Proyecto VOIP

Programación de Sistemas

Fabrizio Cabaleiro – Carlos Ibáñez Ch. – Gabriel Juri M.

EXTRACTO

Informe sobre el proyecto relacionado con comunicación de voz sobre IP para el ramo “Programación de Sistemas”, Abril del 2012

Introducción:

VoIP o Voice over IP comúnmente se refiere a los protocolos de comunicación, tecnologías,metodologías y técnicas de transmisión involucradas en la comunicación de voz y sesiones multimedia a través de IP (Internet Protocol).

En la actualidad no se ha encontrado en Linux una aplicación para realizar llamadas por internet que satisfagan las exigencias de los participantes del proyecto, antes Skype satisfacía esta necesidad, pero, luego de que este programa fue comprado por Microsoft Evil Inc. los desarrolladores de Skype detuvieron su progreso para el sistema operativo Linux.

Se probaron varias aplicaciones libres para realizar VoIP, pero todas ellas tenían problemas que resultaban molestos, una de las aplicaciones libres que mas se utilizo fue Ekiga, esta es bastante buena, pero, tiene un problema muy molesto, este es que por razones desconocidas, Ekiga algunas veces al ser ejecutado, muestra un mensaje de error que dice que el usuario no esta registrado, se ha buscado en la web por este problema, pero, no se encontró una solución satisfactoria.

D-Light VoIP es desarrollado para llenar este agujero que existe en Linux, esta pensado en ser una aplicación muy liviana en donde el usuario sabe todo lo que esta pasando y puede fácilmente encontrar problemas en caso de que existieran.

D-Light VoIP genera una conexión directa entre ambos participantes de la conversación, no es necesario crear una cuenta de usuario o registrarse en algún sitio web, esta aplicación usa bibliotecas comunes y tiene una implementación de bajo nivel.

Visión General:

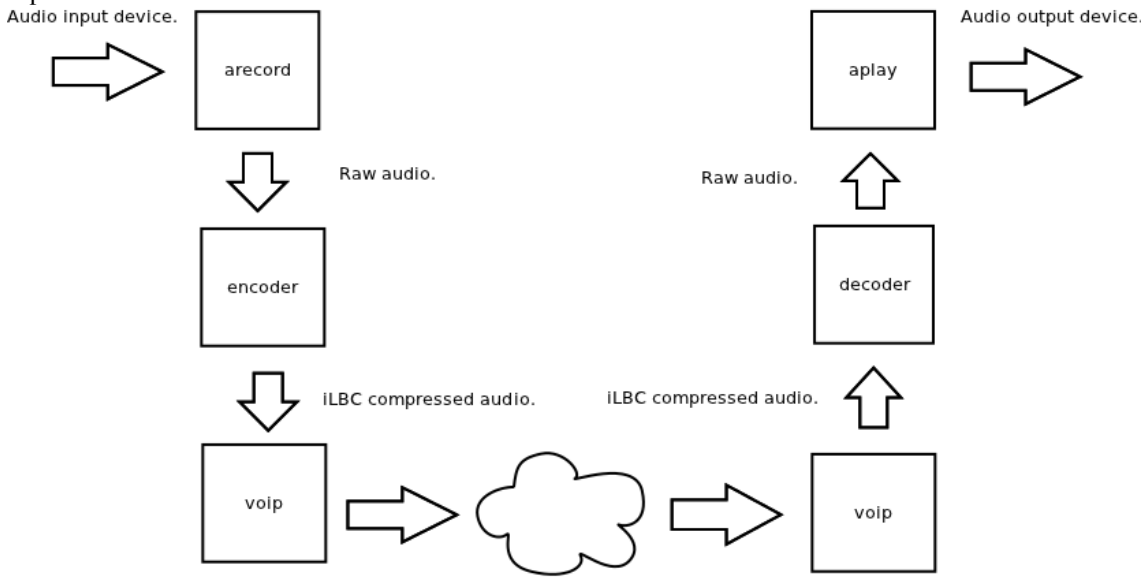
Este proyecto consta de 3 módulos:

1. Encoder: Modulo capas de decodificar el audio de entrada en el formato especificado por el RFC3951,
2. VoIP: Realiza una pipe de entrada entre el programa que captura sonido y el codificador. Realiza una pipe de salida entre el decodificador y el programa reproductor de sonido. Establece conexión con receptor de la llamada. Recibe información codificada desde pipe de entrada y la envía a receptor. Recibe información codificada desde emisor y la envía a la pipe de salida.
3. Decoder: Capaz de recibir información codificada y decodificarla de acuerdo a lo especificado en el RFC3951.

Este proyectos tiene dependencias con el controlador de audio ALSA para LINUX.

En la figura 1, se encuentra un diagrama con el "Overview" del proyecto. Este esquematiza que el audio del cliente es recibido desde el dispositivo correspondiente hacia la aplicación arecord, la que proporciona datos de audio crudos (raw) y posteriormente estos son codificados por el encoder para luego ser enviados al programa principal (voip) el que manda estos datos al servidor correspondiente. Luego el servidor recibe estos datos y los decodifica a través del decoder y los manda en forma cruda hacia la aplicación aplay, la que los saca finalmente por el dispositivo de audio del servidor.

D-Light VoIP, puede también trabajar con captura y reproducción proporcionada por el programa Sox, caso en el cual la captura con la codificación y la reproducción con la decodificación están internamente implementadas.



Funcionamiento General.

D-Light VoIP tiene una serie de parámetros que se le pueden dar por linea de comando o este puede preguntarlos según los necesite o dejarlos por defecto. Los parámetros son:

-lp <local_port> Puerto local que se une al socket que recibe data de audio, por defecto 1111.

-rp <remote_port> Puerto remoto al cual esta unido el socket del servidor remoto, por el cual recibe audio.

-ad <remote_Address> Dirección del servidor remoto con el cual uno se comunica.

-s <msize> Tamaño del paquete de datos que serán enviados por internet, por defecto 500.

-w Cuando esta opción es agregada, el programa espera recibir datos desde el servidor remoto antes de empezar a enviar datos de audio.

-au <program> Se puede optar por usar "alsa" o "sox", por defecto se usa alsa.

-r <sample_rate> Asigna el sample rate para la captura y reproducción de audio exclusivamente al usar sox.

-p <protocol> Asigna el protocolo de comunicación, este puede ser "tcp" o "udp", por defecto udp.

-t Cuando esta opción es agregada, el programa imprime por pantalla información útil para depurar.

-h Cuando esta opción es agregada, el programa imprime por pantalla la ayuda y termina.

Consideración de uso: Cuando se quiere tener una comunicación full duplex usando D-Light VoIP, se tiene que tener en cuenta que ambos usuarios usen el mismo protocolo de comunicación, el mismo programa de audio para grabar o reproducir y el mismo tamaño "msize".

También hay que tener en cuenta que uno de los dos usuarios iniciara el programa primero y este empezara a mandar paquetes antes de que el otro comience, para evitar esto se usa la opción "-w", así cuando el primer usuario comience el programa, este esperara recibir data desde el segundo usuario, lo que significa que este usuario ya tiene su servidor listo para recibir data también.

Funcionamiento con ALSA.

Codecs y Entrada:

Mandar audio por internet de manera cruda “raw” tiene un costo en terminos de ancho de banda bastante grande, por lo que no resulta factible. Para esto se usan ciertos protocolos para realizar la compresión de estos datos y la posterior decompresión. Aquellos programas que son capaces de comprimir y descomprimir esta información, son llamados Codec, que viene de la composición de “compressor-decompressor” (compresor-decompresor) o mas comúnmente “coder-decoder” (codificador-decodificador).

Los codecs en general son diseñados para enfatizar o correctamente procesar datos de cierta índole, por ejemplo, un video digital de un evento deportivo, necesitara comprimir bien el movimiento, en cambio un video de una exhibición de arte necesitara comprimir el color y las texturas de buena manera.

