UIT2003], el protocolo de señalización se encarga de los mensajes y procedimientos utilizados para establecer una comunicación, pedir cambios de tasa de bits de la llamada, obtener el estado de los puntos extremos y desconectar la llamada.

2.2.1.1. H.323

H.323 es un estándar que norma todos los procedimientos para lograr Sistemas Audiovisuales y Multimedios, por lo que engloba varios protocolos y estándares. Uno de estos procedimientos es la señalización de la llamada.

H.323 propone dos tipos de señalización [PAC2006]:

- Señalización de control de llamada (H.225.0): Este protocolo tiene dos funcionalidades. Si existe un *gatekeeper* en la red, define como un terminal se registra con él. Este proceso se denomina RAS (*Registration, Admission and Status*) y usa un canal separado (canal RAS). Si no existiese un *gatekeeper*, define la forma como dos terminales pueden establecer o

terminar llamadas entre sí (Señalización de Llamada). En este último caso se basa en la recomendación Q.931[1].

- Señalización de control de canal (H.245): Una vez que se ha establecido la conexión entre dos terminales usando H.225, se usa el protocolo H.245 para establecer los canales lógicos a través de los cuales se transmite la media. Para ello define el intercambio de capacidades (tasa de bits máxima, *codecs*, etc.) de los terminales presentes en la comunicación.

Se usa RAS siempre y cuando un *Gatekeeper* esté presente en la red. El *Gatekeeper* es un componente opcional cuya función principal es el control de admisión. Es un intermediario entre los puntos terminales que permite el establecimiento de llamadas entre estos. También puede enrutar la señalización hacia otro dispositivo para implementar funciones como desvío de llamadas.

Una llamada H.323 se caracteriza por las siguientes fases de señalización [MAR2006]:

- **Establecimiento de la comunicación.** Primero se tiene que registrar y solicitar admisión al *Gatekeeper*, para lo cual se usan los mensajes RAS. Luego, el usuario que desea establecer la comunicación envía un mensaje de SETUP, el llamado contesta con un mensaje de *CallProceeding*. Para poder seguir con el proceso, este terminal también debe solicitar admisión al *GateKeeper* con los mensajes RAS y, una vez admitido, envía el *Alerting* indicando el inicio del establecimiento de la comunicación. Este mensaje *Alerting* es similar al *Ring Back Tone* de las redes telefónicas actuales. Cuando el usuario descuelga el teléfono, se envía un mensaje de *Connect*.

[1] Esta norma establece el procedimiento de establecimiento de llamada en RDSI

- **Señalización de Control.** En esta fase se abre una negociación mediante el protocolo H.245 (control de canal). El intercambio de los mensajes (petición y respuesta) entre los dos terminales establece quién será maestro y quién esclavo, así como también sus capacidades y los *codecs* de audio y video soportados (Mensajes TCS, *Terminal Capability Set*). Como punto final de esta negociación se abre el canal de comunicación (direcciones IP, puerto) (Mensajes OLC, *Open Logical Channel*).

- **Audio:** los terminales inician la comunicación mediante el protocolo RTP/RTCP.

- **Desconexión.** Por ultimo, cualquiera de los participantes activos en la comunicación puede iniciar el proceso de finalización de llamada mediante los mensajes *Close Logical Channel* (CLC) y *End Session Command* (ESC). Una vez hecho esto, ambos terminales tienen que informarle al *Gatekeeper* sobre el fin de la comunicación. Para ello se usan los mensajes RAS DRQ (*Disengage Request*) y DCF (*Disengage Confirm*).

Las fases de una llamada se ilustran en la figura 2.2 con detalle:

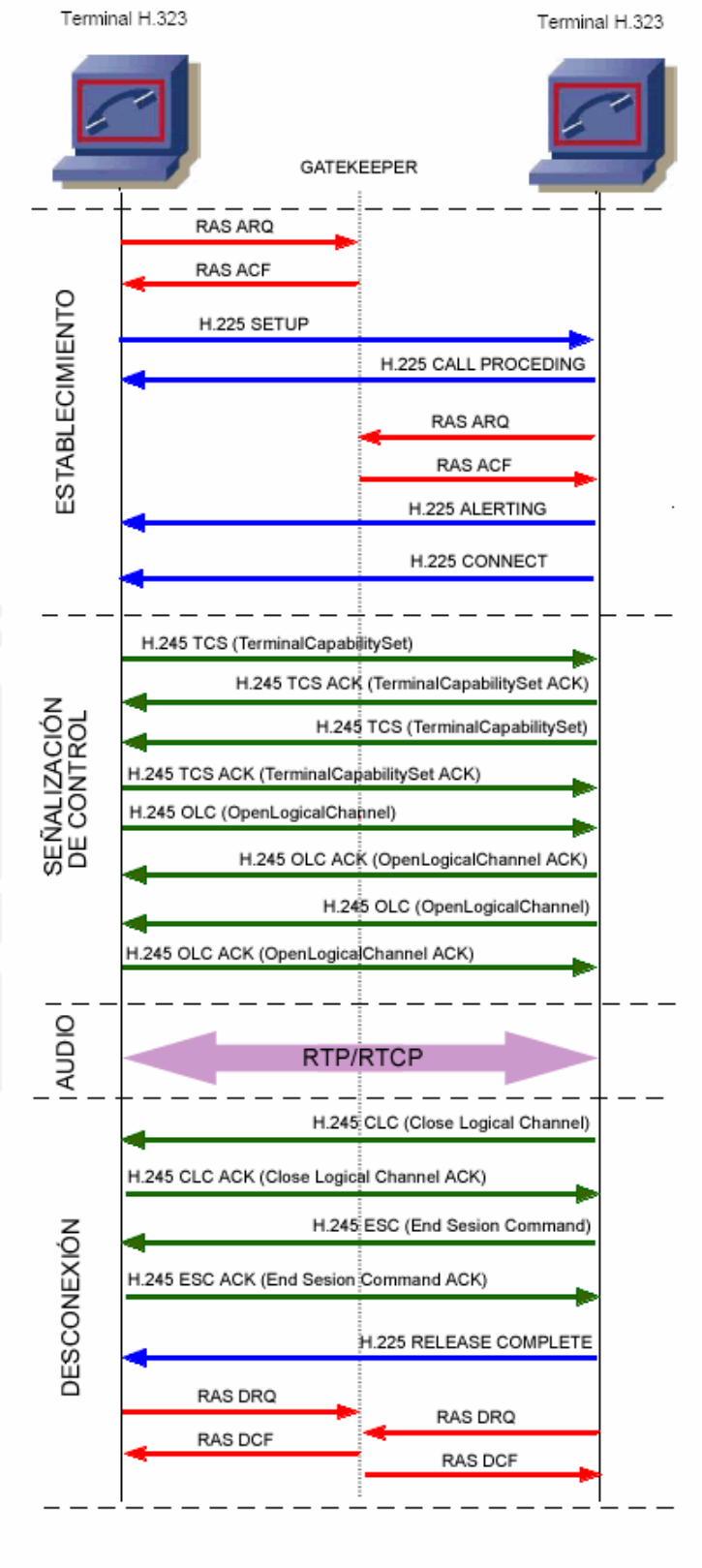

FIGURA 2.2. FASES DE UNA LLAMADA H.323
Fuente: [MAR2006]

2.2.1.2. SIP (Session Initiation Protocol)

A diferencia de H.323, SIP tiene su origen en la comunidad IP, específicamente en la IETF (*Internet Engineering Task Force*); y no en la industria de las Telecomunicaciones (UIT). Este estándar está definido en [RFC2543] y luego con aclaraciones en [RFC3261]. Se tomará esta última RFC como base para el estudio.

SIP es similar al HTTP en muchos sentidos, incluso tiene algunos mensajes de error en común, como el "404 no encontrado" (*404 not found*) y el "403 servidor ocupado" (*403 Server Busy*) [MAR2006] [MOR2002].

Los componentes presentes en SIP son [MAR2006] [MOR2002]:

1. Agentes de Usuario (*User Agent*, UA): Existen dos tipos de agentes de usuario, los cuales están presentes siempre, y permiten la comunicación cliente-servidor:
    a. Agente de usuario cliente (UAC): El UAC genera peticiones SIP y recibe respuestas.
    b. Agente de usuario servidor (UAS): El UAS responde a las peticiones SIP.
2. Servidores SIP: Existen tres clases lógicas de servidores. Un servidor puede tener una o más de estas clases. Estas clases son las siguientes:
    a. Servidor de Redirección (*Redirect Server*): Reencamina las peticiones que recibe hacia el próximo servidor.
    b. Servidor Proxy (*Proxy Server*): Corren un programa intermediario que actúa tanto de servidor como de cliente para poder establecer llamadas entre los usuarios.

c. Servidor de Registro (*Registrar Server*): Hace la correspondencia entre direcciones SIP y direcciones IP. Este servidor solo acepta mensajes REGISTER, lo que hace fácil la localización de los usuarios, pues el usuario donde se encuentre siempre tiene que registrarse en el servidor.

Se define dos tipos de mensajes SIP: Peticiones y Respuestas [RFC3261].

1. Peticiones SIP. Se definen 6 métodos básicos:

   a. INVITE: Permite invitar un usuario a participar en una sesión o para modificar parámetros de una sesión ya existente.

   b. ACK: Confirma el establecimiento de la sesión.

   c. OPTION: Solicita información de algún servidor en particular.

   d. BYE: Indica término de una sesión.

   e. CANCEL: Cancela una petición pendiente.

   f. REGISTER: Registra al Agente de Usuario.

2. Respuestas SIP: Existen también mensajes SIP como respuesta a las peticiones. Existen 6 tipos de respuestas, que se diferencian por el primer dígito de su código. Estas son:

   a. 1xx: Mensajes provisionales.

   b. 2xx: Respuestas de éxito.

   c. 3xx: Respuestas de redirección.

   d. 4xx: Respuestas de fallas de método.

e. 5xx: Respuestas de fallas de servidor.

f. 6xx: Respuestas de fallas globales.

Algunos de estos mensajes se aprecian en el ejemplo de comunicación ilustrado en la figura 2.3:

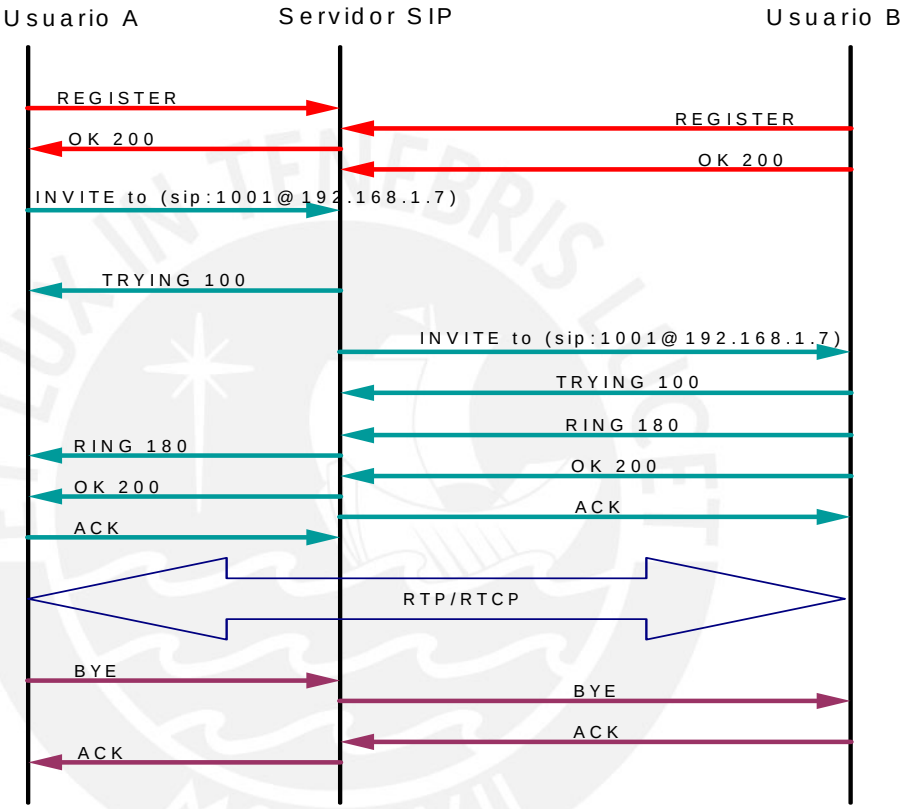

FIGURA 2.3. INTERCAMBIO DE MENSAJES EN SIP
Fuente: [MAR2006]

Las dos primeras transacciones tienen que ver con el registro de usuarios. El punto medio es el servidor que en esta etapa actúa como servidor de registro.

La siguiente transacción establece el inicio de sesión. El Usuario A (llamante) le manda un INVITE al Usuario B (llamado) a través del servidor, que redirecciona la llamada a este último. La sesión se establece cuando ambos puntos mandan la confirmación.

Cuando la sesión se ha establecido, entra a funcionar el protocolo de transporte (RTP, *Real-time Transport Protocol*), que es el encargado del transporte de la voz.

Cuando alguien quiere terminar la comunicación, manda la petición BYE que el servidor lo redirecciona al otro punto. Luego, este último envía la confirmación, terminando así la sesión. Cualquiera de los participantes puede terminar la conversación en cualquier momento.

2.2.1.3. Diferencia entre SIP y H.323

La principal diferencia es la velocidad: SIP hace en una sola transacción lo que H.323 hace en varios intercambios de mensajes. Adicionalmente, SIP usa UDP mientras que H.323 debe usar necesariamente TCP para la señalización (H.225 y H.245), lo que origina que una llamada SIP sea atendida más rápido [HER1999].

Otra diferencia importante es que H.323 define canales lógicos antes de enviar los datos, mientras que una unidad SIP simplemente publicita los *codecs* que soporta, más no define canales, lo que puede generar saturación de tráfico en casos de muchos usuarios, pues no se separa la tasa de bits necesaria para la comunicación [MAR2006].

2.2.1.4. IAX2 (Inter Asterisk Exchange)

Es el protocolo usado por Asterisk. La versión 1 de este protocolo ha caído en desuso, en favor de la versión 2 (IAX2). Al momento de redactar esta tesis, este protocolo se encontraba en borrador para ser un RFC [SPE2006], por lo que no hay mucha información disponible.

El objetivo con el que se creó este protocolo fue minimizar la tasa de bits requerida en las comunicaciones VoIP y tener un soporte nativo para traspasar dispositivos de NAT (*Network Address Translation*). En otras palabras, provee soluciones a los problemas dados en SIP y H.323. Fue creado por Mark Spencer, quien también participó en la codificación de Asterisk.

IAX2 usa un único puerto UDP (4569) para transmitir tanto señalización como datos. El tráfico de voz es transmitido en banda (*in-band*), es decir, los datos de voz van encapsulados en el protocolo; SIP, en cambio, se basa del protocolo RTP para la transmisión de los datos (su transmisión es *out-band*). Esto le permite al protocolo IAX2 prácticamente transportar cualquier tipo de dato.

Otra característica de IAX2 es que soporta *Trunking*; es decir, un solo enlace puede enviar datos y señalización de varios canales. Cuando se hace *Trunking*, un solo datagrama IP puede contener información de varias llamadas sin crear latencia adicional. Esto genera una disminución de la tasa de bits y del retraso de los paquetes debido a que ahorra enviar varias veces la cabecera IP.

Todas estas características del IAX2 se deben a que en su diseño se basaron en muchos estándares de señalización y de transmisión de datos, quedándose solo con lo mejor de cada uno. Algunos protocolos tomados como base para el IAX2 son: SIP, MGCP y RTP (Real-time Transfer Protocol).

2.2.1.5. Otros protocolos:

2.2.1.5.1. MGCP (Media Gateway Control Protocol)

Este protocolo está basado en un modelo cliente/servidor, mientras que SIP y H.323 están basados en un modelo *peer-to-*

peer. Este estándar está descrito en [RFC2705], donde se menciona que “este protocolo está diseñado para usarse en un sistema distribuido que se ve desde afuera como un solo *gateway* VoIP”.

MGCP al igual que SIP usa el Protocolo de Descripción de Sesión (SDP) para describir y negociar capacidades de media. Su funcionalidad es similar a la capacidad H.245 de H.323 [RAM2005].

