HDTV sobre IP en Internet2

Tesista: Ing. Alejandro Cabrera Obed

Director: Ing. Luis Marrone

"Tesis presentada para obtener el grado de Magister en Redes de Datos"

"Facultad de Informática - Universidad Nacional de La Plata"

Diciembre 2005

ABSTRACT

La presente es una tesis teórica que pretende ahondar en el análisis de la transmisión de televisión digital de alta definición sin comprimir sobre Internet2, teniendo en cuenta todos los requerimientos de este tipo de tráfico de alto throughput.

Internet2 es un consorcio liderado por muchas universidades del mundo que trabajan en sociedad con industrias y gobiernos para desarrollar tecnologías y aplicaciones avanzadas de red, para luego transferirlas a la Internet del mañana.

Como punto clave a destacar, la tesis plantea el uso de tecnología estándar Internet para el streaming en vivo de HDTV, por lo cual se descarta el uso de ATM (Asyncronous Transfer Mode) y de tecnologías de transmisión específicas para video. Además se estudiarán principalmente temas como la conveniencia de la implementación de IPv4 o IPv6, la aplicación de calidad de servicio (QoS) y el uso de broadcast o multicast.

Cabe destacar que al tener que tomar decisiones justificadas en cuanto a la elección de las diferentes tecnologías a aplicar en el escenario de la tesis, se descartarán otras alternativas que bien podrían ser consideradas como tema para otros trabajos de investigación futuros.

Finalmente, se tomará un experimento de transmisión de HDTV que haya sido realizado con éxito y se lo cotejará con la solución por mí propuesta.

Índice Temático

1.Introducción

En esta última década hemos sido protagonistas de importantes cambios en nuestra vida cotidiana a raíz de la penetración de Internet en nuestros hogares, escuelas, negocios, etc. Particularmente siempre me he preguntado cuáles serían las limitaciones en la transmisión por la red de redes de tráfico demandante de grandes anchos de banda como por ejemplo la televisión con alta calidad de imagen, y además en este caso particular cuáles serían los beneficios para las personas en general en caso de llevarse a cabo esta idea. En los últimos tiempos, con el avance mundial de las pruebas e implementación de la televisión digital de alta definición con todos sus diferentes estándares, me he interesado en saber cómo se podría transmitir la misma por Internet y cómo se comportaría esta red ante una posible demanda masiva.

La HDTV es la variante de DTV (Televisión Digital) con más alta resolución en su imagen combinada con el sistema de audio Dolby Digital Surround Sound y actualmente se transmite por medios convencionales como lo son el aire, los satélites y el cable. Ahora bien, si nos imaginamos por un momento que la misma se podría transmitir por Internet en una modalidad Webcast, esto implicaría algunos cambios en algunos aspectos de nuestras vidas como por ejemplo:

- La gente común podría recibir películas de cine con imagen de alta calidad en tiempo real, distribuidas por las cadenas de TV
- Los productores de cine entregarían sus copias de películas a los teatros o cines por Internet
- Los productores de TV podrían editar las emisiones en vivo desde estudios alejados geográficamente del evento
- Los estudios de filmaciones distribuirían sus películas digitalizadas a los cines y teatros
- Las productoras de video/cine/TV intercambiarían sus productos con sus clientes, las cuales actualmente lo hacen por medio de satélites o transporte aéreo
- Los equipos médicos podrían interactuar entre ellos transmitiendo operaciones o procedimientos con imágenes en tiempo real y de altísima calidad
- Los científicos podrían trabajar en ambientes de colaboración virtual para procedimientos delicados

Con el objeto de estudiar la relación entre Internet y la televisión de extrema alta calidad de imagen y su distribución en una modalidad de tiempo real o muy cercano al mismo, es que me he decidido a investigar

sobre las características y requerimientos de la HDTV. El video con calidad de estudio posee algunas características destacadas como lo son la alta velocidad de transmisión, la sensibilidad a la pérdida de paquetes de datos y la necesidad de llevar a cabo las transferencias de información en tiempo real o muy cercano. El transporte de la transmisión de HDTV sobre una red IP como Internet es significativo. El video necesita ser transmitido todo de una vez para preservar la calidad de la imagen, pero la tecnología IP típicamente quiebra la transmisión en pequeñas partes, transmite las partes de manera independiente y luego las reensambla en el extremo receptor. Esto hace que se requiera transmitir cantidades masivas de datos con un gran ancho de banda y características de tiempo real. Por tales motivos, la Internet convencional no es un medio adecuado para transmitir HDTV, pero sí lo es Internet2 dado que es una red de muy alta velocidad usada en ambientes académicos y de investigación en todo el mundo sobre la cual ya se han plasmado proyectos similares y algunos relacionados a la telemedicina y realidad virtual –entre otros-. La finalidad de los desarrollos en Internet2 es finalmente poder transferirlos algún día a la Internet convencional que cada día requiere más ancho de banda.

Por lo tanto, en una transmisión típica de HDTV por Internet hay dos puntos destacados desde el punto de vista tecnológico: el primero es la capacidad de convertir el video digital en streams IP y la segunda es el desafío de transmitir los streams de HDTV a muy altas velocidades. En este último punto es importante destacar que cuando la HDTV se comprime mediante CODECs de video, se introducen mayores valores de latencia en la transmisión que si la misma se hiciera en modo “crudo” o sin comprimir pero con la consecuente necesidad de un mayor ancho de banda.

2.HDTV: conceptos básicos

2.1.Qué es HDTV – Formatos

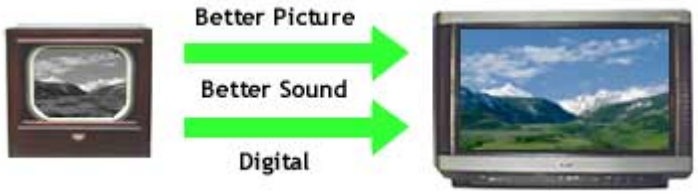
HDTV es la variante de DTV (Televisión Digital) con más alta resolución en su imagen combinada con el sistema de audio de Dolby Digital Surround Sound (AC-3) de cinco canales. HDTV es puramente digital, ya sea en los equipos desde la que se la genera, estaciones de transmisión, su transmisión propiamente dicha y hasta su recepción (aunque en esta última se puedas usar en conversos digital/analógico y seguir usando los receptores de TV analógicos como los actuales).

Una imagen digital no es inherentemente mejor que una imagen analógica, y en algunos casos hasta puede ser peor. Una imagen HDTV tampoco tiene

que ser necesariamente digital; por ejemplo la televisión japonesa se difunde sobre una señal analógica. De hecho, hay diferentes razones para ir hacia la **tecnología de transmisión digital**, que incluyen las siguientes características:

- Mayor cantidad de datos a transmitir, dado que para el mismo ancho de banda se puede transportar mayor cantidad de información en una señal digital que en una señal analógica.
- Mayor consistencia de los datos para ser transmitidos a distancia, comparado con la transmisión analógica, dado que a menso que la señal sea demasiado baja siempre es posible distinguir los 1's y 0's.
- Tipo de señal que puede transportar, que no sólo se remite a audio y video sino también que se introduce el concepto de interactividad.

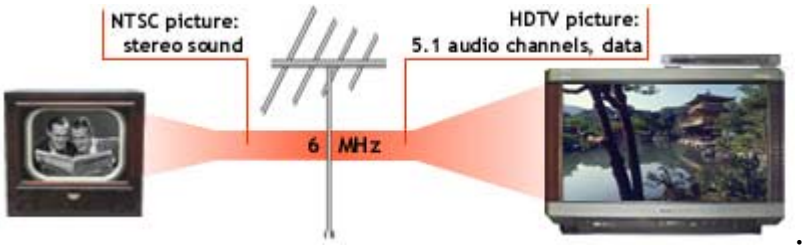
Por lo tanto, la tecnología de transmisión digital combinada con HDTV y sonido digital, esto da como resultado una mejor imagen, mejor sonido y datos en formato digital.



Una de las diferencias más importantes entre la TV digital y la analógica es que al comprimirse la información en formato digital una emisora de TV puede colocar de 6 a 10 canales de programas en el mismo ancho de banda que ocuparía un canal analógico. Además la HDTV provee una imagen entre 5 y 10 veces más clara que la televisión actual, y por supuesto el sonido es de mucho mayor calidad como se mencionó antes.

Una ventaja que la tecnología digital tiene sobre la analógica es que las señales digitales pueden ser comprimidas mejor que las señales analógicas. Para transmitir una imagen de televisión digital, se deben incluir en la señal todos los pixels de dicha imagen. Por ejemplo, la pantalla NTSC estándar incluye 535 líneas por 720 pixels, o sea un total de 378.000 pixels/frame, y esa cantidad de pixels se transmiten por el canal de televisión de 6 MHz de ancho de banda. Una pantalla del estándar ATSC [1] puede tener hasta 1080

líneas por 1920 pixels, lo que representan 2.076.600 pixels/frame, más de 5 veces el ancho de banda de un canal de TV, sin incluir el sonido comprimido ni los datos. Es aquí donde aparece en escena la compresión.



Comparación de la transmisión de una imagen NTSC y HDTV

De los 18 formatos de DTV que existen, 6 son formatos HDTV, 5 de los cuales están basados en escaneos progresivos y 1 en escaneo entrelazado. De los restantes formatos de DTV, 8 son de SDTV (4 formatos de pantalla ancha con una relación de aspecto 16:9 y 4 con formatos convencionales con una relación de aspecto 4:3), y los 4 restantes son formatos "video graphics array" (VGA). Las estaciones de transmisión son libres de elegir en qué formato van a difundir su transmisión.

Aquí están los formatos usados en la televisión digital –definidos por el estándar ATSC A/53- son:

Formato	Formato de imagen	Relación de aspecto	Velocidad de frame
HDTV	1920x1080	16:9	30I, 30P, 24P
HDTV	1280x720	16:9	60P, 30P, 24P
SDTV	704x480	16:9	30I, 60P, 30P, 24P
SDTV	640x480	4:3	30I, 60P, 30P, 24P

ATSC son las siglas de Advanced Television System Comittee [1], y es un organismo gubernamental de EE.UU. encargado de definir los estándares para los nuevos sistemas de televisión digital.

Progresivo (P) y entrelazado (I) se refieren al sistema de escaneo de la imagen.

Como se mencionó antes, las estaciones de transmisión están obligadas a comprimir el detalle de las imágenes y la alta calidad de sonido en el mismo ancho de banda de 6 MHz usado por la TV analógica y para ellos hace falta usar CODEC's MPEG-2.

Resumiendo, la televisión digital tiene las características principales siguientes:

- Imágenes más amplias
- Mucho mayor detalle en las imágenes
- 5.1 (5 + 1 canales = 6) canales independientes CD-quality Dolby Digital (AC-3) surround sound
- Capacidad de enviar datos directamente a una pantalla o una PC
- Formato de pantalla 4:3 o 16:9 (relación ancho por alto)

Los sistemas de transmisión de HDTV usan la técnica de banda lateral vestigial (VSB) de 8 niveles que usa canales de UHF. Del lado del receptor se necesita una antena UHF para recibir la señal HDTV "sobre el aire". En la recepción, uno ve la imagen o no, sin términos medios como en la TV analógica.

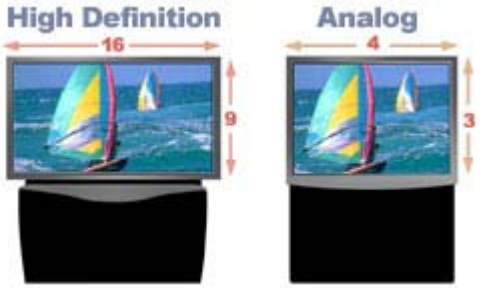
En Internet, HDTV necesita de un gran ancho de banda (mayor que 200 Mbps) y su calidad es igual o superior a la de los DVD. Aunque en su modalidad de Consumer-grade (broadcast) quality esta a 19,2 Mbps y en Contributor quality a 40 Mbps MPEG-2. HDTV es la mayor consumidora de ancho de banda, llegando a consumir aproximadamente 1,5 Gbps en su modalidad sin compresión.

En las redes donde se transmite video y hay pérdida de paquetes, se pueden aplicar técnicas de protección de pérdida de información como FEC (Forward Error Correction) o retransmisión (ARQ, Automatic Repeat Request) para mantener una calidad de video alta hasta incluso en redes con gran pérdida de paquetes. Muchos de los errores se producen a raíz de que la relación señal/ruido (S/N) en la fibra óptica decrece con la distancia.

2.2.Relación de aspecto

En cuanto a la relación de aspecto de 16:9 en HDTV, este formato esta definido así para que las películas calcen su formato lo más preciso posible a cómo fueron diseñadas. En la TV analógica, con relación de aspecto 4:3,

los formatos de las películas deben ser recortados en sus lados izquierdo y derecho y así hay una pérdida del campo visual de la imagen. Otra alternativa que se usa en la TV analógica para mostrar las películas lo más cercano a cómo se diseñaron es poner dos barras negras en las partes superior e inferior para llenar el espacio extra; esta característica se llama “letterbox”.



2.3.Líneas de barrido horizontales

Las líneas de barrido hacen referencia a las porciones horizontales de una imagen de TV que es escaneada sobre la pantalla, de a una línea por vez, para formar una imagen completa. La imagen en una TV analógica está formada por 480 líneas de escaneo horizontales que son pintadas y repintadas en forma rápida y sucesiva, creando la apariencia de un movimiento fluido. En verdad en la TV analógica hay más de 500 líneas horizontales pero sólo se usan 480 para formar las imágenes.

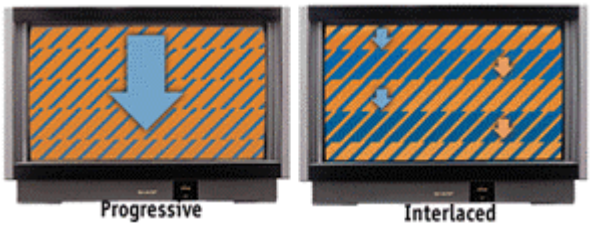
Mientras que la tecnología para utilizar un gran número de líneas horizontales en la pantalla de TV ha existido desde hace décadas, la mayoría del mundo se ha ajustado a los estándares NTSC de hace 50 años atrás. Luego de estos años de espera, la HDTV finalmente provee la habilidad de mostrar imágenes con un mayor número de líneas de escaneo horizontales (720 o 1080).

	Interlaced	Progressive
HDTV	1080 x 1920	720 x 1280
SDTV	480 x 704	480 x 640

2.4.Escaneo progresivo y entrelazado

Estos términos se refieren al sistema de escaneo de la imagen de TV. En el formato entrelazado, la pantalla muestra todas las líneas impares en un escaneo de la pantalla y luego sigue con todas las pares en un segundo escaneo. Cada uno de ellos se llama campos (fields), y se muestran en forma sucesiva. Dado que hay 30 cuadros o frames por segundo, la pantalla muestra una mitad del frame –o sea un field- cada 1/60 de segundo. Para pantallas pequeñas, esto no se hace notar. Pero cuando el tamaño de la pantalla crece, el problema con el entrelazado es el “flicker”.

El formato progresivo muestra la pantalla entera por cada escaneo, las líneas escaneadas se muestran de arriba hacia abajo de un frame completo, lo que trae como consecuencia una imagen más alisada y pareja pero que requiere de un poco más de ancho de banda.



2.5.Pixels

Cada frame en la secuencia de video digital está comprendido por una grilla de pequeños cuadrados de luz coloreada llamados pixels. Pixel es la abreviatura de “Picture Elements”. El término “Resolución” es usado para describir el número de filas y columnas que contienen a estos pixels. El nivel de detalle potencial en una pantalla de HDTV está determinado por el número de líneas de escaneo horizontal –filas de puntos coloreados de luz y direccionables en forma individual-, los cuales pueden corresponder pero a menudo son mayores que la resolución de la fuente de video. Esto hace que la pantalla de HDTV muestre una imagen más realista y aguda.

A contibniación se nota la diferencia entre una imágen análoga y otra de alta definición:

Imagen análoga
211.000 pixels

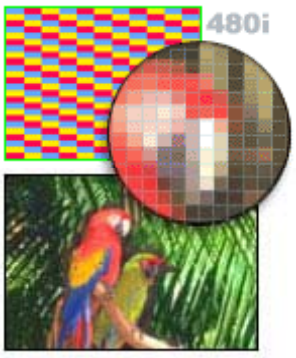
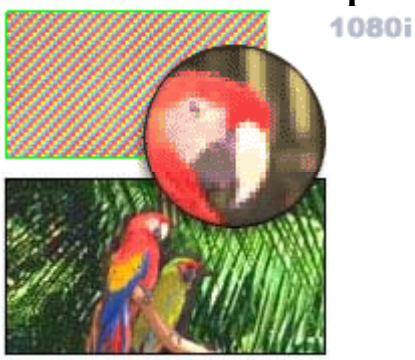


Imagen de alta definición
1.000.000 – 2.000.000 pixels



2.6.Formato de espacio de color

El ojo humano ve los tres colores básicos rojo, verde y azul, y los demás se crean a partir de la combinación de ellos. Además el ojo humano tiene como característica que es más sensible a la luminosidad (cantidad de luz por pixel) que al color. Si centramos la compresión en el color preservando la luminosidad, conseguiremos un primer paso en la reducción del tamaño de la captura sin afectar demasiado la calidad.

HDTV se maneja con el formato de espacio de color YUV, entre otros.

“Y" hace referencia a la luminosidad, mientras que "U" y "V" a la crominancia, o color. Este sistema de compresión suele venir por defecto en un buen número de tarjetas capturadoras sin necesidad de instalar ningún códec y la mayor parte de los CODECs de compresión también lo utilizan como base para compresiones mayores. Hay una gran variedad de formatos YUV, algunos presentan diferencias muy sutiles y otras más importantes. El principal aspecto a tener en cuenta cuando comprimimos en YUV es el "subsampling" o submuestreo.

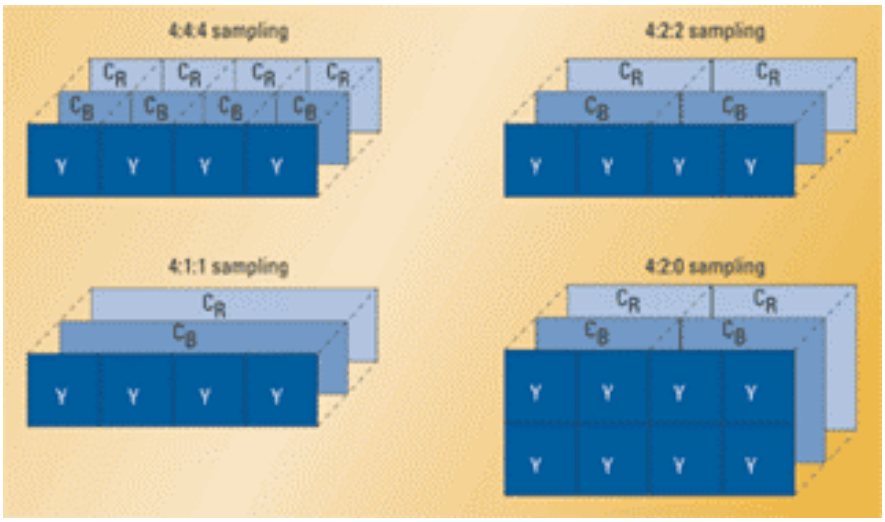
Por lo tanto si el video se tiene que mostrar en dispositivos tales como monitores de computadoras, se debe realizar la conversión al formato de espacio de color RGB (Red Green Blue).

Las relaciones entre ambos formatos son las siguientes:

Y = 0.299R + 0.587G + 0.114B (Luminancia o Brillo)
U = B - Y (Crominancia 1)
V = R - Y (Crominancia 2)

Cada línea de escaneo comprende un número entero de pixels. Cada pixel está representado por un número de muestras. Las muestras se pueden codificar con valores de 8, 10, 12 y 16 bits. En el caso de YUV, una muestra puede representar un componente de color o un componente de luminancia del video. Se pueden compartir muestras de color entre pixels adyacentes. El hecho de compartir muestras de color entre pixels adyacentes se lo conoce somo submuestreo (subsampling). O sea, el submuestreo consiste en reducir la información de color preservando intacta la luminosidad. Eso se expresa de la siguiente manera:

- 4:4:4 mantiene intacta tanto la información de la luminosidad (primer "4") como la del color (los otros dos "4"'s). O sea, la luminancia y las crominancias son muestreadas a la misma frecuencia. No existe ahorro de ancho de banda.
- 4:2:2 reduce el muestreo del color a la mitad. O sea, las señales de crominancia son muestreadas a la mitad de la frecuencia que las de luminancia en la dirección horizontal.
- 4:1:1 reduce el muestreo de color a la cuarta parte. O sea, las señales de crominancia son muestreadas a un cuarto de frecuencia que las de luminancia en la dirección horizontal.
- 4:2:0 elimina uno de los valores de color dejando el otro valor en la mitad. O sea, las señales de crominancia son muestreadas a la mitad de frecuencia que las de luminancia, tanto en la dirección vertical como en la horizontal.



Para realizar el submuestreo se toman muestras, y para cada muestra se puede usar un cierto número de bits, generalmente 8, 10, 12 o 16. Decir que una submuestra es de 4:2:2 equivale a decir que, si por ejemplo se usan 8 bits por muestra, por cada pixel se usarán 16 bits: 8 bits para la luminosidad, 4 bits (la mitad de 8 bits) para el primer componente de color y 4 bits (la mitad de 8 bits) para el segundo componente de color.

En un televisor la imágen se forma de manera entrelazada entre líneas horizontales. Por lo tanto, el "submuestreo" hace referencia a que para comprimir la imágen tan sólo se utiliza el muestreo vertical, puesto que las líneas son horizontales y se cuentan de "arriba a abajo".

Los formatos YUV más comunes son los siguientes:

- YUV2: Es el más utilizado y equivale al 4:2:2. Se obtiene la luminosidad de cada píxel y el color de cada dos pixels. Se obtiene una calidad muy cercana a la RGB24 con una compresión bastante buena.

- YUV8: Se elimina por completo la información de color dejando tan sólo la luminosidad. Es la famosa escala de grises de 8 bits por píxel y 256 tonos.

- YUV9: La luminosidad se toma de cada píxel, mientras que para el color se obtiene un valor medio de una matriz de 3x3 píxels

- BTYUV: Es equivalente a 4:1:1. La luminosidad de cada píxel se conserva. Para el color, se agrupan 4 píxeles por cada línea y se obtiene la media.

- YUV12: Este formato es muy empleado en compresión MPEG y explica el porqué de la frecuente pixelación con flujos de datos bajos, ya que se obtiene un valor medio de luminosidad y color por cada matriz de 2x2 píxeles

Hay muchas más variedades y en numerosas ocasiones un mismo tipo de YUV recibe nombres distintos. De hecho, la compresión YUV puede llamarse YCC, YCbCr, YPbPr o YIQ atendiendo a diversas consideraciones en el tratamiento del color.

En el caso de HDTV 1080i y 720p, se usan 10 bits u 8 bits para representar las muestras de componente de color, según los diferentes formatos de los estándares de la Society of Motion Pictures and Televisión Engineers (SMPTE) [2].

2.7.CODECs de Video

La palabra CODEC viene de COdificador / DECodificador. El codificador de video comprime (codifica) los datos de video eliminando la información superflua o duplicada resultando un archivo que puede ser almacenado o transmitido por streaming a través de equipos de bajo costo de banda ancha. El decodificador descomprime (decodifica) los datos de video habilitando el playback del video. Todo esto envuelve muchos algoritmos muy complejos, que lo son más aún si la velocidad de transmisión es baja y se necesita altas relaciones de compresión de video.

Hay actualmente dos principales organizaciones estándares de video CODECs: la ITU (International Telecommunication Union) [3] y la MPEG (Movie Picture Expert Group) [4]. La mayoría de los estándares intrenacionales fueron escritos por estos dos grupos y los principales estándares fueron aprobados por ambos dos (por ej., MPEG-1, MPEG-2 y MPEG-2 part 10 conocido como H.264). Además hay estándares propietarios como ser DivX (basado en MPEG-4 ASP) y Microsoft Media-9 (similar a H.264).

La tabla siguiente compara los estándares de video CODECs dominantes de la industria actual:

CODEC	Objetivo	Calidad típica	Comentarios
MPEG-1 Estándar Internacional 1992	Alcanza video y audio verosímiles en grabación y transmisión a aprox. 1.5 Mbps	Calidad VHS a 1.5 Mbps. Resolución típica SIF	Casi todas las computadoras soportan archivos MPEG-1. Usada típicamente para baja resolución de video, pero puede ser usada para cualquier resolución. Sólo progresivo.
MPEG-2 Estándar Internacional (1993)	Aplicaciones genéricas de alta velocidad incluyendo TV y broadcast	Alta calidad típicamente con resolución Full D1 (broadcast y DVD son MPEG-2) a 8 Mbps en sus comienzos y 2-4 Mbps en la actualidad	Aplicaciones: Digital TV, HDTV, DVD, Digital cable, satélite y terrestre. Es el estándar más común. Soporta entrelazado.
MPEG-4 (1998)	Muy baja velocidad de CODEC, habilitando bajas velocidades de frame, baja resolución.	Típicamente calidad moderada. En velocidades con full-frame, calidadad moderada/alta requiere una velocidad similar a MPEG-2	Entrega de contenido de video digital sobre Internetv y dispositivos wireless. La video conferencia usando H.263 se basa en este estándar.

H.264 Estándar Internacional Emergente (2003)	Todos los requerimientos desde HDTV de alta calidad hasta bajas velocidades de transmisión para redes celulares.	Variedad de velocidades (bit rates). Por ejemplo, Full D1 30 fps calidad DVD a menos e 1 Mbps. CIF a 256 Kbps	Es el estándar más avanzado y se basa en los estándare previos MPEG.

Inicialmente los CODECs eran por hardware y hoy en día son por software debido a que el poder de procesamiento de las computadoras se ha incrementado notablemente. Esta es una ventaja porque es una solución barata y flexible dado que no se tiene que tener el mismo decodificador por hardware del otro extremo para decodificar la señal.

La DTV se basa en un esquema de compresión y codificación conocido como MPEG-2 (Movie Picture Expert Group 2), que reduce la cantidad de información en aproximadamente una relación de 55:1.

En las técnicas de compresión existe un patrón de video que es una "información" 3D con una dimensión horizontal, otra vertical y la tercera temporal. La compresión en la dimensión de tiempo es diferente que la imagen 2D estática, debido a que el patrón de tiempo necesita sólo transmitir los cambios de un frame a otro. Pero usando un frame de base es posible construir continuos frames mediante la transferencia de pequeñas cantidades de datos que describen las diferencias entre dos frames continuos, hasta que un nuevo frame de base deba ser transferido.

La compresión MPEG-2 es una técnica de interpolación para predecir cómo el patrón es creado desde el comienzo hasta el fin. La compresión MPEG está basada en este método y trabaja con tres clases de frame: I-Frame (intra frame), P-Frame (predicted frame) y B-Frame (bidirectional interpolated frame). La compresión 2D JPEG del frame estático es usada para comprimir el frame base (I-Frame). El algoritmo JPEG se basa en el análisis espectral del frame y se focaliza en los principales componentes de frecuencia de la imagen. La calidad de imagen requerida determina la relación de compresión. Cambiando la compresión en la dimensión temporal permite establecer el número de frames interpoladas (B-Frame) y las frames previstas (P-Frame) que son almacenadas entre un frame de base y otro.

MPEG-2 es el estándar en la industria de los videos DVD y algunos de los sistemas satelitales de transmisión broadcast. La compresión reduce la calidad de la imagen tal como se la ve en la cámara digital del estudio. Por otro lado, este formato es muy bueno filtrando los detalles de las imágenes que el ojo humano ignora de todos modos. La calidad de la imagen es muy buena y significativamente mayor que la calidad de la imagen de la TV analógica. Una TV digital decodifica la señal MPEG-2 y la muestra igual que como lo hace el monitor de una computadora, dándole alta resolución y estabilidad.

2.8.Audio de HDTV

En esta sección se brindan algunos detalles de los diferentes formatos de audio hasta llegar al utilizado por la HDTV.

El sonido es una de las partes más importantes en cualquier instalación de cine. En cierta manera, las limitaciones de la imagen bidimensional, pueden olvidarse ante un buen sistema de sonido envolvente, capaz de tridimensionalizar estas imágenes.

Con la llegada del DVD, y desde hace pocos años aunque muy rápidamente, el sonido ha pasado del clásico "estéreo" al inevitable sonido envolvente, pasando por los sistemas de recreación de ambientes sonoros, hasta los más ambiciosos sistemas de codificación y decodificación digital como el 5.1 actual o los más avanzados 6.1 y 7.1. A continuación se detalla la historia de los distintos formatos:

Mono

El Sonido Mono es el sonido que se reproduce por un único canal. Desde el primer momento en que era posible "grabar" sonido, éste se ha grabado en mono, hasta llegados los años 70 en que las televisiones empezaron a emitir en estéreo. En realidad, se ha utilizado el sonido en mono durante muchísimos años, ciertamente hasta que no ha sido posible grabar dos o más pistas de audio sincrónicamente. Actualmente, encontramos sonido mono en algunas de las emisoras de teledifusión analógicas (muchas veces no es necesario emitir en estéreo, pues el contenido de los programas no invita a ello), y en algunos DVD que son digitalizaciones de películas antiguas. El sonido "mono" no puede convertirse en sonido envolvente.

En cuanto a los dispositivos actuales, es fácil encontrar televisores de bajas prestaciones que sólo utilizan un canal (mono) para la restitución del sonido. Asimismo, los amplificadores de alta gama suelen ser "mono" para

ofrecer al usuario la posibilidad de adquirir tantos amplificadores de un sólo canal como desee (si se quiere una instalación de 5.1, se necesitarán hasta 6 amplificadores mono). Este tipo de amplificadores, más caros, eliminan cualquier interferencia entre canales propia de un amplificador multicanal.

Estéreo

El sonido "estéreo" utiliza dos canales "mono" diferentes sincrónicos. En el oído humano, la diferencia entre un sonido "mono" y otro "estéreo" es extremadamente sensible. No debemos confundir "dos canales" con "dos canales independientes". Si los dos canales contienen exactamente la misma información, no indica que sea "estéreo", sino únicamente la reproducción de la misma información vía dos altavoces.

El oído humano es sensible a las mínimas diferencias entre ambos canales en modo "estéreo", lo que ofrece una mayor dimensión del campo sonoro. La separación de ambos oídos permite oír un sonido procedente de una fuente única con diferencias de tiempo sensibles. Gracias a estas diferencias de tiempo somos capaces de identificar la procedencia de esta fuente (más a la derecha o a la izquierda). Por ejemplo, una fuente de sonido (un altavoz) colocado a 30º hacia la derecha enfrente nuestro, hará que el sonido llegue primero a nuestro oído derecho y milésimas de segundo más tarde en el oído izquierdo. Nuestro cerebro será capaz, mediante estas ínfimas diferencias de microsegundo, analizar ambos sonidos y no sólo ofrecernos un sonido convincente sino además dirigir la atención hacia la fuente pues descubrirá cuál debería ser su posición.

Una pregunta que quizá alguien se hará ahora es ¿cómo nuestro cerebro puede recrear una imagen sonora tridimensional a partir únicamente de dos oídos? Es decir, mediante dos oídos sería fácil distinguir si una fuente está en el centro, a la derecha o a la izquierda, e incluso lo alejada que puede estar ... pero, ¿estará delante o detrás? ¿arriba o abajo?. Aunque no seamos conscientes de ello, nuestra cabeza físicamente está en constante movimiento: nunca tenemos la cabeza literalmente quieta. Estos movimientos, a los cuáles estamos extremadamente habituados, permiten tomar decenas de muestras cada segundo de un mismo sonido. De esta manera no se obtiene únicamente una muestra en estéreo, sino decenas de muestras en estéreo desde diferentes ángulos, suficientes para que nuestro cerebro pueda procesar dichos datos rápidamente y ofrecernos una situación casi exacta.

A partir de aquí, es fácil entender cómo es posible recrear artificialmente un campo sonoro mediante dos canales de audio: las diferencias entre ambos canales permitirá recrear un ambiente sonoro mucho más complejo que si se utiliza un único canal. Y así es cuando oímos una orquestra en estéreo, y somos capaces de colocar casi instintivamente los diferentes instrumentos delante de nosotros.

Algunas grabaciones musicales antiguas en mono han sido reeditadas en estéreo. Naturalmente es imposible volver a grabar un concierto o un grupo si, por ejemplo, hace años que están muertos. Una de las maneras de convertir un señal mono en estéreo es realizando pequeños desajustes entre ambos canales, de tal manera que ciertas frecuencias (a voluntad del ingeniero de sonido) se radiarán primero por un canal y milisegundos más tarde por el otro, permitiendo recrear una imagen sonora antes imposible. Naturalmente, los resultados pueden ser a veces algo decepcionantes.

Dolby Stereo

Los primeros cines que decidieron eliminar la palabra "Mono" de sus salas, se basaron en cuatro fuentes de sonido (dos frontales y dos laterales, éstos para dimensionar la imagen). En ese momento no existía ningún formato doméstico capaz de almacenar cuatro canales de audio sincrónicos, por lo que las grabaciones se realizaban en estéreo.

Dolby Laboratories presentó el sistema Dolby Stereo, que permitía grabar 4 canales de audio para su uso en cines con 4 canales de audio. Incorporaba además un sistema de reducción de ruido, pues las grabaciones resultaban de mala calidad al quedar registradas de manera óptica (no magnética) y así permitir un rango dinámico algo mayor. Algunos cines siguen utilizando este formato. La trascripción doméstica del Dolby Stereo es el formato Dolby Surround. Ambos formatos se graban en dominio digital.

Dolby Surround

En realidad, el Dolby Surround es un formato de dos canales de audio (estéreo) pero que incorpora información de audio "escondida" para el canal frontal central y envolventes. Esta información permanece "escondida" cuando se reproducen las pistas utilizando cualquier sistema en estéreo. Es un formato aun válido para transmisiones de TV analógicas o cintas de VHS, pues ofrece un valor añadido sin tener que aumentar el ancho de banda necesario. Además, es un sistema compatible, es decir, si el usuario no dispone de un sistema de decodificación Dolby Pro-Logic, podrá oír perfectamente el programa. Únicamente cuando las dos pistas de

audio pasan a través de un decodificador Dolby Pro-Logic se puede hacer uso de esta información "escondida".

Dolby Pro-Logic

El Dolby Pro-Logic, por un lado recrea información en mono para los canales envolventes, de muy baja calidad, pero que ayudan a recrear una atmósfera más completa al campo sonoro. Un buen decodificador Dolby Pro-Logic no debe permitir acercar la atención del espectador hacia la información del canal envolvente, ya que este sonido no "define", sino que su bajo nivel ayuda a crear un ambiente idóneo. Por otra parte, el sistema es capaz de crear un canal central con todas sus características.

Dolby Pro-Logic II

Después de la introducción del Dolby Pro-Logic a finales de los 80, el sonido en el cine y en el Cine en Casa ha experimentado grandes avances, que ahora se traducen en el Dolby Digital y el dts, por poner sólo un ejemplo. Estos formatos más modernos que trabajan en el dominio digital, han conseguido apartar el sonido envolvente analógico de características como calidad.

