Trabajo Practico Final

Streaming en vivo desde cámara de celular Android

Materia: Laboratorio de Sistemas Operativos y Redes

Profesor: Jose Luis Di Biase

Integrantes:

Gabriel Gonzalez
Patricio Tella Arena
Juan Pablo Moraes

Contenido

Objetivo:

Transmitir video en vivo desde la cámara de un teléfono móvil con SO Android, usando el protocolo RTSP, utilizando herramientas de Software Libre tanto para el cliente como para el servidor.

Originalmente se iba a utilizar el software FFserver que actuara como servidor para recibir el stream producido por nuestro teléfono móvil, al encontrarnos con varios problemas durante su implementación, decidimos cambiar el software que utilizamos como servidor, eligiendo finalmente el VLC para completar el proyecto y poder mostrar la utilización del protocolo RTSP para realizar streaming en vivo.

Software utilizado:

Utilizamos una serie de softwares libres, entre ellos la colección del proyecto FFmpeg, como así también distintas librerías de codecs libres. Para realizar la transmisión en vivo desde el celular utilizamos una APP llamada RTP Camera.

Introducción:

Protocolo RTP y RTSP:

RTP, es un protocolo de transporte en tiempo real que proporciona las condiciones adecuadas para que aplicaciones que transmiten datos en tiempo real, tales como audio, video, puedan efectuarse.

RTP (y RTSP) han sido diseñados para ser independientes del protocolo de la capa de transporte o capa de red utilizada, por lo cual RTP incluso puede ser utilizado sobre IPV6.

RTP fue creado con los siguientes propósitos:

- Lograr separación entre datos de control de los de datos (audio,video,etc)
- Funcionar independiente del protocolo de la capa transporte o red utilizado
- Escalable
- Seguro al soportar opciones de cifrado

Además RTP posee diversas funciones. Algunas de estas son las siguientes

- Identificación de la fuente
- Segmentación de los datos (generalmente usado UDP)
- Re secuenciación
- Sincronización entre flujos
- Realimentación de la calidad de servicio, para ajustar calidad de envío
- Permite la mezcla de flujos de diferentes partes

FFmpeg

Ffmpeg es un programa sin interfaz gráfica que permite convertir o transformar entre formatos multimedia, tanto de video como de audio. Aunque existen otros programas, algunos sin necesidad de usar comandos, es una de las opciones con más posibilidades y es muy rápida. El paquete viene con tres programas:

1. ffmpeg: ffmpeg es una herramienta en línea de comandos para convertir ficheros de video, flujos de red o la entrada de una capturadora de TV a varios formatos de video.
2. ffserver: es un servidor de flujo para todo lo que ffmpeg pueda usar como entrada (ficheros, flujos, entrada de la capturadora de TV, cámara web, etc)
3. ffplay: es un reproductor de medios muy simple y portable que utiliza las librerías ffmpeg y la librería SDL.

FFserver

FFserver es un complemento anexo del programa FFmpeg, es un servidor de streaming para audio y video que soporta multiples codecs y protocolos, entre ellos RTSP, que es el que vamos a utilizar en nuestro caso.

Instalación:

Para descargar el programa FFserver, del cual se puede leer la documentación en https://ffmpeg.org/**ffserver**.html, se debe copiar el repositorio disponible en GitHub utilizando el comando:

```
git clone git://source.ffmpeg.org/ffmpeg.git ffmpeg
```

una vez descargada esta última versión estable, debemos compilar el repositorio utilizando el comando:

```
./configure; ./make; ./make install
```

Para su uso es necesario conocer acerca de su archivo de configuración:
Se necesita configurar el archivo ffserver.conf que tiene una sección previa antes de configurar los streams de salida del servidor.

```
Port 8090
BindAddress 0.0.0.0
MaxClients 1000
MaxBandwidth 1000
NoDaemon
```

Lo cual determina el puerto usado por ffserver para realizar el streaming, la dirección de bind, la cual es necesaria si tenemos más de una interfaz de red, la máxima cantidad de clientes conectados, el ancho de banda máximo permitido entre todos los clientes y por último si este funcionará como servicio o será iniciado manualmente.