2.2.1.5.2. SCCP (Skinny Client Control Protocol)

Protocolo propietario de Cisco, se basa en un modelo cliente/servidor en el cual toda la inteligencia se deja en manos del servidor (*Call Manager*). Los clientes son los teléfonos IP, que no necesitan mucha memoria ni procesamiento [RAM2005]. El servidor es el que aprende las capacidades de los clientes, controla el establecimiento de la llamada, envía señales de notificación, reacciona a señales del cliente (por ejemplo cuando se presiona el botón de directorio). El servidor usa SCCP para comunicarse con los clientes, y si la llamada sale por un *gateway*, usa H.323, MGCP o SIP.

2.2.2. Protocolos de Transporte

2.2.2.1. RTP (Real-time Transport Protocol)

Este protocolo define un formato de paquete para llevar audio y video a través de Internet. Está descrito en [RFC3550]. Este protocolo no usa un puerto UDP determinado, la única regla que sigue es que las comunicaciones UDP se hacen vía un puerto impar y el siguiente puerto par sirve para el protocolo de Control RTP (RTCP) [BAN2006].

La inicialización de la llamada normalmente se hace por el protocolo SIP o H.323.

El hecho de que RTP use un rango dinámico de puertos hace difícil su paso por dispositivos NAT y *firewalls*, por lo que se necesita usar un servidor STUN (*Simple Traversal of UDP over NAT*, RFC3489). STUN es un protocolo de red que permite a los clientes que estén detrás de un NAT saber su dirección IP pública, el tipo de NAT en el que se encuentran y el puerto público asociado a un puerto particular local por el NAT correspondiente. Esta información se usa para iniciar comunicaciones UDP entre dos *hosts* que están detrás de dispositivos de NAT.

Las aplicaciones que usan RTP son menos sensibles a la pérdida de paquetes, pero son típicamente muy sensibles a retardos, por lo que se usa UDP para esas aplicaciones.

Por otro lado, RTP no proporciona calidad de servicio, pero este problema se resuelve usando otros mecanismos, como el marcado de paquetes o independientemente en cada nodo de la red.

2.2.2.2. RTCP (Real-time Transport Control Protocol)

El protocolo de control RTP se basa en la transmisión de paquetes de control fuera de banda a todos los nodos participantes en la sesión. Tiene 3 funciones principales [BAN2006]:

- Provee realimentación en la calidad de la data.
- Utiliza nombres canónicos (CNAME) para identificar a cada usuario durante una sesión.
- Como cada participante envía sus tramas de control a los demás, cada usuario sabe el número total de participantes. Este número se usa para calcular la tasa a la cual se van a enviar los paquetes. Más usuarios en una sesión significan

que una fuente individual podrá enviar paquetes a una menor tasa de bits.

2.2.3. Codecs

Codec viene de Codificador-Decodificador. Describe una implementación basada en software o hardware para la transmisión correcta de un flujo de datos. Se estudiará solamente los *codecs* de voz.

2.2.3.1. UIT G.711

G.711 tiene una tasa de transmisión alta (64 kbps). Desarrollado por la UIT, es el *codec* nativo de redes digitales modernas de teléfonos.

Formalmente estandarizado en 1988, este *codec*, también llamado PCM, tiene un tasa de muestreo de 8000 muestras por segundo, lo que permite un ancho de banda total para la voz de 4000 Hz. Cada muestra se codifica en 8 bits, luego la tasa de transmisión total es de 64 kbps [WOO1998].

Existen dos versiones de este *codec*: Ley-A (A-law) y Ley-μ (μ-law). La segunda se usa en Estados Unidos y Japón mientras que la primera se usa en el resto del mundo, incluida Latinoamérica. La diferencia entre ellas es la forma como la señal es muestreada. Las ecuaciones de muestreo son las siguientes [HUA2005] y se grafican en la figura 2.4:

- Ley-A:
  - o $y = \frac{Ax}{1+\ln A}$ para $x \le \frac{1}{A}$ (1)
  - o $y = \frac{1+\ln Ax}{1+\ln A}$ para $\frac{1}{A} \le x \le 1$ (2)

- Ley-μ:

  o $$y = \frac{\ln(1+\mu x)}{\ln(1+\mu)} \quad (3)$$

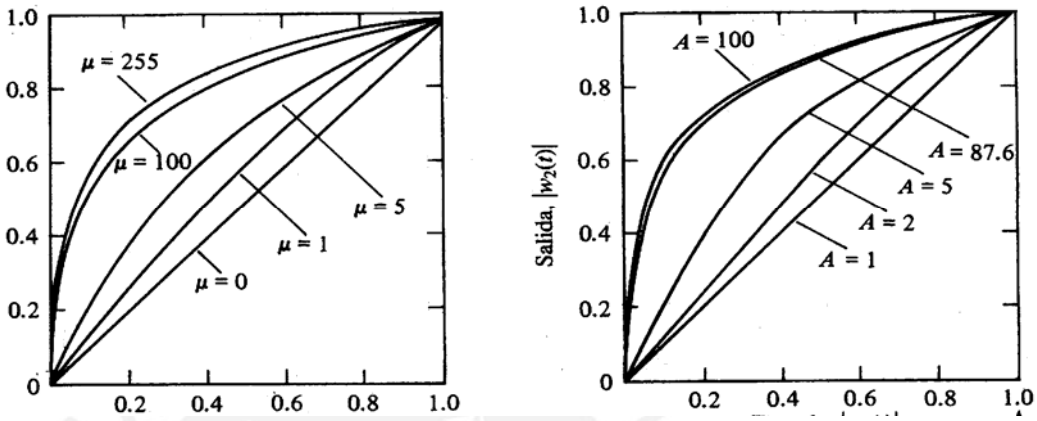

FIGURA 2.4. COMPARACIÓN LEY-μ VS. LEY-A
Fuente: [HUA2005]

Los valores de μ y de A están estandarizados por la UIT y son μ=255 para el caso de la ley-μ y de A=100 para el caso de la ley-A. La forma logarítmica refuerza las muestras más pequeñas de la entrada con el fin de protegerlas del ruido.

El uso de G.711 para VoIP ofrece la mejor calidad (no realiza compresión en la codificación), por lo que suena igual que un teléfono analógico o RDSI. Esto se comprueba con la medida del MOS. El MOS (*Mean Opinion Score*) es una medida cualitativa de la calidad de la voz. Un MOS de 5 indica una comunicación con calidad excelente mientras que un MOS de 0 indica una calidad pésima. G.711 tiene el MOS más alto de todos los *codecs* en condiciones ideales (sin pérdida de paquetes), con un MOS de 4.1.

También presenta el menor retardo debido a que no hay un uso extensivo del CPU (no hay compresión de datos).

El inconveniente principal es que necesita mayor tasa de bits que otros *codecs*, aproximadamente 80 kbps incluyendo toda la cabecera

TCP/IP. Sin embargo, con un acceso de alta velocidad, esto no debería ser mayor problema.

Este *codec* es soportado por la mayoría de compañías de VoIP, tales como proveedores de servicio y fabricantes de equipos.

2.2.3.2. UIT G.729

Este *codec* comprime la señal en períodos de 10 milisegundos. No puede transportar tonos como DTMF o fax.

G.729 se usa principalmente en aplicaciones VoIP por su poca tasa de bits (8 kbps). Existen extensiones de la norma que permiten tasas de 6.4 y 11.8 kbps para peor y mejor calidad de voz, respectivamente. Idealmente presenta un MOS de 3.8.

El uso de aplicaciones usando este *codec* requiere una licencia. Sin embargo existen implementaciones gratuitas para uso no comercial.

2.2.3.3. GSM (RPE-LTP)

Este *codec* se llama oficialmente RPE-LTP (*Regular Pulse Excitation – Long Term Prediction*) pero se conoce mundialmente como GSM debido a que es el *codec* usado en el estándar GSM de comunicaciones móviles.

Tiene una tasa de bits de 13 kbps con un MOS ideal de 3.6 y realiza la codificación generando coeficientes representativos de un intervalo de tiempo determinado. Este intervalo normalmente es de 20 milisegundos de voz [WOO1998]. La descripción de todo el proceso de codificación en forma de diagrama de bloques se encuentra en [VAL2001].

2.2.3.4. iLBC

iLBC (*Internet Low Bit-Rate Codec*) es un *codec* de voz de banda estrecha libre (se puede usar sin el pago de regalías). [RFC3951] describe todo el proceso de codificación y decodificación.

La señal de voz es muestreada a 8 kHz., y el algoritmo usa una codificación predictiva lineal (LPC). Soporta dos tamaños de cuadro: 20 ms a 15.2 kbps y 30 ms a 13.33 kbps. La figura 2.5 muestra un estudio realizado por la empresa DynStat en el cual se comparan los protocolos iLBC, G.729 y G.723.1 en base a su robustez frente a la pérdida de paquetes. Para esto se midió el MOS conforme se iban perdiendo los paquetes. Al inicio de la prueba, iLBC presentó un MOS similar al G.729, y conforme se fueron perdiendo los paquetes presentó una mejor calidad.

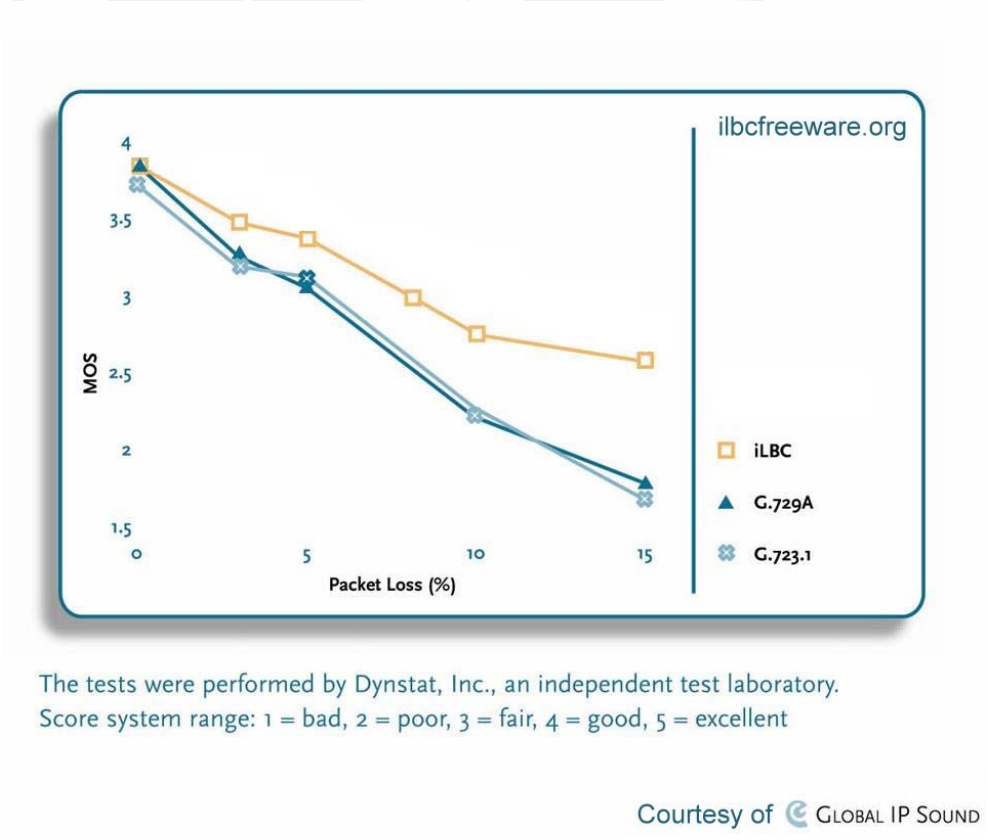

FIGURA 2.5. ROBUSTEZ FRENTE A PÉRDIDA DE PAQUETES
Fuente: [ILB2003]

2.2.3.5. Resumen

La tabla 2.1 muestra un cuadro comparativo entre los *codecs* descritos anteriormente:

TABLA 2.1. COMPARACIÓN CODECS

Nombre	*Org.*	*Descripción*	*Bit Rate (kbps)*	*Frecuencia Muestreo (kHz)*	*Tamaño cuadro (ms)*	*Obs.*	*MOS (ideal)*
G.711	UIT	Pulse Code Modulation (PCM)	64	8	Muestreada	Ley-A Ley-μ	4.1
G.729	UIT	Conjugate-structure algebraic-code-excited linear-prediction (CS-ACELP)	8	8	10	Bajo retardo (15 ms)	3.8
GSM	ETSI	Regular Pulse Excitation – Long Term Prediction (RPE-LTP)	13	8	20	Usado por GSM	3.5-3.7
iLBC		Linear Predictive Coding (LPC)	15.2 13.33	8	20 30		4.1

2.2.4. Hardware usado en los clientes

El hardware mencionado en este punto se refiere a los dispositivos usados por el usuario para comunicarse a través de la red de telefonía IP. Son básicamente dos tipos: Adaptadores analógicos y teléfonos IP propiamente dichos.

2.2.4.1. Adaptadores Analógicos:

Son dispositivos con una interfaz para conectar un teléfono analógico (*slot* para conector RJ-11) y otra interfaz para conectar a la red (*slot* para conector RJ-45). Básicamente su función es la de proveer

señalización FXO a los teléfonos, es decir, se comporta como un dispositivo FXS. Se explicará brevemente estos dos términos [WAL2005]:

FXO: *Foreign eXchange Office*, es la interfaz que se conecta a la red de Telefonía Básica (RTB, PSTN) o a una PBX y normalmente está presente en todos los teléfonos analógicos. Recibe la señalización dada por la FXS.

FXS: *Foreign eXchange Subscriber*, es la interfaz que se conecta directamente a un teléfono analógico y le brinda tono de timbrado y voltaje, entre otras cosas. En un escenario convencional (telefonía analógica), el FXS está en la central de conmutación, brindando señalización al dispositivo FXO (teléfono analógico).

Se tienen dos posibilidades para usar teléfonos analógicos en una red VoIP: Una es que el servidor de VoIP tenga una tarjeta con módulos FXS y la otra es tener en la red ciertos *gateways* que conviertan la señal analógica en datos IP. De esta forma, la PBX IP se comunica con los teléfonos analógicos a través de los *gateways* usando los protocolos de señalización mencionados anteriormente. Un ejemplo de estos *gateways* son los ATAs (*Analog Telephone Adapter*).

2.2.4.2. Teléfonos IP

Son dispositivos que soportan uno o varios protocolos de señalización. Entre las marcas más conocidas se tiene a Atcom, Cisco, Sipura (comprado por Cisco), etc. La gran mayoría soporta como mínimo el *codec* G.711, pudiendo soportar otros más. Adicionalmente pueden tener otras funcionalidades tales como supresión de silencios o conexión redundante a dos servidores. Para las diversas pruebas en la presente tesis se usará el teléfono IP

ATCOM AT-320. Este teléfono soporta IAX2, SIP, H.323 como protocolos de señalización y G.711, G.729, iLBC, GSM como protocolos de codificación (*codecs*). La hoja de datos de este teléfono así como sus manuales de usuario (SIP y IAX2) se encuentran en los anexos 1, 2 y 3; respectivamente.

En la figura 2.6 se muestra como conviven los diferentes tipos de teléfonos en una red VoIP.

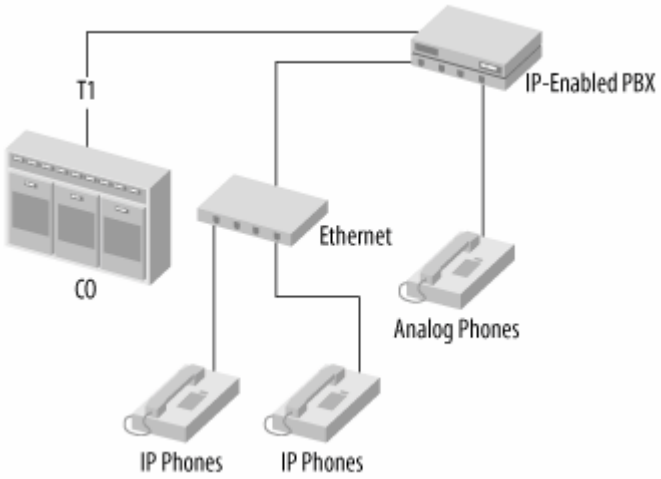

FIGURA 2.6. CONVIVENCIA DE TELÉFONOS IP Y ANALÓGICOS
Fuente: [WAL2005]

3. Análisis Previo

3.1. Ambiente de pruebas

Este capítulo tiene por finalidad hacer una comparación de las diversas tecnologías mencionadas anteriormente. Para ello se cuenta con un servidor de pruebas con Asterisk ya instalado (el proceso de instalación de los servidores está detallado en el capítulo 4). El servidor tiene las siguientes características:

- Procesador Intel Celeron 1.7 GHz
- 512 MB de RAM
- Sistema Operativo: Debian Sarge con Asterisk 1.0.9

Se comparará primero los protocolos de señalización para luego estudiar los diversos *codecs* y por último el hardware.