De reciente aparición, el diseño del Dolby Pro-Logic II se ha basado en su predecesor Dolby Pro-Logic, pero con la mentalidad puesta en las necesidades del momento, eso es conseguir la restitución más parecida en configuraciones 5.1. El Dolby Pro-Logic II consta de cuatro canales: L, R, central y envolvente. El canal envolvente es un canal simple de ancho de banda limitado, diseñado para ser reproducido por dos altavoces envolventes y recrear ambientes sonoros. Este canal más el canal central son "matrizados" dentro de los dos canales en estéreo (L y R). El nuevo sistema de codificación y decodificación utiliza el mismo mecanismo de matriz que el Dolby Surround, pero con un procesado diferente. Mediante la nueva tecnología, los canales envolventes se tratan de manera totalmente independiente, por lo que más bien trabajan en estéreo que en mono (lo que ofrece mayor sensación sonora a oídos del espectador). Además, las señales decodificadas tienen en cuenta que la mayoría de las instalaciones cuentan con un subgrave, por lo que se ha prestado especial atención a este aspecto ofreciendo un sonido en general más dinámico y realista.

Dolby Digital (AC-3)

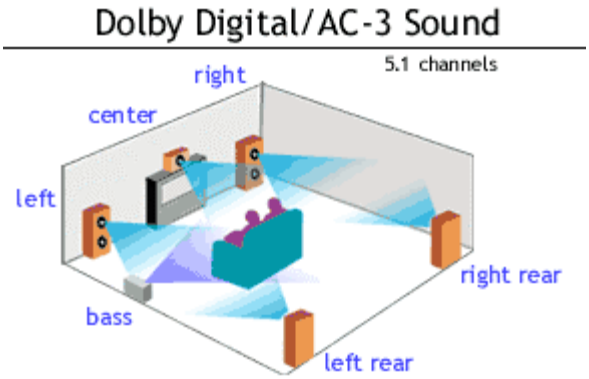
Este es el formato de audio usado en HDTV. El siguiente paso al Dolby Stereo fue el Dolby Spectral Recording (o Dolby SR), que fue sucedido por

el primer formato sobre dominio digital: Dolby SR Digital. En 1992, la película "Batman Returns" fue la primera en incorporar el Dolby Digital, o AC-3 [5]. El AC-3 es un estándar de compresión de audio digital de la ATSC (Advanced television System Commitee). El nombre AC-3 se refiere a que el Dolby Digital es el tercer formato de la generación de codificadores y decodificadores digitales utilizados para su uso en bandas sonoras cinematográficas.

Este estándar tiene las siguientes características:

48 KHz, 44.1 KHz o 32 KHz de frecuencia de muestreo
16-24 bits/muestra
Compresión 2-40:1
5.1 canales digitales
Velocidad máxima de transferencia: 640 Kbps
Velocidad mínima de transferencia: 32 Kbps

El Dolby Digital maneja hasta 6 canales independientes, o canales de audio discretos. Cinco de estos canales son los correspondientes a los diferentes canales que envuelven al espectador: L, R, Central, envolvente L y envolvente R. El sexto canal (en 5.1, sería el 1) tiene un ancho de banda limitado pues sólo incluye frecuencias bajas LFE (Low Frequency Effects). Este canal LFE se utiliza para enviar la información a un altavoz (o varios) subgrave. El sonido de un subgrave independiente es casi inteligible para el oído humano, pero en concordancia con los otros canales ayuda a mejorar el rango dinámico total del espectro audible. Su propiedad de "efecto" la consigue al utilizarse muchas veces como un recurso dramático en escenas de acción.



Dadas sus características técnicas, el Dolby Digital no sólo permite codificar configuraciones 5.1. Por ejemplo puede servir para la codificación de configuraciones de doble estéreo, cuatro canales Dolby envolventes o

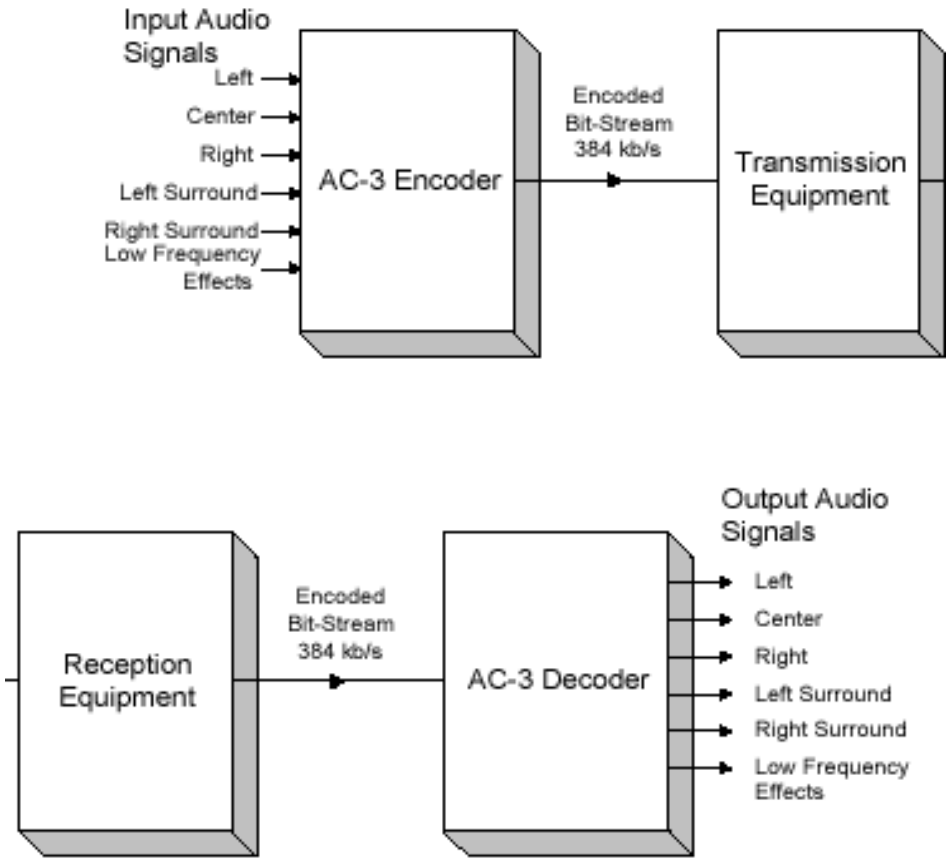
incluso canales mono independientes. La principal diferencia entre el 5.1 y el Dolby Surround es que los canales envolventes trabajan en estéreo y sin limitaciones de ancho de banda.

Un año después de su presentación acompañado de Batman, el Dolby Digital llega al Cine Doméstico. Lo utilizaron diferentes estaciones de TV digital, así como en algunos Laserdisc del momento. Dos años más tarde, en 1995, el DVD escogió el Dolby Digital como el formato para el soporte del audio multicanal.

Cualquier dato digital original puede comprimirse en una relación de compresión adecuada sin que se pierdan datos originales. Para poner un ejemplo práctico, pero para nada real, si obtenemos una información digital como 111110000111, veremos que ocupa 12 bits; en cambio, si decimos 514031 (5 unos, 4 ceros, 3 unos) obtenemos la misma información pero ocupando sólo 6 bits, con lo que conseguimos una relación 2:1 (podemos enviar la misma información con la mitad de espacio). Naturalmente este es un ejemplo práctico y para nada real.

El Dolby Digital debe su éxito a su alta relación de compresión sin comprometer calidad. De esta manera es fácil colocar varias bandas sonoras en diferentes idiomas en un mismo disco DVD, o en el caso del cine, colocar una banda sonora en Dolby Digital y dejar el suficiente espacio para colocar la misma banda sonora en otro formato de audio, normalmente Dolby Stereo.

A continuación se grafican los bloques de transmisión y recepción de audio AC-3 para una velocidad de 384 Kbps:



2.9. Transmisión de HDTV sobre redes IP

2.9.1.Conceptos generales

A pesar que hoy en día se envían grandes cantidades de datos sobre redes propietarias o ATM, se espera que la transmisión sobre redes IP sea el método preferido de entrega, dado que es la tecnología menos costosa y más efectiva para enviar información en forma rápida -incluyendo voz, video y datos- a través de Internet.

La principal ventaja de usar IP como servicio de transporte para audio y video en tiempo real es que nos provee una red convergente y unificada. La misma red puede ser usada tanto para voz, música y audio como para e-mail, web, transferencias de archivos, juegos y mucho más. Y así el resultado es un ahorro importante en implementación, infraestructura, soporte y costos.

Otro beneficio del uso de las redes IP es que proveen multicast, con lo que se puede difundir contenidos multimediales a diferentes grupos de usuarios de manera económica.

En el caso particular de HDTV sin comprimir, se usan generalmente redes propietarias de fibra óptica para enviar el pesado tráfico de video entre dos extremos. Si uno piensa que en un futuro no muy lejano el mismo envío puede hacerse sobre redes IP (Internet), uno fácilmente puede calcular el abaratamiento de costos en instalar, mantener y administrar redes y enlaces privados.

Además, hoy en día se piensa en el envío de todo tipo de información a través de paquetes IP, lo que se conoce como **paquetización de la información**. Y es hacia esa dirección donde apuntan todos los desarrollos tecnológicos de vanguardia.

2.9.2. Interfaces digitales

En las transmisiones sobre IP, las interfaces digitales son la clave para realizar una categorización:

- Studio quality
  - HDSDI
  - SDTI

- Broadcast quality
  - Firewire
  - DVB-ASI
  - ATSC

Estas interfaces de video digital permiten transmitir los formatos existentes de HD, y también los formatos SD. “Studio quality” requerirá mayor ancho de banda que “Broadcast quality”. La principal diferencia entre ellos es que el segundo trata con señales comprimidas, mientras que el primero trata con señales sin comprimir.

Siguiendo la clasificación previa de interfaces, hoy en día existen diferentes sistemas para transmitir HD sobre IP:

Sistema HDSDI (High Definition Serial Data Interface)

Este sistema provee aplicaciones HDTV usando la interface digital HDSDI. La interface HDSDI es una adaptación a HDTV de la interface SDI a 270 Mbps escalando la velocidad (bps) por un factor de 5.5, y alcanzando una velocidad fija de 1.485 Gbps. La interface HDSDI esta estandarizada en el estándar SMPTE 292M (descrito más adelante).

Sistema SDTI (Serial Digital Transport Interface)

Este sistema es un medio para transportar datos sobre una interface normal de broadcast de video digital, la SDI (Serial Data Interface). Los datos que son transportados serán típicamente video comprimido (MPEG o DV) pero igualmente pueden ser cualquier otros datos paquetizados, como HD.

Sistema HDV (High Definition Video)

HDV usa la interface digital IEEE 1394 (FireWire) para proveer transporte a los stream MPEG-2 para que sean enviados a través de la red. HDV es un "formato de video de consumidor de alta definición" propuesto por un consorcio de fabricantes, el núcleodel cual estña compuesto por JVC, Sony, Canon y Sharp.

Sistema DVB (Digital Video Broadcasting) /ATSC (Advanced Televisión System Committee)

Estos sistemas son estándares de broadcast de televisión digital originados en Europa y Estados Unidos.

2.10.Tráfico tiempo real y no-tiempo-real

Las aplicaciones en tiempo real (real-time) generalmente pueden definirse como aquellas con características de "playback" (reproducción). En otras palabras, un stream de datos que es empaquetado en la fuente y transportado a través de la red hacia el destino, donde es desempaquetado y reproducido por la aplicación correspondiente del receptor. Mientras que los datos son transmitidos a través de la red, el retardo datos introducido en esos datos en cada punto del camino es inevitable.

La cantidad de **retardo** o **latencia** ("delay") introducido es variable, porque el retardo es la suma acumulativa de los tiempos de transmisión y los tiempos de espera en las colas de los dispositivos de red (donde los tiempos de espera en las colas puede ser altamente variable). Esta variación en el

retardo se denomina **jitter**; el jitter en la señal de tiempo real es lo que debe ser alisado o suavizado por la reproducción. El receptor compensa este jitter almacenando los datos recibidos en **buffers** durante un periodo de tiempo (un **offset delay**) antes de reproducir el stream de datos, en un intento de negar el jitter introducido por la red. El truco está en hacer un buen cálculo del "offset delay", porque teniendo un offset delay mucho más pequeño que el actual nivel de jitter efectivamente se obtiene una señal diferente a la original en tiempo real.

El escenario ideal es tener un mecanismo que pueda calcular y ajustar en forma dinámica el offset delay en respuesta a las fluctuaciones inducidas en el jitter promedio. Una aplicación que puede ajustar su offset delay es llamada una aplicación "playback adaptable". El rasgo predominante de una aplicación de tiempo real es que no espera por el arribo retrasado de los paquetes cuando reproduce la señal de datos en el receptor; simplemente impone un offset delay anterior al procesamiento.

Opuestas a las aplicaciones de tiempo real, se hallan las aplicaciones elásticas, las cuales siempre esperan el arribo de los paquetes antes de procesar los datos. Por ejemplo: Telnet, FTP y SMTP.

En el caso particular del streaming de HDTV, se lo puede considerar una aplicación de tiempo real pero no tan crítica como lo puede ser la videoconferencia o la telemedecina que tiene exigencias extremas de delay y jitter.

2.11. Introducción al video streaming en red

"Streaming" en el contexto de las comunicaciones significa que el contenido de audio/video es transmitido mientras esta siendo creado, y convertido en el extremo receptor a video y audio continuo. Es decir, es una aplicación de tiempo real.

Hay dos modalidades principales de aplicaciones de video en red:

1) **Video Interactivo:** acá lo importante es que la ***latencia sea pequeña*** para mantener una aceptable interactividad. Se usa en telepresencia (videoconferencia), educación a distancia, operaciones médicas y aplicaciones científicas. El rango de velocidad varía según el escenario usado (tecnología y CODEC) y las expectativas del usuario, y va de unos cientos de Kbps hasta 1.5 Gbps en telepresencia con HDTV.

2) **Video Streaming:** es un servicio donde la interactividad no es la principal característica, y ejemplos de ello son la recuperación de noticias, transmisión de video en vivo, webcasting, Pay-TV, Video On Demand, producciones de estudio basadas en red y HDTV, y en general acceso continuo a contenidos de video en vivo o almacenado para entretenimiento, educación y otros propósitos. Las expectativas del usuario y los escenarios de uso de las aplicaciones varían según la calidad y el throughput deseados:

- Streaming de contenido grabado/vivo con modesta calidad (Real Video, Video for Windows Microsoft Advanced Streaming Format (Microsoft ASF), QuickTime)- actualmente con pocas expectativas de throughput (desde unas decenas de Kbps hasta 1 Mbps)

- Video On Demand (VoD) con calidad igual o mejor que VHS. Se usan los CODECs MPEG-1 y 2. La reciente evolución de MPEG-4 ofrece igual calidad a velocidades de transmisión menores. Los requerimientos de rango de throughput van desde 1-3 Mbps (MPEG-1) hasta 10 Mbps o mayores (MPEG-2)

- HDTV puede ser usado para aplicaciones extremas de gran calidad como producciones de estudio, broadcast de TV de alta calidad, etc. Dependiendo de la compresión usada, el throughput varía desde 19.2 Mbps hasta 1.5 Gbps (raw HDTV)

Por lo tanto, el caso puntual de transmisión de HDTV como video streaming esta encuadrado en esta segunda categoría.

En este punto vale la pena definir al **throughput** como la cantidad de datos transmitidos en forma exitosa desde un lugar a otro en un dado periodo de tiempo; sería la velocidad efectiva de transmisión de datos.

La calidad de video final es una función compleja que depende de diversos factores: la disponibilidad de velocidad del canal determina la calidad de codificación, la velocidad de los frames y la resolución de la imagen de video.

La calidad del video final transmitido a través de una red depende de: variabilidad del ancho de banda, incremento del retardo, jitter y pérdida de paquetes.

Si se compara “calidad de imagen vs. velocidad de transmisión” se observará que la calidad aumenta si aumenta la velocidad de transmisión

(throughput) pero llega un punto de saturación que a pesar que siga aumentando la velocidad la calidad permanece invariable.

En la Internet convencional y con conexiones estándar tipo modem, DSL o cable modem, los reproductores ("players") de video tienen una opción para ajustar la calidad de imagen por medio de la cual el usuario puede elegir ver con mayor calidad de imagen (con la consecuente pérdida de información en el medio y así por ejemplo si el video tiene mucha movilidad en sus imágenes se ve entrecortado) o con mayor velocidad (o sea mayor cantidad de frames por segundo y así la calidad de la imagen estará degradada pero se preserva casi con fidelidad la movilidad de las imágenes del video).

Los efectos de la pérdida de paquetes dependen de un número de factores, entre los cuales están:

- La técnica de compresión usada y la selección de los parámetros de codificación, por ejemplo la relación de compresión
- La velocidad de pérdida de datos
- El patrón de pérdidas
- El tamaño de paquete de datos

El impacto de la pérdida de un paquete es mayor cuanto mayor es el tamaño del paquete. La pérdida de paquetes se incrementa cuando la velocidad de transmisión aumenta.

Cuando se transmite video comprimido, primero se lo codifica y esa codificación introduce un primer nivel de distorsión. Luego, el flujo de bits es transmitido de forma comprimida a través de la red en forma de paquetes y luego, la variación del retardo y la pérdida de paquetes hacen que cierta información no esté disponible para el decodificador. La cuantificación es la principal fuente de distorsión de la codificación.

En la transmisión de video por una red, la pérdida de paquetes hace que no estén disponibles para el decodificador y el retardo extremo a extremo juega en contra de la aplicación. Por lo tanto, el efecto de ambos es la no-disponibilidad de los datos.

En el caso del video, el mismo puede ser interactivo ("two-way video") o no interactivo ("one-way video").

2.12.HDTV sin comprimir

Existen dos formatos para transmitir HDTV sin comprimir, tal como se los describió en el punto 2.1. La transmisión de HDTV sin comprimir es importante para los casos en que sea necesario contar con un bajo delay para la edición de video online.

La velocidad de transmisión de HDTV sin comprimir se calcula de la siguiente manera, teniendo en cuenta el estándar SMPTE 292M del cual se hablará más adelante:

- Estándar SMPTE-292M
- HDTV 1080i a 30 fps (aclaración: el estándar no usa 60 fps porque sería un throughput muy elevado)
- Pixel shape: square
- 1080 x 1920 = 2.073.600 pixels activos/frame
- 1125 x 2200 = 2.475.000 pixels totales (incluye muestras en los intervalos en blanco verticales y horizontales, dado que la mayoría de los equipos de HDTV tienen 1125 líneas de escaneado totales, con 1080 líneas para la imágen)
- Muestreo de componente de color 4:2:2 con 10 bits por componente (20 bits por pixel) para los pixels totales = 49.500.000 bits/frame
- 30 fps = 1.485.000.000 bits/seg
- **1.485 Gbps**
- El payload activo es de 1.244.160.000 bps

En el caso de mi tesis, me voy a abocar al estudio de la transmisión de HDTV sin comprimir y sus variantes, sobre redes IP de alta velocidad.

3.Internet2: conceptos básicos

3.1.Qúe es Internet2

Internet2 [6] es un consorcio sin fines de lucro que nace en EE.UU., conducido por más de 200 universidades que trabajan en conjunto con la industria y el gobierno para desarrollar aplicaciones y tecnologías avanzadas de redes. Internet2 esta recreando la alianza entre el sector académico, la industria y el gobierno de USA que dieron origen a la Internet actual.

Las metas de Internet2 son: crear una red de alta capacidad para la comunidad académica en los países, desarrollar nuevas aplicaciones de Internet y asegurar la transferencia rápida de los nuevos servicios de red y aplicaciones a la amplia comunidad de Internet.

Internet2 no es una red separada físicamente ni reemplazará a Internet. La meta principal es unir los esfuerzos y recursos de las instituciones académicas, la industria y el gobierno para desarrollar nuevas tecnologías y aplicaciones que luego serán extendidas a la Internet global. Así como las herramientas que utilizamos todos los días en Internet son el resultado de la colaboración y las inversiones previas en el área académica y de investigación, se espera que lo mismo ocurra con las nuevas posibilidades que se desarrollan actualmente.

El backbone de Internet2 está constituido por redes de alta velocidad, fundamentalmente Abilene [7], pero también la Red de Ingeniería e Investigación para la Defensa (DREN) [8], la Red de Ciencias de la Energía del Departamento de Energía de los EE.UU. (ESnet) [9], la Red de Servicios Integrados de la NASA (NISN) [10], la Red de Investigación y Educación de la NASA (NREN) [11] y el vBNS [12].

Este backbone de redes atraviesa el país uniendo 53 redes regionales que están estratégicamente dispersas a lo largo de los EE.UU. Cada red regional provee una conexión conocida como Giga POP que vincula Abilene con sus nodos de acceso. Aproximadamente 150 de estos nodos de acceso están distribuidos a lo largo de los EE.UU.

Cabe mencionar que no se trata simplemente de una Internet con mayores anchos de banda, sino que estamos hablando de nuevas aplicaciones que no se pueden realizar en la actualidad, entre ellas se puede citar por ejemplo, bibliotecas digitales, laboratorios virtuales, educación a distancia y tele inmersión.

Paralelamente a esta iniciativa de los EE.UU. surgen proyectos similares en otras partes del mundo, con los mismos objetivos. Podemos citar entre otras: CANARIE con su CA*net4 [13], APAN [14], NORDUnet [15], TERENA [16], GÉANT [17] y otras.

Los ejemplos paradigmáticos de aplicaciones son: la tele-inmersión, los laboratorios virtuales, las bibliotecas digitales y la educación distribuida.

Tele-inmersión

La tele-inmersión permite a usuarios en sitios geográficamente distribuidos colaborar en tiempo real en un ambiente compartido, simulado, híbrido como si estuvieran en la misma habitación física.

Es lo último en síntesis de tecnologías mediáticas:

- Exploración del ambiente en 3-D,
- Tecnologías proyectiva y de visualización,
- Tecnologías de seguimiento,
- Tecnologías de audio,
- Robótica,

y una red de gran poder. Los requisitos considerables para el sistema de tele-inmersión, como gran ancho de banda, latencia baja y variación de la misma (jitter) también baja, lo hacen una de las aplicaciones más desafiantes de la red. Esta aplicación por lo tanto se considera un proyecto piloto ideal a tener en cuenta por los investigadores de la comunidad de Internet2.

Laboratorio Virtual

Un laboratorio virtual es un ambiente heterogéneo, distribuido para solucionar problemas que permite a un grupo de investigadores situados alrededor del mundo trabajar juntos en un conjunto común de proyectos. Como en cualquier laboratorio, las herramientas y las técnicas son específicas al dominio de la investigación, pero los requisitos básicos de infraestructura se comparten a través de las disciplinas. Aunque está relacionado con algunas de las aplicaciones de la tele-inmersión, el laboratorio virtual no asume a priori la necesidad de un ambiente inmersivo compartido.

Un ejemplo es un sistema del pronóstico de clima que incorpora datos basados en satélites, una gran cantidad de información de sensores, y simulaciones masivas para las predicciones del clima de corto y de mediano alcance. Una variación en esto es la predicción de la calidad del aire a través de un laboratorio virtual que relaciona modelos climáticos con modelos de circulación del océano y la química de la contaminación con sensores de partículas terrestres y aéreos. En este tipo laboratorio, los científicos ambientales podrían sugerir, dadas las condiciones actuales, cuándo cerrar temporalmente ciertos tipos de fabricas para evitar una crisis potencial en la calidad del aire. Los laboratorios virtuales se han propuesto

en muchas otras disciplinas incluyendo biología computacional, radioastronomía, diseño de drogas, y ciencia de materiales.

Educación distribuida

Hay muy poco software educacional de la alta calidad disponible para servir como la base de contenido para la instrucción distribuida. La mayoría del software educacional se ha diseñado para el uso independiente, especialmente el que incorpora sonido, imagen o vídeo. Mucho de esto es dependiente de un sólo sistema operativo. Internet2 es una oportunidad de trabajar en una arquitectura para el desarrollo de aplicaciones para el "learningware" y las aplicaciones relacionados con su distribución y uso en la instrucción distribuida.

Las diferencias trascendentales entre la Internet convencional (o comercial) e Internet2 son:

Calidad de servicio: En la actualidad, los paquetes con información que fluyen por las redes tienen la misma prioridad. De este modo, si alguien esta en una videoconferencia y otros usuarios están transfiriendo archivos de datos, las dos aplicaciones disputan el mismo canal. Es altamente probable que los frames de video no fluyan en forma continua, lo que se traducirá en un congelamiento o al menos una disminución en la calidad de la imagen. Por el contrario en la futura Internet se les puede dar prioridad a los paquetes de vídeo, de forma tal que se garantice que la totalidad los cuadros lleguen a tiempo y los paquetes de un archivo de datos se transmitirán únicamente cuando el canal esta libre.

Un ancho de banda mayor: Una de las particularidades principales de Internet2 es el gran ancho de banda del que podrá disponer, actualmente aun con los mejores recursos se dispone de velocidades del orden del los Megabits/segundo, el futuro promete velocidades de Gigabits/segundo y por qué no de Terabits/segundo.

Bajo retardo: Tanto en aplicaciones de tiempo real como de control remoto y otras que son particularmente dependientes de los tiempos de transmisión es importantísimo reducir estos al mínimo. Un gran ancho de banda, la posibilidad de asignar prioridades a las aplicaciones y técnicas modernas de enrutamiento permiten lograr que el retardo sea del orden de unos pocos milisegundos.

Multicasting: Actualmente cuando se debe enviar un paquete de información a varias personas a la vez, como por ejemplo en transmisiones

en vivo, cada frame es enviado a cada persona por separado lo que naturalmente incrementa mucho el tráfico en la red. Con el advenimiento de Internet2 se podrá utilizar una técnica conocida como multicasting con la cual sólo se envía un paquete con toda la información necesaria para que pueda ser recibido por todas aquellas personas que deban hacerlo.

Con protocolos que permiten autenticar completamente el origen de la información y que certifican la integridad y confiabilidad de la misma en Internet2 se verá muy mejorada la seguridad y privacidad de la red.

3.2.Fibra óptica como medio de transmisión de gran capacidad

El medio de transmisión por excelencia con respecto a la capacidad o velocidad de transmisión es la fibra óptica. La disponibilidad de la misma varía mucho con respecto a la ubicación geográfica de cada área en el mundo.

La transmisión de datos por fibra óptica [18], como todo medio de transmisión, tiene un cuello de botella y un umbral de velocidad de transmisión. Pero es interesante notar que la fibra óptica por sí misma no representa el cuello de botella. La ventana de transmisión es la longitud de onda central de la fuente luminosa que utilizamos para transmitir la información a lo largo de la fibra y comprende además una región de longitudes de onda cercanas donde la pérdida óptica es relativamente baja. Un pulso de luz, a medida que viaja por la fibra, se va ensanchando. Este fenómeno se denomina dispersión del pulso y limita la cantidad de información que se puede transmitir. La utilización de una ventana u otra determinará la atenuación que sufrirá la señal transmitida por kilómetro. Las ventanas de trabajo más corrientes son: primera ventana a 850 nm, segunda ventana a 1300 nm y tercera ventana a 1550 nm. La atenuación es mayor si trabajamos en primera ventana y menor si lo hacemos en tercera. El hecho de que se suela utilizar la primera ventana en la transmisión de una señal es debido al menor costo de las fuentes luminosas utilizadas, al ser tecnológicamente más simple su fabricación. La suma de dichas ventanas de transmisión puede, en teoría, ser usada para transmitir entre 50 y 75 Tbps. Lo que representa el verdadero cuello de botella es la electrónica asociada a ambos extremos de la fibra. Actualmente la tecnología de transmisión desarrollada usa sistemas de transmisión de 10 Gbps en cada cable de fibra óptica. Incluso con el desarrollo de la tecnología Wave División Multiplexing (WDM), la velocidad de transmisión por cable alcanzaría entre los 50 y 1000 Gbps, la cual es todavía dos órdenes de magnitud por debajo de la capacidad de transmisión teórica de la fibra.

En el caso de Internet2, siempre se está tratando de romper records en cuanto a velocidades máximas alcanzadas. Por ejemplo, en el mes de Agosto de 2004, se transfirieron 859 GBytes de datos en menos de 17 minutos a través de casi 16 Km de redes a una velocidad de 6.63 Gbps. Este record fue alcanzado por un equipo conformado por miembros del California Institute of Technology y el CERN, usando el stack de protocolos basados en IPv4.

Como conclusión, se puede decir que la fibra óptica es el medio por excelencia para desarrollar backbones de alta velocidad, como efectivamente los usados actualmente por Internet2. Pero lamentablemente es imposible acercar esta tecnología con semejantes velocidades a los usuarios finales.

3.3.Internet2 en Argentina: RETINA2

RETINA [19] es la Red Teleinformática Académica de Argentina, y cuenta con un proyecto de Internet de alta performance denominado RETINA2 [20]. El objetivo de dicho proyecto es iniciar la conexión de la red RETINA a las redes académicas de alta velocidad de EE.UU. y otros países, mediante un acuerdo con el consorcio Internet2. A su vez, esto permitirá el acceso a dichas redes a las instituciones académicas de nuestro país, tanto del ámbito público como el privado, a través de RETINA.

El proyecto RETINA2 se concretará mediante la participación en el proyecto AmPath [21] que promueve la Florida International University (FIU) junto con la empresa Global Crossing [22], con el objeto de permitir que las redes académicas latinoamericanas tengan acceso a los recursos más avanzados en el campo de las comunicaciones por medio de unnodo de acceso ubicado en Miami y de esta manera permitir a la comunidad académica poner en práctica nuevas aplicaciones que no son posibles en la Internet actual.

La conexión entre Argentina y Florida (EE.UU.) es un DS3 (45 Mbps). DS3 es un servicio de línea privada, dedicada y digital dandole gran capacidad para cualquier combinación de aplicaciones de datos, voz o video. La flexibilidad de DS3 lo hace ideal para usuarios intensivos de datos con grandes aplicaciones en el "backbone" o redes privadas que requieren transporte canalizado. La estructura punto a punto del Servicio DS3 proporciona la economía más grande de escala combinando 28 DS1s o 672 canales separados de 64 Kbps en una sola vía de comunicaciones.

El Servicio DS3 lleva los datos sobre líneas privadas y dedicadas entre ubicaciones o entre un negocio y su intercambiador de portadora. El ancho de banda es de 44.736 Mbps.

Por DS3 se puede transmitir videoconferencia de calidad, imágenes de ingeniería CAD/CAM, transmisión de datos seguros, conexiones host a host y conexiones LAN a LAN.

3.4.Backbone de Internet2: caso Abilene

Internet2 tiene muchas redes que conforman su backbone. Abilene [7] [23] es un backbone con tecnología de avanzada, ubicado en EE.UU., que emplea tecnología de transporte óptico y routers avanzados de alta performance, y es el soporte para el desarrollo y expansión de las nuevas aplicaciones que se desarrollan dentro de la comunidad de Internet2. Abilene conecta los "network aggregation points" regionales, llamados gigaPoPs, para soportar el trabajo de las universidades miembros de Internet2, ya que ellas desarrollan aplicaciones avanzadas de Internet. Abilene complementa a otras redes de investigación de alto rendimiento.

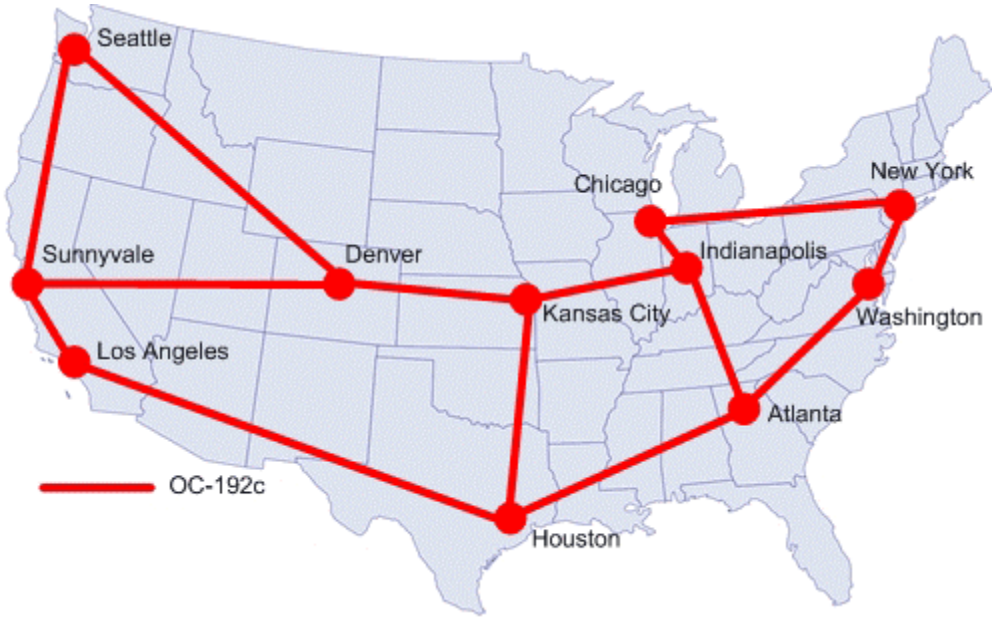
Metas de la Red Abilene

Proporcionar un backbone avanzado de la alta-disponibilidad para satisfacer las demandas de las aplicaciones de investigación avanzada que son desarrolladas por los miembros de UCAID. Para lograr esto, Abilene intentará integrar los servicios de red avanzados que están siendo desarrollados actualmente.

Proporcionar una red separada para permitir probar las capacidades de red avanzadas antes de su introducción en la red de desarrollo de aplicaciones. Se espera que estos servicios incluyan los estándares de calidad del servicio (QoS), multicasting y protocolos avanzados de seguridad y de autenticación.

Proporcionar una red con los recursos necesarios para el estudio de redes, incluyendo diseños alternativos de red capaces de motorizar el avance de la red Abilene y también el estado del arte.

Mapa de Abilene dos años atrás
(con un enlace OC-48c)



Mapa de Abilene actual
(todos sus enlaces a 10 Gbps)

La red Abilene atraviesa el territorio de EE.UU. con una velocidad de backbone de 10 Gbps (OC-192c). Tiene el objetivo de brindar conexiones de 100 Mbps hacia las estaciones de trabajo conectadas a ella. Nuevas conexiones ATM no son ya soportadas, y tampoco se aceptan conexiones

OC-3c (155 Mbps) como regla básica. Abilene soporta tanto redes IPv4 como IPv6 nativas.

SONET es un estándar del Instituto Nacional Americano de Estándares (ANSI) [24] para la transmisión sincrónica de datos sobre medios ópticos. El equivalente internacional de SONET es la Jerarquía Digital Sincrónica (SDH). Juntos aseguran que las redes digitales puedan interconectarse en forma internacional y que los sistemas de transmisión convencionales existentes puedan sacar ventajas de los medios. SONET provee estándares para un número de velocidades de línea hasta la velocidad máxima de línea de 9.953 Gbps. Actualmente son posibles velocidades de línea próximas a 20 Gbps. SONET define una velocidad base de 51.84 Mbps y los múltiplos son conocidos como “Optical Carrier Levels (OCx)”.

Gigabit Ethernet es una tecnología de transmisión de Nivel-2, que opera solamente en full-duplex sobre fibra óptica. Las especificaciones de 10 Gigabit Ethernet están contenidas en estándar IEEE 802.3ae [25] suplementario al 802.3. **Las tecnologías de nivel físico usadas por 10 Gigabit Ethernet incluyen el uso de fibras ópticas monomodo y multimodo, pudiendo ser compatible con SONET STS-192c mediante el encapsulamiento de los frames Ethernet.** También se puede usar sobre “dark fiber” (fibra oscura), que es un cable de fibra óptica que no está en uso y que no está conectado a ningún otro equipo, y está disponible para ser usado; el término “dark” justamente hace referencia a que no se transmiten pulsos de luz por la fibra. En estos casos, son los clientes quienes agregan sus propios equipos a la fibra y tienen así completo conrol sobre su operación.