El resto del archivo de configuración tiene dos importantes secciones:

Feed: Cada Feed contiene una secuencia de Video y/o Audio proveniente de la salida de alguna instancia de ffmpeg. Es en sí una sección del archivo de configuración.
Por ejemplo:

```
< Feed feed1.ffm >
File /tmp/feed1.ffm
FileMaxSize 200K
ACL allow 127.0.0.1
< Feed >
```

Stream: Aquí se definen los parámetros de reproducción de los streams provenientes de los archivos previamente codificados por FFmpeg.

Ejemplo :

```
# ASF compatible

< Stream test.asf >
Feed feed1.ffm
Format asf
VideoFrameRate 15
VideoSize 352x240
VideoBitRate 256
VideoBufferSize 40
AudioBitRate 64
< Stream >
```

Por cada Feed pueden existir varios Streams que determinan varios tipos de formatos o de diferentes características.

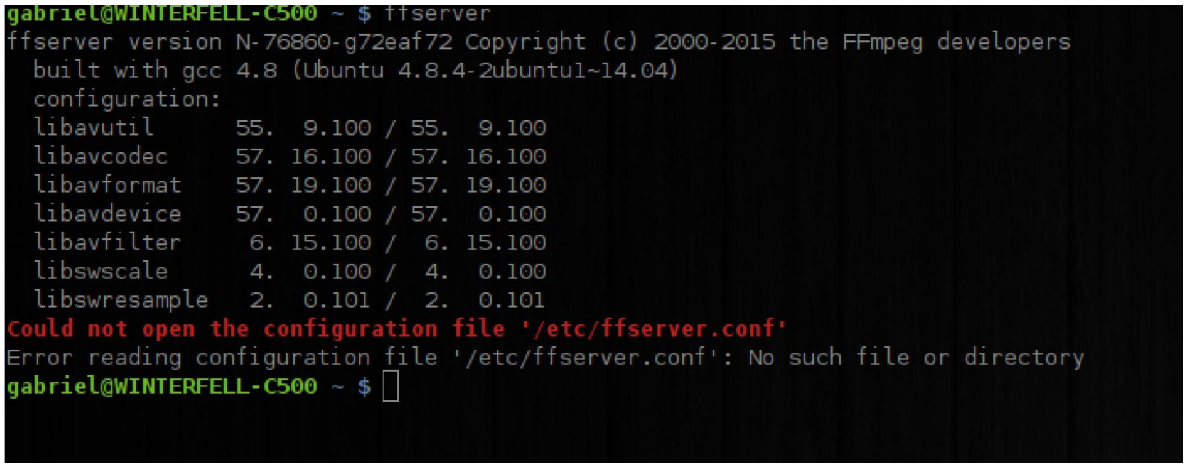
Como iniciar el streaming:

```
./ffserver -f etc/ffserver.conf
```

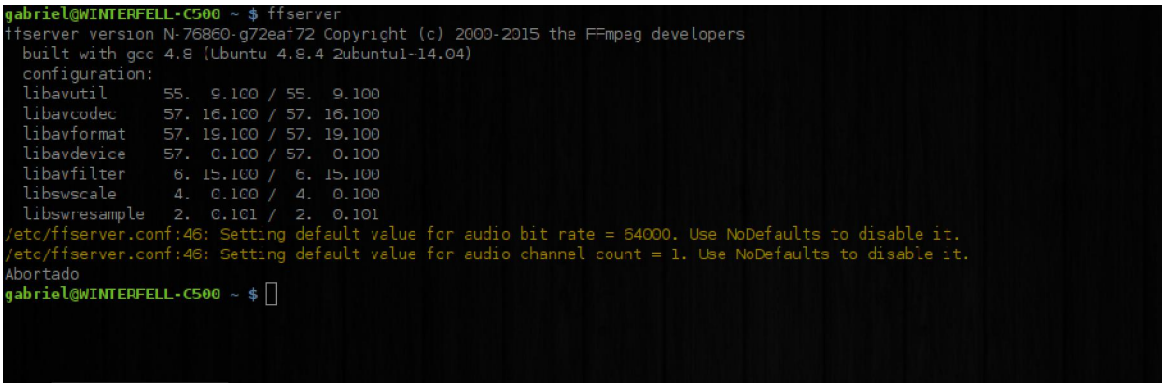
```
./ffmpeg -i INPUTFILE http://localhost:8090/feed1.ffm
```

Problemas:

Al intentar utilizar el FFserver tuvimos varios errores, el primero fue al momento de instalar, ya que FFserver y FFmpeg no se encuentran disponibles para Ubuntu 14.04 (Porgram Deprecated), sino que el mismo recomienda utilizar Libav-tools, por esto debimos descargar el repositorio y compilar el mismo. El siguiente problema fue tratar de hacer funcionar el archivo ffserver.conf



Al intentar ejecutar el FFserver, nos encontramos con algunos de estos mensajes de error o de advertencias.