3.2. Protocolos de Señalización

Para el caso de los protocolos de señalización, se estudiarán básicamente dos protocolos: SIP e IAX2. No se tomará en cuenta a H.323 por ser un protocolo con muy poca documentación en Asterisk, y además requiere licencia para su uso. Para cada protocolo estudiado se verá como es el proceso de registro y establecimiento de una llamada. Para todo ello se hará uso del teléfono IP Atcom AT-320PD, el cual, como se dijo en el capítulo anterior, tiene la particularidad de que su *firmware* es intercambiable para soportar los diversos protocolos. Es decir, existe un *firmware* para SIP y otro para IAX2, los cuales se pueden cambiar de manera gráfica vía web. Para capturar los paquetes se utiliza el software Ethereal que se ejecuta en el servidor. Se evaluará cada protocolo según su comportamiento en las distintas etapas de la señalización.

3.2.1. Registro

3.2.1.1. SIP

Cada dispositivo se autentica en el servidor usando para eso el mensaje REGISTER, como lo muestra la figura 3.1:

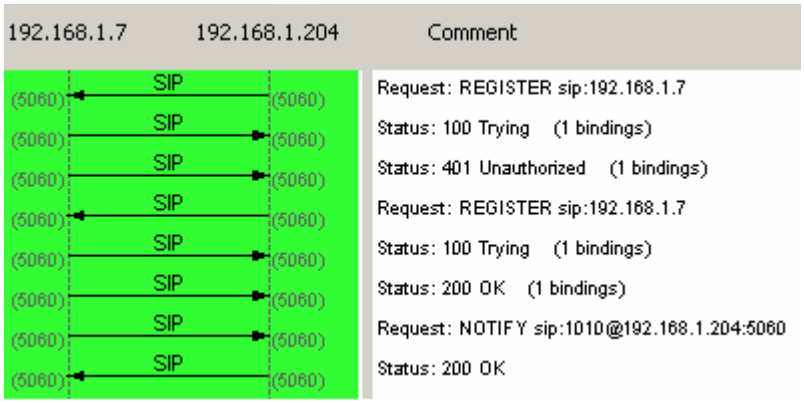

FIGURA 3.1. SIP: FLUJO DE PAQUETES REGISTRO USUARIO

La dirección IP del servidor Asterisk es 192.168.1.7 y la del teléfono IP es 192.168.1.204. El flujo de registro es el siguiente: Primero el usuario que quiere registrarse, en este caso el teléfono IP, envía un mensaje REGISTER solo con el usuario y el IP a donde se redireccionarán las llamadas a esa extensión, más no contiene la contraseña. Debido a esto, el servidor rechaza la petición con el mensaje 401 UNAUTHORIZED, pidiéndole al usuario su contraseña y dándole un método de cifrado. Una vez conocido esto, el usuario vuelve a enviar un REGISTER pero esta vez ya con la contraseña cifrada, con lo que el servidor registra al usuario. Por último, el servidor envía una trama NOTIFY, para asegurarse que todo este en perfecto funcionamiento.

Todo el proceso descrito en el párrafo anterior está bien documentado en [RFC3665], sin embargo, en este documento no se menciona la última parte, siendo ésta opcional en las implementaciones del estándar.

En total fueron intercambiados 8 paquetes dando un total de 2829 bytes. La traza de los paquetes en formato PCAP se encuentra en el anexo 4.

3.2.1.2. IAX2

Se obtuvo el flujo de paquetes ilustrado en la figura 3.2:

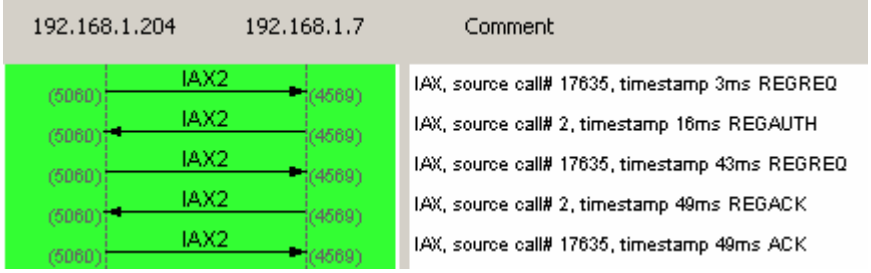

FIGURA 3.2. IAX2: FLUJO DE PAQUETES REGISTRO USUARIO

Primero el teléfono manda un mensaje REGREQ al servidor Asterisk al puerto 4569 (puerto por defecto del servicio de IAX2). Este primer mensaje al igual que el primer mensaje REGISTER de SIP solo contiene el usuario que se quiere registrar, pidiendo el método de cifrado. La central le responde con el REGAUTH indicando al usuario el método adecuado de cifrado. Luego, el usuario vuelve a mandar el mensaje REGREQ, esta vez con la contraseña ya cifrada. Por último el servidor confirma su registro y el usuario responde a esta confirmación.

En total se intercambiaron 5 paquetes dando un total de 404 bytes, 7 veces menos que lo que se obtuvo con SIP.

3.2.2. Establecimiento de Llamada

3.2.2.1. SIP

Para comprobar el comportamiento en una llamada cualquiera se usa un segundo equipo, que tendrá el *softphone* X-lite[2] para conectarse al servidor Asterisk. Este equipo tiene la IP 192.168.1.50. El flujo de establecimiento de llamada se ilustra en la figura 3.3:

[2] Se puede descargar gratuitamente de http://www.xten.com/index.php?menu=download

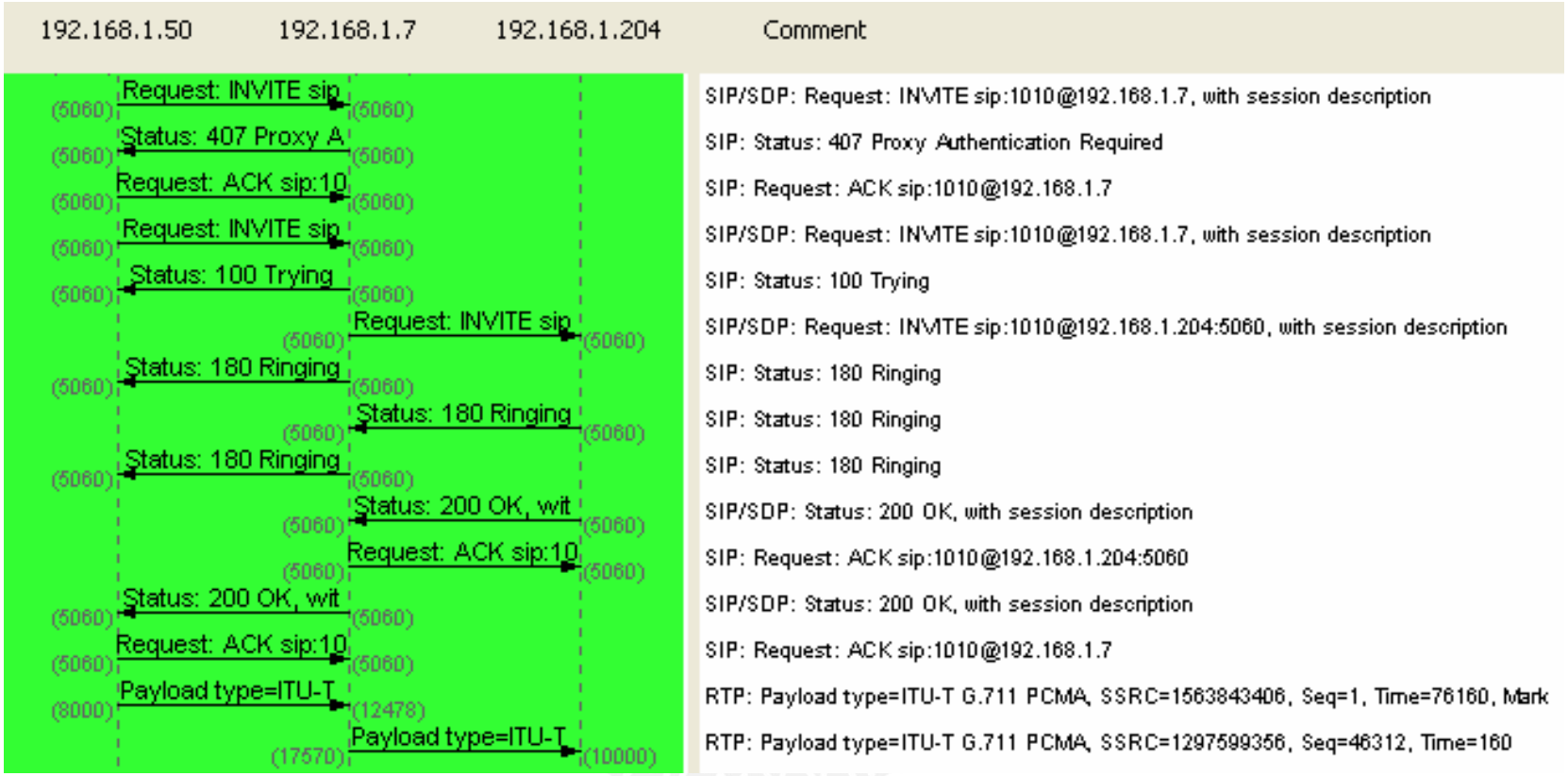

FIGURA 3.3. SIP: FLUJO DE PAQUETES ESTABLECIMIENTO LLAMADA

En este caso pasa algo similar al registro, pues primero se manda el mensaje INVITE al número SIP 1010@192.168.1.7. Como esa extensión pertenece a una dirección IP externa, la central le pide al llamante que se registre primero como cliente, siguiendo el mismo procedimiento que el registro. Cuando vuelve a enviar el INVITE ya con las credenciales, el servidor lo redirecciona a su destino final: el teléfono IP. Durante todo este proceso se intercambian los *codecs* soportados y se elije uno para ser usado en la transmisión de voz, en este caso G.711.

En total, en todo el proceso de registro se intercambiaron 14 paquetes dando un total de 7378 bytes, sin contar los paquetes RTP de la comunicación de voz propiamente dicha.

3.2.2.2. IAX2

Se capturan los paquetes intercambiados en el establecimiento de una llamada. A diferencia de SIP, se usa el *softphone* Idefisk[3], que soporta el protocolo IAX2 y está instalado en el mismo equipo del caso anterior (dirección IP 192.168.1.50). El flujo de establecimiento de llamada usando IAX2 se ilustra en la figura 3.4:

[3] Se puede descargar gratuitamente de http://www.asteriskguru.com/idefisk/free/

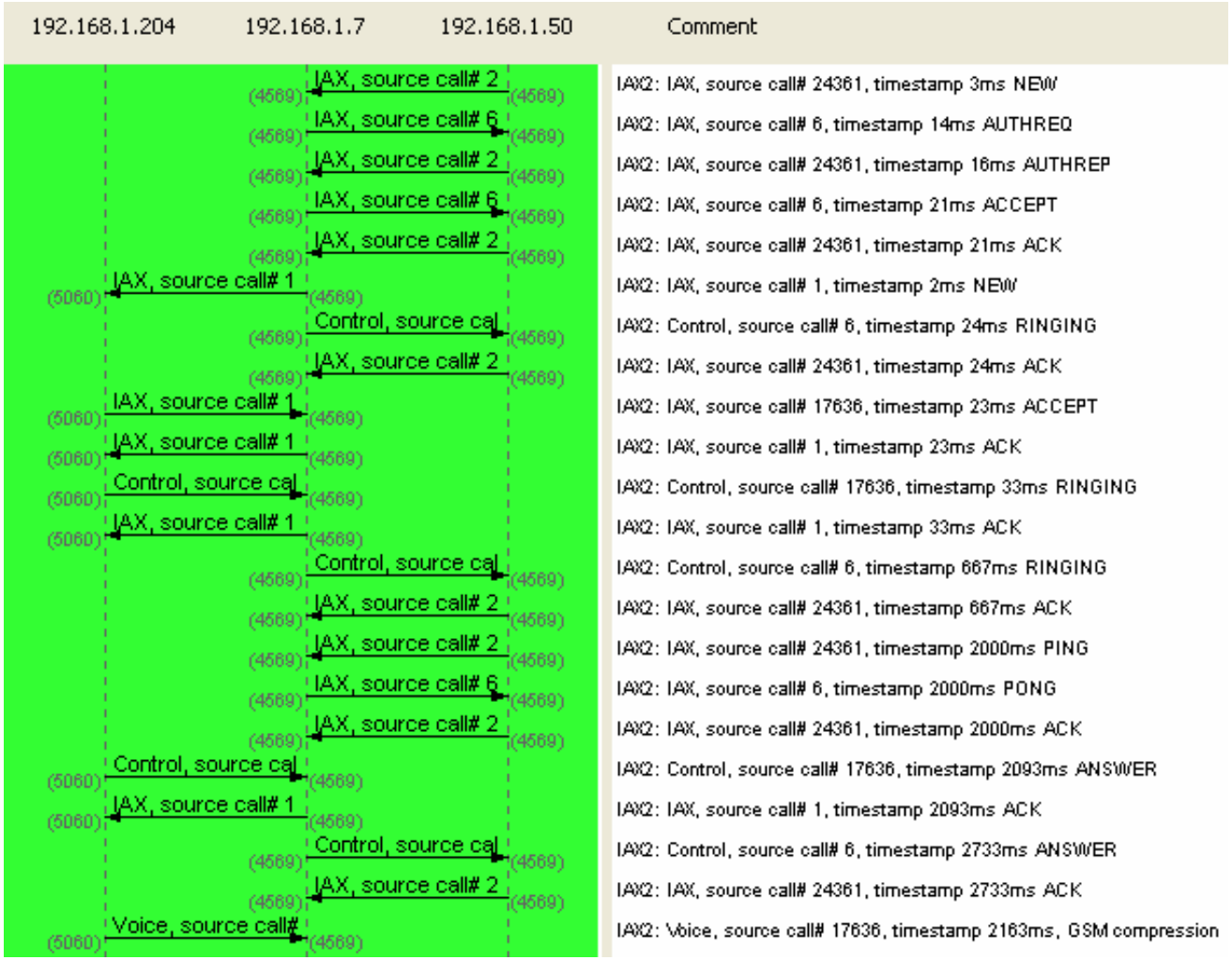

FIGURA 3.4. IAX2: FLUJO DE PAQUETES ESTABLECIMIENTO LLAMADA

Se aprecia que primero el *softphone* manda el mensaje de llamada nueva a la central con el número de extensión del teléfono IP como destino (las tramas completas se encuentran en el anexo 4). Luego, al igual que para el registro, la central desafía al *softphone* pidiéndole su contraseña cifrada, dándole para esto la llave de cifrado a usarse. El *softphone* manda su clave cifrada y la central acepta la llamada. Luego la central, que está actuando como Proxy, redirecciona la llamada al teléfono IP, y le manda tanto al llamante como al llamado el mensaje RINGING, al cual responden con el mensaje ACK. Por último el teléfono IP contesta la llamada y manda el mensaje ANSWER a la central, la que a su vez la reenvía al *softphone*, originándose así la llamada. En total se intercambiaron 21 paquetes dando un total de 1366 bytes, 5 veces menos que SIP.

3.2.3. Tasa de Bits

Uno de los objetivos de esta prueba es determinar la tasa de bits consumida por cada protocolo, usando para ello dos llamadas concurrentes. De esta forma se verá qué protocolo tiene una menor tasa de bits. Para el primer caso las llamadas serán SIP y para el segundo IAX2. El *codec* a usar para esta parte es G.711 (ley-μ), para de esta forma ser más imparcial, ya que si se usaran *codecs* distintos en SIP e IAX2, la tasa de bits se vería influenciada mucho más por el *codec* usado que por el protocolo en sí.