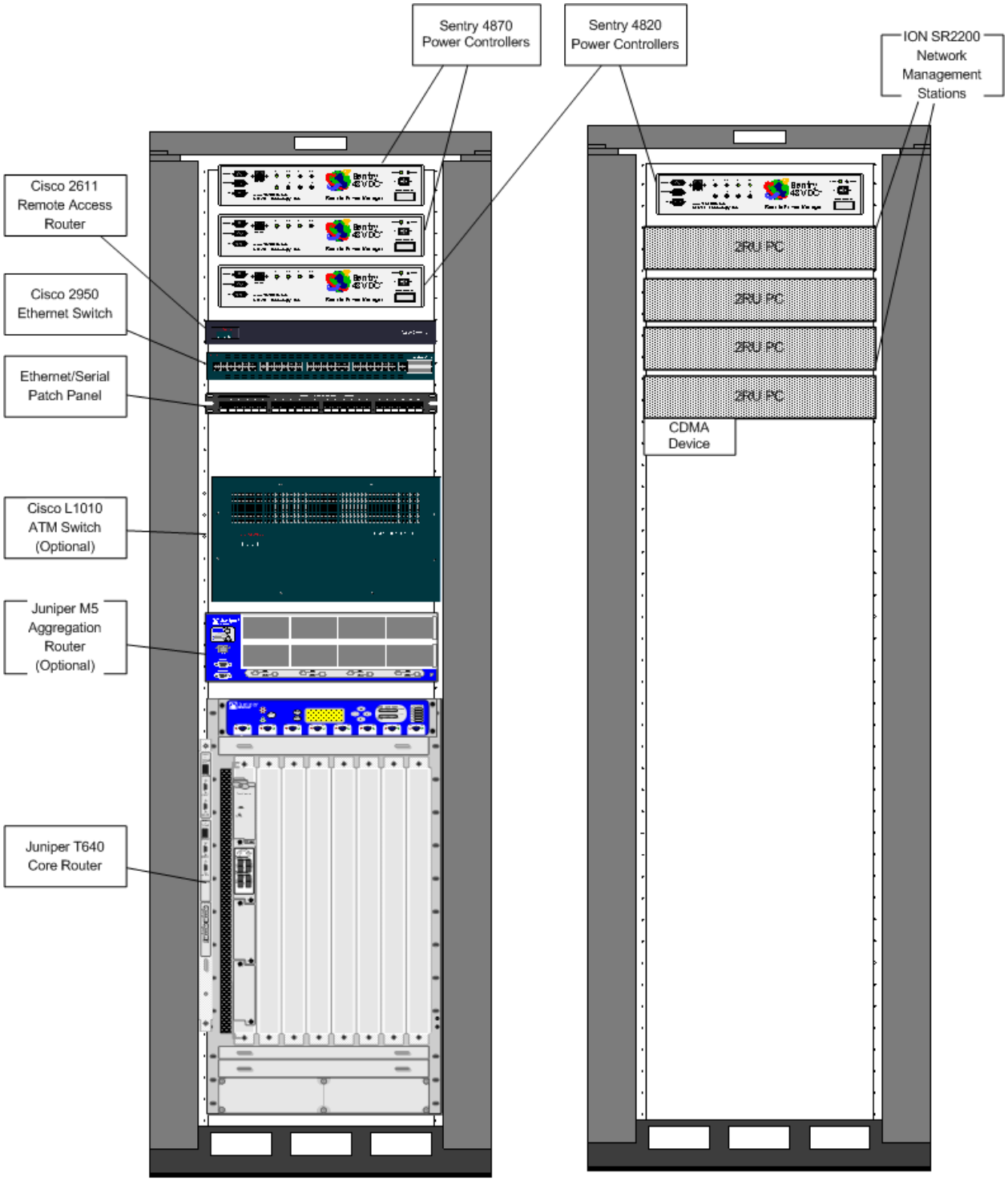
Conectarse a Abilene como uno lo hace habitualmente a Internet mediante un ISP o una red empresarial es imposible, dado que es una red para implementar desarrollos usada por universidades y laboratorios de investigación de Internet2.

El “core router” de Abilene es un router Juniper T640 que soporta velocidades de transferencia 10 Gbps configurado de manera apropiada para cada nodo. La configuración del mismo incluye lo siguiente [26]:

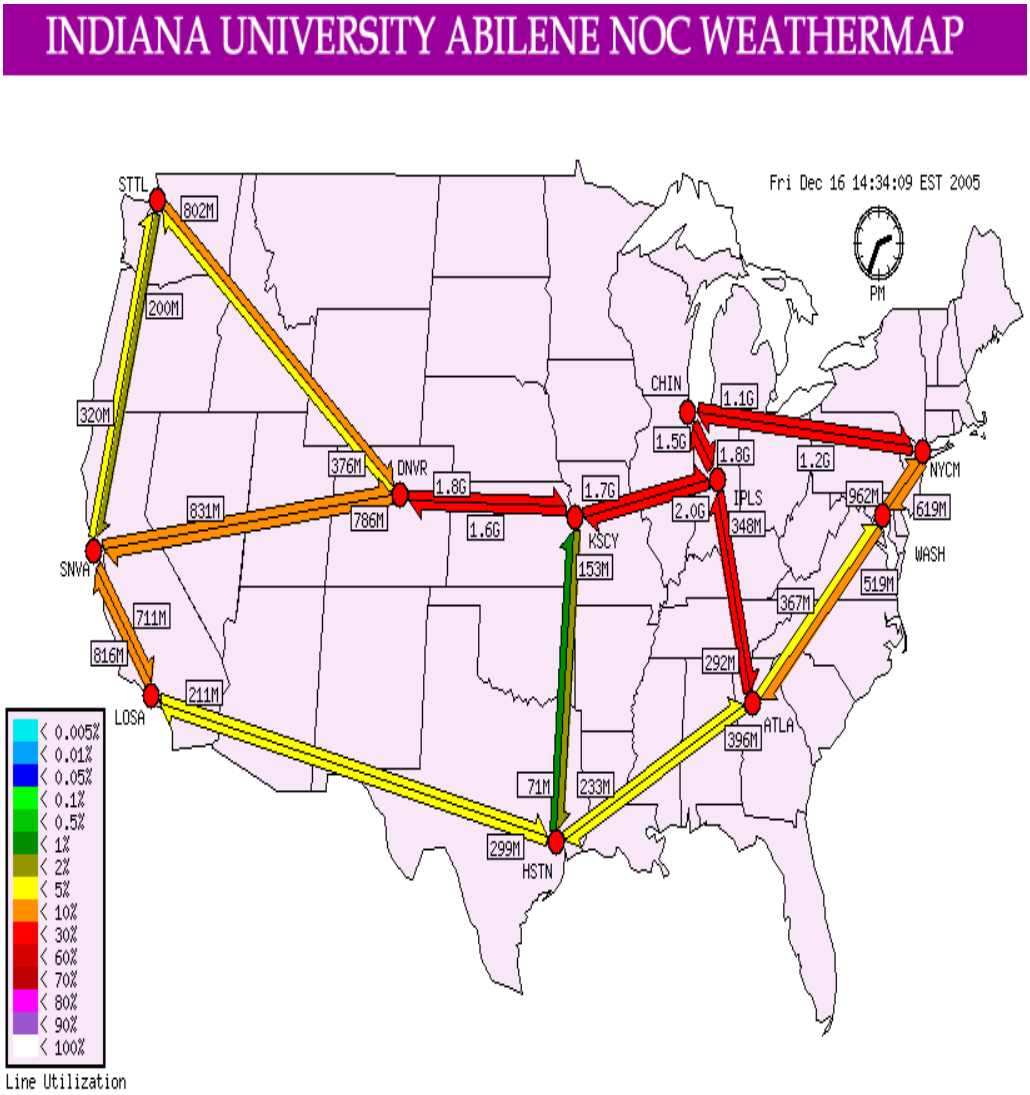
- Un “route engine” o motor de rutas, que mantiene las tablas de ruteo y controla los protocolos de ruteo que corren sobre el router
- Un “control board”, provee control y funciones de monitoreo para el router
- Una “craft interface”, que permite ver el stado y provee información para resolución de problemas

- Un “Conector Interface Panel”, con conectores Ethernet, consola y auxiliares para los “routing engines” y relay de alarmas
- Cinco “switch interface boards”, crea el “switch fabric” para el router
- Un generador de reloj SONET, para las interfaces SONET/SDH del router
- Fuente de alimentación dual de DC
- El router en cuestión no maneja interfaces ATM OC-3 (155.52 Mbps) por lo que debería haber estrategias para que lo pueda hacer

El layout físico del rack donde se montan el router principal y los dispositivos secundarios (access router, consolas snmp, switches Ethernet y eventualmente router o switch ATM) es el siguiente:



A continuación se presenta un mapa del porcentaje de utilización de la red Abilene (Diciembre 2005):



Se encuentran enlaces de los siguientes tipos:

Velocidad del Enlace	**Tecnología1/Tecnología2 (se usa una o la otra)**
OC3 (155 Mbps)	POS/ATM
OC12 (622 Mbps)	POS/ATM
OC48 (2.45 Mbps)	POS/GigE
OC192 (10 Gbps)	Lambda/10GigE

La tecnología Lambda (λ) hace referencia a una longitud de onda de luz totalmente dedicada y que maneja anchos de banda de 1 a 10 Gbps. Obviamente Lambda hace referencia a una tecnología de nivel 1 en el

modelo OSI, sobre la cual puede ir montado un nivel 2 de 10 Gigabit Ethernet.

En la actualidad, como se desprende de un informe específico del sitio web de Abilene [27], **el RTT mínimo promedio de Internet2 está en alrededor de 14-60 mseg**. El delay extremo a extremo se debe principalmente a la velocidad de la luz dentro de la fibra óptica.

3.5.Porcentaje de utilización de los troncales de Abilene

En la actualidad existe un monitoreo de tráfico sobre los enlaces troncales que constituyen el backbone de Abilene. El resultado de dicho monitoreo en el año 2004 [28] muestra diversas estadísticadas tomadas durante las 24 horas a partir del 6 de junio del citado año.

De dicha información se desprende lo siguiente:

- Se detalla el tráfico entrante y saliente hacia/desde Abilene, y el volumen de los mismos (Mbps) es muy similar
- El porcentaje máximo de utilización de un enlace OC-192 llega al 17.3% (aprox. 1,7 Gbps) y se da en el backbone de Chicago
- El porcentaje máximo de utilización de un enlace OC-48 llega al 25.4% (aprox. 635 Mbps) y se da en el enlace MAX

Por lo tanto, se puede decir que Abilene es una red sin problemas de congestión por el momento.

3.6.Tecnologías de conexión a Abilene

La siguiente tabla describe las tecnologías asociadas con las conexiones al backbone Abilene. Detalla tanto los “connector” como los “peers”. “Conectors” se refiere a las redes que se conectan a Abilene en forma directa, y “Peers” se refiere a redes locales o internacionales de investigación y educación que se conectan con Abilene por medio de acuerdos de cooperación. Además en la tabla se incluye la velocidad y el tipo de interface del router Abilene. Además se destacan todos los valores de MTU de todas las interfaces, tanto para IPv4 como para IPv6.

etwork	Router	Speed	Type	Shared	AS	IP	Address	MTU	Multicast
Connectors									
Arizona GigaPoP	Denver	OC-3	SONET	NO	1706	IPv4	192.80.43.21	9000	YES
						IPv6			
Arizona State University	Denver	OC-3	SONET	NO	2900	IPv4	209.147.191.133	9180	YES
						IPv6			
CalREN-2 North	Sunnyvale	OC-12	SONET	NO	11423	IPv4	198.32.249.161	4470	YES
						IPv6			
CalREN-2 South	Los Angeles	10 gbps	Ethernet	NO	2153	IPv4	137.164.25.2	9178	YES
						IPv6	2001:468:ff:144e::2	9000	NO
Drexel University	Washington DC	OC-3	SONET	NO	11834	IPv4	204.238.76.1	4470	NO
						IPv6			
Florida A&M University	Houston	OC-12	ATM	NO	7202	IPv4	192.80.53.58	4470	YES
						IPv6			
Front Range GigaPoP	Denver	OC-12	SONET	NO	14041	IPv4	192.43.217.129	9000	YES
						IPv6	2001:468:ff:445::2	9000	NO
Great Plains Network	Kansas City	OC-12	SONET	NO	11317	IPv4	164.113.238.194	9180	YES
						IPv6	2001:468:ff:1341::2	9180	NO
Indiana GigaPoP	Indianapolis	1 gbps	Ethernet	NO	19782	IPv4	192.12.206.250	1500	YES
						IPv6	2001:468:ff:1244::2	1500	NO
Intermountain GigaPoP	Denver	OC-3	SONET	NO	210	IPv4	205.124.237.9	9000	YES
						IPv6	2001:468:ff:1048::2	9000	NO
Iowa State University	Kansas City	OC-3	SONET	NO	2698	IPv4	192.245.179.249	9000	YES
						IPv6	2001:468:ff:1389::2	9000	NO
Jackson State University	Atlanta	OC-3	ATM	YES	32229	IPv4	192.208.151.13	4470	YES
						IPv6			
LaNet	Houston	OC-3	SONET	NO	13504	IPv4	162.75.0.5	4470	YES
						IPv6			
MAGPI	Washington DC	OC-12	SONET	NO	10466	IPv4	198.32.42.209	9000	YES
						IPv6	2001:468:ff:1858::2	9000	NO
MREN	Chicago	1 gbps	Ethernet	YES	22335	IPv4	198.32.11.98	9000	YES
						IPv6			
Merit	Indianapolis	OC-12	SONET	NO	237	IPv4	192.122.183.9	9000	YES
						IPv6	2001:468:ff:1254::2	9000	NO
Mississippi State University	Atlanta	OC-3	ATM	YES	10546	IPv4	192.208.151.9	4470	YES
						IPv6			
NCNI	Washington DC	OC-48	SONET	NO	81	IPv4	198.86.17.65	9180	YES
						IPv6	2001:468:ff:1855::2	9180	NO
NYSERNet	Chicago	OC-12	SONET	NO	3754	IPv4	199.109.2.1	9180	YES
						IPv6	2001:468:ff:f49::2	9180	NO
NYSERNet	New York City	1 gbps	Ethernet	NO	3754	IPv4	199.109.4.129	9000	YES
						IPv6	2001:468:ff:1549::2	9000	NO
NoX	New York City	OC-48	SONET	NO	10578	IPv4	192.5.89.9	9180	YES
						IPv6	2001:468:ff:1546::2	9180	NO
North Texas GigaPoP	Houston	OC-3	SONET	NO	16905	IPv4	129.110.161.1	9000	YES

						IPv6	2001:468:ff:1157::2	9000	NO
OARnet	Indianapolis	OC-48	SONET	NO	3112	IPv4	192.88.192.133	9180	YES
						IPv6			
OneNet	Kansas City	OC-12	SONET	NO	5078	IPv4	164.58.0.49	9000	YES
						IPv6	2001:468:ff:134a::2	9000	NO
Oregon GigaPoP	Denver	OC-3	SONET	NO	4600	IPv4	198.32.163.13	9180	YES
						IPv6	2001:468:ff:104d::2	9180	NO
Oregon GigaPoP	Sunnyvale	OC-3	SONET	NO	4600	IPv4	198.32.163.9	4470	YES
						IPv6	2001:468:ff:174d::2	4470	NO
Pacific/Northwest GigaPoP	Seattle	10 gbps	Ethernet	NO	101	IPv4	198.107.144.1	9000	YES
						IPv6			
Pittsburgh GigaPoP	Washington DC	OC-48	SONET	NO	5050	IPv4	192.88.115.125	9180	YES
						IPv6	2001:468:ff:1842::2	9180	NO
SoX	Atlanta	OC-48	SONET	NO	10490	IPv4	199.77.193.9	9180	YES
						IPv6	2001:468:ff:e43::2	9180	NO
South Florida GigaPoP	Atlanta	OC-12	SONET	NO	20080	IPv4	198.32.252.253	9180	YES
						IPv6	2001:468:ff:e47::2	9180	NO
Texas GigaPoP	Houston	OC-3	SONET	NO	4557	IPv4	198.32.236.13	9000	YES
						IPv6	2001:468:ff:115a::2	9180	NO
Texas Tech University	Houston	OC-3	SONET	NO	10421	IPv4	129.118.4.85	9000	YES
						IPv6			
Tulane University	Houston	OC-3	SONET	NO	10349	IPv4	129.81.255.1	4470	YES
						IPv6			
UT Southwestern Medical Center	Houston	OC-3	SONET	NO	13998	IPv4	199.165.153.254	9180	NO
						IPv6			
University of Florida	Atlanta	OC-12	SONET	NO	11095	IPv4	192.80.53.46	4470	YES
						IPv6	2001:468:ff:e82::2	4470	NO
University of Hawaii	Seattle	OC-3	SONET	NO	6360	IPv4	205.166.205.218	9000	YES
						IPv6	2001:468:ff:161c::2	9000	NO
University of Louisville	Indianapolis	OC-3	SONET	NO	3714	IPv4	199.120.154.149	4470	YES
						IPv6			
University of Maryland (MAX)	Washington DC	OC-48	SONET	NO	10886	IPv4	206.196.177.1	9180	YES
						IPv6	2001:468:ff:d4c::2	9180	NO
University of Memphis	Kansas City	OC-3	SONET	NO	14048	IPv4	198.32.11.58	4470	YES
						IPv6	2001:468:ff:135b::2	4470	NO
University of Minnesota (Northern Lights)	Indianapolis	OC-12	SONET	NO	57	IPv4	192.42.152.170	9180	YES
						IPv6	2001:468:ff:1259::2	9180	NO
University of Mississippi	Atlanta	OC-3	ATM	YES	25656	IPv4	130.74.3.50	4470	YES
						IPv6			
University of New Mexico	Denver	OC-3	SONET	NO	3912	IPv4	192.65.78.205	9000	YES
						IPv6			
University of South Florida	Atlanta	OC-3	SONET	NO	5661	IPv4	131.247.47.246	4470	YES
						IPv6			
University of Southern Mississippi	Atlanta	OC-3	ATM	YES	16430	IPv4	131.95.1.25	4470	YES
						IPv6			

University of Texas at Austin	Houston	OC-12	SONET	NO	276	IPv4	192.88.12.21	9180	YES
						IPv6			
WiscREN	Chicago	OC-12	SONET	NO	2381	IPv4	205.213.118.1	9180	YES
						IPv6	2001:468:ff:f4f::2	9180	NO
Worcester Polytechnic Institute	New York City	OC-3	SONET	NO	10326	IPv4	130.215.7.1	9180	YES
						IPv6	2001:468:ff:155e::2	9180	NO
Wright-research	Washington DC	10 gbps	Ethernet	YES	668	IPv4	138.18.47.41	9000	NO
						IPv6			
Peers									
6TAP	Chicago	10 gbps	Ethernet	YES	3425	IPv4			
						IPv6	3ffe:3900:a7::1	1482	NO
AARNet (via Pacific Wave)	Seattle	10 gbps	Ethernet	YES	7570	IPv4	198.32.170.45	1500	YES
						IPv6			
ANSP (via AmPath)	Atlanta	OC-12	SONET	YES	1251	IPv4	198.32.252.230	9180	YES
						IPv6			
APAN/TransPAC (via StarLight)	Chicago	10 gbps	Ethernet	YES	7660	IPv4	203.181.249.210	9000	YES
						IPv6			
APAN/TransPAC	Los Angeles	OC-48	SONET	NO	7660	IPv4	203.181.248.130	9180	YES
						IPv6	3ffe:8140:101:1::2	9180	NO
CA*net (via ManLan)	New York City	10 gbps	Ethernet	YES	6509	IPv4	205.189.32.94	9000	YES
						IPv6	2001:410:101:24::1	9174	NO
CA*net (via ManLan)	New York City	10 gbps	Ethernet	YES	6509	IPv4	205.189.32.118	9000	NO
						IPv6	2001:410:101:23::1	9174	NO
CA*net (via Pacific Wave)	Seattle	10 gbps	Ethernet	YES	6509	IPv4	198.32.170.44	1500	YES
						IPv6	2001:468:ff:16c1::3	1500	NO
CA*net (via Pacific Wave)	Seattle	1 gbps	Ethernet	YES	6509	IPv4	198.32.171.44	1500	YES
						IPv6			
CA*net (via StarLight)	Chicago	10 gbps	Ethernet	YES	6509	IPv4	198.32.11.30	9174	YES
						IPv6	2001:410:101:20::1	9174	NO
CA*net (via StarLight)	Chicago	10 gbps	Ethernet	YES	6509	IPv4	205.189.32.98	9174	YES
						IPv6	2001:410:101:21::1	9174	NO
CERN (via StarLight)	Chicago	10 gbps	Ethernet	YES	513	IPv4	192.91.236.198	9000	NO
						IPv6			
CERN (via StarLight)	Chicago	10 gbps	Ethernet	YES	513	IPv4	192.91.246.126	9174	NO
						IPv6			
CERN (via StarLight)	Chicago	10 gbps	Ethernet	YES	513	IPv4			
						IPv6	2001:1458:e100:1000::5:11	9000	NO
CERN (via StarLight)	Chicago	10 gbps	Ethernet	YES	513	IPv4			
						IPv6	2001:1458:e100:1000::5:1	9000	NO
CUDI	Los Angeles	Unknown	Tunnel	NO	18592	IPv4	200.23.60.113	9156	YES
						IPv6	2001:468:ff:14c1::2	4418	NO
DARPA BoSSNet	Washington DC	OC-48	SONET	NO	7082	IPv4	140.173.1.237	9180	YES
						IPv6			
DREN (via Pacific Wave)	Seattle	10 gbps	Ethernet	YES	668	IPv4	198.32.170.37	1500	YES
						IPv6			

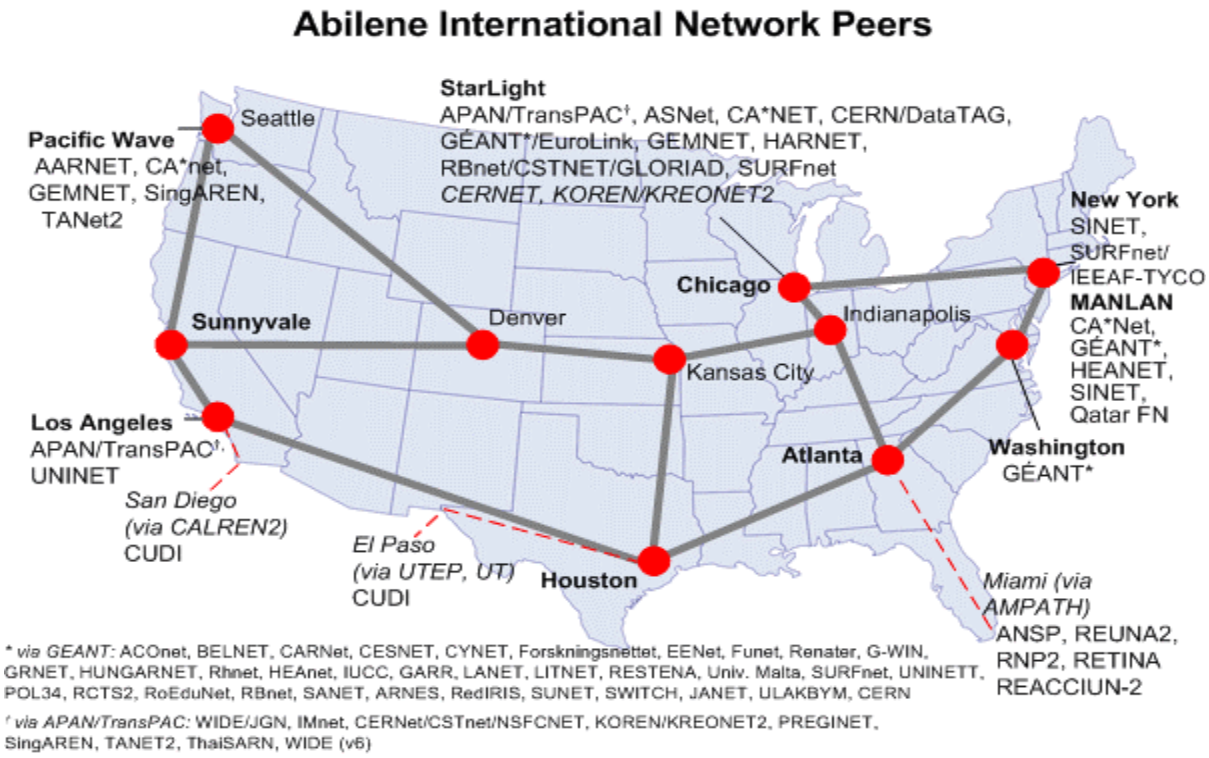
DREN (via StarLight)	Chicago	10 gbps	Ethernet	YES	668	IPv4	138.18.155.13	9000	YES
						IPv6			
DREN (via StarLight)	Chicago	10 gbps	Ethernet	YES	668	IPv4			
						IPv6	2001:468:ff:fc4::2	1500	NO
DREN	Sunnyvale	OC-12	ATM	YES	24	IPv4	192.12.123.197	9180	YES
						IPv6			
DREN	Sunnyvale	OC-12	ATM	YES	668	IPv4	138.18.193.21	4470	YES
						IPv6			
DREN	Sunnyvale	OC-12	ATM	YES	668	IPv4			
						IPv6	2001:468:ff:17c2::2	9180	NO
DREN	Washington DC	10 gbps	Ethernet	YES	668	IPv4	138.18.47.33	9000	YES
						IPv6			
ESnet (via AmPath)	Atlanta	OC-12	SONET	YES	293	IPv4	198.124.216.141	9180	YES
						IPv6			
ESnet	Chicago	OC-48	SONET	NO	293	IPv4	198.125.140.53	9000	YES
						IPv6			
ESnet	New York City	1 gbps	Ethernet	NO	293	IPv4	198.124.216.117	9000	YES
						IPv6			
ESnet	New York City	1 gbps	Ethernet	NO	293	IPv4			
						IPv6	2001:468:ff:15c4::2	9000	NO
ESnet	Sunnyvale	OC-12	ATM	YES	293	IPv4	198.32.11.94	9180	YES
						IPv6			
ESnet	Sunnyvale	OC-48	SONET	NO	293	IPv4	198.129.248.86	9000	YES
						IPv6	2001:468:ff:17c3::2	9000	NO
ESnet	Sunnyvale	OC-12	ATM	YES	293	IPv4			
						IPv6	2001:468:ff:17c1::2	9180	NO
France Telecom (via StarLight)	Chicago	10 gbps	Ethernet	YES	5511	IPv4	198.32.11.14	9000	YES
						IPv6			
GEANT (via ManLan)	New York City	10 gbps	Ethernet	YES	20965	IPv4	198.32.11.62	9000	YES
						IPv6	2001:468:ff:15c3::2	9174	NO
GEANT	Chicago	OC-48	SONET	NO	20965	IPv4	62.40.103.165	9180	YES
						IPv6	2001:798:2022:10aa::d	9180	NO
GEANT	Washington DC	OC-48	SONET	NO	20965	IPv4	62.40.103.253	4470	YES
						IPv6	2001:798:2014:20aa::1	4470	NO
GEMNET (via Pacific Wave)	Seattle	10 gbps	Ethernet	YES	23796	IPv4	198.32.170.35	1500	YES
						IPv6	2001:468:ff:16c1::5	1500	NO
HARNet (via StarLight)	Chicago	10 gbps	Ethernet	YES	3662	IPv4	192.245.196.109	4470	YES
						IPv6	2001:468:ff:12c3::2	4470	NO
HEAnet (via ManLan)	New York City	10 gbps	Ethernet	YES	1213	IPv4	193.1.196.141	9174	YES
						IPv6	2001:770:88:5::1	9174	NO
KOREN/KREONET2 (via Pacific Wave)	Seattle	10 gbps	Ethernet	YES	17579	IPv4	198.32.170.33	1500	YES
						IPv6	2001:468:ff:16c1::6	1500	NO
KOREN/KREONET2 (via StarLight)	Chicago	10 gbps	Ethernet	YES	17579	IPv4	134.75.108.45	9000	YES
						IPv6			
NISN (via StarLight)	Chicago	10 gbps	Ethernet	YES	297	IPv4	192.150.29.5	9000	YES

						IPv6			
NISN	Sunnyvale	OC-12	ATM	YES	297	**IPv4**	192.150.31.5	4470	YES
						IPv6			
NISN	Washington DC	10 gbps	Ethernet	YES	297	**IPv4**	192.84.8.253	9000	YES
						IPv6			
NREN (via StarLight)	Chicago	10 gbps	Ethernet	YES	24	**IPv4**	192.12.123.85	4470	YES
						IPv6	2001:468:ff:fc5::2	4470	NO
NREN	Sunnyvale	OC-12	ATM	YES	24	**IPv4**	192.12.123.201	9180	NO
						IPv6	2001:468:ff:17c4::2	9180	NO
NREN	Washington DC	10 gbps	Ethernet	YES	24	**IPv4**	198.32.11.22	4470	YES
						IPv6	2001:468:ff:18c2::2	4470	NO
Qatar (via ManLan)	New York City	10 gbps	Ethernet	YES	29384	**IPv4**	80.231.134.29	1500	NO
						IPv6			
Qwest Labs	Denver	OC-3	SONET	NO	209	**IPv4**	205.171.4.129	9000	NO
						IPv6	2001:468:ff:10c1::2	9000	NO
RBnet-CSTNET/GLORIAD (via StarLight)	Chicago	10 gbps	Ethernet	YES	5568	**IPv4**	195.209.4.253	1500	YES
						IPv6	2001:6d0:1:3::1	1500	NO
REACCIUN-2 (via AmPath)	Atlanta	OC-12	SONET	YES	20312	**IPv4**	198.32.252.226	9180	NO
						IPv6			
RETINA (via AmPath)	Atlanta	OC-12	SONET	YES	3597	**IPv4**	198.32.252.234	9180	YES
						IPv6			
REUNA2 (via AmPath)	Atlanta	OC-12	SONET	YES	11340	**IPv4**	198.32.252.246	9180	YES
						IPv6	2001:468:700:1804::2	9180	NO
RNP2 (via AmPath)	Atlanta	OC-12	SONET	YES	1916	**IPv4**	198.32.252.238	9180	YES
						IPv6	2001:12f0:0:fe::9	9180	NO
SINET (via ManLan)	New York City	10 gbps	Ethernet	YES	2907	**IPv4**	150.99.200.193	9000	NO
						IPv6			
SINET	New York City	OC-48	SONET	NO	2907	**IPv4**	150.99.199.145	9180	NO
						IPv6			
SURFnet (via StarLight)	Chicago	10 gbps	Ethernet	YES	1103	**IPv4**	145.145.248.129	1500	YES
						IPv6	2001:610:24:8128::1	1500	NO
SURFnet	New York City	OC-48	SONET	NO	1103	**IPv4**	198.32.8.105	9176	YES
						IPv6	2001:468:ff:15c1::2	9176	NO
SURFnet	New York City	OC-12	SONET	NO	1103	**IPv4**	198.32.8.107	9176	YES
						IPv6	2001:468:ff:15c2::2	9176	NO
SingAREN (via Pacific Wave)	Seattle	10 gbps	Ethernet	YES	7610	**IPv4**	198.32.170.47	1500	YES
						IPv6			
SingAREN (via Pacific Wave)	Seattle	1 gbps	Ethernet	YES	7610	**IPv4**	198.32.171.47	1500	YES
						IPv6			
StarLight (via StarLight)	Chicago	10 gbps	Ethernet	YES	10764	**IPv4**	198.32.11.102	9174	YES
						IPv6	2001:468:ff:fc2::2	9174	NO
TANET2 (via Pacific Wave)	Seattle	10 gbps	Ethernet	YES	7539	**IPv4**	198.32.170.34	1500	YES
						IPv6	2001:468:ff:16c1::4	1500	NO
TANET2 (via Pacific Wave)	Seattle	1 gbps	Ethernet	YES	7539	**IPv4**	198.32.171.34	1500	NO
						IPv6			

USGS (via StarLight)	Chicago	10 gbps	Ethernet	YES	1842	IPv4	192.41.213.97	9000	NO
						IPv6			
USGS	Sunnyvale	OC-12	ATM	YES	1842	IPv4	192.41.213.229	4470	NO
						IPv6			
USGS	Washington DC	10 gbps	Ethernet	YES	1842	IPv4	198.187.220.201	4470	NO
						IPv6			
UniNet	Los Angeles	OC-3	SONET	NO	4621	IPv4	63.216.18.54	4470	YES
						IPv6	2001:3c8:e100:1004::5	4470	NO
vBNS	Washington DC	10 gbps	Ethernet	YES	145	IPv4	198.32.11.10	9000	YES
						IPv6			

Se puede observar en esta tabla que las velocidades de conexión van desde OC-3, pasando por OC-12, 1 Gbps, OC-48 hasta llegar al máximo de 10 Gbps; mientras que las tecnologías son ATM, SONET y Ethernet (esta última se usa para las conexiones de 1 Gbps y 10 Gbps).