En audio el caso es tal que para una comunicación por celular por ejemplo, se requiere una baja latencia entre el codificador y la reproducción posterior del audio. En cambio codecs para grabar o hacer una transmisión masiva pueden tener alta latencia a cambio de mayor fidelidad a una baja tasa de transmisión.

Para este caso se utilizo una codificación abreviada iLBC o Internet Low Bitrate Codec, cuya descripción se encuentra en el RFC3951. Esta es adecuada para aplicaciones VoIP como la que se propone en este proyecto. El algoritmo usa una técnica abreviada LPC o Linar Predictive Coding (Codificación de predicción lineal) aplicada a bloques independientes con una elección de marco de datos equivalentes a 20ms y 30ms. En general estos paquetes codificados, son encapsulados en un protocolo adecuado para transporte, usualmente se utiliza RTP (Real-time Transport Protocol). Para efectos de este proyecto, los paquetes son mandados sin este tipo de encapsulamiento, sin embargo se deja propuesta su futura implementación.

Para poder realizar la codificación de manera adecuada, se debió respectar lo estipulado en el RFC3951 con respecto a la entrada al Codec. Este especificaba las siguientes condiciones:

1. Debería estar en formato 16-bit PCM Uniforme.
2. Debería tener frecuencia de muestreo de 8[kHz].
3. Debería estar particionada en bloques de 160 o 240 muestras, representativas de 20 o 30 [ms].

Para este proyecto, se utilizaron bloques de 240 muestras representativas de 30[ms].

Para lograr esto se requirió acceso de bajo nivel al dispositivo de sonido. Esto se logra utilizando las facilidades que entrega el controlador de sonido ALSA para LINUX. Si bien era posible programar esta entrada utilizando la API para C que proporciona ALSA. Se opto por utilizar una aplicación que implementa para el sistema operativo, para captura y reproducción de audio, llamada arecord y aplay, respectivamente.

Controlador ALSA audio:

Una tarjeta de sonido tiene un buffer de hardware que guarda las muestras capturadas. Cuando el buffer esta lo suficientemente lleno, este genera una interrupción. El controlador del kernel de sonido usa DMA para transferir estas muestras a un buffer de la aplicación en memoria. Similarmente para reproducir, otro buffer es llenado y es transferido al buffer de hardware de la tarjeta de sonido usando DMA.

Como se menciono anteriormente, ALSA proporciona 2 aplicaciones en las cuales se puede especificar los parámetros con los cuales la tarjeta de sonido captura y reproduce el audio. En este caso, para lograr lo que requiere el codec, se debió especificar 4 parámetros:

1. Formato de audio crudo “raw”.
2. Formato de audio 16-bit Little Endian.
3. Tiempo del buffer de aplicación 30[ms].
4. Tiempo de muestreo 8000[Hz].

Estos parámetros deberían ser suficientes para lograr el formato adecuado, notar que:

8000*30*10^(-3) = 240 → 240 Bloques de datos requeridos por el codec.

Por lo que los comandos respectivos para realizar este ajuste, corresponden a:

```
arecord -t raw -f S16_LE -r 8000 -B 30000
aplay -t raw -f S16_LE -r 8000 -B 30000
```

Funcionamiento con Sox.

Sox proporciona codecs de audio, de manera nativa, por lo que basta con mandar este audio ya codificado por el software. los comandos correspondientes serian:

rec -r <sample_rate> -p
play -r <sample_rate> -

Para un funcionamiento óptimo se ajusta el buffer para que sea equivalente a una duración de 30[ms]

Programa Principal:

D-Light VoIP tiene un funcionamiento bastante simple, a grandes rasgos los pasos que hace son:

1. Decodifica los parámetros pasados al main.
2. Configura parámetros de conexión.
3. Obtiene la información de la maquina local.
4. Crea un socket y lo une al puerto pasado por parámetro.
5. Crea una nueva hebra que se encargara de todo lo relacionado al servidor de VoIP.
6. La hebra principal espera que la hebra del servidor este en un punto definido como listo.
7. Pregunta por "IP Address" y "Puerto Remoto" en caso de no haberlos pasado previamente por parámetro al main.
8. Obtiene la información de la maquina remota.
9. Crea un socket y une el puerto remoto a este.
10. Crea una instancia del codec y de captura-reproducción de audio en caso de estar trabajando con ALSA. En caso de trabajar con sox, la codificación-decodificación es hecha internamente, por lo que la instancia de audio basta.
11. La hebra actual (main) obtiene datos desde el micrófono y los manda por el socket correspondiente hacia el servidor remoto.

Para la hebra de servidor que fue creada previamente:

1. Crea una instancia de un programa de audio.
2. En caso de estar configurado como TCP, se espera una conexión y se leen datos del socket correspondientes hasta que se cierre la conexión.
3. En caso de estar configurado como UDP, se esperan datos desde el socket correspondiente.
4. En ambos casos al recibir data desde el socket, esta es escrita a la instancia de codificación-decodificación audio para ser reproducida en caso de estar trabajando con ALSA. En caso de trabajar con sox, la codificación-decodificación es hecha internamente, por lo que basta con escribir en una instancia de reproducción de audio.

En la figura 2, se encuentra un diagrama de flujo del programa principal.