Para poder medir la tasa de bits consumida por unidad de tiempo se hace uso de SNMP (*Simple Network Management Protocol*). SNMP es un protocolo que forma parte de la *suite* TCP/IP que permite a los administradores de red monitorear el desempeño de la misma. Esta herramienta se utilizará para monitorear la tasa de bits consumida en el servidor durante las llamadas. Para instalarlo simplemente se siguen los pasos de la tabla 3.1:

TABLA 3.1. INSTALACIÓN SNMP EN DEBIAN

```
voip:~# apt-get install snmpd
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Se instalarán los siguientes paquetes NUEVOS:
  snmpd
0 actualizados, 1 se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 0B/731kB de archivos.
Se utilizarán 856kB de espacio de disco adicional después de desempaquetar.
Seleccionando el paquete snmpd previamente no seleccionado.
(Leyendo la base de datos ...
72257 ficheros y directorios instalados actualmente.)
Desempaquetando snmpd (de .../snmpd_5.1.2-6.2_i386.deb) ...
Configurando snmpd (5.1.2-6.2) ...
Starting network management services: snmpd.
```

Luego se edita el archivo /etc/snmp/snmpd.conf y se añade la línea mostrada en la tabla 3.2.

TABLA 3.2. ARCHIVO /ETC/SNMP/SNMPD.CONF

```
com2sec  readonly  default  public
```

De esta forma se da permiso a que otras máquinas de la red puedan enviar comandos SNMP.

Para poder visualizar los gráficos obtenidos se usa el programa PRTG[4], que preguntará cada cierto tiempo al servidor su tasa de bits (mediante SNMP) y lo graficará en un eje de tiempo.

3.2.3.1. SIP

En este caso se usa la red Gizmo, la cual es una red mundial que usa el protocolo SIP como protocolo de señalización para sus usuarios. Se configura esta red como una troncal en nuestra central, con el usuario y password dados por Gizmo.

Los resultados obtenidos se muestran en la figura 3.5:

[4] Página web: http://www.paessler.com/prtg/

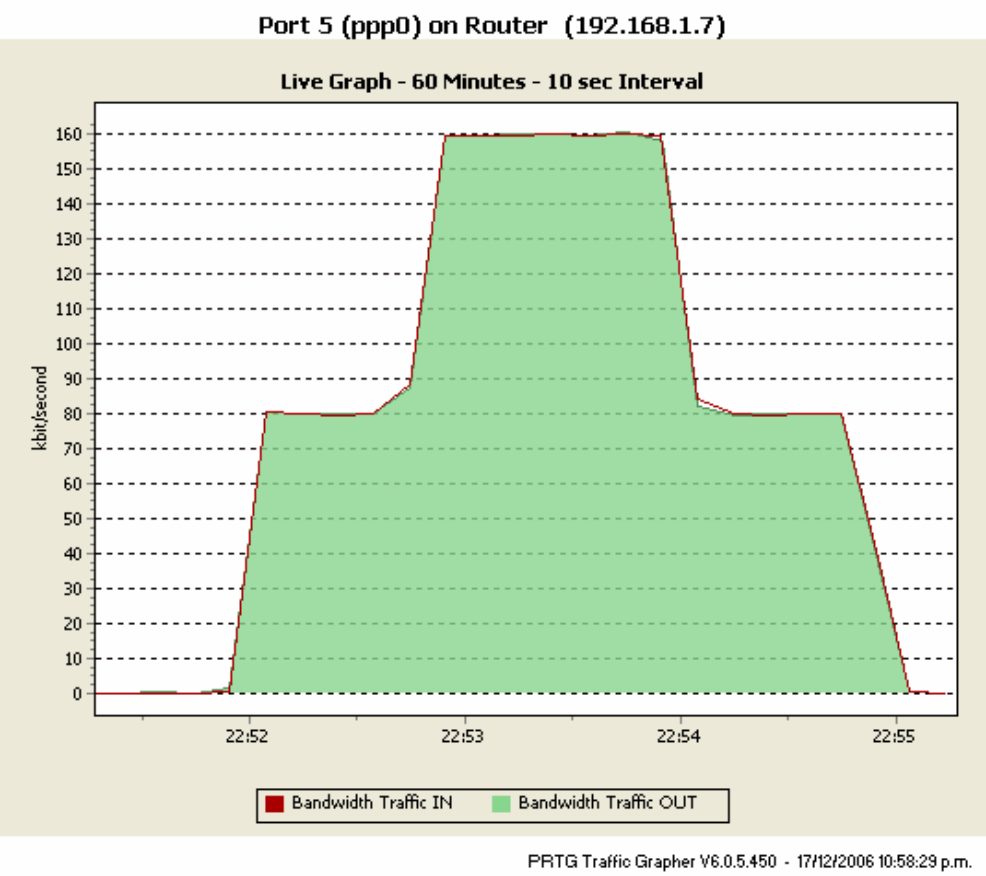

FIGURA 3.5. SIP: TASA DE BITS VS TIEMPO

Se obtuvo alrededor de 80 kbps para una llamada y aproximadamente 160 kbps para 2 llamadas. Cabe destacar que se está usando el *codec* G.711 que ocupa 64 kbps. La diferencia es la cabecera propia del protocolo SIP.

3.2.3.2. IAX2

En este caso se usa la red FWD para saber cuál es la tasa de bits consumida. FWD (*Free World Dialup*), al igual que Gizmo, es una red mundial de VoIP, y se configura como una troncal IAX2. Se obtiene el gráfico mostrado en la figura 3.6:

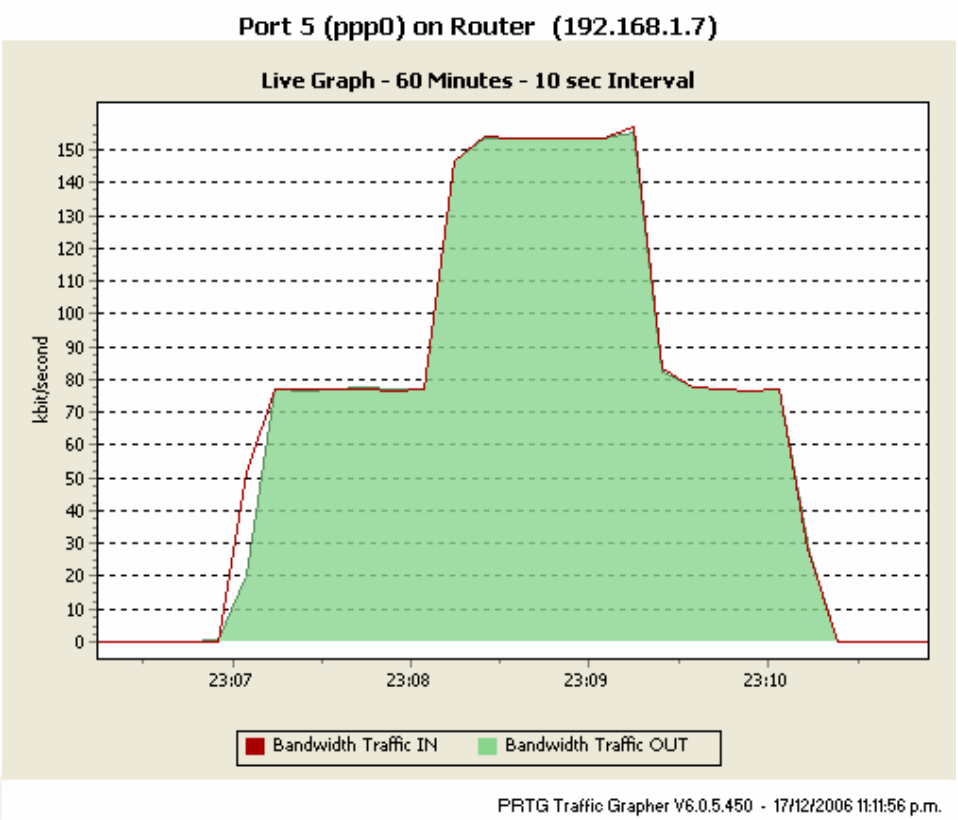

FIGURA 3.6. IAX2: TASA DE BITS VS TIEMPO

La primera llamada consume alrededor de 77 kbps (simétrico, 77 de entrada y 77 de salida), mientras que cuando la segunda llamada se inicia, el tráfico consumido asciende hasta un promedio de 152 kbps.

3.2.4. Conclusiones

Se compararon los dos protocolos principales usados en Asterisk: SIP e IAX2. En todos los aspectos comparados, IAX2 demostró ser más eficiente al transmitir la menor cantidad de paquetes y al tener el menor *overhead*. Sin embargo estas diferencias son muy pequeñas como para inclinarse definitivamente por IAX2, ya que SIP tiene a su favor que es el protocolo más extendido y soportado por las empresas de VoIP.

3.3. Codecs

Para el caso de los codificadores, se usará como base el protocolo IAX2, y sobre ese se cambiarán los distintos *codecs* que existen. Solo se compararán los *codecs* G.711 e iLBC. Se eligen estos dos *codecs* debido a que ofrecen el mejor MOS y en cuestión de consumo de CPU son totalmente opuestos (G.711 no comprime e iLBC es el que usa más CPU para comprimir).

El resultado obtenido se muestra en la figura 3.7:

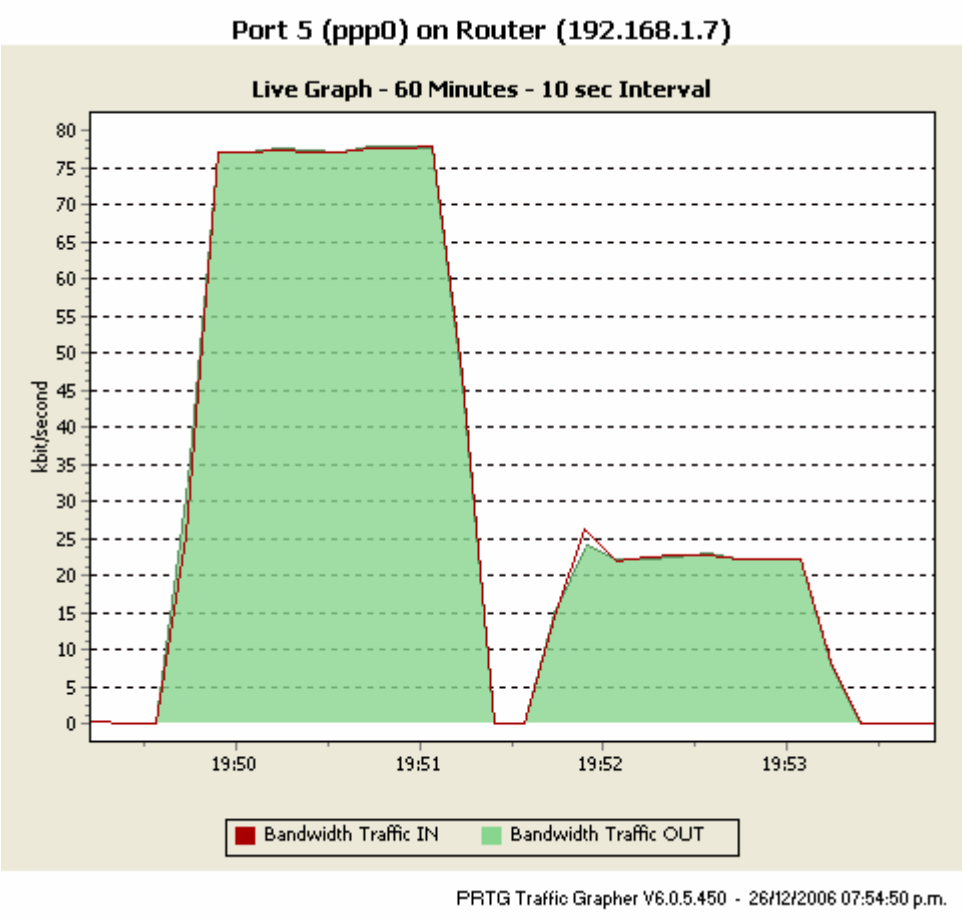

FIGURA 3.7. LLAMADAS G.711 E ILBC

3.3.1. G.711

La figura 3.7 muestra que una llamada en G.711 consume alrededor de 77 kbps. Esta codificación transmite la voz sin generar retardo alguno. Es decir, muestrea la señal de voz y va transmite. Se descubre que hay un *overhead* dado por el protocolo, pues según la teoría G.711 solo usa 64 kbps.

3.3.2. iLBC

Se obtuvo un resultado de 23 kbps, mucho menor que G.711. Esto debido a que iLBC usa compresión para la codificación de la voz. Sin embargo, se comprueba experimentalmente que la codificación iLBC genera cierto retardo ya que se espera que lleguen un grupo de muestras para poder codificar la voz y transmitirla (usualmente 20 ms). Al igual que G.711, existe *overhead* brindado por el protocolo, por lo que no sale el valor teórico, el cual es 15 kbps.

3.4. Hardware

Se pueden usar dos clases de dispositivos: teléfonos analógicos y teléfonos IP, cada uno con ventajas y desventajas particulares, que se detallan a continuación:

3.4.1. Teléfonos Analógicos

Para poder usar teléfonos analógicos se debe tener hardware especial instalado en el servidor. Se trata de módulos FXS que brindan señalización y energía a los teléfonos, tal como lo hace una central pública a un usuario convencional. Estos módulos FXS pueden estar directamente conectados a la central de conmutación (PBX) o a la red

LAN, en cuyo caso el dispositivo se denomina ATA (*Analog Telephone Adapter*). La principal diferencia entre un ATA y simplemente un módulo FXS es que el ATA aparte del módulo FXS tiene un puerto Ethernet, por lo que transforma al teléfono analógico en un teléfono IP. En cambio, el FXS simplemente hace una conmutación analógica interna para las comunicaciones. Como el ATA es casi lo mismo que un teléfono IP, no se tomará en cuenta para la comparación. Un ejemplo de ATA es el IAXy, mostrado en la figura 3.8.

FIGURA 3.8. ADAPTADOR IAXY
Fuente: [DIG2006]

El reuso de teléfonos analógicos tiene la principal ventaja que no hay latencia en la transmisión de la voz, al no pasar por un codificador, sino de frente se conmuta en la central. Las principales desventajas son su costo (alrededor de US$300 por tarjeta con 4 puertos) y el requerimiento de hardware adicional cuando se acaben los *slots* de las tarjetas. Sin embargo, el inconveniente principal es que se necesita cablear desde la central a cada teléfono analógico (no se reusa la red de datos).

3.4.2. Teléfonos IP

Existen dos tipos de teléfonos IP: *softphones* y *hardphones*. En cuanto a características y funcionalidades son idénticos, por lo que no se hará distinción entre uno u otro. La única diferencia que tienen es que mientras el *hardphone* es un dispositivo físico (teléfono tangible), el *softphone* funciona de la misma forma que cualquier programa en la computadora. Un ejemplo de *hardphone* es el teléfono IP AT-320 que se muestra en la figura 3.9.

FIGURA 3.9. TELÉFONO IP ATCOM AT-320
Fuente: [ATC2006]

La principal ventaja de un teléfono IP es la movilidad, es decir, se puede mover el equipo en cualquier punto de la red y se mantiene su mismo número de extensión, esto no es posible con los teléfonos analógicos, donde cada ranura identifica a un número. Esto conlleva un reuso de la infraestructura de datos para pasar voz, abaratando costos a largo plazo.

Sin embargo también existen desventajas. La principal es el retardo producido por el proceso de codificación – transporte – descodificación en la comunicación entre la red pública y cualquier extensión IP, el cual produce eco en los teléfonos al oírse los usuarios a sí mismos luego de cierto período de tiempo. Esto se soluciona eficazmente con canceladores de eco.

3.4.3. Conclusiones

Vistas las principales diferencias se recomienda comenzar a trabajar con teléfonos IP, debido a que en su versión de software (softphone) son gratuitos y no requiere cableado especial para ellos (se usa el cableado de la red de datos). Entre los *softphones* y los *hardphones* no hay ninguna diferencia en capacidades y funcionalidades, por lo que es indiferente el uso de uno u otro. En lo concerniente al precio, un *hardphone* cuesta alrededor de US$100. Si bien el precio es relativamente alto, se tiene la ventaja de la comodidad y de no tener que depender de una PC para poder realizar llamadas. Si solo se quiere la funcionalidad, lo ideal es trabajar con *softphones* gratuitos.