En el caso de RETINA, se conecta a Abilene en la ciudad de Atlanta (EE.UU.) a través de Ampath, como lo detalla la siguiente figura:



Puntos de peering internacionales de Abilene

3.7.Tipos y porcentajes de tráficos en Internet2

La tabla siguiente [28] [34] detalla los tipos de tráfico usados en Internet2 hoy en día y el porcentaje de utilización cada uno de ellos:

Traffic type	Octets		Packets	
Data Transfers	---	---	---	---
http	15.79%	75.41T	16.89%	103.3G
NNTP	9.98%	47.65T	7.64%	46.72G
FTP	2.91%	13.91T	2.70%	16.53G
Rsync	2.04%	9.739T	1.66%	10.14G
Measurement	---	---	---	---
Iperf	15.64%	74.70T	2.91%	17.81G
ICMP	0.41%	1.955T	4.06%	24.84G
IPMP	0.01%	25.90G	0.06%	359.8M
File Sharing	---	---	---	---
Shoutcast	3.10%	14.79T	4.00%	24.44G
BitTorrent	2.31%	11.04T	2.49%	15.20G
Gnutella	1.21%	5.790T	2.33%	14.24G
Hotline	0.66%	3.163T	0.53%	3.248G
eDonkey2000	0.55%	2.610T	0.58%	3.569G
Audiogalaxy	0.52%	2.488T	0.55%	3.338G
FastTrack	0.02%	76.64G	0.02%	107.8M
Neo-Modus	0.01%	48.82G	0.01%	58.54M
WinMX	0.00%	20.62G	0.00%	29.15M
Carracho	0.00%	10.35G	0.00%	15.93M
Blubster	0.00%	8.677G	0.02%	134.2M
Freenet	0.00%	2.659G	0.00%	3.635M
Direct Connect++	0.00%	1.981M	0.00%	38.38k
Encrypted Traffic	---	---	---	---
SSH	3.86%	18.41T	3.74%	22.86G
HTTPS	0.95%	4.522T	1.34%	8.201G
IPsec ESP	0.06%	298.2G	0.08%	489.2M
IPsec AH	0.01%	34.65G	0.01%	56.50M
IPsec IKE	0.00%	1.745G	0.00%	8.462M

Advanced Apps	---	---	---	---
UNIDATA LDM	3.50%	16.71T	3.59%	21.94G
BBCP	0.19%	906.0G	0.18%	1.094G
IBP	0.07%	352.1G	0.05%	322.5M
McIDAS	0.07%	328.5G	0.06%	350.1M
BBFTP	0.01%	32.79G	0.01%	89.88M
GsiFTP	0.00%	21.65G	0.01%	43.32M
Audio/Video	---	---	---	---
Any-Source Multicast	1.06%	5.041T	0.92%	5.626G
Real Player	0.71%	3.368T	0.65%	3.989G
Windows Media	0.05%	254.8G	0.06%	367.8M
H.323 Signaling	0.02%	72.61G	0.02%	111.1M
Backbone Radio	0.01%	55.27G	0.01%	76.73M
StreamWorks	0.01%	46.72G	0.01%	62.78M
Subset of VoIP	0.00%	7.185G	0.00%	17.75M
Camarades webcams	0.00%	1.513G	0.00%	5.066M
Single-Source Multicast	0.00%	0.000	0.00%	0.000
Misc	---	---	---	---
Mail	0.79%	3.755T	1.08%	6.594G
X11	0.32%	1.509T	0.37%	2.260G
Squid	0.21%	982.3G	0.30%	1.848G
DNS	0.14%	653.7G	0.88%	5.359G
Port 0	0.12%	565.8G	0.08%	502.4M
AFS	0.05%	236.4G	0.09%	566.6M
IRC	0.04%	171.7G	0.07%	445.1M
MS Windows	0.03%	152.2G	0.28%	1.711G
AOL AIM	0.02%	91.56G	0.02%	122.7M
NFS	0.01%	59.72G	0.02%	112.1M
Telnet	0.01%	58.68G	0.04%	246.0M
SOCKS	0.01%	43.09G	0.01%	82.61M
NTP	0.01%	34.60G	0.07%	453.7M
IDENT	0.00%	23.15G	0.01%	76.68M
SNMP	0.00%	4.313G	0.01%	36.90M
RPC Portmapper	0.00%	238.0M	0.00%	2.830M
RTIP	0.00%	31.87M	0.00%	185.2k

Games	---	---	---	---
DirectX	0.54%	2.592T	0.69%	4.242G
Battlenet	0.09%	430.6G	0.15%	892.4M
Half-Life	0.07%	327.1G	0.46%	2.788G
Quake	0.03%	134.0G	0.04%	225.9M
Asheron	0.01%	29.12G	0.01%	52.21M
Starsiege Tribes	0.00%	8.150G	0.00%	21.85M
Spy Arcade	0.00%	5.278G	0.00%	11.58M
Unidentified	---	---	---	---
Unidentified	31.83%	152.0T	38.70%	236.6G
Total	---	---	---	---
Total	100.00%	477.5T	100.00%	611.5G

3.8.Necesidad de transmitir HDTV sobre Internet2

Cada día hay más necesidad de transmitir por Internet contenidos que consumen mayores anchos de banda como el video y el audio integrados, como es el caso de la televisión. En particular, la HDTV puede emplearse en telemedicina, laboratorio virtual y hasta en el mundo del entretenimiento.

Repito que “streaming” en el contexto de las comunicaciones significa que el contenido de audio/video es transmitido mientras esta siendo creado, y convertido en el extremo receptor a video y audio continuo. Uno puede aquí agregar el concepto de “tiempo real”: el tiempo de retardo entre el extremo generador del audio y video (por ejemplo, sonido desde un micrófono y imagen desde una cámara de video) hasta el extremo receptor del otro lado de la línea de comunicaciones, debería ser lo suficientemente pequeño para habilitar una interacción normal en la cual ambos extremos no se verían afectados por dicho retardo.

Las tecnologías de streaming deben diseñarse para hacer frente principalmente al problema de la limitación del ancho de banda. Además, si se piensa en un streaming en forma masiva existe la limitación que todos los usuarios de la Internet convencional no tienen el mismo ancho de banda dado que están conectados a la red de redes por medio de diferentes tecnologías como Modem telefónico, ISDN, ADSL, LAN, etc. Y en el caso particular de HDTV en su modalidad Broadcast, la misma tiene una

necesidad de 19.2 Mbps, velocidad de transferencia muy lejana a las que proveen las tecnologías actuales de conexión a Internet.

Como un simple ejemplo, supongamos el caso de transferir sonido de una computadora a otra vía Internet. Para tomar muestras (samples) de la voz, tomamos como entrada un micrófono conectado a una placa de sonido estándar que usa un solo canal de audio, con una tasa de muestreo (sample rate) de 8 KHz y 8 bits audio. Esto quiere decir que se van a tomar **8000 muestras por segundo y se va a codificar cada una de dichas muestras en 8 bits, y por lo tanto la placa de sonido genera 64000 bps**. Entonces se observa que hay una necesidad de comprimir el audio para que sea transmitido por tecnologías básicas como modem analógico digital. El extremo receptor luego decodificará la información y así alimentará la placa de sonido en forma continua.

En el caso de un archivo de sonido con calidad de sonido mayor como CD-quality por ejemplo, la tasa de muestreo debería haber sido de 44100 Hz, 16 bits por canal. En el caso de tener un solo canal, se hubiesen necesitado 705.600 bps. Es decir, se producen una gran cantidad de datos antes de la etapa de compresión para utilizar el enlace de comunicaciones en forma apropiada.

Cuando se trata de transferir video en vivo (live video), el problema del ancho de banda es más crítico. Por lo tanto se necesitará un factor de compresión mucho mayor para su transferencia y la única manera que se pueda alcanzar la correcta transferencia será llevando a cabo cambios fundamentales en los anchos de banda de los canales de comunicación dado que la compresión se agotará.

En el caso del video, vale la pena aclarar que la resolución del ojo humano es 100 veces mayor aproximadamente a la resolución de una cámara de video y que el ángulo de visión del ojo humano es de por lo menos el doble que el de una cámara.

Hoy en día los sitios web de Internet ofrecen streaming de audio y video según el tipo de conexión (velocidad) del usuario a Internet o Internet2. En general se especifican así:

- Modem: conexiones dialup de 56 Kbps o mayores
- DSL: conexiones de 256 Kbps o mayores, como DSL e ISDN
- Conexiones mayores a 1300 Kbps, como LAN, xDSL, cable modem o T1

- LAN: conexión a Internet2 de 3.5 Mbps o mayores
- MPEG-2: conexión a Internet2 a través de Ethernet de 5.6 Mbps o mayores

Los players de audio y video necesitan un mínimo de almacenamiento de datos en buffer para poder ejecutar el streaming, razón por la cual el ancho de banda juega un papel fundamental dado que si el mismo escasea no se van a poder llenar esos buffers en forma adecuada y la reproducción no será la deseada. Por ejemplo, MPEG-2 Multicast requiere un mínimo de conexión de 3.5 Mbps para funcionar adecuadamente.

Existen algunos players de audio y video que son los más usados por el público de Internet e Internet2 en general, y ellos son:

- Windows Media Player
- QuickTime
- VLC Media Player (conexiones a Internet2 de 3.5 Mbps o mayores)
- VideoCharger (para conexiones a Internet2 de 5.6 Mbps o mayores)

La Internet convencional es una red de transmisión basada en paquetes de datos, pero no es confiable si se requiere una transmisión en tiempo real. Carga muy pesada de tráficos y problemas de transmisión internos pueden ocasionar retardos que pueden quedar fuera de control. Esto puede resultar en una reproducción entrecortada de audio y video del lado del receptor. A medida que Internet crezca en su ancho de banda global, este problema será menos evidente.

Además, la Internet convencional no maneja calidad de servicio (QoS) dado que se basa en el criterio del “best effort” (mejor esfuerzo) para la entrega de los paquetes. La Internet actual no puede garantizar que los paquetes de un dado flujo no se van a perder, ni que van a llegar ordenados ni que van a llegar a tiempo. No se puede predecir hoy en día el comportamiento de los routers de Internet porque no tienen una configuración estándar, y por lo tanto no se puede aplicar QoS extremo a extremo. La calidad de servicio sin agregar protocolos al respecto se puede llevar a cabo en forma específica a través del uso de señalización User-to-Network (UNI) y mecanismos de ruteo Private Network-to-Network Interface (PNNI), ambos de ATM.

Por lo tanto se necesita un escenario de características técnicas superiores a las que ofrece Internet hoy en día para experimentar con aplicaciones que exijan grandes performances de red y grandes transferencias de datos, y así aparece la necesidad de una Internet de nueva generación representada

entre otras por Internet2 para implementar este tipo de pruebas y transferirlas en un futuro a la Internet comercial y masiva.

4.Jumbo Frames

4.1.Jumbo Frames e Internet2

Antes de comenzar a hablar sobre Jumbo Frames o Jumbograms [29] [30] voy a hacer un breve resumen de los distintos PDU (Packet Data Unit) utilizados en Internet/LAN:

En TCP [41], al PDU se lo llama “segmento”
En UDP [42], al PDU se lo llama “datagrama”
En IP [43], al PDU se lo llama “datagrama” o “paquete”
En Ethernet, al PDU se lo llama “frame”

Los frames IP grandes, también llamados “jumbogramas” o “jumbo frames”, han sido recomendados para redes de alto rendimiento como Abilene como un factor significativo para aumentar la performance a través de distancias largas. En general el factor limitante en el tamaño del frame o paquete es el **MTU** (Media Transmission Unit), el cual en Internet es de 1500 Bytes.

El MTU se especifica en octetos y es el tamaño máximo del paquete o frame que se puede enviar por una interface de red (y su correspondiente driver), tal el caso de Internet. TCP utiliza el MTU para determinar y optimizar el tamaño de los paquetes en cualquier transmisión.

La capa de enlace (link layer o nivel 2 en el modelo OSI) es la responsable de descubrir el MTU y reportarlo a los protocolos de niveles superiores.

El MTU se configura en forma independiente para cada interface con un flujo de datos IP, y cualquier interface puede limitar el tamaño del paquete fragmentándolo. El método “Path MTU Discovery” [31], intenta descubrir el MTU óptimo para el camino de manera de evitar la fragmentación de los paquetes. Para cada destino se asume inicialmente el MTU del link del primer salto. El Path MTU Discovery está deshabilitado en la mayoría de los sistemas operativos actuales, y existe un MTU fijo.

Con el Path MTU Discovery existen problemas, documentados en el RFC 2923 [32]. El mecanismo es frágil porque depende del mensaje ICMP “Can’t Fragment” de la red. Cuando hay un “agujero negro” de ICMP los

mensajes no se envían, causando que falle el Path MTU Discovery y así se suspenda la conexión de TCP.

Recordemos que el MTU fue una manera de protegernos de las tasas de errores de bit (bit error rate) en los primeros tiempos de Ethernet, en sus componentes de Nivel 1. Pero actualmente se ha incrementado el poder de procesamiento de las computadoras y se usa una Ethernet switcheada sobre UTP o Fibra Óptica, lo que implica una baja significativa en los errores en Ethernet.

La recomendación de Internet2 es trabajar hacia un IP MTU "extremo-a-extremo" de por lo menos 9000 bytes por ahora, y eventualmente más grande. Con respecto a esto, Abilene actualmente trabaja con valores de IP MTU de 9180 Bytes y 9000 Bytes. Estos valores de MTU se usan tanto para IPv4 como para IPv6.

Vale la pena aclarar que en el párrafo anterior se habla de "IP MTU" y no de MTU, dado que se quiere hacer notar que **en Internet2 se considera al MTU como el tamaño máximo de paquete IP sin fragmentar que puede atravesar una red, sin importar qué tecnologías de framing estén por debajo**. O sea, es deseable trabajar con un valor extremo a extremo de 9000 Bytes de MTU a nivel IP o layer-3. Y por supuesto, **dicho MTU incluye al header IP.**

Abilene y sus conectores trabajan todos en conjunto con dicho tamaño de MTU, y las instituciones dentro de Internet2 configuraron el MTU también a 9 KB dentro de sus redes locales o por lo menos hasta las estaciones de alto rendimiento.

En cuanto a las redes Gigabit Ethernet, las mismas soportan tamaños de frame mayores a 1500 bytes, y esto ha sido tema de debate durante años y la decisión de adoptar o no este criterio afectará la performance de Internet en los próximos años.

Ethernet ha usado tamaños de frame de 1500 bytes desde que fue creada alrededor de 1980. Para mantener la compatibilidad hacia atrás, las redes Fast Ethernet (100 Mbps) y las redes Gigabit Ethernet (1000 Mbps) estándares usan el mismo tamaño de frame, y por lo tanto no hay ninguna fragmentación o reensamblaje de Nivel 2 (modelo OSI). En este punto vale aclarar que, teniendo en cuenta el modo de transmisión half-duplex y con el objeto de detectar las colisiones, en Ethernet y Fast Ethernet el tamaño mínimo de frame es de 64 Bytes, mientras que en Gigabit Ethernet existen tamaños mínimos de frame definidos de acuerdo a si se transmite por fibra

o cable. El estándar 802.3z para transmisión sobre fibra óptica (1000Base-X) tiene un tamaño mínimo de frame de 416 Bytes, mientras que el estándar 802.3ab para transmisión sobre cable de par trenzado (1000Base-T) tiene un tamaño mínimo de frame de 520 Bytes (provee más tiempo para que el dispositivo transmisor reciba una notificación de colisión en el caso de un evento de congestión). En base a ello en Gigabit Ethernet se usa en campo de extensión variable para rellenar (padding) los frames que son más cortos que el tamaño mínimo. Los tamaños mínimos de frame son necesarios para que el transmisor detecte una colisión antes de enviar el último byte del frame; de esta manera los frames deberán ser lo suficientemente largos para llenar completamente el medio y así, si una colisión sucede, el transmisor la detectará y arrancará el proceso de retransmisión antes de esperar por un ACK. Vale decir, el tamaño mínimo de frame está relacionado al control de colisiones. A manera de comentario, Gigabit Ethernet 1000Base-X usa 416 bytes en lugar de los 520 bytes debido a que esta tecnología codifica y transmite 10 bits por cada byte. Es importante aclarar que half-duplex es el modo de transmisión y recepción por el mismo medio pero no al mismo tiempo, hay contención por un medio compartido y así ocurren colisiones. Se usa el protocolo CMSA/CD (Carrier Sense Multiple Access with Collision Detection). Por otra parte, full-duplex es el modo de transmisión y recepción de datos por el mismo medio y al mismo tiempo. En este último caso existen dos enlaces punto a punto (uno en un sentido y otro en el opuesto) que conectan dos estaciones de transmisión y dado que en este esquema de trabajo no hay contención por un medio compartido, no pueden ocurrir colisiones y así no es necesario el uso del protocolo CMSA/CD.

Los “jumbo frames” extienden Ethernet a 9000 Bytes, primero porque Ethernet usa un CRC (Cycling Redundancy Checking o chequeo de redundancia cíclica) de 32 bits, que es un algoritmo que pierde efectividad por encima de los 12000 Bytes [33]. En verdad no hay ninguna demostración teórica de la citada pérdida de efectividad, pero ha sido bien demostrado en forma práctica en tamaños de frames mayores a 12 KB. EL CRC es un método de chequeo de errores en los datos que se transmiten por un canal de comunicaciones. El emisor aplica un polinomio de 16 o 32 bits al bloque de datos a transmitir y agrega el CRC resultante a dicho bloque. El receptor luego aplica el mismo polinomio a los datos y compara el resultado con el CRC agregado por el emisor; si los resultados concuerdan entonces los datos han sido recibidos en forma exitosa, de lo contrario el emisor puede serc notificado para renviar los datos con errores. El CRC de 16 bits detecta todos los errores de un solo bit y bits dobles (dos errores aislados de un solo bit) y asegura una capacidad de detección del 99,998% de todos los posibles errores de ráfagas de 18 bits o mayores. Este nivel de detección es

considerado suficiente para frames de 4 KB o menos. Para paquetes de mayores tamaños se usa un CRC de 32 bits.

En segundo lugar, un paquete de 9000 Bytes es un tamaño adecuado para transportar un datagrama de 8 KB (como por ejemplo los de una aplicación como NFS) más el overhead del header del paquete.

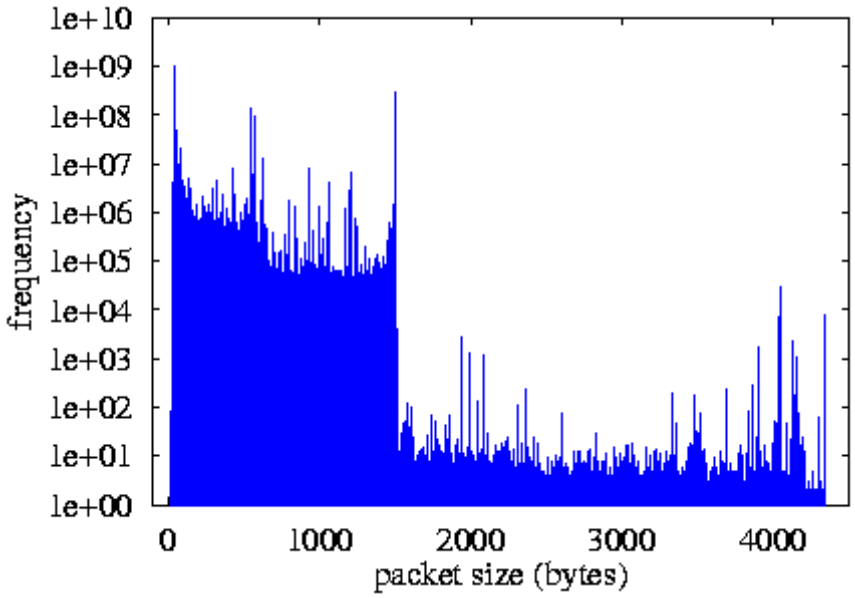
El límite de un datagrama IPv4 es de 64 KB, en tanto que IPv6 permite paquetes de hasta 4 GB de tamaño**. Para Ethernet el límite de CRC 32 bits es muy difícil de cambiar, por lo tanto no se puede esperar tamaños de frame Ethernet mayores a los 9000 Bytes en lo inmediato.**

La coexistencia de los "jumbo frames" y frames de 1500 bytes es posible en las siguientes circunstancias:

- En un principio de comunicación puerto a puerto, donde todo el flujo de downstream desde un dado puerto es conocido para soportar jumbo frames

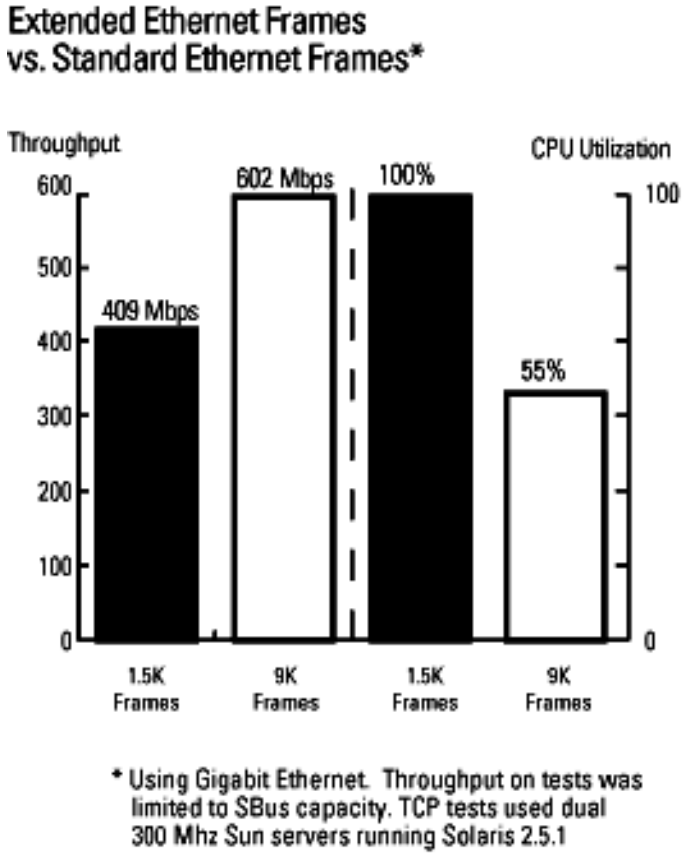
- Usando el estándar 802.1q Virtual LANs, donde los dispositivos jumbo frame y no jumbo frames son separados hacia diferentes VLANs

Un estudio del tamaño de los frames que más se usan en la actualidad dio este resultado:



El gráfico muestra la distribución de diferentes tamaños de paquetes que fluyen a través de un backbone OC3. Hay mucha más densidad de uso de paquetes de hasta 1500 Bytes (Ethernet estándar) pero también hay tráfico por encima de los 4000 Bytes (que es el MTU de FDI). Y aquí hay que destacar un hecho: mientras que el número de paquetes de tamaño mayor a 1500 Bytes es más pequeño, más del 50% de los Bytes fueron transportados por tales paquetes a consecuencia de su mayor tamaño. Muchos de los flujos de datos de alta velocidad han sido alcanzados sobre enlaces ATM WAN con 9180 Bytes de MTU.

Los frames más pequeños implican más interrupciones de CPU y más overhead de procesamiento para un dado tamaño de transferencia de datos. Un análisis comparativo entre jumbo frames Ethernet y frames Ethernet de 1500 bytes muestra el siguiente resultado:



Los jumbo frames proveen casi 50% más de throughput con casi el 50% menos de carga de CPU que utilizando frames de 1500 Bytes. A pesar de que este overhead local puede reducirse en parte con diseños de sistemas mejorados, descarga de trabajo hacia las interfaces NIC, etc., lo que más importa es el enlace WAN.

Las últimas estadísticas [28] reportan que aproximadamente el 6% del tráfico de Internet2 es UDP y que el 5.42 % de los paquetes son mayores de 1500 Bytes (jumbo frames).

4.2.Jumbo Frames: performance TCP

La performace de TCP sobre redes WAN, como lo es el caso de Internet, ha sido estudiada y modelada en forma extensa. Un importante paper de referencia [35] explica cómo el throughput TCP tiene un límite superior basado en los siguientes parámetros:

Throughput <= ~0.7 * MSS / (RTT * sqrt(Packet_Loss))

donde:

MSS: Maximum Segment Size, es la máxima cantidad de datos TCP enviados sobre un único datagrama IP, que el sistema local puede aceptar y reensamblar. En la práctica, MSS = MTU (1500 Bytes)- headers TCP e IP (40 Bytes) = 1460 Bytes

RTT: Round Trip Time, es el tiempo viaje de un paquete saliendo de una determinada interface hacia un determinado host, y volviendo a la misma interface

Packet_Loss: es la pérdida de paquetes en una red, cuanto mayor es el tamaño de los paquetes mayor es la probabilidad de que haya pérdidas de los mismos, en cierta manera debido a problemas de buffers llenos que deben descartar paquetes. La pérdida de paquete se produce principalmente por congestión de la red, y también por ruido en las líneas. Aquí vale la pena aclarar que **el término "packet_loss" de la fórmula incluye también al *reordenamiento de paquetes,*** que ocurre a causa de que los paquetes de un mismo flujo de datos toman diferentes caminos determinados por routers que deben balancear la carga y también por el uso inherente de dispositivos de red tales como switches y routers.

En el citado paper se explica la derivación de la fórmula anterior.

Aclaración: el mismo autor del paper -Matt Mathis- reformuló esta fórmula el año pasado para los casos en que el tamaño del MTU (o MSS) sean mucho mayores a 12000 Bytes.
En esos casos la fórmula queda así:

Throughput <= 0.7 * sqrt(MSS) / (RTT*sqrt(Packet_Loss))

Pero mi investigación se centra en tamaños de MTU cercanos a los 9000 Bytes, tal como los que soporta Internet2, así que no voy a considerar esta última fórmula.

Por lo tanto, teniendo en cuenta la primer fórmula, el máximo throughput TCP es directamente proporcional al MSS. Si todas los demás valores siguen siendo igual, se puede doblar el throughput doblando el tamaño del paquete. La pérdida de paquetes suele incrementarse con el tamaño del MSS, pero en una relación sub-lineal, y en cualquier caso tiene un efecto sobre el throughput dado por la inversa de su raíz cuadrada, y así el MSS todavía es el factor predominante sobre el throughput.

Tomemos por ejemplo una conexión entre Los Angeles y Nueva York. El RTT es de alrededor de 40 mseg, y la pérdida de paquetes de alrededor del 0.1% (0.001). Con un MTU de 1500 Bytes (MSS de 1460 Bytes), la fórmula anterior nos dice que el throughput tiene un límite máximo de 6.5 Mbps. Con frames de 9000 Bytes, el throughput TCP alcanza un límite máximo de alrededor de 40 Mbps.

Y ahora veamos el ejemplo anterior en términos de pérdida de paquetes. Consideremos el mismo RTT, pero digamos que queremos alcanzar un throughput de 500 Mbps con frames de 9000 Bytes y 1500 Bytes respectivamente. Para hacer esto con frames de 9000 Bytes, se necesitaría un porcentaje de pérdida de paquetes no mayor que $1*10^{-5}$ (0.00001). Con frames de 1500 Bytes, el porcentaje de pérdida de paquetes debería ser inferior a $2.8*10^{-7}$ (0.00000028). Por lo tanto, hay que notar que mientras los jumbo frames son 6 veces mayores en tamaño que los de 1500 Bytes, nos permiten un throughput igual a costa de tener 36 veces más pérdida de paquetes.

4.3.Jumbo Frames y el tráfico multimedia

Si uno piensa en principio que el tráfico multimedia es sensible al delay, jitter y pérdida de paquetes, entonces la idea de usar frames grandes no sería tan bien vista. E inclusive uno se puede preguntar si los frames grandes atentan contra las aplicaciones que hacen uso de frames de menor tamaño. Este último es un tema de “slot time”, es decir, cuánto va a retardar un paquete largo el/los tiempo/s para transmitir los paquetes pequeños.

Un paquete GigE de 9000 Bytes toma la misma cantidad de tiempo en ser transmitido que un paquete de 900 Bytes en FastEthernet o que un paquete

de 90 Bytes en 10 Mbps Ethernet. Por lo tanto los frames de 9000 Bytes en GigE a lo sumo agregan menos variación de delay que los frames de 1500 Bytes en redes Ethernet más lentas. Esto quiere decir que en el primer caso se envían 9000 bytes de datos por un enlace GigE, y en el otro caso se están enviando 1500 bytes de datos por enlaces más lentos y eso implica un mayor retardo entre paquetes enviados.

4.4.Utilización de GigE en los NAP (Network Aggregation Point)

Los enlaces GigE no se deberían usar en los NAP si es que ellos reducen el MTU disponible. Los NAP están en el núcleo mismo de Internet, donde se conectan las redes de área amplia. Si los NAP ponen una limitación sobre el MTU, entonces esa limitación aparecerá en todas las WAN, LAN y sistemas finales que atraviesan dichos NAP. No hay nada que un sistema final pueda hacer para elevar el límite de performance impuesto por el MTU de un NAP. Por su ubicación en la estructura de Internet, los NAP deberían hacer todo lo posible para remover los cuellos de botella.

Muchos NAP hoy en día están corriendo sobre redes switcheadas FDI (100 Mbps, 4 KBytes MTU), y se están quedando sin capacidad. Un upgrade hacia ATM OC3 (155 Mbps, 9KB MTU) es difícil de justificar dado que sólo provee un 50% de incremento de ancho de banda. Y tratar de instalar un switch que soporte más de 50 puertos ATM OC12 (2.5 Mbps) es muy costoso. Un switch GigE de 64 puertos es más económico y además posee más ancho de banda por puerto. El problema sin embargo es la transmisión de paquetes de 1500 Bytes MTU, pero GigE con jumbo frames podría permitir transmitir en forma completa los MTU de FDDI y sólo reducir levemente el MTU de IP sobre ATM (9180 Bytes). (En el evento Supercomputing '99, Microsoft y NCSA transmitieron HDTV sobre TCP a una velocidad mayor que 1.2 Gbps con paquetes de 9000 Bytes MTU y debieron bypassear los switches en el NAP).

4.5.Recomendaciones de uso de MTU en Internet2

A partir del año 2003 ya se establecieron ciertas recomendaciones de uso de IP MTU para Internet2, tanto para backbones como para GigaPoPs y LANs de campus. El mismo se estableció en 9000 Bytes.

Por ejemplo, se recomendó a todos los conectores de Abilene que incrementen el tamaño de su MTU. Para ello deberían contactar al NOC (Network Operation Center) de Abilene y así coordinar los cambios apropiados. Además es importante que los conectores puedan utilizar los

MTU de su propia infraestructura y no sólo los de Abilene, para conservar la compatibilidad hacia atrás.

La razón de esta recomendación incluye los siguientes puntos:

- Las aplicaciones, que incluyen pero no se limitan al tráfico “bulk” TCP, se benefician al enviar 8 KB de payload más varios headers (un MTU de 9 KB satisface esta condición). Básicamente, mantener los datos de usuario en un tamaño de 8192 Bytes es una condición importante para la mayoría de los sistemas operativos, y hasta inclusive con IPv6, un paquete de 9 KB es suficiente para trasnportar un payload de 8 KB.

- Un número creciente de NICs en routers, switches y hosts aceptan paquetes IP de al menos 9000 Bytes.

- Un pequeño número de NICs en routers, switches y hosts aceptan paquetes IP de más de 9500 Bytes. Por lo tanto hay menos interés en el uso de paquetes IP mucho mayor que 9000 Bytes.

- Hay evidencias que el protocolo Path MTU Discovery sería más confiable si existe un valor acordado de MTU usado comúnmente. Esto se relaciona a la debilidad en la actual tecnología de Path MTU Discovery.

Esto es una recomendación actual, y no quita que los ingenieros de Internet de la próxima generación exploren los beneficios del uso de MTUs mucho más grandes, hasta los 64 Kbytes permitidos por un datagrama IPv4. Los 64 Kbytes de tamaño máximo permitido para un paquete IPv4 salen de que el campo de tamaño del payload del header es de 16 bits, entonces 2**16 = 65536 Bytes.

4.6.MTU en los routers de Abilene

Existe una herramienta web muy interesante [36] que permite ejecutar comandos online en los routers del backbone de Abilene. Estos comandos son por ejemplo: show ip, show IPv6, show interface, traceroute, mtrace, ping, etc., y nos brindan una información muy rica en detalles técnicos.

A continuación se transcribe parte de la salida del comando “show interface” ejecutado en el router NYCMng Juniper T640 de Nueva York:

Physical interface: so-3/1/0, Enabled, Physical link is Up
Interface index: 158, SNMP ifIndex: 87
Link-level type: Cisco-HDLC, MTU: 9192, Clocking: Internal, SONET mode,
Speed: OC192, Loopback: None, FCS: 32, Payload scrambler: Enabled
Device flags : Present Running
Interface flags: Point-To-Point SNMP-Traps
Link flags : Keepalives
Keepalive settings: Interval 10 seconds, Up-count 1, Down-count 3
Keepalive: Input: 22592 (00:00:01 ago), Output: 22602 (00:00:06 ago)
CoS queues : 8 supported
Last flapped : 2004-06-22 23:38:50 EST (2d 13:55 ago)
Input rate : 1256913496 bps (76433 pps)
Output rate : 566225784 bps (99312 pps)
SONET alarms : None
SONET defects : None

Logical interface so-3/1/0.0 (Index 88) (SNMP ifIndex 88)
Description: BACKBONE - OC-192 to CHINng
Flags: Point-To-Point SNMP-Traps Encapsulation: Cisco-HDLC
Input packets : 25153433424
Output packets: 41011884565
Protocol inet, MTU: 9180
Flags: User-MTU
Addresses, Flags: Is-Preferred Is-Primary
Destination: 198.32.8.82/31, Local: 198.32.8.83
Protocol iso, MTU: 1497
Flags: User-MTU
Protocol inet6, MTU: 9180
Flags: User-MTU
Addresses, Flags: Is-Preferred Is-Primary
Destination: 2001:468:ff:f15::/64, Local: 2001:468:ff:f15::2
Addresses, Flags: Is-Preferred
Destination: fe80::/64, Local: fe80::280:42ff:fe11:6f91
Protocol mpls, MTU: 9180
Flags: User-MTU

Se ve claramente que los IP MTU tanto IPv4 como IPv6 de las interfaces consultadas tienen actualmente un tamaño de 9180 Bytes.

4.7.Jumbo Frames: consideraciones finales

Si se necesita alta performance en las transmisiones de datos, el diseño de red que sea usado –independientemente de la tecnología de red implementada- deberá tener en cuenta lo siguiente:

Cada host extremo deberá tener un camino entre él y la red de área amplia de manera tal que:

- *No reduzca el ancho de banda del enlace, y además*
- *No reduzca el MTU*

Por ejemplo, si un host tiene una interface ATM OC3 corriendo IP Clásico, deberá haber un camino extremo a extremo entre él y la WAN de al menos una velocidad OC3 y que soporte al menos 9000 Bytes MTU. En el caso de los Jumbo Frames, deberá haber un camino que soporte Jumbo Frames desde y hacia Internet.

Es importante destacar que siempre debe haber un balance entre el throughput y la pérdida de paquetes.

En las redes WAN, el RTT (Round Trip Time) y la pérdida de paquetes son parámetros difíciles de cambiar. Por lo tanto nos queda el tamaño del paquete para poder aumentar el throughput, de allí la decisión de usar jumbo frames.

Como conclusión final, si uno desea transmitir datos desde una LAN a muy alta velocidad, la dinámica del TCP requiere usar tamaños de paquetes IP grandes. Sin ellos, la tasa de pérdida de paquetes sobre un dado enlace deberá ser extremadamente baja. La infraestructura del núcleo de Internet, desde los backbones de campus hasta los NAPs, debería tener mucho cuidado en no limitar el MTU permitido a 1500 Bytes. En el largo tiempo no hay razón para limitar el tamaño del paquete a 9000 Bytes, pero actualmente dada la limitación del CRC de Ethernet es un buen comienzo para tasas de transferencias del orden del gigabit.

5.Características de la transmisión de HDTV sobre Internet2

5.1.Disponibilidad

La disponibilidad ideal de un enlace es del 100%. Normalmente el porcentaje alcanzado es de 99.8 %, y la tendencia es llegar al 99,9999% (“six nines”).

5.2.Ancho de banda

El ancho de banda de un enlace de comunicaciones es la cantidad de bits que pueden pasar por unidad de tiempo.

Acá vale la pena hacer notar que el término ancho de banda no es lo mismo que throughput, dado que este último es la cantidad efectiva de datos (sin tener en cuanta el overhead) por unidad de tiempo que se transmiten desde un host hacia otro a través de un dado enlace de comunicaciones. El proveedor de servicios (ISP) usualmente garantiza una tasa mínima de transferencia, llamada CIR (Commited Information Rate).

5.3.Latencia

Es el tiempo que toma el dato para viajar desde la fuente al destino. En Internet la latencia depende de los retardos en los routers, tránsito entre los enlaces y congestionamiento de redes. En el caso de las redes ultra rápidas de fibra óptica, como lo es Internet2, la mayoría del delay se debe a la velocidad de la luz en la fibra que es de aproximadamente 200 Km/mseg.

5.4.Jitter

El jitter es la variación de la latencia. Las causa que lo provocan son las variaciones en la longitud de las colas de los routers, la variación en la velocidad de procesamiento de paquetes que llegan fuera de orden y la variación en el tiempo necesario para reensamblar paquetes que fueron segmentados.