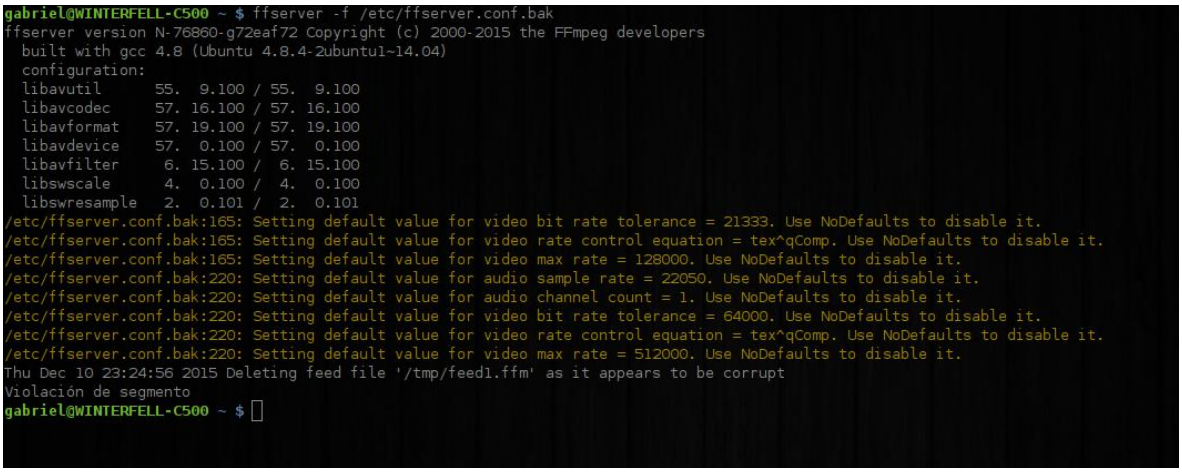
Se debió modificar el archivo de configuración, y utilizar distintos codecs, ya que no todos funcionaban, seguíamos teniendo problemas para ejecutar el servicio debido a que no podíamos utilizar el archivo feed1 el cual funciona como buffer para la transmisión.

Una solución a este inconveniente fue darle permisos de escritura, ejecución y lectura con el siguiente comando:

```
chmod 777 /tmp/feed1.ffm
```

De esta forma pudimos levantar el servicio, y seguimos con las pruebas para enviarle al servidor que ya estaba escuchando, el stream deseado para su transmisión.

Descubrimos, que luego de estos pasos, al reiniciar el equipo nos figuraba el siguiente error:



Lo que se debió hacer fue recompilar sin ser root y volver a hacer todos los pasos anteriores.