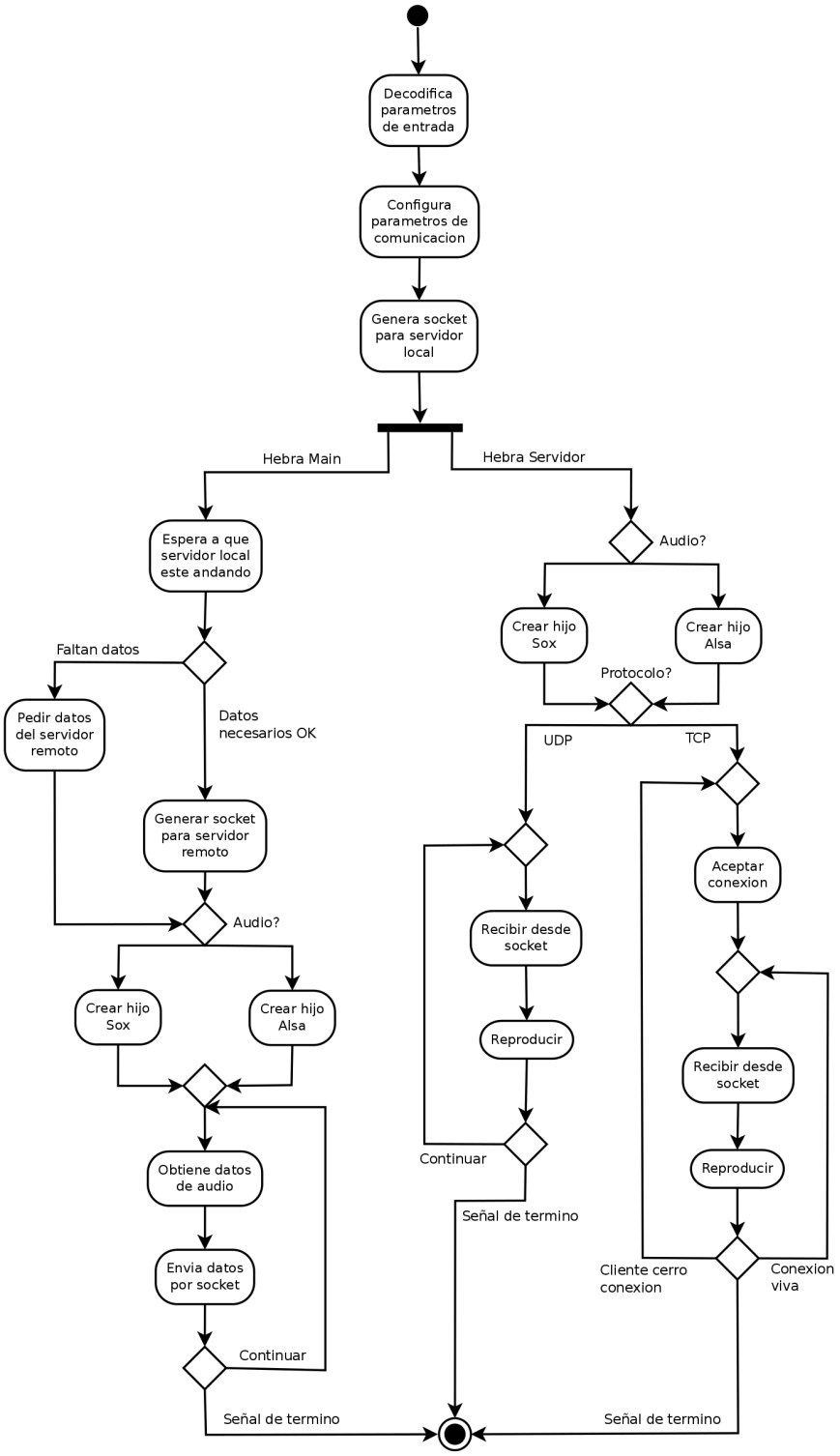


Figura 2: Diagrama de flujo de programa principal.

Herramientas Utilizadas:

GIT

Al desarrollar todo tipo de proyecto en grupo, uno se encuentra con la problemática de que los demás desarrolladores trabajan en el código y este de alguna manera tiene que ser actualizado, esto genera problemas muchas veces ya que dos personas trabajan sobre un mismo proyecto que va tomando diferentes formas, para solucionar esta problemática el equipo de D-Light VoIP opto por usar un controlador de versión llamado "Git".

Git es un control de versión desarrollado por Linus Torvalds, este programa almacena el estado de los archivos fuentes durante el tiempo, pudiendo mostrar diferencias entre un mismo archivo en tiempos diferentes, diferencias entre archivos de maquinas diferentes, puede hacer fusión entre archivos fuentes que han sido actualizado por diferentes personas y una serie de funciones mas.

Git puede ser alojado en un servidor web, de esta forma los desarrolladores del proyecto D-Light VoIP pueden trabajar remotamente sobre los archivos fuentes que están constantemente actualizados.

Para este proyecto se creo un repositorio en github.com y se agrego a todo el equipo como usuarios del repositorio para que pudieran hacer todas las operaciones necesarias sobre los archivos contenidos en el repositorio. Esto fue de gran ayuda para llevar a cabo el proyecto.

Conclusión:

Problemas a Solucionar:

Dentro del marco del proyecto, algunos problemas que faltarían solucionar sería implementar un protocolo de envío de audio mas universal, debido a que ALSA se encuentra presente solo en sistemas UNIX y sox debe estar instalado para poder funcionar.
Si uno quisiera correr esto en un computador con sistema operativo microsoft, la comunicación de audio sería imposible.
Se esperaría también mejorar la transmisión de voz para alcanzar el retardo de una conversación a tiempo real (< 200 [ms]).

A Futuro:

Se espera poder implementar video conferencia con uno o mas destinatarios. Para poder hacer esto se debe indagar en los metodos de compresión temporal y espacial usados por ejemplo en la transmisión de TV Digital o parecidos, para lograr un retardo aceptable, además de variar la calidad de la imagen dependiendo de la calidad del enlace y perdida de paquetes.

Trabajos citados

http://en.wikipedia.org/wiki/Internet_Low_Bit_Rate_Codec
http://en.wikipedia.org/wiki/Real-time_Transport_Protocol
http://rfc-ref.org/RFC-TEXTS/3951/contents.html
http://www.linuxjournal.com/article/6735
http://es.wikipedia.org/wiki/Linear_Predictive_Coding