4. Implementación de los Servidores

4.1. Arquitectura

La plataforma sobre la cual funciona la red de voz sobre IP está conformada por dos servidores y sus clientes están ubicados en la RAAP; y cualquier persona con conexión a esta red puede ser también usuario del sistema de VoIP. Como se mencionó anteriormente, se tendrán dos servidores, ambos redundantes. Estos servidores están localizados en la PUCP. El diagrama de red propuesto se muestra en la figura 4.1:

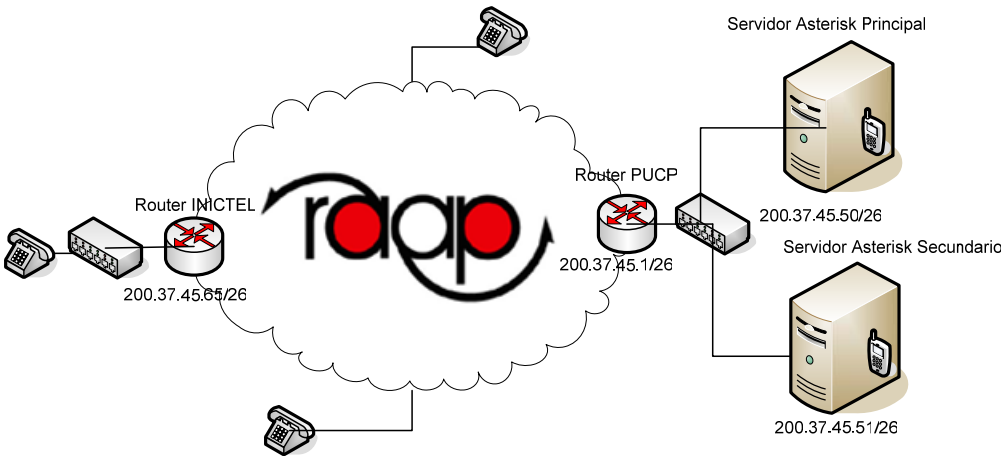

FIGURA 4.1. ARQUITECTURA DE RED

4.2. Alta Disponibilidad

Alta disponibilidad es garantizar la operatividad del servicio evitando que exista un punto único de falla. Se plantea la redundancia del hardware, de tal manera que la configuración del servicio de VoIP (Asterisk) sea idéntica y transparente al usuario final y este no note que servidor le está brindando el servicio.

Para lograr este objetivo se aplicará las recomendaciones dadas por el Proyecto de Alta Disponibilidad en Linux [ROB2006]. Uno de los componentes principales del núcleo de este proyecto es el programa HeartBeat, el cual provee detección de muerte de nodo, administración de comunicaciones y clusters en un solo proceso.

La configuración detallada de HeartBeat se encuentra en la sección 4.5.2.2.

4.3. Sistema Operativo

Se eligió el sistema Operativo GNU/Linux Ubuntu Server debido a su capacidad en el reconocimiento del hardware a ser usado, como por

ejemplo los discos SATA y el RAID (*Redundant Array of Independent/Inexpensive Disks*).

4.4. Hardware

4.4.1. Servidor Principal

El servidor principal es un servidor IBM cuyo procesador tiene las características técnicas mostradas en la tabla 4.1:

TABLA 4.1. PROCESADOR SERVIDOR PRINCIPAL

```
dquintana@gst:~$ cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 15
model           : 4
model name      : Intel(R) Xeon(TM) CPU 3.00GHz
stepping        : 3
cpu MHz         : 3000.733
cache size      : 2048 KB
fdiv_bug        : no
hlt_bug         : no
f00f_bug        : no
coma_bug        : no
fpu             : yes
fpu_exception   : yes
cpuid level     : 5
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep
mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2
ss ht tm pbe nx lm pni monitor ds_cpl cid cx16 xtpr
bogomips        : 6006.12
```

Estas características se obtienen revisando el archivo /proc/cpuinfo. Adicionalmente cuenta con 1 GB de memoria RAM. Este valor se saca del archivo /proc/meminfo.

Este servidor tiene 2 interfaces de red: 1 conectada a la red pública de Internet con la dirección IP 200.16.6.206 y otra conectada a la RAAP con dirección 200.37.45.50. La razón de tener una dirección IP de la red

pública es poder realizar las configuraciones desde cualquier lugar conectado a Internet, por ejemplo desde cualquier conexión ADSL.

Para la red pública esta máquina se llama gst y pertenece al dominio telecom.pucp.edu.pe. La RAAP, al momento de escribir esta tesis, no tiene registros DNS, por lo que este servidor se localiza en esta red académica únicamente mediante su dirección IP.

4.4.2. Servidor Secundario

El servidor secundario tiene un procesador con las características técnicas mostradas en la tabla 4.2 (tomado de /proc/cpuinfo):

TABLA 4.2. PROCESADOR SERVIDOR SECUNDARIO

```
root@git:~$ cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 15
model           : 2
model name      : Intel(R) Xeon(TM) CPU 2.80GHz
stepping        : 9
cpu MHz         : 2795.711
cache size      : 512 KB
fdiv_bug        : no
hlt_bug         : no
f00f_bug        : no
coma_bug        : no
fpu             : yes
fpu_exception   : yes
cpuid level     : 2
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep
mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2
ss ht tm pbe cid xtpr
bogomips        : 5591.20
```

Este servidor cuenta con 1 GB de memoria RAM y al igual que el servidor gst, tiene 2 interfaces de red: 1 conectada a la red pública con la dirección IP 200.16.6.205 (git.telecom.pucp.edu.pe) y otra conectada a la RAAP (dirección IP 200.37.45.51). Al igual que en el servidor principal, la dirección IP pública es usada únicamente para la configuración del sistema.

4.5. Software

El software a utilizar para la realización del presente trabajo de tesis es el siguiente:

- Asterisk: Servidor de VoIP
- AMPortal: *Asterisk Management Portal*, es la interfaz de usuario para configurar el Asterisk vía web.
- HeartBeat: Es el servicio mediante el cual se logra la alta disponibilidad en el sistema.

4.5.1. Instalación

4.5.1.1. Asterisk y AMPortal

Se usarán los repositorios de la distribución Xorcom Rapid, basada en Debian Sarge. Esta distribución provee los archivos de instalación binarios (*.deb) para ser instalados directamente en cualquier distribución basada en Debian. Se descartan los repositorios oficiales de Ubuntu porque no cuentan con una interfaz de configuración web amigable. En cambio, los repositorios de Xorcom sí cuentan con ella (AMPortal). Para instalarlo se agrega el siguiente repositorio a los repositorios de APT (*Advanced Packaging Tool*), localizados en el archivo /etc/apt/sources.list, tal como lo muestra la tabla 4.3:

TABLA 4.3. REPOSITORIO XORCOM RAPID

```
#Asterisk AMPortal
deb http://updates.xorcom.com/rapid sarge main
```

APT es una herramienta de administración de paquetes creada por el proyecto Debian, que simplifica en gran medida el proceso de instalación y desinstalación de paquetes en sistemas GNU/Linux.

Para no interferir con el resto de paquetes instalados por Ubuntu, se modifica el archivo /etc/apt/preferences para indicarle a APT que solo busque en este repositorio los paquetes relacionados a Asterisk: zaptel, libpri1, libpq3, asterisk-sounds-main, asterisk y amportal. [NOR2003] especifica la manera como hacer esto.

El archivo debe quedar como lo muestra la tabla 4.4:

TABLA 4.4. ARCHIVO /ETC/APT/PREFERENCES

```
root@gst:~# cat /etc/apt/preferences
Package: asterisk
Pin: release o=Xorcom
Pin-Priority: 999

Package: zaptel
Pin: release o=Xorcom
Pin-Priority: 999

Package: libpri1
Pin: release o=Xorcom
Pin-Priority: 999

Package: libpq3
Pin: release o=Xorcom
Pin-Priority: 999

Package: asterisk-sounds-main
Pin: release o=Xorcom
Pin-Priority: 999
```

A continuación se explica brevemente el significado de cada opción escrita:

- Package: Indica el paquete al cual se quiere aplicar la regla.
- Pin: Indica alguna característica del paquete a tener en cuenta. En este caso los paquetes tienen como origen al repositorio de Xorcom. Este valor se obtiene siguiendo el procedimiento mostrado en la tabla 4.5 [NOR2003]:

TABLA 4.5. ORIGEN REPOSITORIO DE PAQUETES

```
root@gst:~# cat
/var/lib/apt/lists/updates.xorcom.com_rapid_dists_sar
ge_main_binary-i386_Release
Archive: stable
Version: 1.1
Component: main
Origin: Xorcom
Label: Debian-All
Architecture: i386
Description: Xorcom Rapid
```

- Pin-Priority: Indica la prioridad. Un mayor número indica mayor prioridad:
  - 0: El paquete nunca será instalado.
  - 0 – 100: paquetes no instalados y sin versiones disponibles
  - 101 – 1000: Paquete será actualizado o instalado con las actualizaciones del sistema
  - >1000: Paquete será instalado de todas maneras, así sea una versión anterior de la ya instalada.

Para instalar los paquetes se ejecuta el comando "*apt-get install amportal asterisk*" con el usuario root. Este usuario es el único que tiene permiso en el sistema para instalar o desinstalar paquetes.

4.5.1.2. HeartBeat

Este programa se instala desde los repositorios oficiales de Ubuntu con el comando "*apt-get install heartbeat*" que instala adicionalmente todas sus dependencias.

4.5.2. Configuración

La configuración es idéntica en ambos servidores por lo que solo se detallará lo hecho en el servidor principal (dirección IP 200.37.45.50).

4.5.2.1. Asterisk y AMPortal

Una vez instalado el Asterisk y el AMPortal se crean las bases de datos “asterisk” (configuración de la PBX) y “asteriskcdrdb” (*Call Detail Record* - detalle de llamadas) en mysql, en caso no ocurriese eso se verifica los permisos del usuario debian-sys-maint en mysql. La razón de esto es que en Ubuntu este usuario no tiene los permisos necesarios, y como el paquete es de Debian, se debe hacer esa única modificación para que funcione. Para ello se usa la aplicación web PHPmyadmin que se encarga de la configuración del motor de base de datos (usuarios y bases de datos).

Adicionalmente Ubuntu cuando inicia borra el contenido del directorio /var/run. Esta carpeta contiene un subdirectorio asterisk importante para el funcionamiento de la PBX. Debido a esto se debe modificar el archivo ejecutable de Asterisk (/etc/init.d/asterisk). Los detalles de cómo hacer esto se encuentran en el anexo 5.

Una vez instalado el AMPortal, desde cualquier máquina de la RAAP se podrá configurar el servidor. Para ello, desde cualquier navegador se accede a la dirección http://200.37.45.50/amportal/admin. El usuario y la contraseña por defecto son admin/admin. Una vez dentro se cambia el password por seguridad. Esta herramienta se utilizará para configurar las extensiones y las troncales que se tendrá en nuestra red de voz, como lo muestra la figura 4.2:

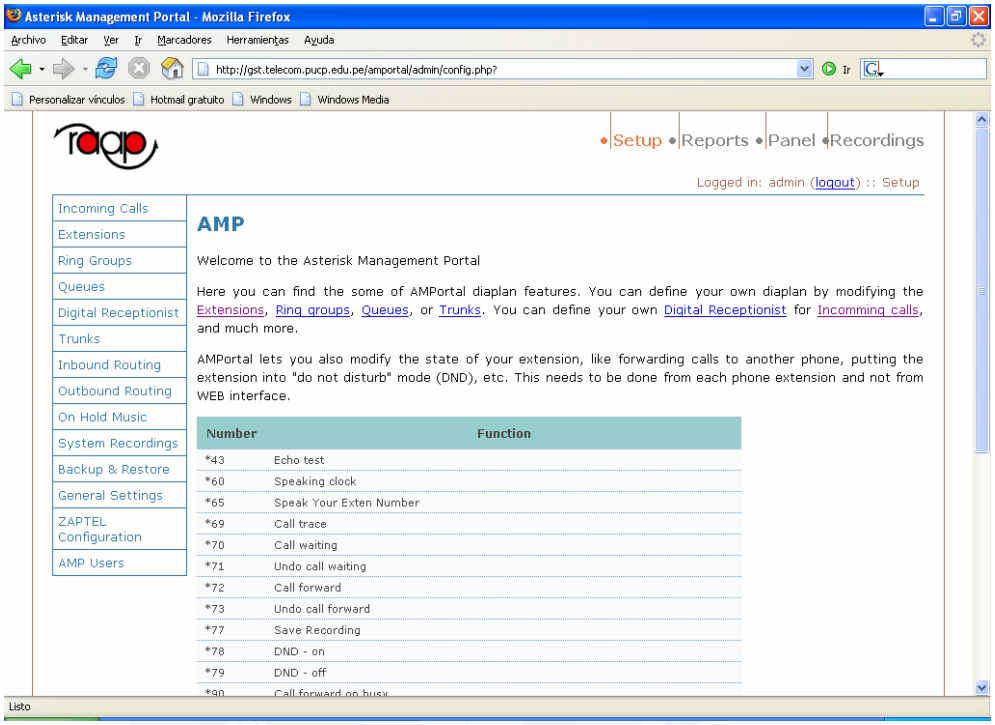

FIGURA 4.2. INTERFAZ AMPORTAL

El AMP contiene las siguientes opciones en el menú:

- *Incoming calls:* Permite configurar como serán tratadas las llamadas entrantes por cualquiera de las troncales al sistema.
- *Extensions:* Permite crear extensiones SIP, IAX2 o analógicas.
- *Ring groups:* Permite crear grupos de timbrado. De esta forma marcando un solo número se pueden llamar a varios usuarios.
- *Queues:* Permite crear colas de llamadas. Esta funcionalidad es propia de un *call center*. No se usará.
- *Digital Receptionist:* Permite configurar una grabación que dé la bienvenida a los usuarios externos para luego pedir el número de extensión.

- *Trunks:* Esta opción permite crear troncales, ya sea SIP, IAX2 o ZAP (analógicas). Útil para conectarse a otras redes de VoIP o incluso a la PSTN.

- *Inbound Routing:* En esta opción se indica que un número en particular vaya dirigido a una extensión específica si se tienen varios números asignados por un proveedor.

- *Outbound Routing:* Define el plan de marcado (*dialing plan*), es decir, se indica que de acuerdo al número marcado se vaya por una determinada troncal u otra.

- *On Hold Music:* En esta opción se agrega música en espera en formato wav o mp3.

- *System Recordings:* Permite subir un archivo de bienvenida para cualquier extensión. A diferencia de *Digital Receptionist*, al final del mensaje se llama a la extensión marcada.

- *Backup & Restore:* Permite hacer un respaldo de la configuración en la base de datos así como también restaurarla.

- *General Settings:* Son las configuraciones generales, como por ejemplo el número de segundos a timbrar, o antes de enviarlo al correo de voz.

- *ZAPTEL Configuration:* Permite la configuración del hardware analógico en la central.

- *AMP Users:* Permite crear usuarios de la administración web con los permisos que se deseen.

4.5.2.2. HeartBeat

Los archivos de configuración tanto en el servidor principal como en el secundario son idénticos. A continuación se describe cada uno de ellos [ROB2006]:

- /etc/ha.d/ha.cf: Este archivo contiene las configuraciones generales, como por ejemplo donde almacenar los archivos de *logs* y los miembros del sistema de alta disponibilidad. Los miembros deben escribirse tal cual aparecen ejecutando el comando "*uname –n*". El archivo a usarse en el cluster de alta disponibilidad se muestra en la tabla 4.6:

TABLA 4.6. ARCHIVO /ETC/HA.D/HA.CF

```
root@gst:~# cat /etc/ha.d/ha.cf
debugfile /var/log/ha-debug
logfile /var/log/ha-log
logfacility local0
keepalive 200ms
deadtime 2
warntime 1
initdead 120
udpport 694
bcast eth0
node gst
node git
```

- /etc/ha.d/haresources: Este archivo sirve para definir cual va a ser la IP virtual del sistema y que servicios se levantarán en el secundario cuando se caiga el servidor principal (solo se levantará el servicio asterisk). También se define cual es el servidor principal. Este archivo se muestra en la tabla 4.7:

TABLA 4.7. ARCHIVO /ETC/HA.D/HARESOURCES

```
root@gst:~# cat /etc/ha.d/haresources
gst 200.37.45.52 asterisk
```

- /etc/ha.d/authkeys: En este archivo se define la clave que usarán ambos servidores para intercambiar *heartbeats*. La tabla 4.8 muestra como debería quedar el archivo.