5.5.Pérdida de paquetes

Los dispositivos de red (routers, switches, etc.) deben encolar datos en sus buffers, y a veces cuando las colas están saturadas hay paquetes que se descartan y se pierden. En una red bien administrada la pérdida de paquetes deberá ser **menor al 1% promediada en forma mensual**.

Hay dos cosas que una aplicación puede hacer ante la pérdida de un paquete: tratar de corregirlo (error correction) o tratar de ocultarlo (error concealment).

5.6.Reordenamiento de paquetes

Este fenómeno se debe a que los paquetes de un mismo flujo atraviesan diferentes caminos de red (debido, por ejemplo, a balanceo de carga por paquetes en un router), y cada paquete experimenta diferentes delays. También se debe a los dispositivos de una red como switches y routers que reordenan el stream de paquetes.

Por la primera causa del reordenamiento de paquetes, algunos operadores de red a menudo desarrollan un balanceo de carga por origen/destino de los paquetes, y así garantizan que el mismo flujo siempre atraviese el mismo path. La mayoría de los routers modernos soportan esta característica que aunque esta implementación presente otras ineficiencias, resuelve el problema del reordenamiento de paquetes.

El reordenamiento de paquetes es un problema “silencioso”, dado que la mayoría de los routers contabilizan pérdidas de paquetes y puede mostrar este tipo de información a los operadores, pero el reordenamiento de paquetes es invisible para estos dispositivos y no dejan trace alguno, y así los operadores de red no pueden identificar la potencial causa de degradación del throughput del tipo TCP. El reordenamiento de paquetes combinado con un cierto nivel de pérdidas de paquetes, originen múltiples invocaciones del mecanismo de “fast recovery” [37] de la ventana de TCP, lo que lleva a una cierta pérdida de la utilización del enlace y del throughput de las aplicaciones debido a que cualquier paquete fuera de orden que arriba al receptor hace que el mismo dispare un ACK duplicado en el cual el receptor repite al ACK del último segmento recibido fuera de orden. Cualquier otro segmento fuera de orden causará otro ACK duplicado. El emisor sin embargo considera que si recibe tres ACKs duplicados es señal que el segmento se perdió y dispara la retransmisión del segmento perdido. Las retransmisiones ocurren sin esperar un timeout, esto se conoce como “fast retransmit” [38]. Una vez que las retransmisiones toman lugar, el emisor entra en la fase de “fast recovery”. Todo esto hace que el throughput de las aplicaciones TCP disminuya.

En el caso de tráfico UDP con un mecanismo de control de congestión como se verá más adelante, el reordenamiento de paquetes es visto como pérdidas de paquetes y esto disminuye el throughput dado que éste es función inversa de las mismas.

5.7.Medición de ancho de banda, delay, jitter y pérdida de paquetes

Para realizar experiencias con aplicaciones que requieren grandes anchos de banda y que pueden tener consecuencias sobre otros flujos simultáneos, se deben usar herramientas para medir el ancho de banda extremo a extremo.

Una de las herramientas que más se usa se llama Iperf [39] desarrollada por la Universidad de Illinois de EE.UU. Iperf fue desarrollada como una moderna herramienta para medir la performance de ancho de banda TCP y

UDP. Por ejemplo esta herramienta se usó para medir el ancho de banda con paquetes UDP en una experiencia de transmisión de HDTV sin comprimir sobre un enlace ultra rápido de la red SuperNet del programa Next Generation Internet entre la ciudades estadounidenses de Arlington y Pittsburg.

Es una herramienta para medir el máximo ancho de banda TCP permitiendo el tuning de varios parámetros y las características UDP. Mide también con modalidad multicast y paquetes IPv6. Iperf reporta ancho de banda, delay, jitter y pérdida de paquetes. Corre sobre sistemas operativos Unix y Windows, y necesita la instalación de un servidor y un cliente. El servidor se instala en un extremo del enlace y el cliente en el otro.

En el caso de UDP –usado preferentemente por las aplicaciones que necesitan tiempo real- crea un stream a una velocidad (bit rate) constante. Uno puede ajustar el tamaño de datagrama según el utilizado por la aplicación. El server detecta pérdida de datagramas por el ID asignado a cada datagrama. Usualmente un datagrama UDP se transmite en diversos paquetes IP. Por lo tanto, perdiendo un único datagrama IP se perderá el datagrama entero. Por lo tanto, para medir pérdida de paquetes en lugar de pérdida de datagramas, hay que hacer el tamaño del datagrama lo suficientemente pequeño para que entre en un paquete IP. También da un reporte de los datagramas fuera de secuencia o reordenados.

El cálculo de jitter en UDP, es computado en forma permanente en el server, como lo especifica RTP [40]. El cliente graba un timestamp de 64 bit sec/microsec en el paquete. El server computa el tiempo de tránsito relativo como la diferencia entre el tiempo de recepción del server y el tiempo de envío del cliente. Los relojes del server y cliente no necesitan estar sincronizados, cualquier diferencia es restada del cálculo del jitter. El jitter es el término medio alisado de la diferencias entre tiempos de tránsito consecutivos.

Aquí va un ejemplo:

```
node2> iperf -s -u -i 1
------------------------------------------------------------
--
Server listening on UDP port 5001
Receiving 1470 byte datagrams
UDP buffer size: 60.0 KByte (default)
------------------------------------------------------------
--
[  4] local <IP Addr node2> port 5001 connected with <IP
Addr node1> port 9726
[ ID] Interval       Transfer     Bandwidth       Jitter
Lost/Total Datagrams
[  4]  0.0- 1.0 sec   1.3 MBytes  10.0 Mbits/sec  0.209
ms    1/  894 (0.11%)
[  4]  1.0- 2.0 sec   1.3 MBytes  10.0 Mbits/sec  0.221
ms    0/  892 (0%)
[  4]  2.0- 3.0 sec   1.3 MBytes  10.0 Mbits/sec  0.277
ms    0/  892 (0%)
[  4]  3.0- 4.0 sec   1.3 MBytes  10.0 Mbits/sec  0.359
ms    0/  893 (0%)
[  4]  4.0- 5.0 sec   1.3 MBytes  10.0 Mbits/sec  0.251
ms    0/  892 (0%)
[  4]  5.0- 6.0 sec   1.3 MBytes  10.0 Mbits/sec  0.215
ms    0/  892 (0%)
[  4]  6.0- 7.0 sec   1.3 MBytes  10.0 Mbits/sec  0.325
ms    0/  892 (0%)
[  4]  7.0- 8.0 sec   1.3 MBytes  10.0 Mbits/sec  0.254
ms    0/  892 (0%)
[  4]  8.0- 9.0 sec   1.3 MBytes  10.0 Mbits/sec  0.282
ms    0/  892 (0%)
[  4]  0.0-10.0 sec  12.5 MBytes  10.0 Mbits/sec  0.243
ms    1/ 8922 (0.011%)

node1> iperf -c node2 -u -b 10m
------------------------------------------------------------
--
Client connecting to node2, UDP port 5001
Sending 1470 byte datagrams
UDP buffer size: 60.0 KByte (default)
------------------------------------------------------------
--
[  3] local <IP Addr node1> port 9726 connected with <IP
Addr node2> port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec  12.5 MBytes  10.0 Mbits/sec
[  3] Sent 8922 datagrams
```

En el ejemplo siguiente se nota un jitter más grande debido al reensamble de datagramas cuando se usan datagramas mayores a 32 KBytes, los que se dividen en 23 paquetes de 1500 Bytes. La mayor pérdida de datagramas que se ve aquí puede ser debida al “burstiness” (característica de ráfaga) del tráfico, el cual es de 23 paquetes back-to-back y luego una larga pausa, más bien que a paquetes individuales espaciados uniformemente.

```
node2> iperf -s -u -l 32k -w 128k -i 1
------------------------------------------------------------
--
Server listening on UDP port 5001
Receiving 32768 byte datagrams
UDP buffer size:  128 KByte
------------------------------------------------------------
--
[  3] local <IP Addr node2> port 5001 connected with <IP
Addr node1> port 11303
[ ID] Interval       Transfer     Bandwidth       Jitter
Lost/Total Datagrams
[  3]  0.0- 1.0 sec   1.3 MBytes  10.0 Mbits/sec  0.430
ms    0/   41 (0%)
[  3]  1.0- 2.0 sec   1.1 MBytes   8.5 Mbits/sec  5.996
ms    6/   40 (15%)
[  3]  2.0- 3.0 sec   1.2 MBytes   9.7 Mbits/sec  0.796
ms    1/   40 (2.5%)
[  3]  3.0- 4.0 sec   1.2 MBytes  10.0 Mbits/sec  0.403
ms    0/   40 (0%)
[  3]  4.0- 5.0 sec   1.2 MBytes  10.0 Mbits/sec  0.448
ms    0/   40 (0%)
[  3]  5.0- 6.0 sec   1.2 MBytes  10.0 Mbits/sec  0.464
ms    0/   40 (0%)
[  3]  6.0- 7.0 sec   1.2 MBytes  10.0 Mbits/sec  0.442
ms    0/   40 (0%)
[  3]  7.0- 8.0 sec   1.2 MBytes  10.0 Mbits/sec  0.342
ms    0/   40 (0%)
[  3]  8.0- 9.0 sec   1.2 MBytes  10.0 Mbits/sec  0.431
ms    0/   40 (0%)
[  3]  9.0-10.0 sec   1.2 MBytes  10.0 Mbits/sec  0.407
ms    0/   40 (0%)
[  3]  0.0-10.0 sec  12.3 MBytes   9.8 Mbits/sec  0.407
ms    7/  401 (1.7%)

node1> iperf -c node2 -b 10m -l 32k -w 128k
------------------------------------------------------------
--
Client connecting to node2, UDP port 5001
Sending 32768 byte datagrams
UDP buffer size:  128 KByte
------------------------------------------------------------
--
[  3] local <IP Addr node2> port 11303 connected with <IP
Addr node1> port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec  12.5 MBytes  10.0 Mbits/sec
[  3] Sent 401 datagrams
```

Aquí vale la pena tener en cuenta que en el ejemplo anterior se mide la performance del ancho de banda con datagramas UDP de 32 KB pero transportados con paquetes IP de 1500 Bytes, no con Jumbo Frames.

6.Streaming de multimedia sobre IP

6.1.Uso de IP para tráfico de tiempo real

A continuación se detalla un esquema del stack del protocolo IP según el modelo OSI de siete capas:

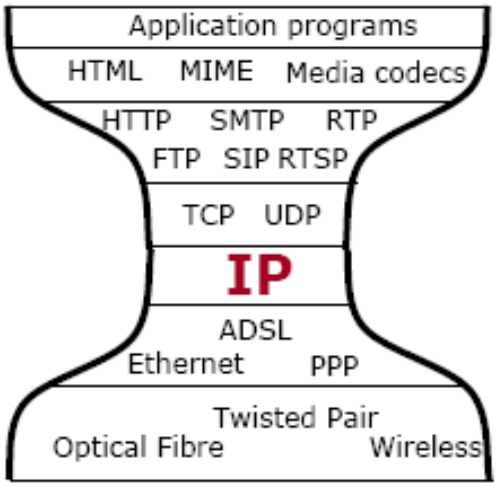


Diagrama del trabajo “RTP: multimedia streaming over IP” – Colin Perkins – USC ISI

IP constituye una capa de abstracción, sobre la cual están las aplicaciones los protocolos de transporte, y bajo la cual están las tecnologías de enlace.

Las aplicaciones no pueden “ver” las capas de enlace, sólo ven la performance de la capa IP. La capa de enlace no puede conocer las necesidades de las aplicaciones, sólo ven una serie de paquetes de datos que bajan a ella.

El modelo de servicio IP está limitado a:

a) Transporte de paquetes “best-effort”

En este modelo de transporte, la performance no está garantizada. Los paquetes pueden ser perdidos, retardados, reordenados, duplicados y corruptos (la capa de transporte debería compensar esto). Y además se aplica checksum para verificar los errores en los bits.

Estos problemas pueden tener diferentes causas como ser:

- La congestión puede causar pérdidas
- La corrupción de paquetes puede causar pérdidas
- Los cambios de rutas pueden causar pérdidas
- Las colas de paquetes pueden provocar delay
- El ruteo IP multipath puede provocar reordenamiento
- Las retransmisiones espúreas y klos bugs en los routers

Todo esto causa además una deformación en el timing de la transmisión y recepción de los paquetes, que se ve de la siguiente manera:

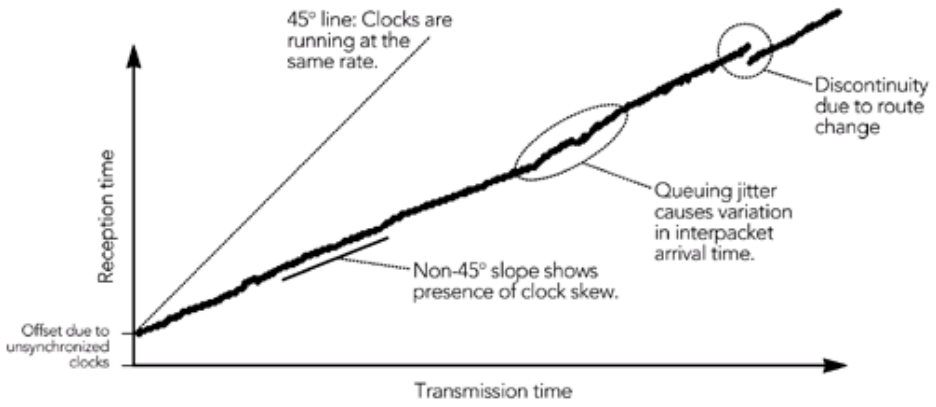


Diagrama del trabajo “RTP: multimedia streaming over IP” – Colin Perkins – USC ISI

b) Fragmentación de paquetes

Existe una fragmentación de los paquetes que exceden el MTU (Media Transmisión Unit) de la red. Esto causa un efecto multiplicador en las pérdidas que es indeseable, dado que la pérdida de un fragmento significa que los otros fragmentos del mismo paquete deben también ser descartados. Lo deseable es que la aplicación realice la fragmentación y genere así pequeños paquetes IP.

El siguiente gráfico muestra el efecto multiplicador de las pérdidas de fragmentos:

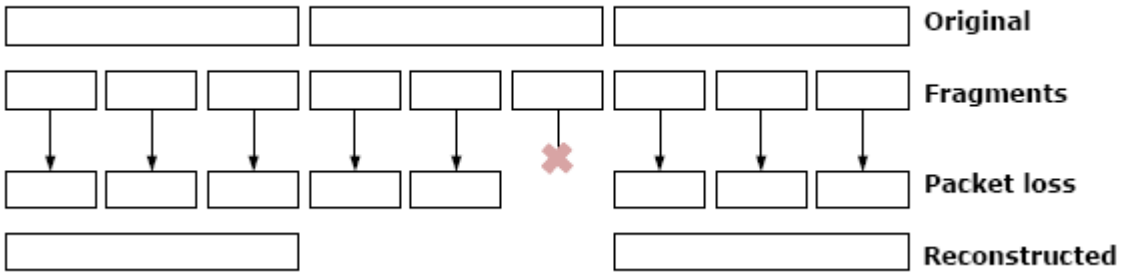


Diagrama del trabajo “RTP: multimedia streaming over IP” – Colin Perkins – USC ISI

c) Routing y addressing

La mayoría de las comunicaciones son unicast (punto a punto), y son ineficientes para el caso que haya muchos receptores, debido a que el servidor debe generar N copias de datos para N clientes. Así se puede generar un cuello de botella significativo. En muchos casos se usa multicast, pero esto introduce más complejidad y además recibe diferentes características de pérdidas. El término “addressing” hace referencia al uso del espacio de direcciones IP.

6.2.Protocolos de transporte

Por lo tanto, se puede decir que el servicio IP por sí mismo es muy limitado, dado que sólo trata de enviar paquetes. Siempre debe estar acompañado por un protocolo de transporte como TCP, UDP u otros en estado de desarrollo.

En el caso de **TCP** (Transport Control Protocol) [41], este protocolo incrementa el servicio brindado por IP de la siguiente manera:

- Orientado a conexión y punto a punto
- Selección de servicios a través de puertos
- Entrega confiable y en orden, asegurado por las retransmisiones
- Adaptación de la velocidad de transmisión mediante diferentes tamaños de ventana de congestión para acomodarse a la capacidad de la red; existe una alta utilización de los enlaces y una compartición justa de la capacidad de los mismos entre los diferentes flujos

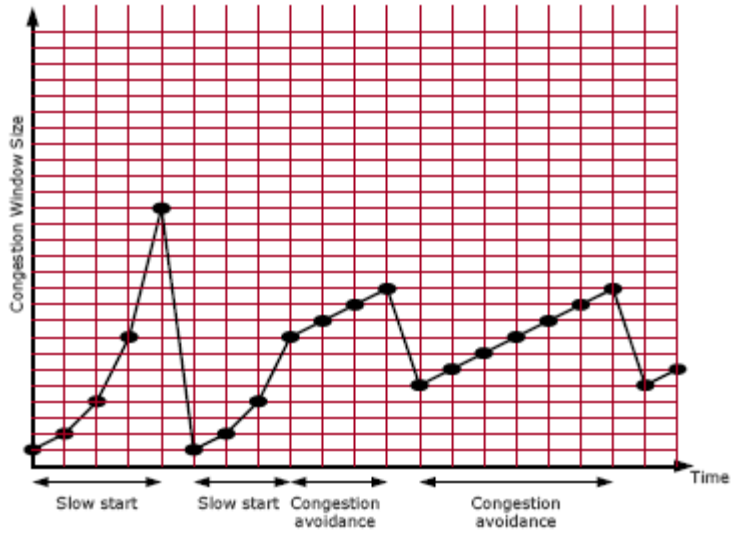


Diagrama del trabajo “RTP: multimedia streaming over IP” – Colin Perkins – USC ISI

En el caso de **UDP** (User Datagram Protocol) [42], también usa los servicios de IP, y tiene las siguientes características:

- Transporte “best-effort” (no confiable) pero a tiempo
- Fragmentación de paquetes
- Routing y addressing
- Unicast y multicast
- Provee puertos, en adición al direccionamiento IP, pero no otros servicios

Por lo tanto, existe un trueque entre confiabilidad y puntualidad en la entrega de paquetes. El siguiente gráfico lo ilustra y muestra además la

ubicación del protocolo RTP (Real Time Protocol) dentro de estas variables:

Diagrama del trabajo “RTP: multimedia streaming over IP” – Colin Perkins – USC ISI

Los sistemas multimedia eligen su transporte en forma cuidadosa:

- TCP para señalización (por ej., H.323 para teleconferencia, SIP para telefonía, RTSP o SAP para streaming)
- UDP para los datos multimedia

Pero los protocolos a nivel aplicación pueden hacer difuso ese límite, como por ejemplo RTP para los datos multimedia.

RTP es un protocolo que se ubicaría en el Nivel de Aplicación en el modelo TCP/IP, y en la Capa de Sesión en el Modelo OSI.

Una vez que la sesión multimedia se establece, los datos fluyen. En el caso de RTP, se puede decir que es un protocolo flexible, soporta muchos CODEC’s y tipos de datos multimedia, y es extensible a nuevos escenarios.

En general, el marco de trabajo para los protocolos multimedia puede ser este:

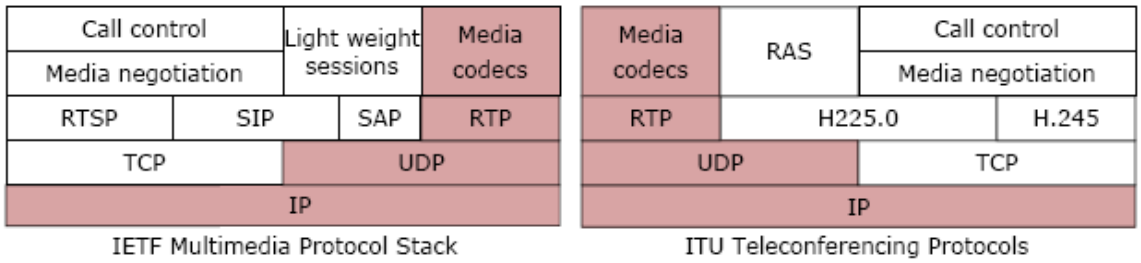


Diagrama del trabajo “RTP: multimedia streaming over IP” – Colin Perkins – USC ISI

6.3.RTP: protocolo de transporte en tiempo real

El estándar de transporte en tiempo real fue propuesto por la IETF allá por el año 1996 a través de las RFC 1889 [44] -hoy actualizada por la RFC 3550 [40] en carácter de estándar- y la RFC 1890 [45] -hoy actualizada por la RFC 3551 [46] en carácter de estándar-. Actualmente es el estándar para transporte en tiempo real sobre redes IP, para casos como streaming de audio y video y voz sobre IP –entre otros-. Fue adoptado por la ITU como parte del estándar H.323 y usado ampliamente en herramientas comerciales de streaming de audio y video de las empresas QuickTime, Real y Microsoft.

El objeto de RTP es construir un mecanismo para la entrega de datos multimedia de manera robusta y en tiempo real, sobre una capa de transporte no confiable e impredecible, y sin cambiar dicha capa, permitiendo un amplio rango de CODEC’s, detección de problemas de red y recuperación del timing de los datos multimedia.

La filosofía de RTP es tener aplicaciones que sean inteligentes y “network-aware” (enterada de las limitaciones de la red subyacente, adaptando su transmisión para que corresponda a dichas limitaciones), capaces de reaccionar a los problemas extremo a extremo. Tanto el servidor como el receptor son inteligentes y la red es “muda” pudiendo ser no confiable. Esta filosofía encaja muy bien con el modelo de servicios IP.

Esta filosofía contrasta con algunas aplicaciones tradicionales como el caso de la red de telefonía, donde la red es inteligente y los extremos son "mudos".

A continuación se detallan los componentes del protocolo:

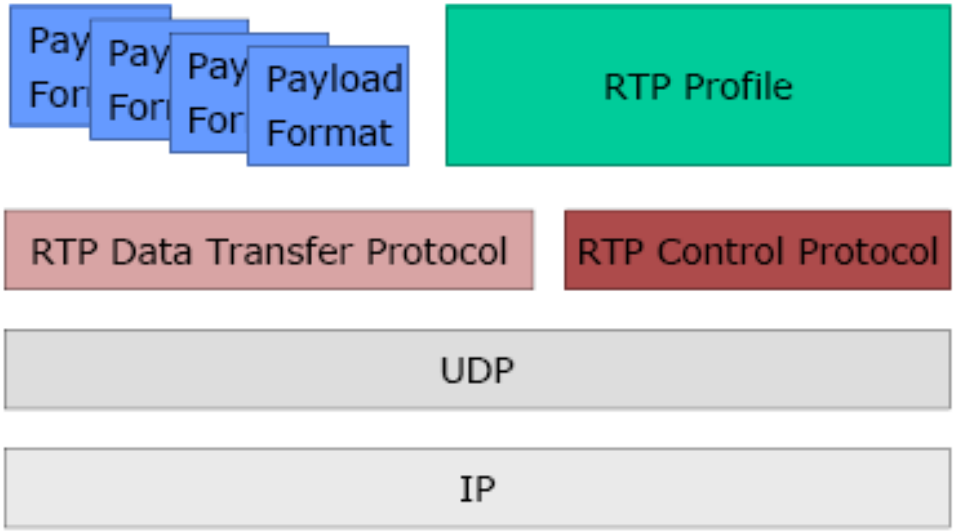


Diagrama del trabajo "RTP: multimedia streaming over IP" – Colin Perkins – USC ISI

Protocolo RTP para transferencia de datos

Transporta unidades de datos de aplicaciones de audio y video, y hace falta un stream o sesión RTP para cada tipo de datos multimedia. Por lo tanto, una sesión multimedia puede comprender varias sesiones RTP.

Cada sesión RTP implementa un perfil RTP particular e incluye un flujo de datos RTP que transporta un tipo único de multimedia acorde a uno o más formatos de payload (por ej., video que usa MPEG).

Además incluye un flujo de control RTP con información de calidad, usuario, etc.

Una sesión multimedia queda definida por las direcciones IP de origen y destino, y por un par de ports UDP –uno para RTP y otro para RTCP-.

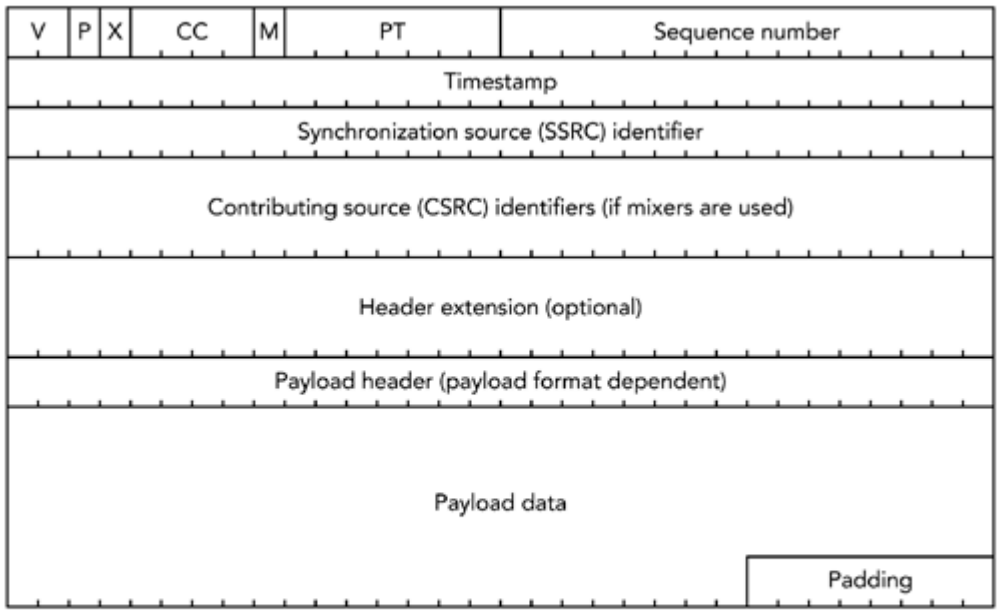
Este protocolo envía un único stream multimedia desde un transmisor hacia uno o más receptores, y usualmente corre sobre UDP/IP. Además, típicamente se implementa en una aplicación o como una librería. Por lo tanto, trabaja a nivel de usuario y no forma parte del kernel del sistema operativo.

Provee identificación del origen, identificación del payload (multimedia), secuenciamiento de los datos multimedia, transporte multimedia (si es necesario aplica padding y marcación de eventos significativos) y recuperación del tiempo (timing recovery). Y además, las extensiones permiten la corrección de errores.

El formato del paquete de transferencia de datos RTP está formado por cuatro partes:

1.- EL RTP header (obligatorio)
2.- Una extensión del header (opcional)
3.- Un payload header (opcional y dependiente del formato de payload usado)
4.- El payload data

A continuación se detalla el formato de un paquete de transferencia de datos RTP:



V = version number
P = padding
X = extensions
CC = count of contributing sources
M = marker
PT = payload type

El **RTP Header** del paquete de datos es de carácter fijo y consiste de los siguientes campos:

- Payload Type
- Sequence Number
- Timestamp
- Synchronization Source
- Contributing Sources
- Marker
- Padding
- Version Number

El **Header Extension** se presenta luego del RTP Header fijo pero antes de cualquier Payload Header y Payload en sí mismo, y se usan para casos que se necesita más información de header de la que provee el mismo header fijo de RTP. Las extensiones que requieren información de header adicional e independiente del formato de payload son escritas mejor como un RTP Profile. Si se requieren headers adicionales para un formato de payload en particular, los mismos no deberían usar un aexdtensión de header y en su lugar deberían ser transportados en la sección del payload del paquete como un Payload Header.

El **Payload Header** es dependiente del formato de payload. EL RTP Header fijo provee información que es común a todos los formatos de payload. En muchos casos un formato de payload necesitará más información para una operación óptima; esta información forma un header adicional que es definido como parte de la especificación del formato de payload. La información contenida en el Payload Header puede ser estática o dinámica, y esto está indicado en la especificación del formato de payload. Por ejemplo se puede especificar un Payload Header para proveer resiliencia al error para aquellos formatos que no fueron diseñados para ser usados sobre redes que experimenten pérdidas de paquetes.

El **Payload Data** se ubica a continuación del Payload Header y contiene una o más frames de datos multimedia. El tamaño y formato del Paylaod Data dependen del formato de payload y formato de los parámetros elegidos durante el establecimiento de la sesión RTP. Muchos formatos de paykload permiten que múltiples frames de datos sean incluídas en cada paquete. Hay dos puntos importantes a considerar a la hora de elegir la cantidad de payload data a inclur en cada paquete: el MTU del path de transmisión y la latencia inducida por la espera de más datos a ser producidos para llenar un paquete más largo (un paquete no puede ser enviado hasta que el último octeto que el contendrá sea producido, y así ls

datos al comienzo del paquete son retrasados hasta que se complete el paquete).

A continuación se describen algunos de los conceptos y campos más importantes del paquete de datos RTP:

Identificación del origen: cada paquete lleva una fuente de sincronización de 32 bits, elegida en forma aleatoria en el arranque, con detección de colisiones. Además provee una capa de transporte independiente del identificador que soprta gateways, IPv4, IPv6 y ATM. También identifica un único flujo multimedia sincronizado en el tiempo mapeado a un identificador persistente usando RTCP. Cada paquete puede incluir una lista de orígenes contributivos (CSRC) que permite identificar hasta 16 orígenes y cada CSRC es el SSRC de cada "mixed participant". Además la identificación de origen permite que RTP soporte mezcladores y traductores (los mezcladores combinan diversos flujos dentro de uno como el caso de la conferencia MCU) y los traductores cambian el formato de un flujo o hacen de gateway entre diferentes redes, por ej. pasando a velocidades de transmisión más bajas o haciendo de gateway entre unicast y multicast.

Identificación de la multimedia: cada paquete transporta un campo de tipo de payload (PT) de 7 bits. Este campo se mapea a un formato de payload durante el establecimiento de una sesión, permitiendo una señalización flexible para los tipos de CODEC's y parámetros; además el mapeo puede ser estático si el perfil lo permite. **Cada flujo transporta sólo un tipo de multimedia: sólo audio o sólo video**. El tipo de payload permite al emisor intercambiar entre un conjunto de formatos de payloads, por ej. un flujo que transporta sólo audio pero puede intercambiar entre fax y voz en cualquier momento.

Transporte de los datos multimedia y formatos de payload: los paquetes contienen un bloque de datos de payload, descritos por el formato de payload. Los formatos de payload describen el mapeo entre la salida del CODEC y los paquetes RTP, elegido de forma tal que cada paquete sea decodificado en forma independiente y además existe un framing a nivel aplicación. Los datos de payload típicamente incluyen un payload header de fácil análisis.

Transporte Multimedia - Bit de marca (marker): cada paquete incluye un bit para marcar eventos significativos, como el comienzo de una conversación en audio o el último paquete de un frame en video.

Transporte Multimedia – Padding: cada paquete puede tener un padding (relleno) más allá de su tamaño natural. Este campo es usado rara vez, y se necesita en algunos algoritmos de encriptado como DES en modo CBC operando con bloques de 64 bits.

Secuenciamiento: cada paquete contiene un número de secuencia de 16 bits, que toma un valor inicial aleatoreo y luego se incrementa en forma monotónica con cada paquete enviado y que se acerca alrededor de cero cuando el límite es alcanzado. Se usa para detectar pérdida de paquetes y no es usado para determinar el orden de reproducción de la multimedia. El RTP básico no provee corrección de error, por lo tanto el receptor deberá encubrir el error y seguir con el procesamiento; en cambio, las extensiones del protocolo proveen FEC (forward error correction) y retransmisiones limitadas.

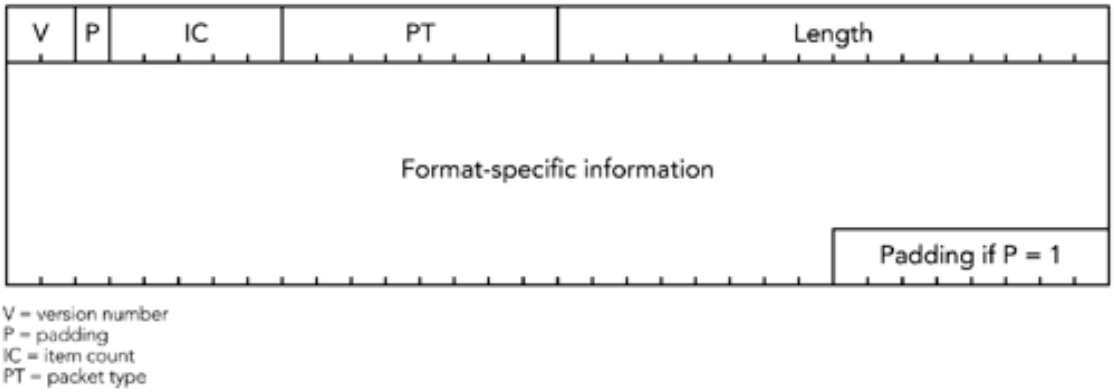
Recuperación del Timing (timestamp): cada paquete contiene un timestamp de 32 bits, que refleja el instante de muestreo del primer octeto en el paquete de datos RTP. Los timestamps son asignados por frame, y si un frame necesita ser fragmentado en diferentes paquetes RTP, entonces cada paquete que conforme el frame tendrá el mismo timestamp. El instante de muestreo debe derivarse de un reloj que se incremente en forma monolítica y lineal en el tiempo para permitir así calcular la sincronización y el jitter. Por este campo se determina el orden de reproducción. La velocidad del clock (clock rate) está definida por el formato de payload o perfil especificados. No hay requerimientos de estabilidad ni exactitud del reloj, lo que implica que hay una adaptación del receptor. El emisor es el responsable por la elección de un reloj apropiado y estable para la aplicación correspondiente. El clock de referencia es compartido por todos los flujos multimedia que deban ser sincronizados. En el caso del video, el clock que se usa generalmente es de 90 KHz, porque da incrementos de timestamps enteros para formatos de video y tasas de frames comunes. Para la mayoría de los formatos de payload de audio, el incremento del timestamp RTP para cada frame es igual al número de muestras (y no de octetos) leidas desde el dispositivo de captura. El código de tiempo no se transporta directamente sino que se mapea a un “reloj de pared” (wall clock) vía reportes RTCP del emisor. El receptor conoce la tasa de clock nominal pero usualmente no tiene otro conocimiento relativo a la exactitud. Por lo tanto las aplicaciones deben ser robustas frente a las variaciones del reloj, tanto en el emisor como en el receptor. Los timestamps en los paquetes de datos RTP y en los Sender Reports RTCP representan el timing de la multimedia en el emisor: el timing del proceso de muestreo y la relación entre el proceso de muestreo y un clock de referencia. Se espera entonces que el receptor reconstruya el timing de la multimedia con esta

información. Se hace notar que el modelo de timing RTP no dice nada acerca de cuándo los datos multimedia deben ser reproducidos. Los timestamps en los paquetes de datos brindan el timing relativo y los RTCP Sender Reports proveen una referencia para la sincronización entre streams, pero RTP no dice nada acerca de qué cantidad de buffer hace falta en el receptor. Aunque el modelo de timing RTP está bien definido, la especificación no hace mención a ningún algoritmo específico para reconstruir el timing: esto es intencional ya que los algoritmos de reproducción dependen de las necesidades de la aplicación y esa es un área donde los vendedores se diferencian con sus productos.