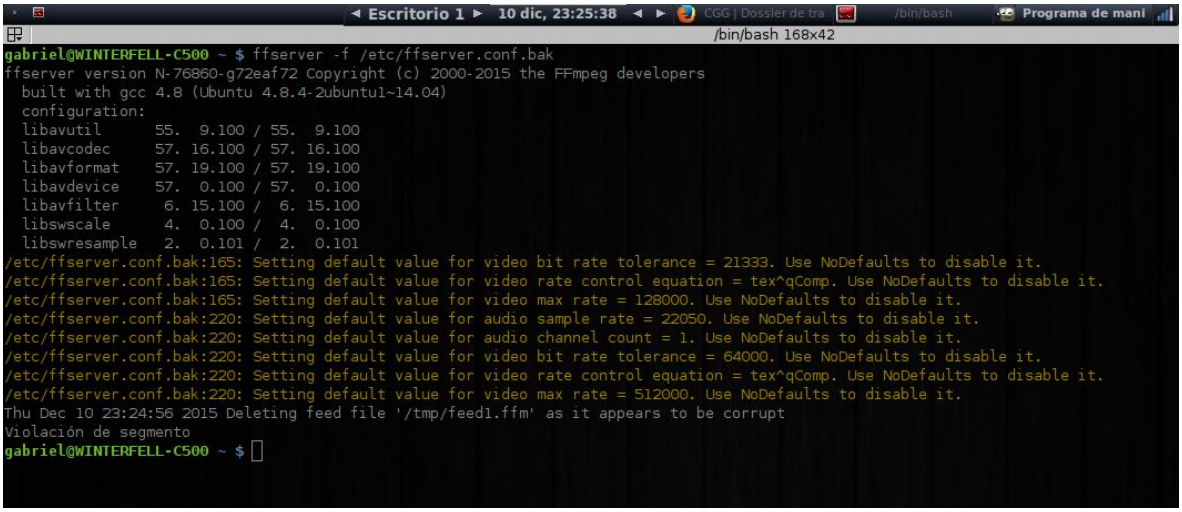
Al tener nuevamente el servicio ejecutándose, intentamos enviar el stream deseado al servidor utilizando el siguiente comando:

```
./ffmpeg -i INPUTFILE http://localhost:8090/feed1.ffm
```

Al realizar esto obtuvimos el siguiente error:

```
HTTP error 404 Not Found

http://xxxxxxx:8090/feed1.ffm: Input/output error
```



Tras varios intentos para transmitir distintos archivos comprimidos con disntos codecs (mpeg, h264, mjepg, etc.)seguimos sin tener éxito y al buscar estos problemas en la red, nos encontramos con varias personas que tenían el mismo problema, sin soluciones por parte de FFmpeg ya que según lo que informan la compatibilidad con RTSP no es 100%

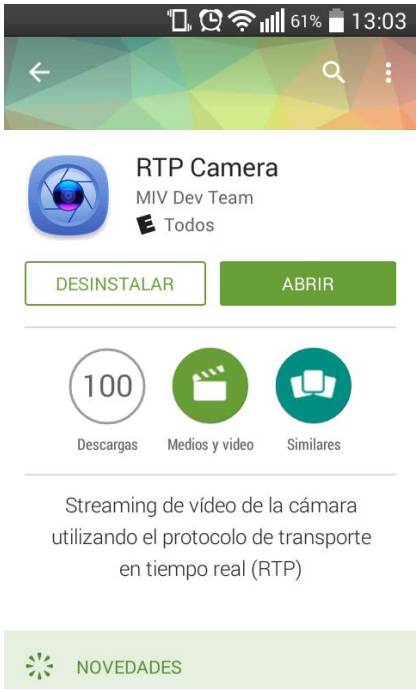
estable y existen muchos problemas con varias librerías de codecs, ya que hay muchas de ellas privativas.

Tratando de lograr nuestro objetivo, decidimos cambiar el software que utilizaríamos para la transmisión de video, y decidimos utilizar VLC que a pesar su fácil implementación comparada a FFserver pudimos lograr el objetivo final para mostrarlo en la presentación.

La diferencia principal es que el FFserver queda funcionando como un servicio y de esta forma “escuchando” para recibir streams, corriendo en segundo plano, en cambio, para hacerlo con el VLC hay que ejecutarlo manualmente.

Ejecución:

Descargar e instalar el RTP Camera desde “Google Play” en nuestro Smartphone



Instalar el VLC Player, cuya documentación se encuentra en la web http://videolan.org

```
Sudo apt-get install vlc
```

Una vez realizados estos pasos, debemos ejecutar la APP en nuestro smartphone, para comenzar a transmitir.

En nuestro servidor ejecutamos el VLC utilizando el siguiente comando:

```
Vlc -vvv http://”IP SMARTPHONE”:8080/camera.sdp --sout
'#standard{access=http,mux=mpeg,dst=”IP LOCAL”:8080}'
```

Reemplazando “IP SMARTPHONE” por la IP que tenga nuestro smartphone en la red e “IP LOCAL” por la IP que tenga nuestra computadora donde estemos ejecutando el VLC.

Una vez realizado esto, simplemente abrimos un navegador web e ingresamos a la ip de nuestro servidor, utilizando el puerto 8080. En caso que estemos dentro de nuestro mismo servidor podríamos utilizar http://127.0.0.1:8080