TABLA 4.8. ARCHIVO /ETC/HA.D/AUTHKEYS

```
root@gst:~# cat /etc/ha.d/authkeys
auth 1
1 sha1 SuPerS&cretP@$$guord
```

HeartBeat funciona haciendo uso del protocolo ARP. Se vale de una IP virtual que puede ser asignada al servidor principal o al secundario de acuerdo a si el principal está caído o no. Para esto se requiere que ambos servidores estén en el mismo segmento de red y configurar a los clientes con la IP virtual. De esta forma, cuando el switch o router pregunte por la dirección física de la IP virtual, responderá el servidor principal en caso esté funcionando y en caso contrario responderá el servidor secundario. El servidor secundario se da cuenta que el servidor principal se cae porque constantemente está escuchando los "*heartbeats*" que produce este último. De esta forma si ya no recibe "*heartbeats*", cuando llegue una petición ARP, el secundario contesta con su MAC.

4.5.2.3. Mejora en la eficiencia

El servicio asterisk en el secundario solo debe comenzar en el momento que el servidor se cae y no en el inicio (*boot*). Para lograr esto se usa el comando sysv-rc-conf. Si no está instalado se instala con apt-get. Una vez ejecutado el comando, se quita el servicio asterisk de todos los *runlevels*. El término runlevel se refiere al modo de operación de cualquier sistema operativo que implemente el estilo de inicialización (*init*) del sistema Unix V (versión 5). Se definen 7 *runlevels*, del 0 al 6. Cuando un *host* entra al *runlevel* 0, se apaga; cuando entra al 6, se reinicia. Los *runlevels* intermedios se

diferencian en las particiones montadas y en los servicios que se inician. Los niveles más bajos usualmente son usados para mantenimiento y reparaciones de emergencia, ya que solo los servicios más necesarios son inicializados.

4.5.2.4. Réplica de la base de datos

Las configuraciones del servicio de VoIP se harán a través de la interfaz web provista por el servidor primario. Pero el servidor secundario debe estar al tanto de las modificaciones hechas en el primario, por lo que se debe replicar la base de datos en el servidor secundario cada día. De esta forma, cuando se caiga el principal, el secundario pueda cargar la configuración del Asterisk sin ningún tipo de problema. Como se mencionó anteriormente, se tienen dos tablas MySQL usadas: asterisk y asteriskcdrdb. Se copiarán ambas bases de datos en el secundario con los comandos mostrados en la tabla 4.9 (ejecutados en el servidor principal) [MYS2006]:

TABLA 4.9. IMPORTACIÓN/EXPORTACIÓN BASES DE DATOS

```
dquintana@gst:~$ mysqldump --databases asterisk --user
asteriskbackup -p9HPGsZucKDpxvfUQ | mysql --host=200.37.45.51
-C asterisk –user asteriskbackup -p9HPGsZucKDpxvfUQ

dquintana@gst:~$ mysqldump --databases asteriskcdrdb --user
asteriskbackup -p9HPGsZucKDpxvfUQ | mysql --host=200.37.45.51
–C asteriskcdrdb --user asteriskbackup -p9HPGsZucKDpxvfUQ
```

Las dos secuencias de comandos son similares, solo cambia la base de datos. Para poder ejecutar los comandos previamente se creó el usuario asteriskbackup (contraseña 9HPGsZucKDpxvfUQ) con acceso ilimitado a las dos bases de datos.

El primer comando (mysqldump) hace la exportación de la base de datos. Este resultado se pasa como parámetro al siguiente comando (mysql) que hace que se sobrescriba la base de datos remota. Entre

los dos comandos se tiene el operador “|” que hace las veces de pasarela entre ambos, permitiendo usar el resultado del primer comando como entrada del siguiente. Usando el servicio Cron de Linux se puede hacer que se ejecuten esos comandos automáticamente una vez al día. Cron es un demonio que tiene la funcionalidad de poder ejecutar programas a intervalos regulares (por ejemplo cada minuto, hora, día o mes). Los procesos que deben ejecutarse y la hora en la que deben hacerlo se especifican en el archivo /etc/crontab, el cual se modifica en el servidor principal de la forma descrita en la tabla 4.10:

TABLA 4.10. ARCHIVO /ETC/CRONTAB

```
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file.
# This file also has a username field, that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user  command
17 *    * * *   root    run-parts --report /etc/cron.hourly
25 6    * * *   root    test -x /usr/sbin/anacron || run-parts --report /etc/cron.daily
47 6    * * 7   root    test -x /usr/sbin/anacron || run-parts --report /etc/cron.weekly
52 6    1 * *   root    test -x /usr/sbin/anacron || run-parts --report /etc/cron.monthly
#
13 4     * * *  dquintana mysqldump --databases asterisk --user asteriskbackup -p9HPGsZucKDpxvfUQ |
mysql --host=200.37.45.51 -C asterisk --user asteriskbackup -p9HPGsZucKDpxvfUQ && mysqldump --databases
asteriskcdrdb --user asteriskbackup -p9HPGsZucKDpxvfUQ | mysql --host=200.37.45.51 -C asteriskcdrdb --
user asteriskbackup -p9HPGsZucKDpxvfUQ
```

5. Pruebas de Desempeño

5.1. Introducción

Este capítulo consiste en obtener el número máximo de llamadas que puede soportar el servidor principal de acuerdo a la codificación que se usa. Existen varios estudios sobre pruebas de desempeño hechas. Se tomará como base el descrito en [XOR2006]. Primero se compararán los *codecs* G.711 e iLBC para luego estudiar la alta disponibilidad y como el sistema responde a probables cortes de servicio.

Estas pruebas consisten en usar el programa SIPp [JAC2006] que permite realizar llamadas a un servidor SIP e ir cambiando el *codec* usado en la comunicación. El funcionamiento de SIPp se muestra en la figura 5.1:

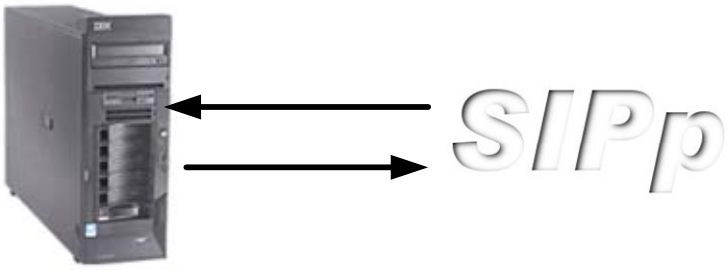

FIGURA 5.1. FUNCIONAMIENTO SIPP

El servicio SIPp irá generando llamadas a una tasa de 1 llamada por minuto. Estás llamadas serán hechas a una extensión que lo único que hace es reproducir un archivo de música de alrededor de 300 minutos. El servidor SIPp responde repitiendo todo lo que le llega. De esta manera se simula una llamada convencional entre dos usuarios cualesquiera, pues también se está enviando tráfico RTP. Se hará la prueba con dos de los *codecs* descritos en capítulos anteriores (G.711 e iLBC) y se obtendrá la capacidad máxima de llamadas concurrentes para cada uno. Los parámetros a analizar y que serán graficados en un eje de tiempo en la prueba son los siguientes:

- Número de llamadas concurrentes: Se verá como aumenta el número total de llamadas concurrentes conforme corra la simulación, de acuerdo a la tasa de generación (1 llamada por minuto).

- Tasa de Bits: Se verá la tasa de bits consumida por el total de llamadas por cada minuto de tiempo transcurrido.

- Uso de CPU: Se verá como aumenta el uso de CPU conforme aumentan las llamadas. Se verán dos tipos de uso de CPU: del sistema y de usuario. El primero mide la cantidad de CPU usada por los procesos propios del sistema operativo mientras que el segundo mide el consumo de CPU usado por los procesos propios de los usuarios.

- Carga promedio: Es la medida que indica la carga que están produciendo los procesos que están usando la CPU en un momento

determinado [MAR2000]. Si se ejecutara un solo proceso que consuma el 50% de CPU se tendría una carga de 0.50. Si se tuviera un proceso que consuma el 100% de CPU se tendría una carga del sistema de 1.0. Ahora, si se tuviera dos procesos con las mismas características del último, se consumiría todo el CPU disponible (100%) pero la carga del sistema ya no sería 1.0 sino 2.0.

Para la obtención de todos los datos arriba mencionados se usará el software de monitoreo Cacti con el *plugin* de Asterisk habilitado. La configuración del sistema de monitoreo se encuentra detallada en [DIA2007].

Por otro lado, el SIPp necesita de un usuario en el Asterisk (usuario sipp) para poder realizar las llamadas. Se debe crear el archivo /etc/asterisk/sip_custom.conf (este archivo se encuentra incluido en el archivo de configuración de Asterisk sip.conf) tal como lo muestra la tabla 5.1:

TABLA 5.1. ARCHIVO /ETC/ASTERISK/SIP_CUSTOM.CONF – SIPP

```
[sipp]
type=friend
host=dynamic
context=from-internal
```

Se asigna el contexto *from-internal* al usuario sipp para tratarlo como si fuera cualquier otro usuario de la centralita, y de esta forma poder llamar a la extensión 6000.

5.2. G.711

5.2.1. Configuración

Para lograr la generación de llamadas con este *codec* se usa la configuración por defecto del SIPp como UAC (*User Agent Client*). Para

esto se llama al SIPp con la secuencia de comandos mostrada en la tabla 5.2:

TABLA 5.2. SIPP G.711

```
dquintana@git:~$ sipp -sn uac -d 15000000 -s 6000 -mp 10001 -i
200.37.45.51 -r 0.016666666666666 200.37.45.50
```

Los parámetros elegidos para interactuar con el SIPp son los siguientes:

- **sn:** Carga un escenario por defecto. Los más usados son UAS (servidor) y UAC (cliente)
- **d:** Indica la duración en milisegundos de cada llamada. En este caso es de 15000 segundos.
- **s:** Indica la extensión a ser llamada. En este caso es 6000, la cual es una extensión que solo reproduce un archivo de música de duración aproximada de tres horas.
- **mp:** *Media Port*, indica por que puerto se quiere recibir todo el tráfico RTP del servidor. En este caso se elige el puerto 10001.
- **i:** Indica cual es la dirección IP del cliente, es decir, la dirección IP de la PC donde se ejecuta el SIPp (200.16.6.205).
- **r:** Es la tasa de creación de llamadas. En este caso se genera una llamada cada minuto (0.016 llamadas cada segundo).
- Por último se especifica la dirección IP del servidor Asterisk, donde se registrará primero al usuario sipp y luego ya podrá realizar llamadas.

5.2.2. Resultados

Las llamadas comenzaron el día 24/09/2006 a las 11:00 a.m. Se generó una nueva llamada cada minuto. La simulación terminó a las 14:19 del mismo día.

5.2.2.1. Número de Llamadas Concurrentes

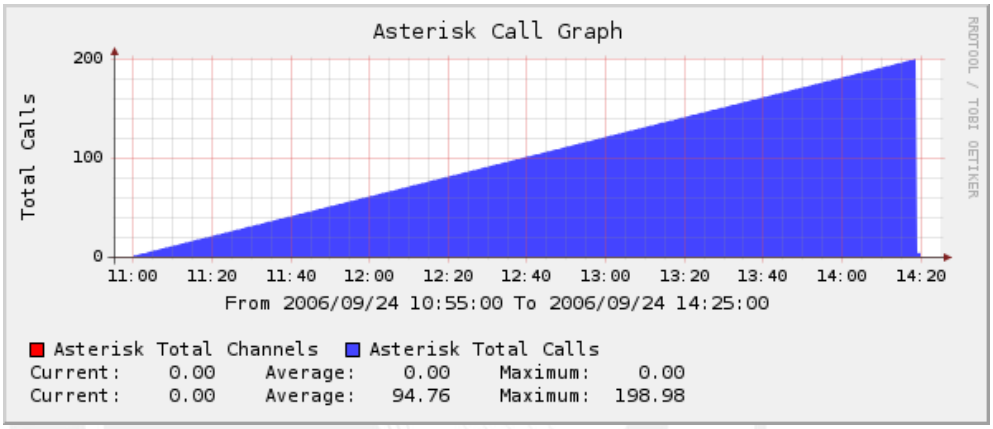

FIGURA 5.2. NÚMERO DE LLAMADAS CONCURRENTES G.711

En la figura 5.2 se muestra como van aumentando las llamadas hasta llegar a 199 llamadas concurrentes. Cuando llega la llamada 200, el servidor rechaza la petición y corta el servicio, ocasionando la pérdida de las demás llamadas.

5.2.2.2. Tasa de Bits

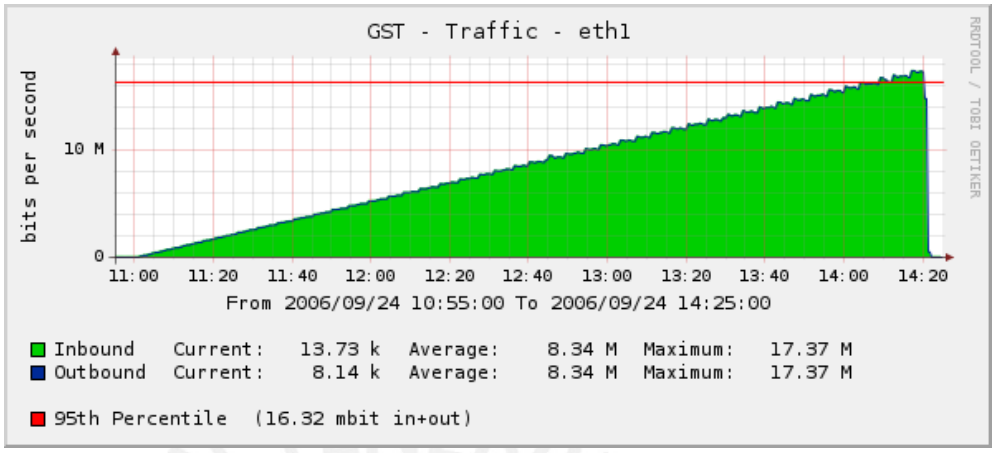

FIGURA 5.3. TASA DE BITS G.711

De la teoría se sabe que cada llamada SIP con el *codec* G.711 consume aproximadamente 80 kbps. Si eso se multiplica por el total de llamadas concurrentes en cada período de muestreo se obtienen los valores mostrados. Al igual que en la figura 5.2, la figura 5.3 muestra que a partir de la llamada 199 ya no se pueden procesar las llamadas y la tasa de bits cae a 0 kbps.

5.2.2.3. Uso de CPU

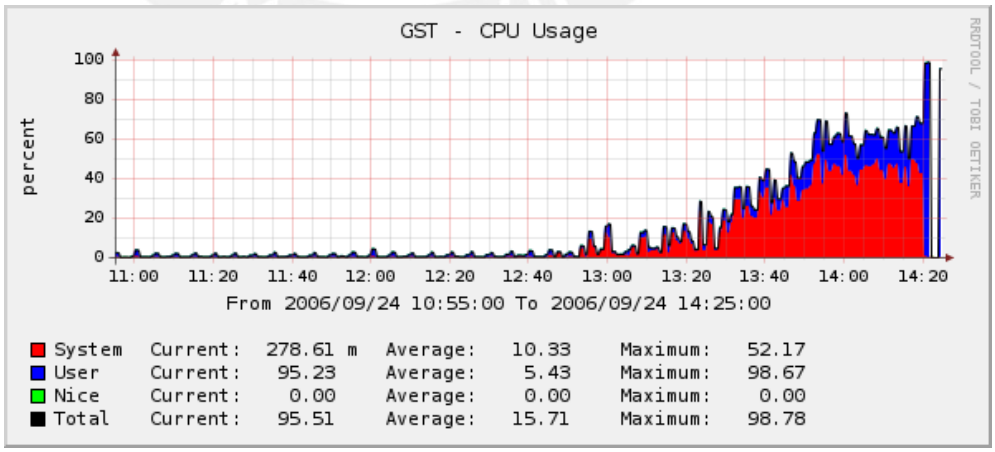

FIGURA 5.4. CONSUMO DE CPU G.711

En la figura 5.4 se observa que alrededor de la llamada 170 (13:50) se llega al limite de consumo de CPU, y las llamadas que siguen

están rondando ese valor, hasta que colapsa el sistema, llegando a tener incluso un uso de CPU de 100%. Se observa un consumo de uso de CPU mucho mayor por parte del sistema que por parte del usuario en la parte final del gráfico. Esto es debido a que la codificación G.711 no hace uso de recursos de CPU (envía la información tal cual fue muestreada). El uso de CPU del sistema (color rojo) indica el procesamiento ejecutado por el propio Asterisk.