6.4.RTCP: protocolo de control RTP

Cada flujo de datos RTP tiene un flujo de control asociado, definido por RTCP [40]. El flujo de control provee administración de la base de tiempo, retroalimentación de la calidad de servicio y administración e identificación de miembros. Cada sesión RTP está identificada por una dirección de red y un par de puertos: uno para los datos RTP y otro para los datos RTCP. El puerto de datos RTP debería ser par y el puerto de datos RTCP uno por encima e impar, por ejemplo UDP/5004 y UDP/5005 respectivamente.

A continuación se grafica el formato básico de paquete RTCP:



Administración de la base de tiempo

Los timestamps mapean entre la línea de tiempo RTP y el tiempo del “reloj de pared” NTP (Network Time Protocol) [107]. Si se usa un reloj NTP en común para múltiples streams, un receptor puede sincronizarlos.

No hay transporte explícito de códigos de tiempo SMPTE (o similares). Estos pueden ser derivados de los timestamps NTP. Esto trae aparejada una

exactitud limitada por la resolución NTP, a menos que se provea un clock externo. El protocolo RTSP (Real Time Streaming Protocol) provee una función de mapeo.

También permite que los receptores puedan estimar la velocidad de datos/paquetes y posibles variaciones del clock.

Retroalimentación de la calidad de servicio

El feedback de la calidad de servicio desde cada receptor provee información sobre la fracción de pérdida, el número acumulativo de paquetes perdidos, el número de secuencia más alto recibido, el jitter entre arribos y el RTT (round trip time).

Esta información puede tener muchos usos, por ejemplo: la tasa de pérdidas puede ser usada para elegir la cantidad de FEC (Forward Error Correction) a emplear y el jitter brinda una estimación del delay del buffer de reproducción en el receptor.

Administración de miembros

RTCP provee un nombre canónico, mapeando el SSRC a un identificador persistente; son usados para asociar streams para la sincronización. RTCP opcionalmente puede entregar datos de descripción del origen como ser: nombre, dirección de mail, número de teléfono y ubicación.

Intervalos de reportes RTCP

RTCP es un protocolo de baja velocidad, que no está diseñado para usos que requieran un feedback instantáneo, y es escalable a sesiones muy grandes.

Los paquetes son enviados en forma periódica, y el intervalo entre paquetes se ajusta al límite RTCP de uno cada 5 segundos o al valor recomendado de 5% de la velocidad de datos (o sea, 5% del ancho de banda de sesión). El envío de estos reportes se hace en forma aleatorizada para evitar sincronización y así no llegar a la saturación de la capacidad de procesamiento del emisor y/o receptor.

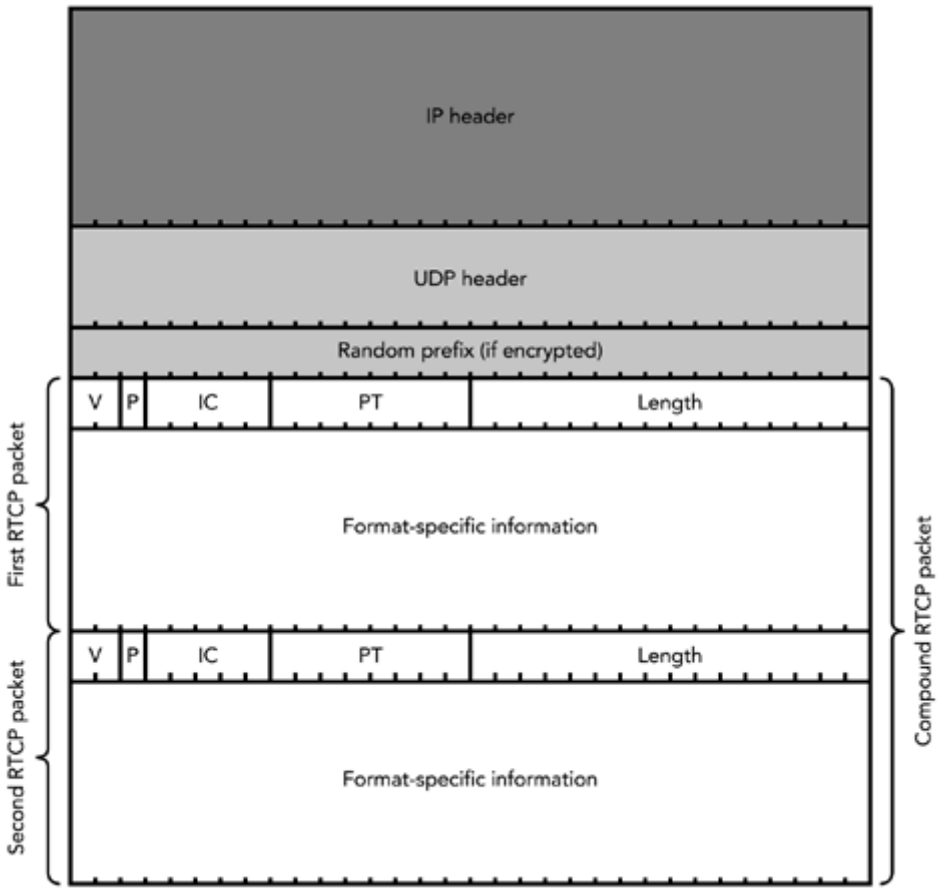
En algunos casos es deseable enviar paquetes RTCP con una frecuencia mayor a l intervalo mínimo predeterminado. Por ejemplo, puede ser el caso que la tasa de datos es alta y la aplicación demande una recepción a tiempo de las estadísticas de calidad de servicio. Por tal motivo, RTP permite –a

partir de una de sus revisiones- un intervalo mínimo reducido para estos casos que se calcula de la siguiente manera:

Mínimo intervalo de reporte RTCP = 360 / (ancho de banda de sesión en Kbps)

Los paquetes RTCP nunca se transportan de a uno sino que se transportan formando grupos. Cada grupo de paquetes RTCP se encapsula un un solo paquete IP/UDP para ser transportado. El tamaño promedio incluye no sólo los datos RTCP sino también los tamaños de header IP y UDP (se agregan 28 octetos por paquete compuesto RTCP para una implementación IPv4 típica).

A continuación se grafica lo antes dicho:



6.5.Robustez del protocolo RTP

RTP opera sobre UDP/IP, con entrega “best-effort”, y los paquetes pueden ser perdidos, retrasados, reordenados, duplicados, etc.

Las aplicaciones son responsables por la correcta reproducción, y aquí intervienen conceptos como recuperación del timing, ocultamiento de errores y control de congestión.

Recuperación del timing

La red puede puede llegar a interrrumpir de manera importante el timing de la multimedia. Los receptores deben incluir un buffer de compensación del jitter para reconstruir los datos multimedia para la reproducción correcta.

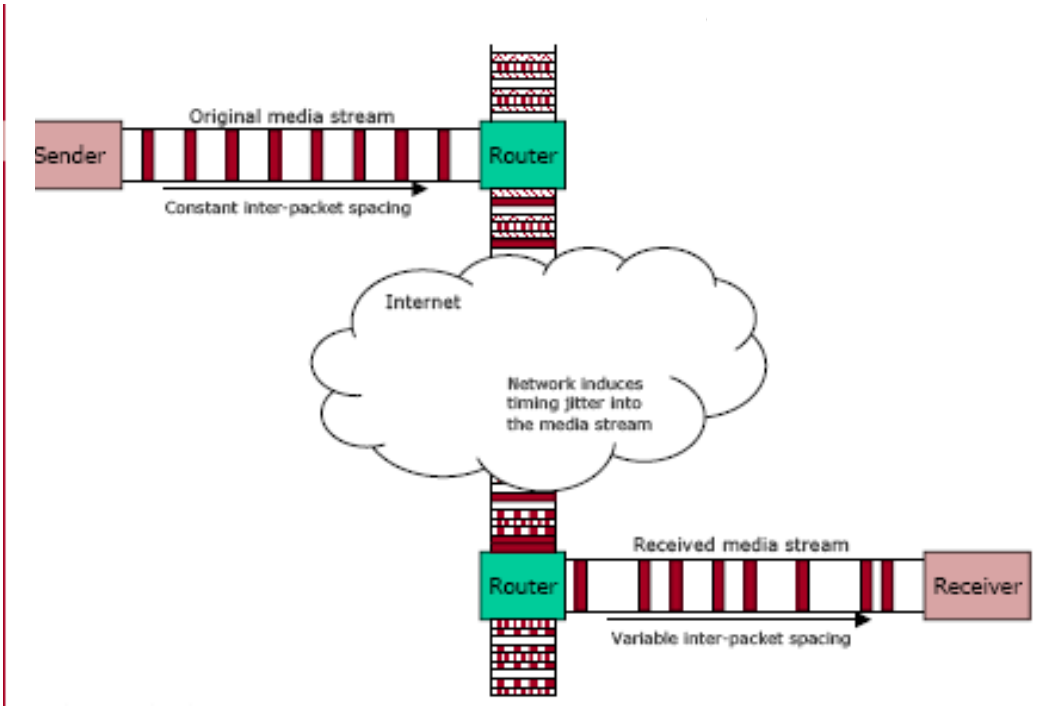


Diagrama del trabajo “RTP: multimedia streaming over IP” – Colin Perkins – USC ISI

Reproducción y corrección del timing

RTP no especifica un buffer de reproducción estándar o un algoritmo de reconstrucción del timing, sino que provee la información necesaria permitiendo la diferenciación de los productos basados en él.

Existen muchos trueques a tener en cuenta, tales como: latencia vs. jitter, velocidad de reacción para el cambio y habilidad para hacer buffering.

Dentro de los diseños típicos, las aplicaciones de streaming usan grandes delay (del orden de los 10 segundos) y las aplicaciones interactivas tratan de mantener bajo el delay (decenas de milisegundos).

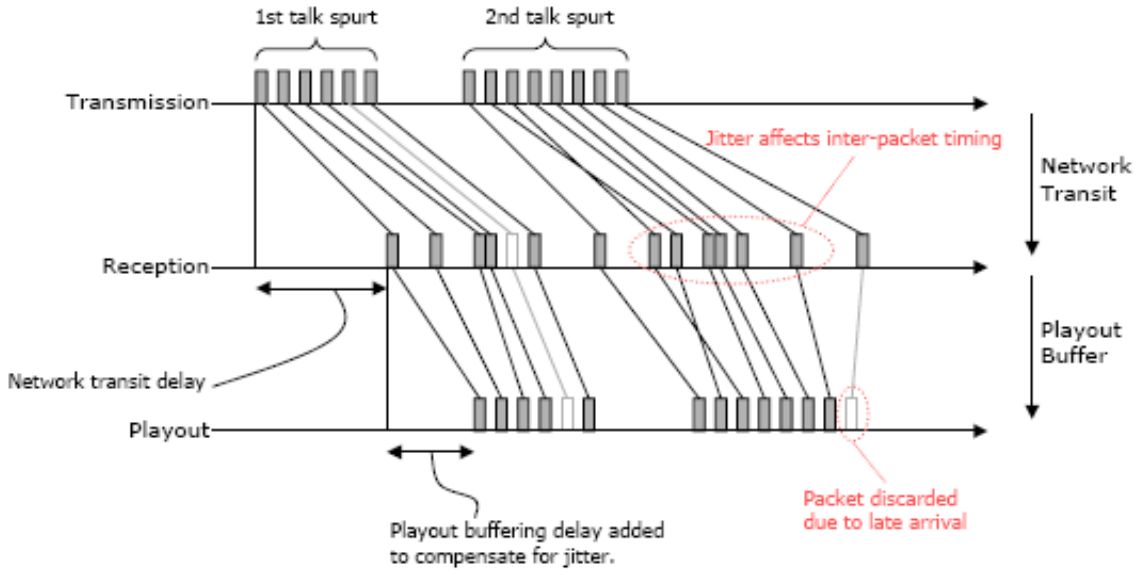


Diagrama del trabajo “RTP: multimedia streaming over IP” – Colin Perkins – USC ISI

Corrección de error

Para corregir los errores se puede usar retransmisiones de manera limitada, Forward Error Correction (independiente de la multimedia), protección de error desigual (unequal error protection) e interleaving. Todo esto agrega una confiabilidad limitada al protocolo RTP.

A continuación se describen algunos métodos o características:

- *Retransmisión en RTP*

Las retransmisiones deben ser limitadas, dado que la confiabilidad se opone a la entrega a tiempo, además no se pretende reinventar el protocolo TCP. Las retransmisiones se implementan mediante el back channel provisto por RTCP y además se puede modificar las reglas de timing de RTCP para permitir feedbacks más tempranos.

- *Perfil de feedback RTCP*

Los reportes RTCP se mandan de manera usual. El feedback puede ser enviado tempranamente: se ignora la regla de los 5 segundos, se pide prestado ancho de banda del siguiente intervalo de reporte, retrasa el siguiente reporte y envía un paquete mínimo de reporte.

Existe un feedback de capa de transporte mediante números de secuencia NACK y ACK.

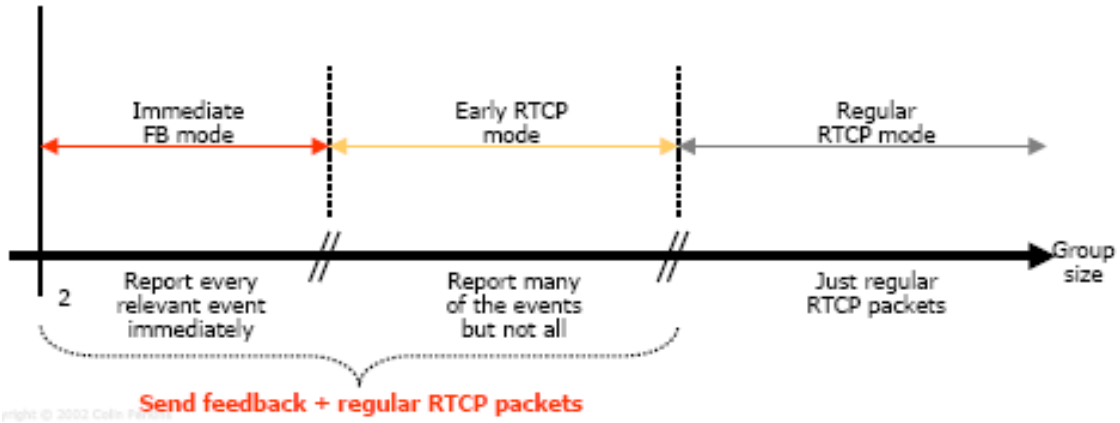


Diagrama del trabajo “RTP: multimedia streaming over IP” – Colin Perkins – USC ISI

- *Forward Error Correction*

Las retransmisiones se basan en el feedback de los receptores, por lo tanto se debe requerir que los paquetes perdidos sean renviados.

En muchos casos trabaja bien, pero hay dos escenarios donde la retransmisión es ineficiente:

- Cuando el RTT es grande
- Cuando existen muchos receptores y los eventos de pérdida son independientes

Una alternativa es que el emisor agregue datos redundantes al stream multimedia, y que los receptores usen esta información para corregir los errores sin necesidad de contactar al emisor.

FEC es una técnica bien conocida en la capa de enlace, y que también puede ser aplicada en el nivel IP o niveles superiores.

- *FEC específico para la multimedia*

  Algunos CODEC's pueden ser naturalmente tolerantes a las pérdidas, y se diseñan los formatos de payload para sacar provecho de esta característica, como el caso de la redundancia de audio/video. Por ejemplo, es el caso del formato de payload RTP para datos redundantes de audiodante [108].

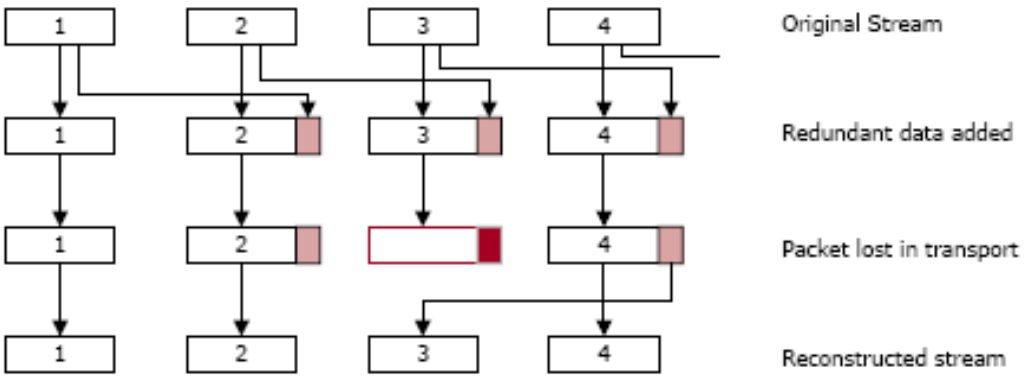


Diagrama del trabajo "RTP: multimedia streaming over IP" – Colin Perkins – USC ISI

- *FEC de paridad*

Los códigos de paridad son para proteger las comunicaciones seriales, como es bien sabido. Se puede aplicar la misma técnica a redes de paquetes y generar paridad por paquete.

Existe un estándar para el FEC de paridad [109] que permite una operación de paridad flexible. Además existen estándares para codificación Reed-Solomon.

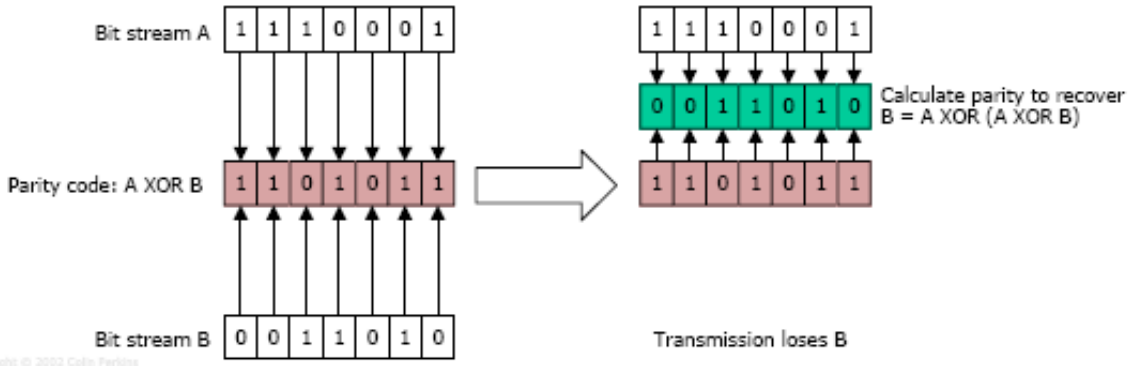


Diagrama del trabajo “RTP: multimedia streaming over IP” – Colin Perkins – USC ISI

- *Protección de error desigual*

No todos los datos en los paquetes son igualmente importantes: los headers y las actualizaciones de los estados del CODEC son vitales, y los datos multimediales son de menor importancia.

Algunos enlaces tienen una alta tasa de error, causada por la corrupción de los paquetes, detectada por checksum a nivel UDP y en consecuencia hay paquetes que se descartan.

De lo de arriba expuesto, se desprende que hay un checksum parcial a nivel UDP.

- *Interleaving (interpolación)*

El ocultamiento y la corrección de la pérdida de paquetes trabaja mejor cuando la pérdida es en forma aislada, como ser el caso de pérdidas en un solo paquete.

En el caso de Internet, la pérdida de paquetes es explosiva (bursty). En este caso, se puede aplicar interleaving para hacer que las explosiones de pérdidas aparezcan como pérdidas aleatorias a pesar que esto agregue un delay considerable.

Este método es popular con las aplicaciones de streaming y forma parte de muchos formatos de payload para audio.

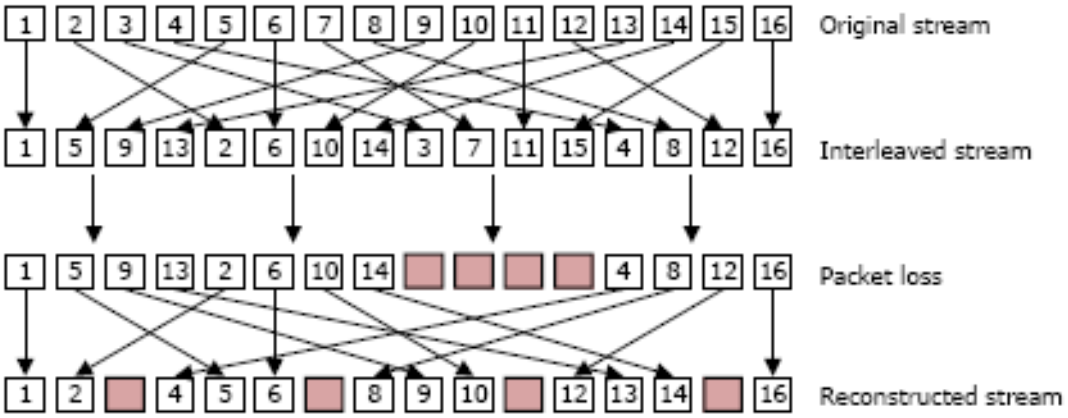


Diagrama del trabajo “RTP: multimedia streaming over IP” – Colin Perkins – USC ISI

- *Conclusiones acerca de la corrección de errores*

Existen muchas investigaciones que se están llevando a cabo y elaboración de estándares acerca de la corrección de errores en RTP, como ya se ha dicho anteriormente.

La red no tiene que ser perfecta; si es adecuada para la transmisión de datos probablemente sea adecuada para la transmisión de video, a menos que uno tenga requerimientos muy estrictos.

Además, es aceptable el uso de “overprovisionig” de la red para lograr calidad de servicio, y no existe una necesidad real de implementar técnicas de QoS tales como RSVP o DiffServ.

Las aplicaciones bien diseñadas pueden tolerar pérdidas significativas, a menudo del orden del 5%.

Control de Congestión

En todo lo que se mencionó recién se asumió que el tráfico se comporta de una buena manera, o bien que los flujos responden a la congestión en la red o bien que se emplea QoS o control de admisión de flujos.

Una red IP, como dije antes, provee un servicio “best-effort” de paquetes switcheados, sin control de admisión, la red acepta todos los paquetes por igual y trata de enviarlos a sus destinos. Sin embargo, no hay garantía de entrega de los mismos, dado que si el enlace estuviese congestionado habría un exceso de paquetes descartados. El protocolo de transporte deberá detectar pérdidas y entonces reducir su velocidad para permitir que la congestión desaparezca; TCP lo hace automáticamente pero RTP no.

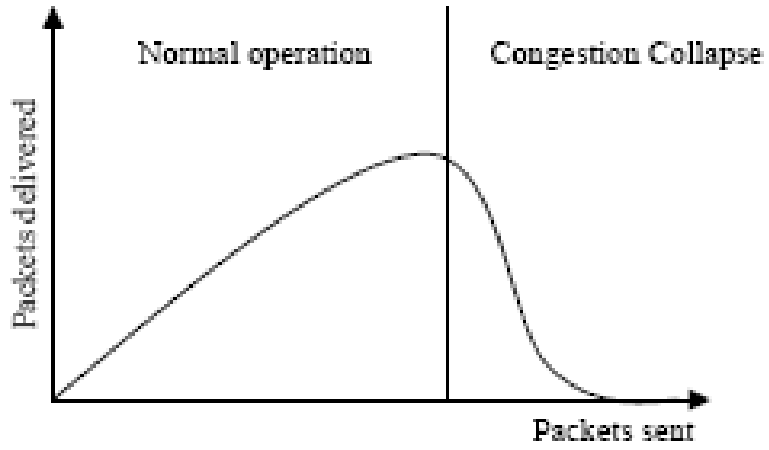


Diagrama del trabajo “RTP: multimedia streaming over IP” – Colin Perkins – USC ISI

Como RTP no tiene un mecanismo para reducir la congestión, entonces la adaptación deberá ser hecha por la aplicación. Así los posibles cambios de velocidad dependen del CODEC y existirá un loop de feedback complejo entre el CODEC y la red. Para RTP, implica que los emisores deberán observar el feedback proveniente de los receptores: si la fracción de pérdida no es igual a cero habrá que considerar enviar menos paquetes, y cuando la pérdida decrezca habrá que considerar aumentar la velocidad de envío de paquetes.

Un método de control de congestión es el denominado TCP Friendly Rate Control (TFRC), mediante el cual es posible predecir el throughput promedio a largo plazo de TCP:

$$X = \frac{s}{R\sqrt{\frac{2p}{3}} + 3p(1+32p^2)\cdot T_{rto}\sqrt{\frac{3p}{8}}}$$

Se obtiene el througput (X) en términos de la tasa de pérdidas observada (p), el RTT y el tamaño de paquete (s).

TFRC es la mejor práctica actual que se puede usar como esquema de control de congestión para flujos multimedia.

Pero también existen limitaciones del control de congestión, que son las siguientes:

- Los algoritmos “amigables” con TCP son nuevos y están en etapa de evolución, por lo tanto hay un desarrollo muy limitado y no está en claro si han alcanzado su forma final

- Las interacciones entre CODEC y red no están muy bien definidas, dado que no está claro cuán lento responde este esquema

- Además hay aspectos de factores humanos que juegan un rol clave, como ser el control de congestión implica una calidad variable, que puede ser molesta a menos que la velocidad de cambio sea lenta

Protocolo de control RTP (RTCP)

Se encarga de la recepción del feedback con parámetros de calidad de servicio, como la fracción de pérdida de paquetes y el jitter promedio. Cuenta además con descripciones opcionales del origen como nombre, ubicación, dirección de mail y número de teléfono. Además implementa un mapeo del reloj de los datos multimedia hacia una base de tiempo externa, e implementa una administración de membresía.

Formato de Payload

Provee la capa de adaptación entre un CODEC en particular y RTP, y está optimizado para tener mayor robustez frente a la pérdida de paquetes. Existen muchos formatos de payload, como por ejemplo: H.261, H.263, M-JPEG, MPEG-2, MPEG-4, BT.656 y SMPTE-292 –entre otros-.

Perfil RTP

Define el uso de RTP en escenarios particulares de aplicación. Tiene parámetros razonables por defecto y hace una adaptación a condiciones inusuales como multicast con origen único, operación sin back channel (enlace de comunicaciones usado por el cliente para enviar comandos de control del video stream) y operaciones seguras y autenticadas –entre otras características-. Además provee un espacio de nombres para los formatos de payloads.

6.6.Perfiles RTP

RTP es un protocolo incompleto y así tiene limitaciones. En primer lugar, el estándar no especifica algoritmos para la reproducción multimedia y regeneración del timing, sincronización entre streams multimedia, ocultamiento y corrección de errores, o control de congestión. Estas características son propias del diseño de la aplicación. Lo que sí hace RTP es proveer la información adecuada para que estos algoritmos cumplan su función en forma correcta.

En segundo lugar, algunos detalles del transporte son dejados de manera abierta para que sean modificados por perfiles y formatos de payload. Estos incluyen características tales como resolución del timestamp, marcado de eventos significativos dentro del sream multimedia, y uso del campo de tipo de paylaod (PT). Las características que pueden ser especificadas por los perfiles RTP incluyen las siguientes [47]:

- Mapeo entre el identificador del tipo de payload del hedear RTP y las especificaciones de formato de payload las que describen cómo se usan los CODECs multimedia con RTP. Cada perfil referenciará múltiples formatos de payload y puede indicar cómo los protocolos de señalización en particular –como por ejemplo SDP- son usados para describir el mapeo.

- El tamaño del campo identificador del tipo de payload del header RTP, y el número de bits usados para marcar eventos de interés dentro del stream multimedia.

- Agregados al header fijo del protocolo RTP, si es que el header no es suficiente para una clase de aplicación particular.

- El intervalo de reportes RTCP, por ejemplo para tener un feedback con mayor frecuencia a expensas de tener un overhead adicional.

- Limitaciones en los tipos de paquetes RTCP que se usen, en el caso que alguna de la información provista no sea de provecho para la clase de aplicación en cuestión. Además, un perfil puede definir extensines a RTCP para reportar información adicional.

- Mecanismos de seguridad adicional, como por ejemplo algoritmos nuevos de autenticación y encriptación.

- Mapeo de RTP y RTCP sobre protocolos de transporte de bajo

6.7.Formatos de payload RTP

Los formatos de payload definen cómo son transportados los tipos de multimedia en particular dentro de RTP [47]. Los formatos de payload son referenciados por los perfiles RTP y además pueden definir ciertas propiedades del protocolo de transferencia de datos RTP.

La relación principal entre un formato de payload RTP y un perfil es un espacio de nombre (namespace), aunque el perfil puede además especificar algunos comportamientos generales para los formatos de payload. El namespace relaciona el identificador de tipo de payload en los paquetes RTP con las especificaciones de formato de payloads, permitiendo que una aplicación relacione los datos a un CODEC multimedia en particular. En algunos casos el mapeo entre tipo de payload y format de payloas es estático, y en otros es dinámico a través de un protocolo de control fuera de banda. Por ejemplo, el perfil RTP para audio y video conferencias con mínimo control define un conjunto de asignaciones estáticas de tipo de payload y un mecanismo para mapear entre un tipo MIME qaue identifica un formato de payload y un identificador de tipo de paylaod que usa SDP (Session Description Protocol).

Un formato de payload especificará el uso de ciertos campos del header RTP y además puede definir un header de payload adicional. La salida

producida por un CODEC multimedia es traducida en una serie de paquetes de datos RTP –algunas partes mapeadas sobre el header RTP, otras en el header del payload, y la mayoría dentro del payload data-. La complejidad de este proceso de mapeo depende del diseño del CODEC y del grado de resiliencia de error requerido.

De una manera simple, un formato de payload sólo define el mapeo entre el media clock y el timestamp RTP, y ordena que cada frame de la salida del CODEC se ubique directamente dentro de un paquete RTP para ser transportado.

Han sido definidos muchos formatos de payloads que se corresposnden con la diversidad de CODECs en uso hoy en día, y muchos más están en desarrollo.

También hay formatos de payload que especifican esquemas de corrección de error.

7.Experiencias en el mundo en HDTV sobre redes Internet de próxima generación

En esta sección voy a describir algunos de las experiencias más destacadas en el mundo sobre streaming de HDTV sobre redes Internet de próxima generación.

En este punto vale la pena aclarar el concepto de “streaming de audio y video”: es la capacidad de reproducir sonido y video en tiempo real al tiempo que son transferidos sobre una red –en este caso Internet de próxima generación-, en contraposición con la descarga y el almacenamiento de los mismos en un archivo para luego ser ejecutado.

7.1.Universidad de Washington

En esta experiencia [48] voy a hacer mayor hincapié porque es tomada como referencia obligada por todos los proyectos actuales relacionados al tema.

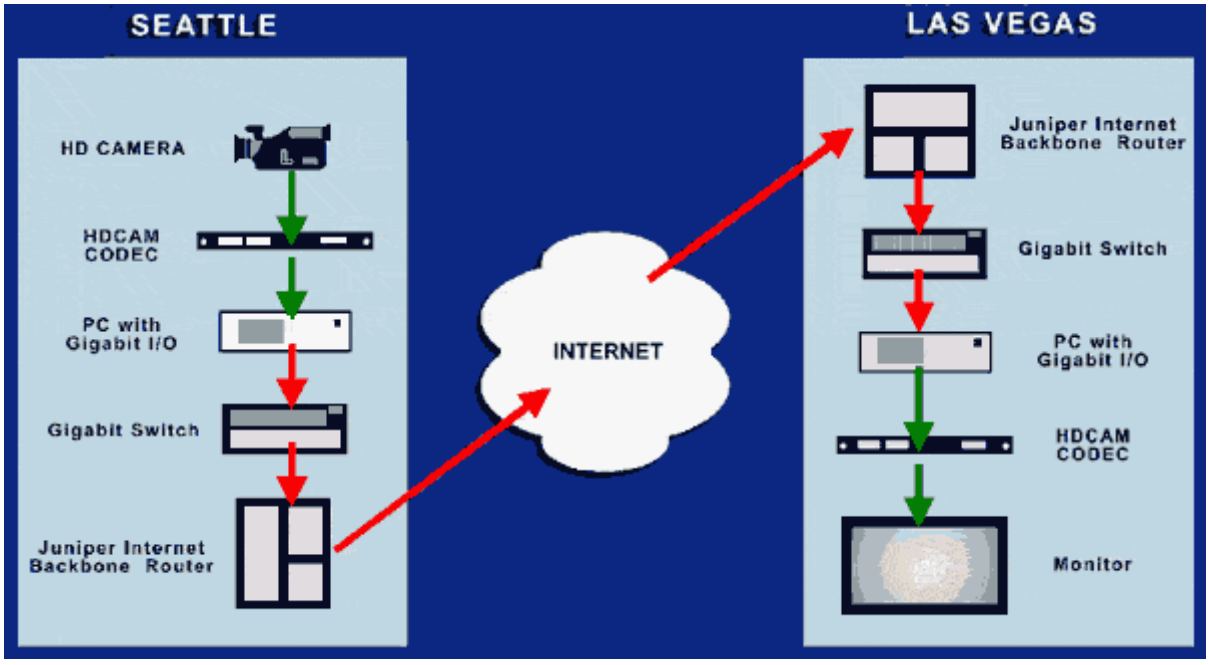
Con un esfuerzo en conjunto de la Universidad de Washington [49], el Research Channel [50] y Sony Electronics [51], se implementó la nueva tecnología propietaria denominada por ellos Internet HDTV, un software desarrollado para enviar señales de HDTV con studio-quality a través de Internet2. Esta calidad de video implica el uso de un algoritmo de compresión.

La National Association of Broadcaster [52] conference –NAB 2000- sirvió de lugar de reunión para la demostración del evento en cuestión: una producción en vivo de audio y video de alta definición donde todas las fuentes de audio/video, instalaciones de producción y la estación transmisora están conectados mediante enlaces Internet de área amplia únicamente.

Ya en el evento de la Super Computing 1999 se había desarrollado una demostración de la UW en el que se transmitió video comprimido en paquetes IP.

Años más tarde, el 13 de Abril de 2000, la aplicación Internet HDTV envió múltiples streams HDTV a más de 200 Mbps –se llegó a 270 Mbps- desde los estudios KING 5 en Seattle y el campus en Seattle de la Universidad de Washington hasta la plataforma de post-producción de Sony en la sala de exposición de la NAB 2000 en las Vegas. Los streams fueron switcheados para para producir un programa de televisión en vivo para la presentación de Sony en la NAB 2000 y fueron enviados de regreso a Seattle para ser difundidos en el canal de televisión digital de KING 5, creando de esta manera la primera difusión de televisión en vivo de un programa de HDTV producido y transmitido sobre enlaces de Internet de alta capacidad.

Esquema de la demostración:



conexiones SDTI
conexiones Gigabit Ethernet

El proyecto Internet HDTV resultó la primer colaboración en banda ancha (gigabit) entre la industria del broadcast y la comunidad de Internet. Este sistema fue especialmente diseñado para trabajar con equipamiento Sony usado en muchas instalaciones de broadcast de HDTV: HDCAM HDW-500 VTRs, HD Switchers HDVS7000, HDCAM CODECS HKPFE/D270, HD Studio Cameras HDC750, HD Portable DXC-H10 Cameras y HD Monitors BVM-D20EIU.

El sistema hizo uso exclusivo de tecnología estándar de transporte en Internet, lo que quiere decir que no hizo falta utilizar tecnologías de networking ATM (Asyncronous Transfer Mode) ni específicas para video. El software Internet HDTV integró dos disciplinas y demostró que las redes Internet son capaces de transportar hasta incluso el tráfico multimedia más demandante.