5.2.2.4. Carga Promedio

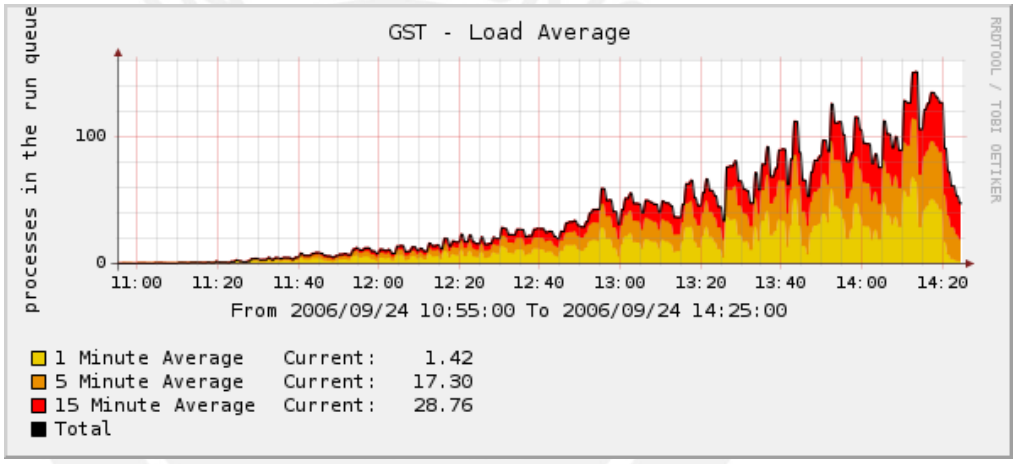

FIGURA 5.5. CARGA PROMEDIO G.711

La figura 5.5 muestra que alrededor de la llamada 170 (13:50) se llega al limite de la carga del sistema. Sin embargo, se pudo seguir sin mayores complicaciones hasta la llamada 199 que hizo colapsar al sistema.

5.3. iLBC

5.3.1. Configuración

En la parte concerniente al Asterisk, se debe modificar la configuración del usuario sipp, para que solo acepte el *codec* iLBC. Para ello se usa la directiva allow=iLBC en la sección del usuario sipp.

Por la parte relativa al SIPp como programa, se debe modificar el escenario para que se use iLBC en la sesión. Para esto, tomando como base el que viene por defecto de UAC, se crea un escenario nuevo. Se obtiene el archivo xml del usuario UAC ejecutando el comando mostrado en la tabla 5.3:

TABLA 5.3. OBTENCIÓN ESCENARIO POR DEFECTO

```
dquintana@git:~$ sipp -sd uac > escenariotesis.xml
```

Una vez con el escenario copiado, se observa que en la línea de invitación de la sesión SIP aparece el *codec* a ser usado, el cual está en la línea destacada del archivo. Se debe cambiarlo al valor de iLBC, que puede ser entre 96 y 127, de tal forma que quede como muestra la tabla 5.4:

TABLA 5.4. ARCHIVO XML ILBC

```
...
      INVITE sip:[service]@[remote_ip]:[remote_port] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      From: sipp <sip:sipp@[local_ip]:[local_port]>;tag=[call_number]
      To: sut <sip:[service]@[remote_ip]:[remote_port]>
      Call-ID: [call_id]
      CSeq: 1 INVITE
      Contact: sip:sipp@[local_ip]:[local_port]
      Max-Forwards: 70
      Subject: Performance Test
      Content-Type: application/sdp
      Content-Length: [len]

      v=0
      o=user1 53655765 2353687637 IN IP[local_ip_type] [local_ip]
      s=-
      c=IN IP[media_ip_type] [media_ip]
      t=0 0
      m=audio [media_port] RTP/AVP 0
      a=rtpmap:96 iLBC/8000
...
```

Una vez con el archivo terminado, se llama al SIPp con el comando mostrado en la tabla 5.5:

TABLA 5.5. SIPP ILBC

```
dquintana@git:~$ sipp -sf ilbc.xml -d 72200000 -s 6000 -mp
10001 -i 200.37.45.51 -r 0.01666666666666 200.37.45.50
```

En este caso se usa una duración de llamadas menor debido a que se espera que el servidor soporte menos llamadas que en el caso de G.711, los demás parámetros quedan iguales.

5.3.2. Resultados

Las llamadas comenzaron a las 17:00 horas del 24/09/2006. La simulación terminó a las 18:45 del mismo día. Las gráficas obtenidas se muestran a partir de la figura 5.6 hasta la figura 5.9:

5.3.2.1. Número de Llamadas Concurrentes

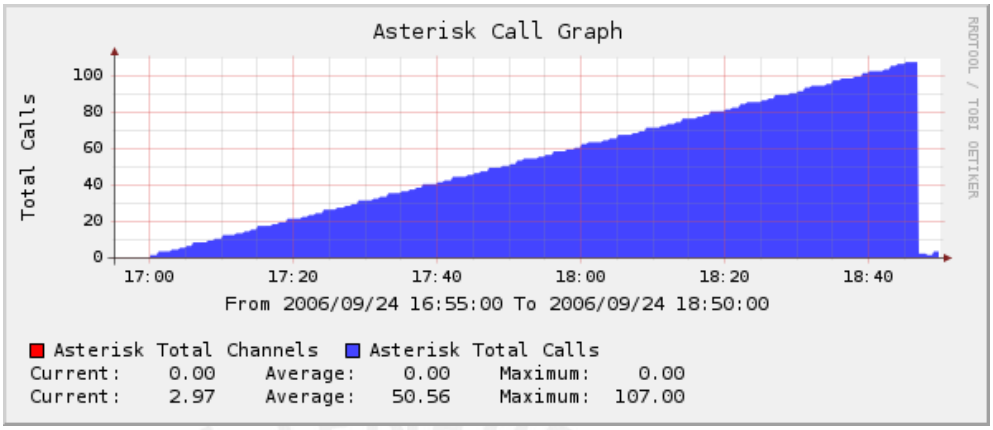

FIGURA 5.6. NÚMERO DE LLAMADAS CONCURRENTES ILBC

En la figura 5.6 se muestra como van aumentando las llamadas hasta llegar a 107, luego del cual se caen todas las llamadas. Sin embargo este número 107 es un número engañoso, se verá luego por qué.

5.3.2.2. Tasa de Bits

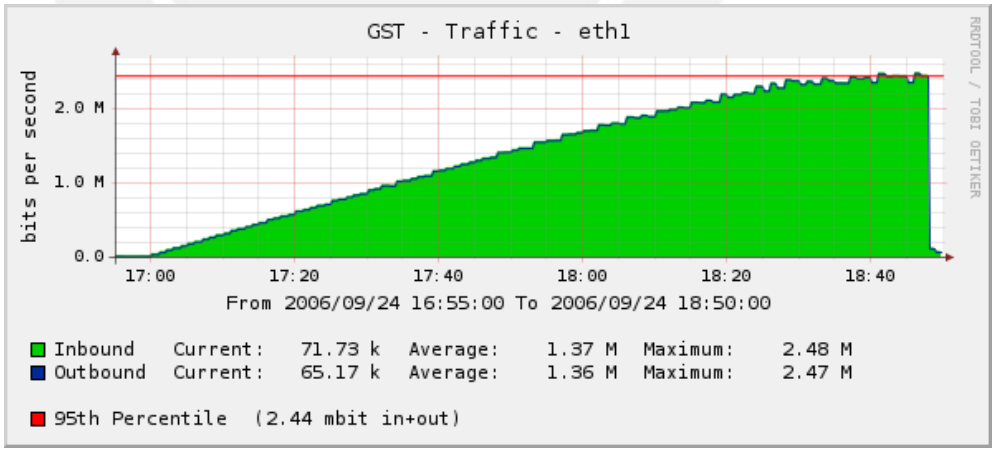

FIGURA 5.7. TASA DE BITS ILBC

De la teoría se sabe que cada llamada SIP con el *codec* iLBC consume alrededor de 25 kbps. Si eso se multiplica por el total de llamadas concurrentes en cada período de muestreo se obtienen los valores mostrados. Al igual que en la figura 5.6, la figura 5.7 muestra

que a partir de la llamada 107 ya no se pueden procesar las llamadas y la tasa de bits cae a 0 kbps. Sin embargo, se observa también que la gráfica comienza a perder linealidad alrededor de la llamada 80. Esto se explicará más adelante.

5.3.2.3. Uso de CPU

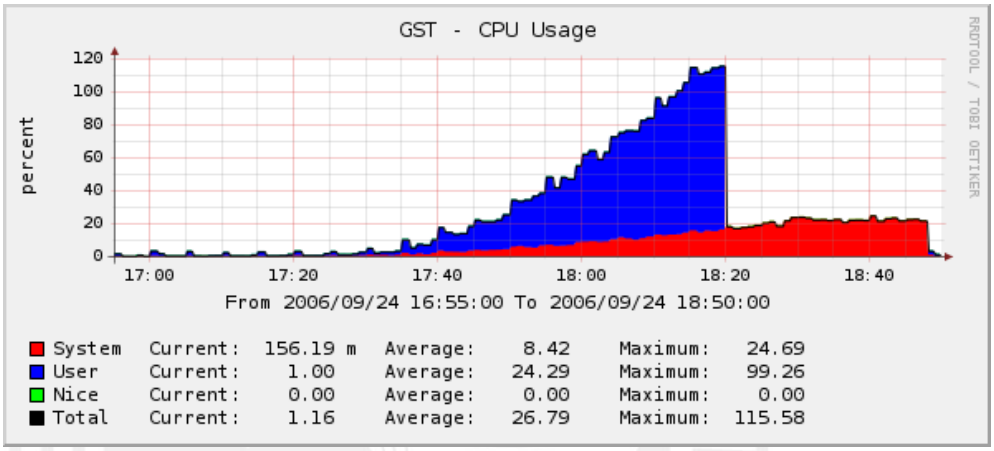

FIGURA 5.8. CONSUMO DE CPU ILBC

En la figura 5.8 se deben distinguir el uso de CPU del Sistema (*System CPU Usage*) del uso de CPU del usuario (*User CPU Usage*). El uso de CPU del sistema se refiere al uso de CPU por parte de los procesos originados por el sistema, en este caso es el Asterisk y la creación de llamadas, más no de la compresión de la voz, de la cual se encarga un proceso de usuario. Es por eso que en G.711 no existe un CPU de usuario considerable al no haber compresión de la voz, sino más bien la voz pasa tal cual fue muestreada.

Si se observa el *System CPU Usage*, este es casi el mismo que en G.711 cuando no están saturados ambos sistemas, por lo que el uso adicional de CPU es el dado por la compresión iLBC.

Se observa en la figura 5.8 que exactamente luego de la llamada 80, cae todo el uso de CPU de usuario a 0, llegando a tener un pico de más del 100%, por lo que se ha visto, luego de esa llamada no

debería haber compresión, o debería escucharse cualquier cosa. Cuando se hizo la prueba, se intentó llamar a una extensión alrededor de la llamada 90, y se escuchaba entrecortado, con una calidad bajísima, casi ininteligible, eso explicaría también la gráfica de tasa de bits, que luego de la llamada 80 pierde su linealidad, debido justamente a que los chorros de sonido ya no están siendo enviados a una tasa de bits constante.

5.3.2.4. Carga Promedio

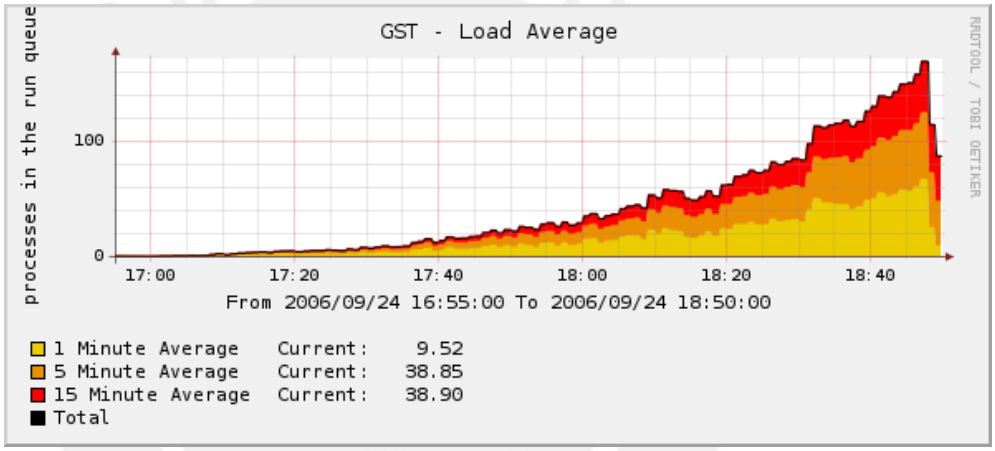

FIGURA 5.9. CARGA PROMEDIO ILBC

En la figura 5.9 se observa que alrededor de la llamada 90 (18:30) se llega al limite de la carga del sistema. Sin embargo, no hubo mayor complicación sino hasta la llamada 107 que hizo colapsar el sistema. Comparada con el uso de CPU, ésta gráfica brinda menos información pues concuerda con el Uso de CPU de Sistema, dejando de lado el Uso de CPU de Usuario.

5.4. Alta Disponibilidad

Con las pruebas anteriores se ha determinado el número de llamadas que el servidor principal puede soportar. Sin embargo no se ha comprobado aún

el funcionamiento de la alta disponibilidad, que se pasará a detallar a continuación:

5.4.1. Configuración

En la parte concerniente a la configuración, se usará la misma que para G.711. Para llamar al SIPp se usa el comando mostrado en la tabla 5.6:

TABLA 5.6. SIPP ALTA DISPONIBILIDAD

```
dquintana@git:~$ sipp -sn uac -d 15000000 -s 6000 -mp 10001 -i
200.37.45.51 -r 0.033333333333333333333333333 200.37.45.52
```

La principal diferencia con el comando de G.711 es que ahora se crea 1 llamada cada 30 segundos y la dirección IP del servidor es la dirección IP virtual.

5.4.2. Resultados

Para esta parte solo se tomarán en cuenta las gráficas de llamadas concurrentes tanto en el servidor principal como en el secundario. De esta forma se puede apreciar el comportamiento del sistema en caso de falla del servidor principal. Para simular la caída del servidor principal simplemente se apaga, y se deja que el secundario se encargue de las llamadas, luego se vuelve a prender el principal, y este retoma el control de las llamadas. La prueba se realizó el 01/11/2006.