Para la demostración de la NAB 2000, la presentación de Internet HDTV estuvo completamente integrada dentro de un ambiente de producción de broadcast. Cada cámara en los estudios de KING 5 fue conectada a una computadora usando el software Internet HDTV y luego se enviaron los streams a Las Vegas.

Cámaras y servidores de video adicionales en el campus de la Universidad de Washington también transmitieron hacia Las Vegas, donde todos los streams aparecieron en la plataforma de post-producción de Sony. El video switcher Sony HDVS7000 fue usado para crear un programa de televisión que luego fue enviado de regreso a Seattle para ser difundido en vivo por el canal DTV (Digital TV) de KING 5.

En la presentación de NAB 2000 se utilizó equipamiento comercial disponible para originar y crear los HD video streams. La tecnología fue necesaria para tomar los HD streams estándares, colocarlos dentro de datagramas Internet y manejar la corrección de errores. Para lograr este objetivo, los ingenieros de la UW combinaron hardware de video especializado y computadores personales de alto rendimiento con un software original desarrollado a lo largo del año 2001.

El software Internet HDTV fue diseñado para lograr un balance entre el número de paquetes de datos enviados, el tamaño de los paquetes de datos y el tipo y cantidad de corrección de error incluida. El requerimiento fue estar habilitado para recuperarse de los niveles de pérdida de paquetes encontrados en condiciones normales de redes.

La señal original fue de 1080i HD de payload (1920 pixels por 1080 líneas, entrelazadas a 60 frames por segundo), transportada en un enlace HD-SDI link de 1.5 Gbps. Todo el audio usado en la transmisión fue sin comprimir, y de esta manera se redujo la latencia. Los dispositivos fuente fueron Sony HD video cámaras y Sony HDW-500 HDTV VTRs (VTR: Video Tape Recorder, un dispositivo que graba video). Los payloads fuente de audio y video fueron introducidos en sistemas Sony HDCAM los cuales usaron técnicas digitales de reducción de velocidad y compresión de video para producir un HDCAM video stream de 140 Mbps (más aproximadamente 3 Mbps para la información de audio). **El video HDCAM con el audio embebido** fue colocado en un enlace SMPTE 305M Serial Data Transport Interface (SDTI).

SMPTE son las siglas de Society of Motion Picture and Television Engineers [2], es una organización que provee un estándar que permite transferencias de velocidades mayores que tiempo real, entre varios servidores y entre sistemas de edición basados en discos, tapes de adquisición y servidores, soportando velocidades de 270 Mbps y 360 Mbps. Con tasas de transferencias típicas de video comprimidas y en tiempo real en el rango de 18 Mbps a 25 Mbps a 50 Mbps, el payload de SDTI de 200+ Mbps puede acomodar transferencias hasta cuatro veces la velocidad normal. El estándar SMPTE 305M describe el ensamble y desensamble de un stream de palabras de datos de 10 bits que conforman a reglas SDI. Las palabras de datos del payload pueden ser de hasta 9 bits. El 10º bit es un complemento del 9° bit para prevenir la ocurrencia de valores ilegales de SDI. El payload básico se inserta entre SAV (Start of Active Video, una señal de sincronización usada en componentes de video digital) y EAV (End of Active Video). El header inmediatamente después de EAV provee una serie de flags y ID de datos para indicar lo que esta por venir así como también contar líneas y CRCs para chequear la continuidad de los datos.

Se usó una computadora con una tarjeta SDTI para leer los datos y prepararlos para la transmisión sobre Internet. La computadora agrega header e información sobre corrección de error antes de colocar el payload SDTI en paquetes IP/UDP (User Datagram Protocol). La computadora luego envía aquellos paquetes a través de una interface Gigabit Ethernet, produciendo un stream de datos de 2000+ Mbps. Las estaciones de trabajo transmisora y receptora cooperan para la recuperación de errores y retransmiten los paquetes perdidos (dropped packets).

A lo largo de la transmisión de estos streams, se transmitieron streams adicionales desde servidores de video desarrollados en la UW. Siguiendo el

mismo principio delineado en el párrafo anterior, los servidores de video fueron usados para granar video a velocidades HDCAM en los meses anteriores. Durante la presentación, los servidores de video leyeron datos desde sus discos en lugar de una fuente externa. Los frames de video comprimidos fueron entonces empaquetados y transmitidos sobre la red.

Se usó equipamiento de red basado en IP para rutear los datos desde Seattle a Las Vegas. Las estaciones de trabajo transmisoras fueron conectadas a puertos Gigabit Ethernet en King 5 o en la red de campus de la UW, los cuales fueron conectados de a turnos al Pacific/Northwest GigaPoP en el centro de Seattle vía conexiones de Internet del tipo OC-48c Packet over SONET provistas por Electric Lightwave. Desde Seattle, los datos atravesaron otra conexión OC-48c SONET –esta misma montada arriba de un circuito de fibra óptica Dense Wave División Multiplexing (DWDM) y provista por Enron Broadband Services- y finalmente arribaron a un router IP provisto por Juniper Networks ubicado en el piso de exhibición del NAB 2000 en Las Vegas. DWDM es una variedad de WDM que usa múltiples longitudes de ondas (o canales) espaciados muy cercanamente en la región de 1550 nm del espectro infrarrojo.

En Las Vegas, los paquetes de datos fueron recibidos por otro conjunto de computadoras especialmente configuradas. Luego de tener en cuenta cualquier corrección de error y requerimientos de retransmisiones, los datos HD comprimidos son enviados a través de la interface SDTI a decodificadores externos Sony HDCAM. Una vez descomprimidos, las señales HDTV fueron provistas a un equipo HD de postproducción.

Al año siguiente, precisamente el 11 de Noviembre de 2001, la UW junto a Tektronix entre otros, llevó a cabo la transmisión de HDTV sin comprimir desde los laboratorios de la UW en Seattle (Washington) hasta Denver donde se llevaba a cabo el evento Super Computing 2001. Este trabajo está mejor desarrollado en el apartado 4.3.

7.2.2NetFX

Un producto de la empresa 2NetFX [53], denominado Thundercastip Advanced Media Server, fue desarrollado para enviar HDTV comprimido sobre IP. El sistema usa el algoritmo de compresión MPEG-2 a 19.2 Mbps (HDTV Broadcast) usando el payload estándar RTP [54].

El uso de compresión agrega latencia y hace que este sistema no sea adecuado para entornos de edición de video o donde se necesite una alta calidad de imagen como por ejemplo en telemedicina.

7.3.Tektronix

Un prototipo desarrollado por Tektronix [55], denominado Universal Network Access System (UNAS) [56] usa hardware propietario para enviar HDTV sobre una interface OC-48 POS. El sistema desarrolla una emulación de circuito de SMPTE-292M sobre IP a 1.5 Gbps usando un payload RTP específico [70] desarrollado en forma conjunta con la UW. En este momento existe un draft más actualizado de este payload, desarrollado por el IETF.

Volviendo al sistema de transmisión de HDTV sobre IP, el mismo fue demostrado en la conferencia Super Computing 2001, en el mes de Noviembre.

El desarrollo fue parte de los proyectos UNAS y USC/ISI's Next Generation Internet Multimedia Applications and Architecture de DARPA NGI. Esta demostración se basó en el trabajo de la UW que fue pionera en la transmisión de HDTV sobre IP, en la presentación de Super Computing 1999.

Se usaron para las pruebas las redes Pacific Northwest GigaPoP [58] y Mid-Atlantic Crossroads y el backbone Abilene de Internet2. Durante la presentación **se envió contenido de video a 1.5 Gbps** desde los laboratorios de la UW en Seattle –Washington- hasta el extremo receptor en el evento SC2001 en Denver, a través de una red IP sobre fibra óptica de la empresa Level 3 [59]. El formato de las señales HDTV de origen fue el SMTPE 292M, estándar universal que define el intercambio de HDTV sin comprimir entre varios tipos de equipamientos de video (cámaras, codificadores, VTRs, etc.).

El Research Channel proveyó el contenido de streaming HD usando la red de Pacific Northwets GigaPoP, y la UW aportó streams de video de sus servidores multimedia. Por su parte la tecnología desarrollada por Tektronix permitió que el video fuera procesado como paquetes de datos para ser enviados, y luego recibidos y compilados como streams de video nuevamente. Esta misma tecnología además fue usada para comparar paquetes en la entrada y la salida para determinar si los paquetes se perdieron o fueron reordenados durante la transmisión.

El proyecto UNAS DARPA/ITO prevé ser un elemento de red configurable que resida en los bordes de Internet y se adapte a los muchos protocolos de red existentes, activando el desarrollo de nuevas aplicaciones y servicios.

La tecnología UNAS actúa como una rampa para aplicaciones que van desde la computación distribuida hasta la telemedicina.

Tektronix tuvo un rol primario en la ingeniería y desarrollo del Universal Network Access Engine (UNAE) para el sistema. El UNAE es un elemento importante para la construcción de dispositivos de borde de red tales como adaptadores de terminales, multiplexores de servicios, switches de borde y equipo de monitoreo de QoS. La flexibilidad de UNAE ayuda a los arquitectos de Internet a diseñar y testear nuevos protocolos optimizados para redes ópticas.

A la fecha de la demostración, la arquitectura UNAS soportaba velocidades de hasta 2.5 Gbps, una velocidad apropiada para el experimento dado que el tráfico del video payload y de encapsulado fue mayor que 1.5 Gbps.

7.4.NTT (1ra. Experiencia)

La experiencia que a continuación se describe [60], demostrada en Tokio (Japón) en Octubre de 2001, fue la primera en el mundo en transmitir HDTV sin comprimir a una velocidad de transferencia de 1.5 Gbps a lo largo de una larga distancia -20 Km- y sobre una red Internet. Un mes después se realizó una experiencia similar pero por parte de la UW y Tektronix, citada en el párrafo anterior.

NTT Laboratorios [61] desarrolló un sistema sobre una PC comercial Pentium-III multi-procesador (2 x 1 Ghz) corriendo el sistema operativo Linux 2.4.2, con una tarjeta de captura HDTV comercial, pero usando una tarjeta de red propietaria de super alta velocidad.

El sistema fue probado entre NTT Musashino R&D Center (Musashino Center in Tokio) y la Universidad de Electro-Comunicaciones, ambas separadas por una distancia de 20 Km. Las imágenes HDTV fueron transmitidas sobre una conexión IP Internet usando una línea de fibra óptica de 2.4 Gbps, y se verificó con éxito que la transmisión fue recibida en forma estable.

Las imágenes fueron obtenidas de una cámara de video HDTV, luego empaquetadas en datagramas IP en la PC transmisora, luego transmitidas por la conexión de súper alta velocidad, luego recibidas y desempaquetadas por la PC receptora, y finalmente mostradas en un monitor HDTV.

A los efectos de lograr que el sistema soporte streams de alta velocidad es que se acondicionó el sistema operativo y el programa de aplicación: se

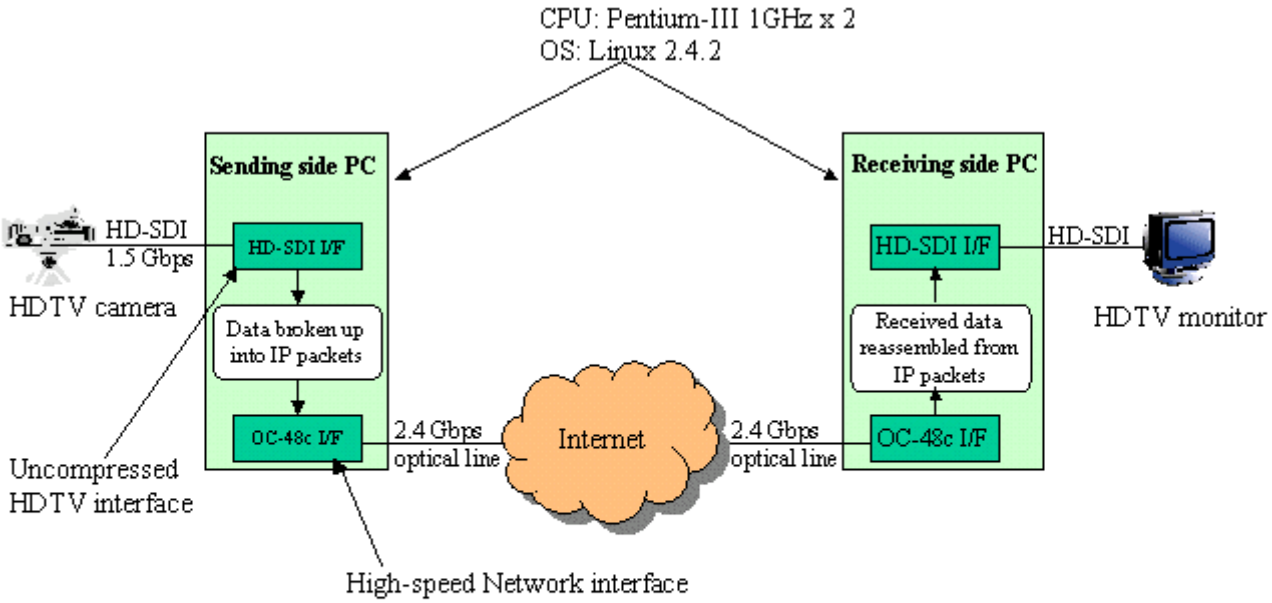
optimizaron el scheduling del bus interno y el acceso a memoria, y se adoptó un procesamiento paralelo con múltiples procesadores. Con este desarrollo se logró éxito en la reducción de la carga de procesamiento por medio de la transmisión de paquetes IP más grandes (MTU mayores a 1500 Bytes) y también se demostró que se es capaz de transmitir datos a tasas de transferencia del orden de los Gbps usando PCs disponibles en el mercado.

La interface de red fue desarrollada por NTT Network Innovation Laboratories basada en IP y soporta una super alta velocidad (2.4 Gbps), ajustándose al estándar SDH (Synchronous Digital Hierarchy). También se ajusta al protocolo MAPOS (Multiple Access Protocol over SONET/SDH) desarrollado por NTT para habilitar capacidades de acceso multi-punto. Adicionalmente, el protocolo IP fue usado en las capas superiores de la red, y de esta manera se pueden conectar múltiples PCs a través de switches o routers por medio de conexiones IP, y el video HDTV sin comprimir puede ser combinado o mezclado con cualquier otro tipo de datos para ser transmitidos.

Vale la pena destacar que SDH es un protocolo de Nivel 1 o Físico (modelo OSI), que trabaja sobre líneas digitales ópticas y acepta distintas jerarquías de multiplexación de velocidades de transmisión: 155 Mbps, 622 Mbps, 2.4 Gbps y 10 Gbps.

Por otro lado, MAPOS (Multiple Access Protocol over SONET/SDH) es un protocolo de Nivel 2 o de Enlace (modelo OSI) para comunicaciones de datos de súper altas velocidades usando SONET/SDH para la capa física. De esta manera MAPOS, por medio del uso de SONET/SDH, puede ser usado como un protocolo de red de alta velocidad que soporta diversas aplicaciones de redes LANs hasta WANs.

Esquemas del sistema:



The system used in the present trials used PCs with an interface board for uncompressed HDTV and a high-speed network interface. The high-speed network interface developed by NTT accommodates 2.4-Gbps fiber-optic lines, a capacity that is equivalent to 18,576 ISDN lines. The pilot implementation essentially consisted of two PCs providing Internet connectivity over a fiber-optic line that was deployed for the trial. Video content shot with an HDTV videocam at the Musashino R&D Center was broken up into IP packets by the PC on the sending side, and output on the fiber-optic line. Received by the PC at the University of Electronic Communication, the data in IP packets was reassembled, and displayed on an HDTV monitor. Actual operability of the system and the stable delivery of video contents were verified by this work.

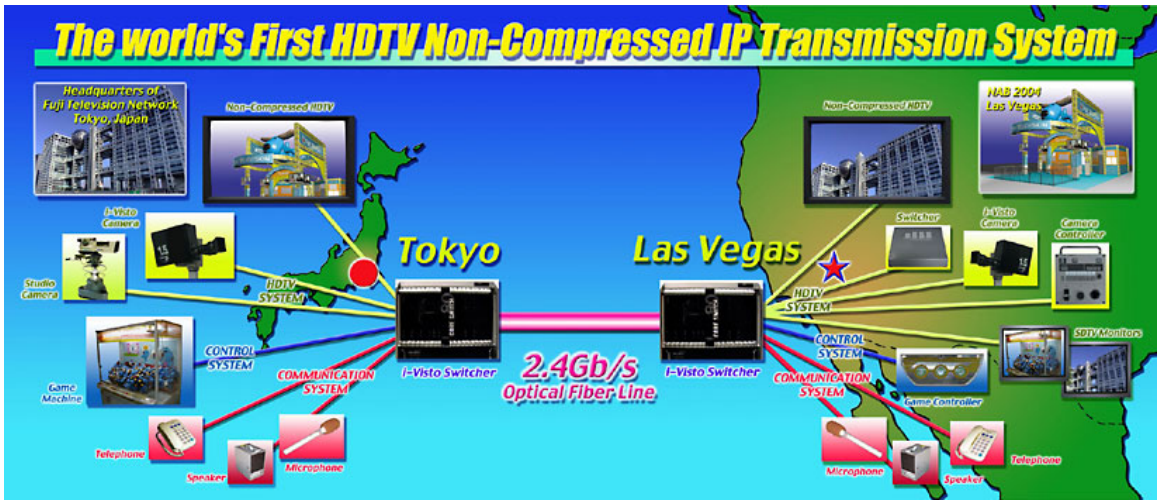
7.5.NTT (2da. Experiencia)

NTT Communications Corporation [61] y Fuji Televisión Network [62] realizaron un estudio de factibilidad para transmisiones bilaterales en vivo de banda ancha de **HDTV sin comprimir** usando las líneas de fibra óptica de 2.4 Gbps de NTT Com.

Finalmente la conexión se estableció el 19 de Abril de 2004 entre la central de Fuji TV y la conferencia de NAB 2004 en Las Vegas. Este experimento tomó lugar entre EE.UU. y Japón sobre una distancia de 15.000 Km, y representa la primera experiencia de una transmisión de HDTV a través de redes IP entre dos diferentes países.

El contenido broadcast en vivo de HDTV se transmitió usando las líneas de fibra óptica de NTT Com desde Japón hacia la conferencia NAB 2004 con calidad superior y virtualmente sin retardo. El broadcast utilizará el sistema de red IP “I-Visto” (Internet Video Studio System for HDTV Production) [63].

Esquema de la transmisión:



Además, ambas compañías instalaron en Las Vegas unos dispositivos de control que habilitaron a los usuarios de la conferencia a controlar las cámaras de HDTV IP compatibles instaladas en Japón (movimientos hacia la derecha, izquierda, arriba y abajo, y también switchear entre la primera y la segunda cámara a través de la señal IP sobre una única línea de comunicaciones).

El concepto del sistema I-Visto es aplicar la tecnología de redes IP de alta velocidad a los sitios de producción de video dentro de la industria del broadcasting. Las características más importantes de I-Visto son las siguientes:

- Trabaja con redes IP de alta velocidad y bajo retardo, basadas en tecnología MAPOS (Múltiple Access Protocol Over SONET/SDH)
- Hardware y software basado en TCP/IP.
- Soporta redes de 2.4 Gbps de velocidad de transmisión.
- Con este sistema, los miembros de la industria del broadcast tienen la posibilidad de trabajar sólo con redes basadas en tecnología IP que soportan multi-formatos y transmisiones de video en tiempo real. Hasta hoy tienen que construir y administrar diferentes redes que pueden soportar diferentes requerimientos de señales como HD-SDI, SDI, voz y control.
- Fácil integración con otros servicios como la Web. Y además, usando el protocolo MAPOS de NTT, I-Visto se puede conectar a redes SONET.

7.6.USC Information Sciences Institute de EE.UU.

En marzo de 2002 se llevó a cabo una demostración [64] de **HDTV sin comprimir** sobre una red IP y en el marco del proyecto Ultragrid [65] el cual trata sobre colaboración a distancia en un ambiente de alta definición en imagen.

Esta demostración se basó en una red IP comercial con “best-effort”. Vale decir que no hay garantía que la red descarte, duplique, retrase o desordene paquetes de datos de un mismo flujo.

El esquema de protocolos usados fue el siguiente:

- IPv4
- UDP: no tiene control de congestión y por lo tanto la entrega de paquetes se aproxima más al tiempo real si las pérdidas de paquetes son muy bajas
- RTP: provee el framing del video, identifica el origen y tipo del payload, y permite “timing-recovery” y detección de pérdida de paquetes. Típicamente corre sobre redes IP/UDP con sus limitaciones inherentes: entrega no confiable (no posee control de congestión) y con “best-effort”. Los receptores usan la información en los headers RTP para corregir la pérdida de paquetes y reconstruir el “media timing”.
- Nivel de aplicación: la salida del CODEC de video se paquetiza y fragmenta en forma inteligente, de acuerdo al formato del payload, y así cada paquete RTP puede ser decodificado en forma independiente. Por lo tanto se debe hacer con cuidado el diseño de los receptores dado que tienen la responsabilidad primaria de corregir la reproducción del video descompaginado por las características de una red IP con best-effort.

Un área crítica para este sistema fue que RTP no provee control de congestión, por lo cual hubo que elegir entre desarrollar un mecanismo de control de congestión o bien correr la aplicación en una red con el suficiente ancho de banda. **Se optó por elegir una red que tenga la suficiente capacidad para transportar el flujo de HDTV, transmitiendo sólo video**. Hay redes que demostraron suficiente capacidad para estos experimentos, como Abilene y DARPA SuperNET [66].

El test inicial fue hecho sobre la red Supernet entre ISI East (Arlington, VA), y CMU (Pittsburg, PA). El path incluye 9 hops en cada dirección y

tiene un RTT de 10 mseg aproximadamente. Se usaron enlaces OC-48 compartidos con tráfico IP, sin calidad de servicio ni control de congestión.

Tanto el transmisor como el receptor usaron placas de red Gigabit Ethernet conectadas a una estructura switcheada con tecnología Gigabit Ethernet también. La red Gigabit Ethernet se conectó a un router de borde Juniper o Cisco, según sea el extremo, los cuales se conectaron a la WAN vía una intreface OC-48 POS (2.5 Gbps Packet over SONET). Esta WAN está formada por una mezcla de routers Juniper y Cisco con interfaces OC-48 y OC-192 POS.

Para conocer la capacidad de la red, primero se realizaron mediciones de tráfico TCP y UDP con la herramienta Iperf [39]. La performance fue dependiente de la carga de la red pero se llegó a velocidades de transmisión satisfactorias.

El sistema para transportar HDTV sobre IP se diseñó con RTP/RTCP como transporte, pero se debió desarrollar un formato de payload RTP específico para tal fin.

El sistema acepta señales digitales de video SMPTE-292M y las encapsula dentro de RTP para transmitirlas sobre IP. **Se eligió el mecanismo de empaquetameinto nativo** (en lugar de emulación de circuito) para llevar a cabo la experiencia dado que no se quería regenerar la señal SMPTE-292M en el extremo receptor, sino mostrarla en un monitor de una workstation. La opción para el transporte local de HDTV fue SMPTE-292M.

Aclaro que el empaquetamiento nativo define un formato de payload RTP para transportar el video en forma directa. El empaquetamiento nativo mira el contenido del stream SMPTE-292M y actúa sobre los datos de video dentro de él. Por lo tanto, se debe definir un formato nativo para cada resolución de video.

Por otro lado, la emulación de circuito provee el envío transparente del stream HDTV, adecuado para ingresarlo en otros dispositivos.

Se usó una versión actualizada de la biblioteca RTP del proyecto UCL Robust-Audio Tool [67] para proveer el núcleo de las funciones de red del sistema. Es una implementación RTP completa, que soporta IPv4, IPv6 y Multicast.

Los requerimientos del sistema fueron que se transmita y se reciba en máquinas separadas.

En el diseño e implementación del sistema se usaron componentes de hardware comerciales disponibles en el mercado. El núcleo del sistema fue una PC de gran performance, con una placa capturadora de HDTV y otra placa Gigabit Ethernet.

A continuación se detalla alguna información relevante sobre la tecnología usada y algunos resultados logrados:

- Unicast
- Se transmitió el video stream a una velocidad de 615 Mbps
- Se envió información de HDTV de 1280x720 pixels a 45 fps
- Se usaron 8 bits por componente de color (subsampling a 24 bits)
- Se midió la pérdida de paquetes y la misma fue de aproximadamente 0.3 %, con la mayoría de las pérdidas ocurridas en paquetes aislados (no consecutivos)
- Se usó la red suavemente cargada, por lo que no hubo en forma significativa un jitter como para impactar el timing de los paquetes
- Se produjo un pequeño grado de reordenamiento de paquetes en el path, que fue aproximadamente del orden de 0.05 %, y con la mayoría de los eventos ocurridos en paquetes adyacentes. En raras ocasiones se observaron paquetes que fueron entregados con 2 o 3 lugares fuera de secuencia
- Para mejorar la calidad de HDTV sin comprimir, hubiera hecho falta un throughput de 850 Mbps y 60 fps, y 1.03 Gbps para color completo (full color) con 10 bits por componente de color

7.7.Breves conclusiones

Los sistemas de los puntos 7.3 y 7.4 han sido implementados usando hardware propietario, o sea, especialmente desarrollado. Esto hace que sean onerosos e inflexibles, comparados con los sistemas que usan componentes comerciales.

Pero la ventaja de los primeros es que tienen una mejor performance, o por lo menos hasta el año 2002 en el cual se desarrollaron las pruebas descritas.

8.Elección de un experimento real para analizar: caso TEKTRONIX

8.1.Justificación de la elección de este experimento

Elegí el experimento de Tektronix en colaboración con la Universidad de Washington, la Universidad de Southern California Information Sciences Institute (USC/ISI) y Level 3 Communications, principalmente porque transmitió HDTV sin comprimir, que es el tipo de tráfico que a mí me interesa analizar en cuanto a su comportamiento sobre una red IP y usando a Abilene como parte del backbone de la transmisión. Además está explícitamente detallado el uso del transporte RTP sobre UDP, con un formato de payload adecuado. Esto hace del experimento algo digno de ser estudiado, debido a que estoy interesado en analizar el uso de payloads RTP dedicados al video y audio.

Dado que Tektronix usó una tecnología propietaria para llevar a cabo la transmisión de HDTV, estoy interesado –como contrapartida-, en proponer finalmente un esquema de transmisión de HDTV con tecnologías estándares actuales.

8.2.Reseña del experimento

Como ya lo comenté antes, el prototipo desarrollado por Tektronix, denominado Universal Network Access System (UNAS) [56] usa hardware propietario para enviar HDTV sobre una interface OC-48 POS. El sistema desarrolla una emulación de circuito de SMPTE-292M sobre IP a 1.5 Gbps usando un payload RTP específico [70] desarrollado en forma conjunta con la Universidad de Washington. También repito que hoy existe un draft más actual del citado formato de payload, desarrollado por el grupo de audio y video de la IETF [68].

Volviendo al sistema de transmisión de HDTV sobre IP, el mismo fue demostrado en la conferencia Super Computing 2001, en el mes de Noviembre.

El desarrollo fue parte de los proyectos UNAS y USC/ISI's Next Generation Internet Multimedia Applications and Architecture de DARPA NGI. Esta demostración se basó en el trabajo de la UW que fue pionera en la transmisión de HDTV sobre IP, en la presentación de Super Computing 1999.

Se usaron para las pruebas las redes Pacific Northwest GigaPoP y Mid-Atlantic Crossroads y el backbone Abilene de Internet2. Durante la presentación se envió contenido de video a 1.5 Gbps desde los laboratorios de la UW en Seattle –Washington- hasta el extremo receptor en el evento SC2001 en Denver, a través de una red IP sobre fibra óptica de la empresa Level 3. El formato de las señales HDTV de origen fue el SMTPE 292M, estándar universal que define el intercambio de HDTV sin comprimir entre varios tipos de equipamientos de video (cámaras, codificadores, VTRs, etc.).

El Research Channel proveyó el contenido de streaming HD usando la red de Pacific Northwets GigaPoP, y la UW aportó streams de video de sus servidores multimedia. Por su parte la tecnología desarrollada por Tektronix permitió que el video fuera procesado como paquetes de datos para ser enviados, y luego recibidos y compilados como streams de video nuevamente. Esta misma tecnología además fue usada para comparar paquetes en la entrada y la salida para determinar si los paquetes se perdieron o fueron reordenados durante la transmisión.

El proyecto UNAS DARPA/ITO prevé ser un elemento de red configurable que resida en los bordes de Internet y se adapte a los muchos protocolos de red existentes, activando el desarrollo de nuevas aplicaciones y servicios. La tecnología UNAS actúa como una rampa para aplicaciones que van desde la computación distribuida hasta la telemedicina.

Tektronix tuvo un rol primario en la ingeniería y desarrollo del Universal Network Access Engine (UNAE) para el sistema. El UNAE es un elemento importante para la construcción de dispositivos de borde de red tales como adaptadores de terminales, multiplexores de servicios, switches de borde y equipo de monitoreo de QoS. La flexibilidad de UNAE ayuda a los arquitectos de Internet a diseñar y testear nuevos protocolos optimizados para redes ópticas.

A la fecha de la demostración, la arquitectura UNAS soportaba velocidades de hasta 2.5 Gbps, una velocidad apropiada para el experimento dado que el tráfico del video payload y encapsulado fue mayor que 1.5 Gbps.

8.3.Estándar de video SMPTE-292M

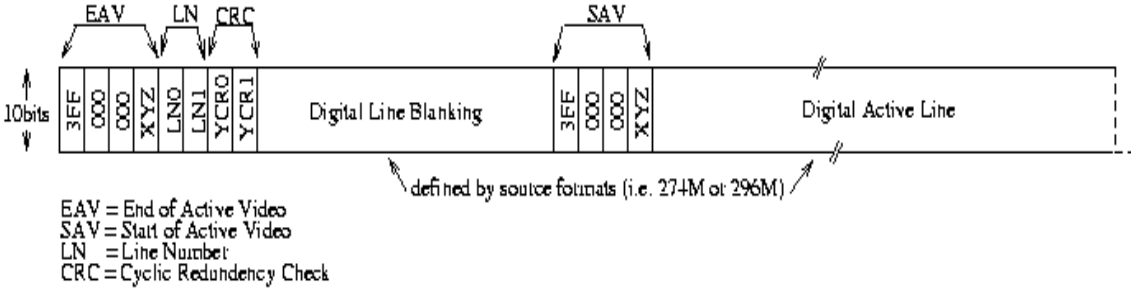
SMPTE-292M es el estándar para para la interface digital bit-serializada para los sistemas de alta definición (Bit-Serial Digital Interface for High Definition Systems).

Este estándar define una interface bit-serial digital coaxil y de fibra óptica para señales de video de HDTV operando a velocidades de datos de 1.485 Gbps y 1.485/1.001 (en la primera expresión el divisor M toma el valor 1 y en la última expresión toma el valor 1.001). Los datos bit-paralelos derivados de formatos de fuentes especificados son multiplexados y serializados en un stream de datos serial.

El **estándar SMPTE-292M** define un medio universal de intercambio para HDTV sin comprimir entre varios tipos de equipamiento de video. Un stream SMPTE-292M comprime dos streams entrelazados con datos de video, uno que contiene las muestras de luminancia (Y) y el otro las muestras de crominancia (Cr y Cb). Dado que la crominancia es sub-muestreada horizontalmente (codificación 4:2:2) las longitudes de los dos streams se corresponden. Y en forma adicional, los streams además de tener la misma longitud tienen la misma estructura. Cada stream se divide en cuatro partes:

1. Referencia de tiempo del final del video activo (EAV), número de línea y CRC
2. Digital line blanking
3. Referencia de tiempo de comienzo de video activo (SAV)
4. Digital active line

SAV y EAV contienen palabras de estado ("XYZ" en la figura de abajo) que indican qué campo de video entrelazado está presente, y la ocurrencia del campo blanking interval (field blanking interval).



Cada stream se comprime en palabras de 10 bits. Actualmente los formatos de video que pueden ser transferidos por SMPTE-292M incluyen SMPTE-260M, 295M, 274M y 296M.

Por lo tanto, cada formato de origen (source format) comprende una estructura de datos y una documentación que definen la entrada de bits en

paralelo hacia el proceso de serialización para un dado sistema de televisión de alta definición.

A continuación se detalla una tabla con los diferentes parámetros de los formatos de origen:

Reference SMPTE standard	260M		295M	274M								296M	
Format	A	B	C	D	E	F	G	H	I	J	K	L	M
Lines per frame	1125	1125	1250	1125	1125	1125	1125	1125	1125	1125	1125	750	750
Words per active line (each channel Y C_B/C_R)	1920	1920	1920	1920	1920	1920	1920	1920	1920	1920	1920	1280	1280
Total active lines	1035	1035	1080	1080	1080	1080	1080	1080	1080	1080	1080	720	720
Words per total line (each channel Y C_B/C_R)	2200	2200	2376	2200	2200	2640	2200	2200	2640	2750	2750	1650	1650
Frame rate (Hz)	30	30/M	25	30	30/M	25	30	30/M	25	24	24/M	60	60/M
Fields per frame	2	2	2	2	2	2	1	1	1	1	1	1	1
Data rate divisor (see 4.5)	1	M	1	1	M	1	1	M	1	1	M	1	M

8.4.Transporte con emulación de circuito

Un sistema para transportar HDTV sobre IP aceptará una señal de video digital y la encapsulará dentro de RTP para transmitirla sobre IP. En el lado receptor, la señal SMPTE-292M podrá ser regenerada o el video puede ser reproducido directamente. Existen diferentes alternativas para llevar a cabo este proceso, dependiendo del objetivo del transporte. Si el objeto es unir equipos existentes la aproximación más correcta podrá ser la **emulación de circuito**, donde la señal SMPTE-292M es mapeada en forma independiente a su contenido. La alternativa es una **paquetización nativa**, donde se define un formato de payload RTP para transportar el video directamente y usando SMPTE-292M en forma local.

La emulación de circuito provee una entrega transparente del bit-stream de HDTV, acorde para ingresarlo en otros dispositivos. Soporta cualquier formato soportado por SMPTE-292M, sin tener que estar adaptado a los detalles del formato. La principal desventaja es que la paquetización no tiene conciencia de los datos multimedia, no pudiendo ser optimizados en base al formato de video. Esto hace que la emulación de circuito sea algo intolerante a las pérdidas. El bit-stream SMPTE-292M no incluye marcadores de límite de frame/line (boundary markers) dado que ellos

están ocultos dentro del stream en lugar de estar en una manera explícita. La emulación de circuito permite interconectar equipos existentes dado que la red IP puede emular el circuito SMPTE-292M.

La paquetización nativa mira dentro del contenido del stream SMPTE-292M, y actúa sobre los datos de video dentro de él. De esta manera, se deben definir formatos nativos para cada posible resolución de video. Además expone el contenido de los datos para que sean manipulados por sistemas finales, en lugar de ocultarlo dentro de otra capa de framing.

En el caso de la experiencia de Tektronix se eligió la emulación de circuito para transportar el video SMPTE-292M, basándose en que se quiso interconectar equipos de HDTV ya existentes.

8.5.Formato de payload RTP para HDTV sin comprimir

Se usó un Internet draft que definía un formato de payload para el video SMPTE-292M [69], desarrollado por el IETF para encapsular HDTV sin comprimir como el definido por el estándar de la Society of Motion Picture and Televisión Engineer (SMPTE 292M). En la actualidad, constituye la RFC 3497 [70] y está catalogada como un "estándar propuesto según el RFC Editor que es el órgano que publica las RFC's y es responsable por su revisión.