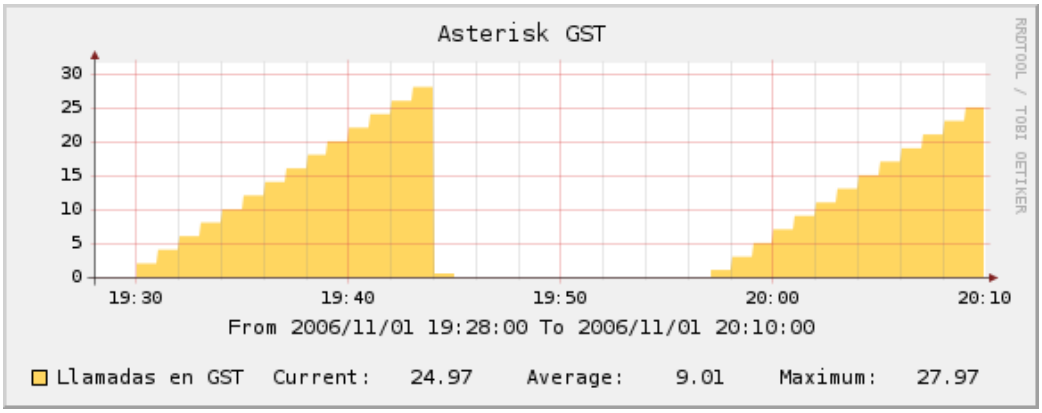

FIGURA 5.10. LLAMADAS SERVIDOR PRINCIPAL

La figura 5.10 muestra las llamadas atendidas por el servidor principal. De la gráfica se desprende que a las 19:44 se apagó el servidor principal y regresó a las 19:57. Durante ese tiempo estuvo funcionando el servidor secundario, como se muestra en la figura 5.11:

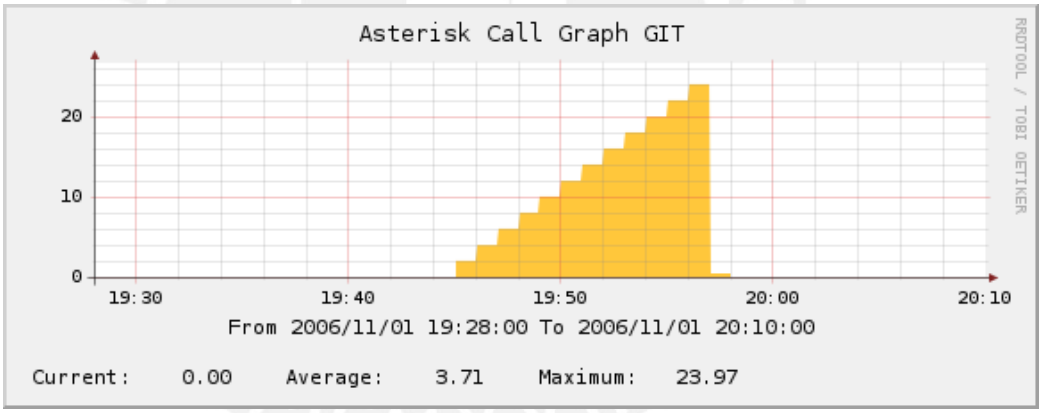

FIGURA 5.11. LLAMADAS SERVIDOR SECUNDARIO

Si se combinan ambas gráficas en una sola se obtiene la figura 5.12:

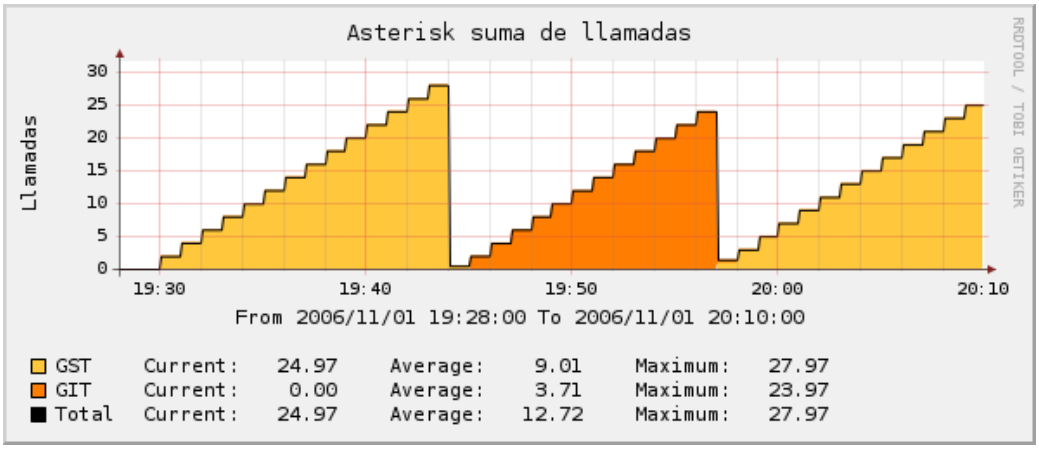

FIGURA 5.12. TOTAL DE LLAMADAS

Se aprecia que si bien las llamadas concurrentes en el momento que se cae el servicio se pierden, el MTTR (*Mean Time To Repair*, Tiempo Promedio Para Reparar) es el mínimo, a lo sumo un par de segundos. De esta forma, se aumenta la disponibilidad del sistema, puesto que el tiempo en el que no hay servicio es casi cero. El MTTR es un concepto muy distinto del MTBF (*Mean Time Between Failures*, Tiempo Promedio entre fallas). La disponibilidad de un sistema se define como:

$$A_i = \frac{MTBF}{MTBF + MTTR} \qquad (4)$$

De la ecuación se desprende que si el MTTR es casi cero o muy pequeño en comparación con MTBF, la disponibilidad es muy cercana al 100%.

5.5. Conclusiones

De las pruebas realizadas se llega a la conclusión que, si bien con iLBC se tiene una menor tasa de bits, se tienen también pocas llamadas concurrentes. Como en la RAAP la tasa de bits no es mayor inconveniente y, en cambio, sí se necesita la mayor cantidad posible de llamadas

concurrentes, se recomienda a la RAAP la utilización de G.711 como *codec* de voz.

Adicionalmente se comprobó la utilidad de la redundancia en los servidores, ya que se redujo al mínimo el MTTR, que es el tiempo que se demora un sistema que ha colapsado en repararse. De esta manera no se pierde el servicio en el tiempo, aunque sí se pierden las llamadas que en ese momento estén llevándose a cabo. Se propone una nueva investigación que aborde precisamente este tema, y ya no se pierdan las llamadas concurrentes. Esto se especifica con mayor detalle en la sección 6.3, donde se mencionan los trabajos futuros.

6. Conclusiones Finales, Recomendaciones y Trabajos Futuros

6.1. Conclusiones finales

Al terminar el presente trabajo de Tesis se puede concluir lo siguiente:

- Como resultado de la comparación de las diversas tecnologías disponibles, se concluye que las mejores alternativas son usar los protocolos IAX2 y SIP, así como hardware basado en IP. El *codec* a ser usado es el G.711 por las razones expuestas en la sección 6.2 (Recomendaciones).

- Se implementó la red piloto de telefonía IP haciendo uso de dos servidores redundantes con clientes tanto en la PUCP como en INICTEL. Al haber redundancia se logra una mayor eficiencia en la disponibilidad del sistema, lo que genera en los usuarios confianza para comenzar a usar la red de telefonía IP.

- Se midió la capacidad real del sistema realizando pruebas de desempeño para ver el número de llamadas concurrentes de acuerdo al *codec* y para comprobar realmente el rendimiento del cluster de alta disponibilidad. Todas las pruebas fueron satisfactorias, encontrándose que la capacidad real del sistema es de 199 llamadas concurrentes para el caso del *codec* G.711. De esta forma se puede dar servicio a los usuarios de la RAAP con una calidad alta (MOS de 4.1).

6.2. Recomendaciones

Luego de examinar minuciosamente cada tecnología usada, se puede decidir fácilmente cuales se recomiendan para ser usadas en la fase de producción definitiva del sistema.

Las tecnologías recomendadas son las siguientes:

- Protocolo: Se comprobó que el protocolo IAX2 es más robusto, además que traspasa ambientes de NAT. Esto es importante para los sitios de investigación en los que haya un laboratorio saliendo a la RAAP por una sola dirección IP. Sin embargo, este protocolo todavía no está muy difundido por lo que SIP también puede usarse. En este último caso para traspasar NATs se puede usar protocolos auxiliares como STUN.

- Hardware: Se recomienda el uso de hardware basado en IP. El tipo de hardware depende mucho de cada institución miembro de la RAAP, del presupuesto con el que se cuente y de la infraestructura existente. Por ejemplo, una institución que solo cuente con teléfonos analógicos preferirá conservar esos teléfonos y convertirlos a IP haciendo uso de los famosos ATA (para el caso del protocolo IAX2, se tiene el IAXy descrito en el capítulo 3). Otra institución con un poco más de presupuesto preferirá invertir en teléfonos IP (hardware)

cuyo costo es de alrededor de US$100 cada uno. Otra opción es usar teléfonos por software (*softphones*) y de esta manera no se gasta nada. Se recomienda esta opción para la primera fase de producción. Para los servidores se recomienda continuar con los que ya viene trabajando la red piloto.

- *Codec*: Como la prioridad en la red no es la tasa de bits, y sí lo es el brindar la mayor cantidad de comunicaciones con la mejor calidad posible, se recomienda usar el *codec* G.711 que demostró dar la mejor calidad y la mayor capacidad de llamadas concurrentes.

6.3. Trabajos Futuros

Muchos trabajos pueden ser derivados de la presente monografía entre los que se tienen:

- Implementar la no pérdida de comunicación cuando se caiga el servidor principal: Este tema por si solo es un tema de tesis debido a que involucra una sincronización a un mayor nivel de la que se plantea en este trabajo. Para ello el Asterisk por sí solo no basta y se hace necesario el uso de un programa auxiliar llamado OpenSER. Al usar OpenSER, la conversación se vuelve punto a punto y se usa el servidor únicamente para señalización. En caso cayera el servidor, las comunicaciones se mantienen sin ningún problema.

- Implementar nuevos servicios: Estos servicios pueden incluir video conferencia, además de números de servicios automatizados, como por ejemplo, predicción del tiempo. Para esto se debe investigar en el tema de IVRs (*Interactive Voice Response*) y síntesis de voz (TTS, *Text To Speech Synthesis*). Un IVR es un Sistema Automatizado de Respuesta de voz. TTS, en cambio, es la conversión de texto a voz, usado en IVRs. Adicionalmente se puede hacer *streaming* de audio y video, como por

ejemplo transmisión de canales de radio o TV de Internet a los usuarios de la red de VoIP.

- Implementar el cluster de alta disponibilidad en distintos segmentos de red: Actualmente el cluster está funcionando gracias a las peticiones y respuestas ARP, como se indicó en los capítulos anteriores. Estas peticiones solo funcionan en un determinado segmento de red. Por esta razón, los servidores pertenecientes al cluster se encuentran en un mismo segmento de red. Sin embargo, con *proxies* ARP, se puede hacer que estas peticiones no necesariamente vayan dirigidas al mismo segmento de red, pudiendo ser dirigidas a segmentos distintos. De esta forma se podría tener el servidor principal en INICTEL y el secundario en la PUCP. Se requiere para ello una configuración adecuada en los routers de la red.

BIBLIOGRAFÍA

[ATC2006] ATCOM Technology LTD. URL: http://www.atcom.cn/. Información de contacto: sales@atcom.cn. Consultado en Julio de 2006. Sección: *Products.*

[BAN2006] Banerjee K. '*Introduction to RTP*'. URL: http://geocities.com/intro_to_multimedia/RTP Información de contacto: intro_to_multimedia@yahoo.co.in Consultado en Junio de 2006. Sección: RTP.

[DIA2007] Díaz A. Tesis de grado: *'Diseño e Implementación del Centro de Operación y Gestión de la Red Académica Peruana en Software Libre'*. 2007.

[DIG2006] Digium ®, The Asterisk Company. URL: http://www.digium.com/. Información de contacto: info@digium.com. Consultado en Julio de 2006. Sección: Hardware.

[HER1999] Hersent O, Gurle D, Petit J. 2000. *'IP Telephony: Packet Based Communications Systems'*. Addison-Wesley. pp. 121-161.

[HUA2005] Huapaya, J. 2005. *'Separatas de Clase del Curso Telecomunicaciones Digitales'*. PUCP.

[ILB2003] Proyecto iLBCfreeware.org. 2003. URL http://www.ilbcfreeware.org/. Información de contacto: info@ilbcfreeware.org. Consultado en Julio de 2006.

[JAC2006] Jacques O. 2006. *'SIPp Project Page'*. URL http://sipp.sourceforge.net/. Información de contacto: jacques2@gmail.com. Consultado en Agosto de 2006.

[MAR2000] Martínez R, et al. 2000. *'FAQ sobre Linux'*. URL http://www.linux-es.org/Faq/Files/Html/FAQ_Linux_V2.0.2.html. Información de

Contacto: rafael@viewpoint.no. Consultado en Septiembre de 2006.

[MAR2006] Martínez F. '*Foro VoIP'*. URL: http://www.voipforo.com Información de contacto: info@voipforo.com Consultado en Junio de 2006. Secciones: H.323, SIP y *Codecs*.

[MOR2002] Moreno M, Álvarez M, Vinyes J. '*Una primera aproximación al protocolo SIP'*. URL: http://www.ahciet.net/comun/portales/1000/10002/10007/10302/docs/009.pdf. Revista de Telecomunicaciones 91 AHCIET. 2002. Consultado en Junio de 2006.

[MYS2006] MySQL AB. *'Manual de Referencia de MySQL 5.0'*. URL: http://dev.mysql.com/doc/refman/5.0/es/index.html. 2006. Sección consultada: *8.7 El programa de copia de seguridad de bases de datos mysqldump.* Consultado en Septiembre de 2006.

[NOR2003] Noronha G, Mora H. '*APT HOWTO – Capítulo 3'*. URL: http://www.debian.org/doc/manuals/apt-howto/ch-apt-get.es.html. Consultado en Septiembre de 2006.

[PAC2006] Packetizer INC. URL: http://www.packetizer.com Información de contacto: webmaster@packetizer.com Consultado en Junio de 2006. Sección: H.323.

[RAM2005] Ramesh K et al. 2005. '*Cisco IP Telephony: Planning, Design, Implementation, Operation, and Optimization*'. Cisco Press. pp. 12-22.

[RFC2543] Handley M, Schulzrinne H, et al. '*SIP: Session Initiation Protocol'*. 1999. URL: http://www.ietf.org/rfc/rfc2543.txt.

[RFC2705] Arango M, Duran A, et al. '*Media Gateway Control Protocolo (MGCP)'*. 1999. URL: http://www.ietf.org/rfc/rfc2705.txt.

[RFC3261] Rosenberg J, Schulzrinne H, et al. '*SIP: Session Initiation Protocol*'. 2002. URL: http://www.ietf.org/rfc/rfc3261.txt.

[RFC3550] Schulzrinne H, Casner S, et al. '*RTP: A transport Protocol for Real-Time Applications*'. 2003. URL: http://www.ietf.org/rfc/rfc3550.txt.

[RFC3665] Johnston A, Donovan S, et al. '*Session Initiation Protocol (SIP) Basic Call Flow Examples*'. 2003. URL: http://www.ietf.org/rfc/rfc3665.txt.

[RFC3951] Andersen S, Duric A, et al. *'Internet Low Bit Rate Codec'*. 2004. URL: http://www.ietf.org/rfc/rfc3951.txt.

[ROB2006] Robertson, A. 'The High-Availability Linux Project'. URL: http://www.linux-ha.org. Información de contacto: alanr@unix.sh Consultado en Agosto de 2006. Secciones: Principal y HeartBeat.

[SPE2006] Spencer M, Capouch B, et al. *'IAX: Inter-Asterisk eXchange Version 2'*. 2006. URL: http://www.ietf.org/internet-drafts/draft-guy-iax-01.txt.

[UIT2003] Recomendación UIT-T H.323. 2003. '*Sistemas de comunicación multimedios basados en paquetes'.* UIT.

[VAL2001] Valero, L. 'Curso Audivisual Interactivo de Codificación de Voz'. 2001. URL: http://ceres.ugr.es/~alumnos/luis/codif.htm. Consultado en Junio de 2006. Sección: RPE-LTP (GSM).

[WAL2005] Wallingford T. 2005. '*Switching to VoIP'*. O'Reilly

[WOO1998] Woodard, Jason. 1998. '*Standard Speech Codecs'*, URL: http://www-mobile.ecs.soton.ac.uk/speech_codecs/. University of Southampton. Información de contacto: jpw@ecs.soton.ac.uk. Consultado en Julio de 2006. Secciones: PCM, GSM

[XOR2006] Xorcom Ltd. 2006. '*TS-1 Performance and Load Test Results*', URL: http://www.xorcom.com/pdfs/AB004_TS1_Tests.pdf Información de contacto: info@xorcom.com. Consultado en Agosto de 2006.

Relación de Anexos

- Anexo 1: Hoja de datos Teléfono AT-320
- Anexo 2: *AT-320 SIP Phone User Manual*
- Anexo 3: *AT-320 IAX2 Phone User Manual*
- Anexo 4: Tramas Paquetes Capturados
- Anexo 5: Solución de Problemas en Ubuntu
- Anexo 6: Enlaces de Interés.