SMPTE es el principal órgano de estandarización en la industria de imágenes en movimiento y el estándar SMPTE-292M define una interface digital serializada en bits para el transporte de HDTV en área local.

El estándar SMPTE 292M define un medio universal de intercambio para HDTV sin comprimir entre varios tipos de equipos de video como cámaras, codificadores, VTR's, etc. Este estándar estipula que la fuente de datos sea con palabras de 10 bits y la velocidad total sea de 1.485 Gbps o 1.485/1.01 Gbps. El cálculo es así:

- Estándar SMPTE-292M
- HDTV 1080i a 30 fps (aclaración: el estándar no usa 60 fps porque sería un throughput muy elevado)
- Pixel shape: square
- 1080 x 1920 = 2.073.600 pixels activos/frame
- 1125 x 2200 = 2.475.000 pixels totales (incluye muestras en los intervalos en blanco verticales y horizontales, dado que la mayoría de los equipos de HDTV tienen 1125 líneas de escaneado totales, con 1080 líneas para la imágen)

- Muestreo de componente de color 4:2:2 con 10 bits por componente (20 bits por pixel) para los pixels totales = 49.500.000 bits/frame
- 30 fps = 1.485.000.000 bits/seg
- Total: 1.485 Gbps
- El payload activo es de 1.244.160.000 bps

El uso de conexiones seriales dedicadas es apropiado en ambientes de estudio, pero es deseable usar redes IP para lograr una conectividad de gran ancho de banda para permitir un eficiente envío del contenido de video SMPTE-292M. Por tal motivo se desarrolló un formato de payload RTP para el formato de video SMPTE-292M.

Es importante hacer notar que los streams de video SMPTE-292M tienen una velocidad de transmisión alta y constante, y no tienen mecanismos de control de congestión. Por dicha razón, el uso de este formato de payload RTP debe ser restringido a redes privadas donde hay mucho control del tráfico o aquellas redes que puedan proveer reservación de recursos o un grado avanzado de QoS.

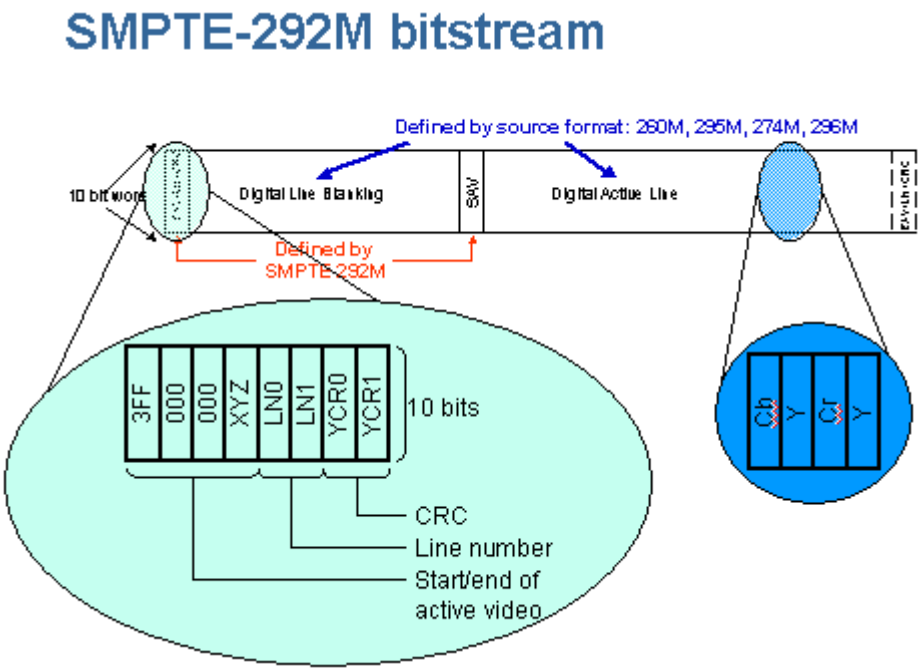
Este formato de payload RTP está destinado únicamente a HDTV sin comprimir, y no a HDTV comprimida donde mayormente se usa el algoritmo MPEG-2.

El protocolo RTP es un protocolo de "transporte" que extiende el multiplexado de puertos básicos y las funcionalidade de datagramas que provee el protocolo normal de transporte, principalmente UDP. Esta funcionalidad consiste de las siguientes características:

- Número de secuencia (sequence number) para recuperar el orden de transmisión
- Timestamp para indicar cuando la multimedia deberá ser reproducida
- Multiplexado de origen (source multiplexing) en el transmisor para permitir múltiples orígenes con la misma dirección
- Identificación del formato de payload, usando el campo PT (payload type)

En este punto es importante aclarar que mientras el timestamp de RTP es usado para colocar los paquetes de audio y video entrantes en el orden de tiempo correcto para la reproducción, el número de secuencia RTP es usado principalmente para detectar pérdidas. Además, mientras que los números de secuencia se incrementan de a uno por cada paquete RTP transmitido, los timestamps se incrementan por el tiempo cubierto por un paquete.

El campo PT es un campo de 7 bits y es válido solo dentro de cada sesión RTP para su duración definida.



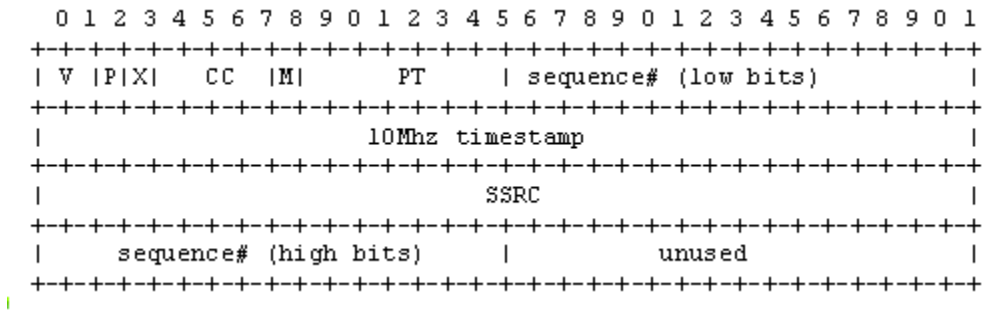
Los formatos fuente de video son: 260M, 295M, 274M y 296M.

Se hace una paquetización para su envío por medio de RTP. Una única línea SMPTE 292M es colocada dentro de un número variable de paquetes RTP:

- Un paquete RTP puede contener datos de dos líneas de video consecutivas
- Un paquete RTP puede contener datos de dos frames de video consecutivos

Las líneas de escaneo deben ser fragmentadas en pares de muestras debido a los componentes de color Y, Cb y Cr, usando submuestreo de color 4:2:2 con 10 bits por muestra (20 bits por píxel). Por lo tanto, esto resulta en una longitud del payload que es múltiplo de 40 bits (2 muestras de video).

El siguiente gráfico muestra un paquete RTP con headers y payload SMPTE-292M:



…… aquí debajo continúa con los datos SMPTE-292M …...

Es importante destacar que **los tamaños de los paquetes son constantes para una dada sesión.**

Header RTP:

Lo siguientes campos del header fijo de RTP son usados para encapsular SMPTE-292M (los otros campos del header RTP son usados de manera usual):

Payload Type (PT): 7 bits
Es un campo tipo payload asignado dinámicamente para designar el payload como SMPTE-292M.

Timestamp: 10 MHz
429 seg. de roll-around

Marker bit (M): marca el final de un frame

Sequence number (low bits): son 16 bits usados como contador de secuencias RTP

Header del payload:

Sequence number (high bits): son 32 bits para el contador de secuencias RTP

F (1 bit) y V (1 bit): son para señales de timing SMPTE-292M

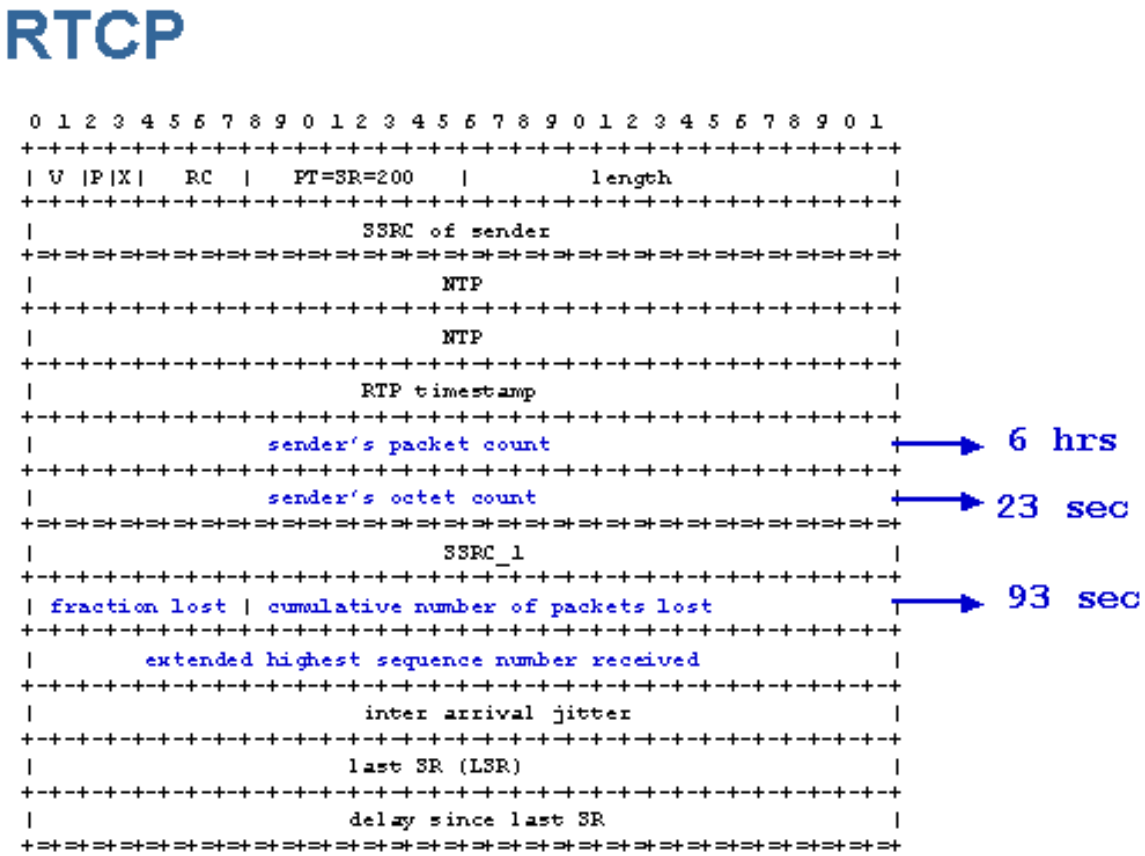
La sección de payload debe contener como mínimo 8 muestras de video (20 bytes), para que los timestamps sean únicos.

8.6.Intervalos de transmisión RTCP (Real Time Control Protocol)

El valor recomendado para un intervalo mínimo RTCP es de 5 seg. o 360/(ancho de banda de la sesión en kbps).

Para el caso de un stream SMPTE-292M sería: 360/1450195 ~ 0.2 mseg o 4028 paquetes por segundo.

El intervalo más reducido debería ser el usado, dado que en caso contrario las estadísticas se tornan menos confiables.



8.7.Registración tipo MIME

MIME Type: video/SMPTE292M

Required parameters: none

Optional parameters: length

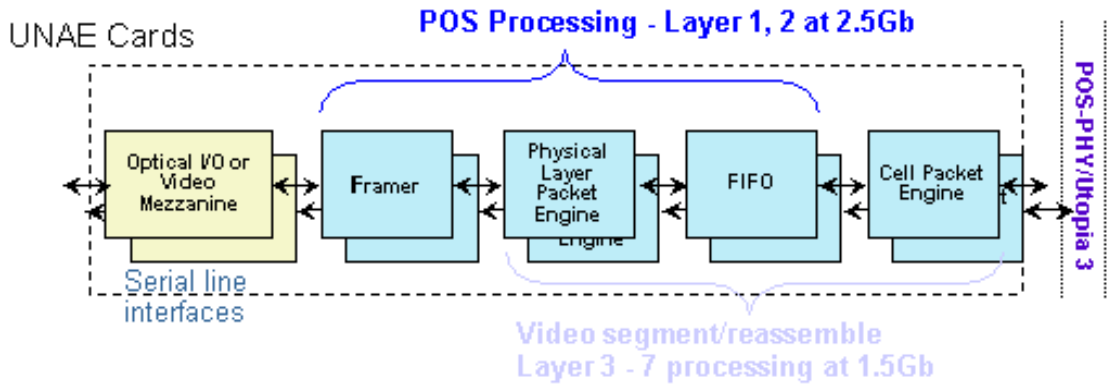
Uso de SDP (Session Description Protocol): es un protocolo para describir los parámetros de inicialización del streaming multimedia que incluye información acerca del tipo de multimedia, protocolo de transporte y formato de la multimedia, y fue publicado por la IETF a través de la RFC 2327 [71]. SDP usa el mismo espacio de nombres que el e-mail, la web y muchos otros protocolos con el objeto de identificar los diferentes formatos multimedia.

En este caso se describe así:

m=video 23456 RTP/AVP 111
a=rtpmap:111 SMPTE292M/1000000
a=fmtp:111 length=686

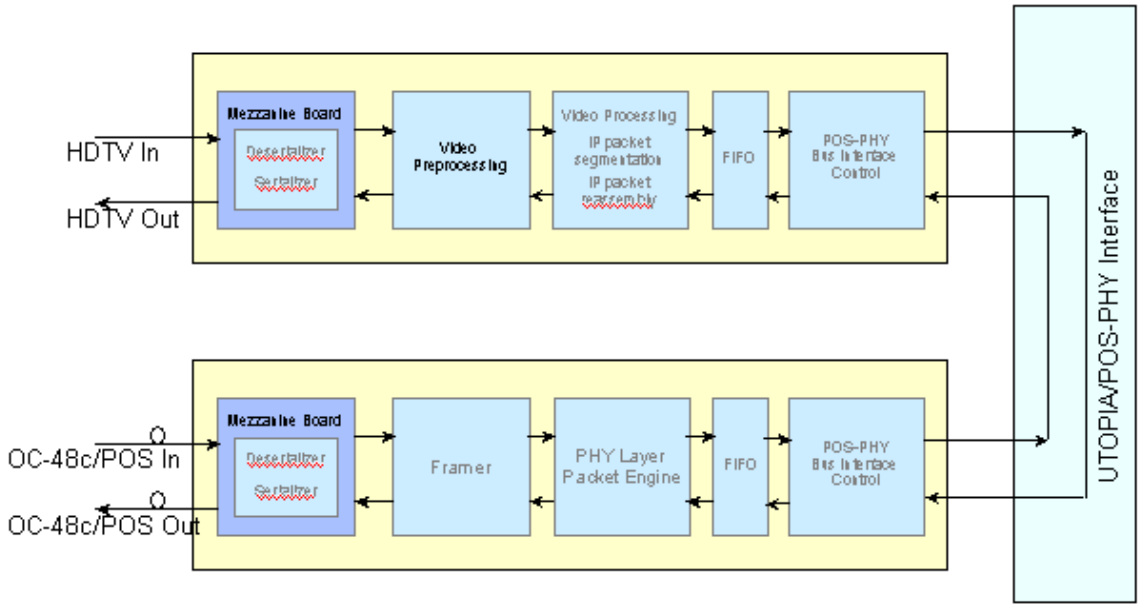
8.8.Implementación

- 1.5 Gbps video serial y interfaces de línea OC-48c (2.5 Gbps)
- Full rate packet stream a través de un bus POS-PHY
- Recuperación del clock de video desde los paquetes entrantes en el Framer
- Detección de paquetes perdidos (dropped) o reordenados por la red
- La arquitectura permite diferentes interfaces de video, audio y red
- Esquema propietario

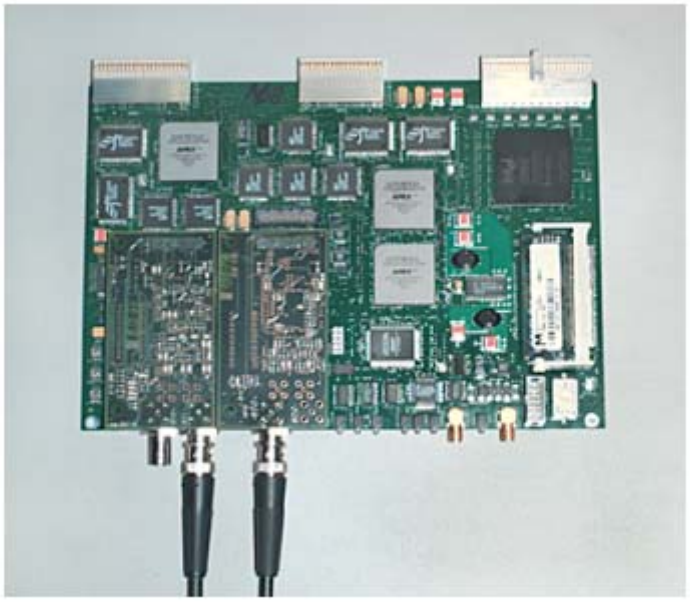


Como se mencionó antes, el experimento de Tektronix se basó en el Universal Network Access Engine (UNAE), que es un elemento importante de carácter propietario para la construcción de dispositivos de borde de red tales como adaptadores de terminales, multiplexores de servicios, switches de borde y equipo de monitoreo de QoS. La flexibilidad de UNAE ayuda a los arquitectos de Internet a diseñar y testear nuevos protocolos optimizados para redes ópticas.

A continuación se presenta el esquema:



A continuación se presenta el esquema de la tarjeta UNAE con la entrada y salida de HD:

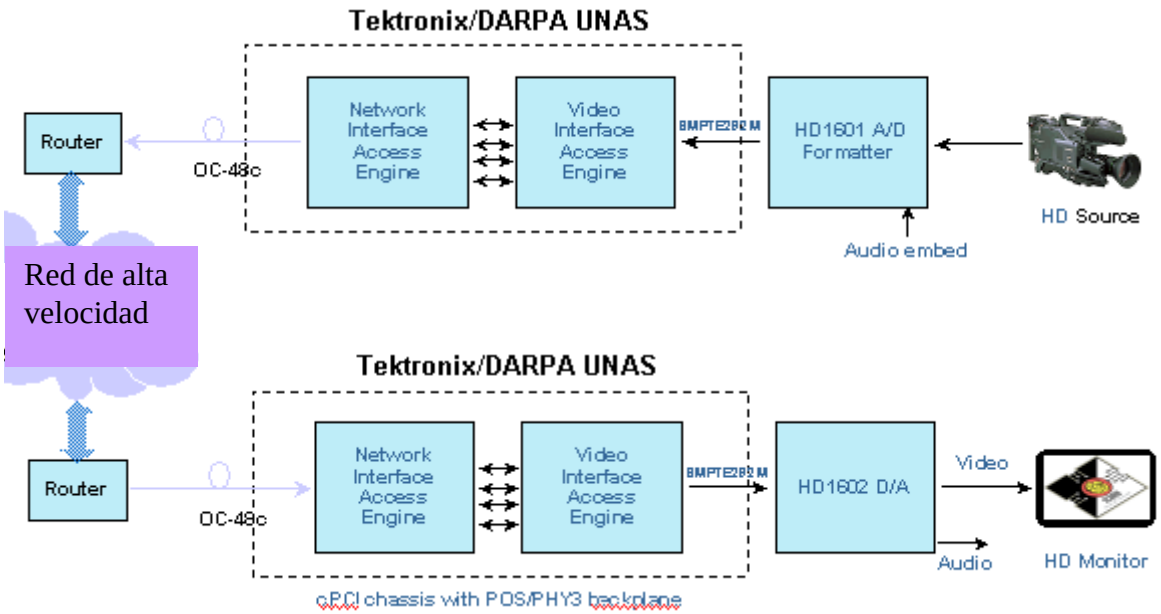


El estado de la implementación fue el siguiente:

- 4 placas UNAE operacionales
- Placas de HD operacionales (Mezanine boards)
- Software de interface de usuario sobre Windows NT
- Códigos completos de Drivers y VxWorks
- Sesión PPP establecida a través de la tarjeta PCI
- Video loopback HD establecido a través del bus POS-PHY
- Inicialmente se registraron algunos problemas de configuración en la interface de transmisión FIFO/PLPE

Demostración año 2001:

- Contenido de video HD enviado desde los laboratorios de la Universidad de Washington en Seattle (Washington) hasta la sala de exhibición de SuperComputing 2001 en Denver, a través de una red de fibra avanzada de óptica sobre IP perteneciente a Level3.
- Se usaron las redes de alta velocidad de Pacific Northwest Gigapopo y Mid-Atlantic Crossroads así como también el backbone Abilene de internet2.
- Research Channel fue el encargado de proveer el contenido de video HD



Conclusiones de la demostración:

- Se envió video HD sin comprimir a más de 1.5 Gbps
- Quedó abierta la posibilidad de probar mecanismos como corrección de errores, control de congestión, etc.

9.Propuesta de un nuevo esquema de transmisión de HDTV sobre IP

9.1.Motivación

En la demostración de Tektronix se transmitió HDTV sin comprimir a una velocidad de 1.485 Gbps, pero usando una solución tecnológica absolutamente propietaria basada en UNAS (Universal Network Access System), y en consecuencia muy onerosa.

La idea interesante que propongo para esta tesis sería la de transmitir HDTV sin comprimir usando hardware que se encuentra disponible en el mercado y software open source, sin basarse en soluciones propietarias que impongan ciertas barreras tecnológicas y económicas a la hora de la implementación.

Por otro lado, también sería adecuado usar una tecnología de red que sea usada de manera popular para llevar a cabo la citada transmisión.

9.2.Definiciones básicas del nuevo esquema de transmisión

Las especificaciones básicas a las que deberá ajustarse el sistema de transmisión de HDTV sin comprimir son en principio:

- Streaming en vivo de HDTV sin comprimir (video + audio)
- Red IP
- Unicast
- Internet2 como red principal de transporte (backbone Abilene)
- Uso de componentes estándares del mercado

9.3.Pasos a seguir para definir la plataforma final de transmisión

En esta instancia del trabajo de tesis, es adecuado comenzar a investigar con profundidad las diferentes variables que entran en juego en la transmisión de HDTV sin comprimir sobre una red IP, para finalmente definir la plataforma final propuesta.

Por tal motivo, en los puntos siguientes del trabajo se analizarán las diferentes variantes que se presentan a la hora de transmitir tráfico de streaming multimedia de alta velocidad, para luego descartar las opciones que no son aptas y seleccionar las opciones óptimas que se ajustan a mi propuesta.

10.IPv4 e IPv6

10.1.Introducción

Internet2 trabaja tanto con los protocolos IPv4 e IPv6. IPv4 es el protocolo IP actual que se trabaja en Internet convencional. Hasta el momento no hay una red de Internet nativa global sobre IPv6, sólo hay redes de esa clase individual o unida por routers IPv6. IPv6 está definido en el RFC 2460 [72].

Para que haya interoperabilidad entre los usuarios de IPv4 e IPv6, existe un método que es el *encapsulamiento o tunelizado IPv6 sobre IPv4*. Vale decir, los paquetes IP de redes que trabajan sobre IPv6 que quieren alcanzar otros hosts pertenecientes a redes IPv6 pero que están físicamente conectadas por redes que manejan el protocolo IPv4, van a tener que ser encapsulados dentro del payload de los paquetes IPv4 y que también pueden atravesar routers que manejan IPv4.

Sólo por nombrar, un ejemplo de protocolo de túneles que pueden encapsular IPv6 sobre IPv4 es el GRE originalmente desarrollado por Cisco [73].

La nueva versión del Protocolo de Internet, IPv6, escala el espacio de las direcciones IP de 32 a 128 bits, aumentando el número de direcciones disponibles a más de 35 billones además de presentar otras ventajas como las capacidades de autoconfiguración y transmisión de Mobile IP.

En lo que respecta a QoS, IPv6 permite establecer clasificación y prioridades entre ciertos tipos de flujos sensibles al tiempo. Esto está integrado en el header de IPv6. Ambas características están disponibles para IPv4 pero con algunas diferencias a favor de IPv6 que más tarde detallaré.

IPv6 está diseñado para permitir una transición gradual desde IPv4, tomando en cuenta los dispositivos de red, sistemas operativos y las

aplicaciones. Estándar doble quiere decir que los dispositivos están programados con IPv4 e IPv6.

Para migrar en una manera suave a redes nativas IPv6 es que se diseño el 6BONE [74]. 6bone es un testbed (banco de pruebas) del IETF IPng para IPv6 que opera lógicamente con direcciones de Internet IPv6 y con el objeto de reemplazar las actuales direcciones de Internet IPv4. El 6bone comenzó a operar sobre redes Internet basadas en IPv4 (con túneles/encapsulamiento IPv6 sobre IPv4) y de a poco va migrando a enlaces con IPv6 nativo.

Existen varias técnicas que ayudan a que las redes IPv4 e IPv6 puedan coexistir:

a) Traducción de direcciones: esta técnica interpreta las direcciones IPv4 a IPv6 y viceversa, de modo que ambas redes se puedan comunicar entre sí.
b) Red de estándar doble: estas redes manejan ambos estándares IPv4 e IPv6, de modo que se puedan comunicar con cualquier tipo de red.
c) Puenteo IPv6: el puenteo encapsula los paquetes IPv6 con direcciones IPv4, de modo que puedan pasar por ruteadores IPv4 en Internet.

En resumen los principales beneficios de IPv6 son:

- Capacidad extendida de direccionamiento
- Jerarquía estructurada para administración del crecimiento de las tablas de ruteo
- Autoconfiguración y reconfiguración (plug’n play)
- Soporte mejorado de opciones y extensiones de protocolos
- Soporte a Calidad de Servicio (QoS)
- Multicast y Anycast (no existe Broadcast)
- Movilidad
- IPSec integrado

Otra cosa importante: IPv6 implementa en forma mandatoria el Path MTU Discovery, para descubrir los MTU de los diferentes links y asi evitar la fragmentación de paquetes. En IPv4, el Path MTH Discovery es posible pero opcional sobre TCP.

Para poder usar IPv6 hacen falta algunos cambios como el agregado de registros AAA a los servidores DNS, las interfaces de sockets deben ser

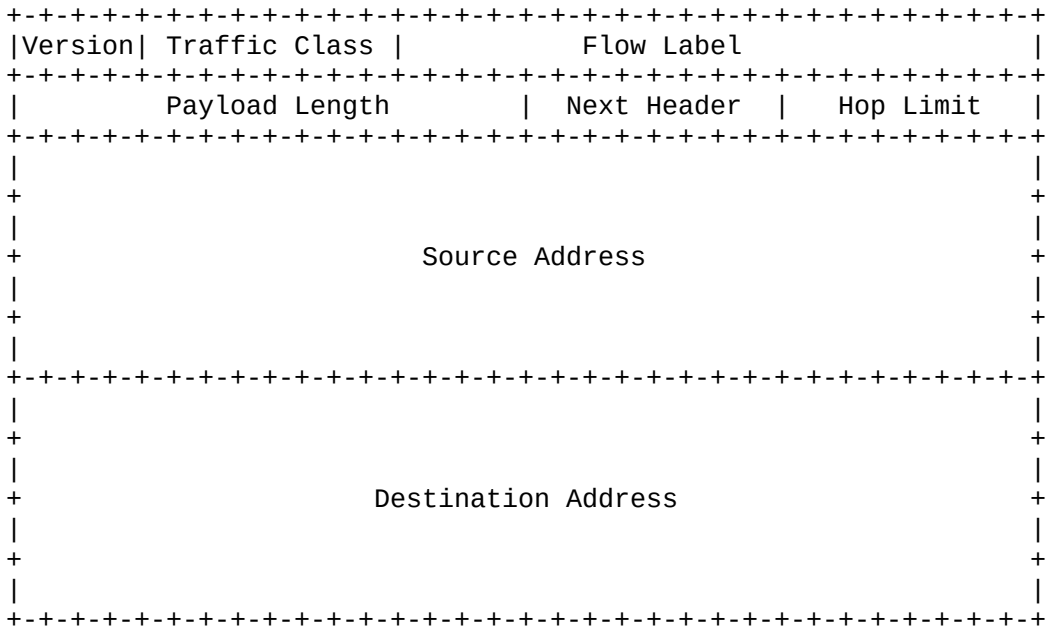
actualizadas al nuevo formato de direcciones de 128 bits y por lo tanto IPv6 necesita otras versiones de las aplicaciones de uso común como web browsers, clientes FTP y SSH, etc.

En el caso de los sistemas operativos Linux, el kernel implementa un mecanismo de dual-stack (dos stacks implementados) con IPv4 e IPv6 juntos al mismo tiempo. Por lo tanto se pueden tener aplicaciones servidores y clientes en ambas versiones corriendo al mismo tiempo en el mismo host. El host en cuestión puede tener una dirección IPv4 configurada manualmente y direcciones IPv6 configuradas por el mecanismo de autoconfiguración propio del protocolo usando las direcciones MAC.

Con respecto a los protocolos de las capas de transporte y aplicación, no deben ser modificados significativamente para trabajar sobre IPv6. Es decir, habrá unos pocos cambios pero la implementación básica del protocolo permanece igual. Por ejemplo, las capas de aplicación y transporte deberán entender las direcciones IPv6 de 128 bits.

10.2.Formato del Header IPv6

El header de IPv6 tiene 8 campos y un total de 40 Bytes, como se detalla a continuación:



Header IPv6:

Version (4 bits): Internet Protocol versión número 6.

Traffic Class (8 bits): clase de tráfico.

Flow Label (20 bits): campo para identificar flujos.

Payload Length (16 bits): longitud del payload IPv6 en octetos. Se debe hacer notar que cualquier extensión del header se debe considerar parte del payload, y por lo tanto se incluye en el conteo de la longitud.

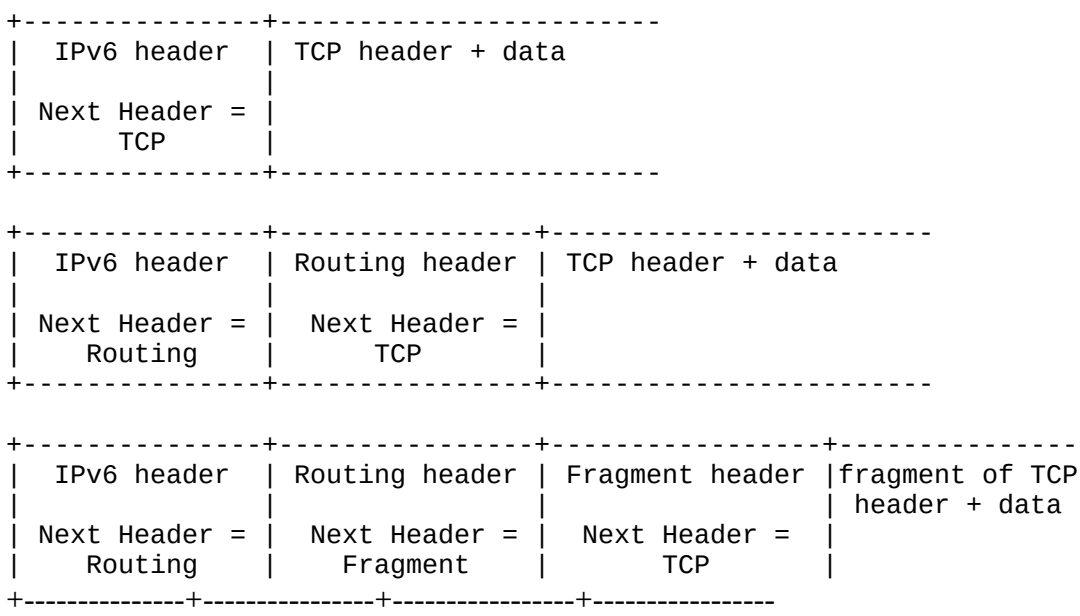
Next Header (8 bits): identifica el tipo de header que sigue de manera inmediata al header IPv6. Usa los mismos valores que el campo correspondiente del protocolo IPv4.

Hop Limit (8 bits): es decrementado por uno por cada nodo que forwardea el paquete. El paquete se descarta si el Hop Limit es decrementado a cero.

Source Address (128 bits): es la dirección del origen del paquete.

Destination Address (128 bits): es la dirección del destino del paquete. Posiblemente no sea la dirección del destino final del paquete, si está presente un header de Routing.

Extensiones del Header de IPv6:



IPv6 no limita el tamaño del paquete a 64 KB como en el caso de IPv4.

10.3.Soporte IPv6 en los backbones de Internet2

Para conectarse a una red de investigación o educación de Internet2 con IPv6, primero hay que suscribirse a Abilene o vBNS.

En el caso de Abilene, la topología del backbone IPv6 es la misma que la soportada para IPv4, aunque no todos los conectores están habilitados para trabajar con IPv6. Todos los routers “core node” de Abilene soportan IPv6 y ofrecen conexiones IPv6 nativas hacia las redes pares (peers networks), excepto en el nodo Indianápolis que es el único nodo del backbone donde se aceptan las conexiones tunelizadas.

Abilene es la encargada de asignar espacios de direcciones IPv6 a los conectores.

Si el router de borde (edge router) que se conecta a Abilene o el upstream del conector de Abilene soportan IPv6, entonces se está en condiciones de establecer una conexión IPv6 nativa. Si el router de borde o el upstream del conector de Abilene no soporta IPv6, entonces es posible establecer un tunel IPv6 en IPv4 entre el router IPv6 de un campus dado y el router que soporta túneles en Indianápolis (perteneciente al backbone de Abilene). Un conector IPv6 necesita correr Multiprotocol-BGP (MBGP) con Abilene.

10.4.IPv6 y QoS

IPv6 presenta muchas mejoras con respecto a IPv4 [18] (las características de QoS de IPv4 se analizan en el apartado de QoS), por las cuales muchos han preferido migrar, pero lamentablemente la característica de QoS no es la razón más convincente para hacerlo. Es un concepto erróneo que la especificación del protocolo IPv6 incluye un soporte intrínseco de QoS. Aunque la estructura de protocolo de IPv6 es significativamente diferente a la de su predecesor IPv4, la funcionalidad básica es aún muy similar.

Dos componentes significativos de IPv6 pueden proveer un método para entregar Clases de Servicios (CoS) diferenciadas. El primer componente es un **campo de Prioridad de 4 bits** (Priority Field) en el header IPv6, el cual es funcionalmente equivalente a los bits de **IP Precedence** en IPv4. El campo de Prioridad puede usarse para identificar y discriminar tipos de tráficos basado en los contenidos de ese campo. El segundo componente es el **Flow Label de 24 bits**, el cual fue agregado para habilitar el rotulado de los paquetes que pertenecen a flujos particulares para los cuales el emisor requiere u tratamiento especial, como el de tiempo real.

En el caso del uso del campo Flow Label, un **flujo** es una secuencia de paquetes enviados desde un origen particular hacia un destino particular (unicast o multicast) para el que el origen desea el ratamiento especial del que en el párrafo anterior. **Un flujo es identificado unívocamente por la combinación de una dirección origen y un flow label diferente de cero**. Los paquetes que no pertenecen a ningún flujo tienen un flow label de cero. Todos los paquetes pertenecientes a un mismo flujo deben enviarse con la misma dirección origen, dirección destino y flow label.

El campo Traffic Class del header de IPv6 fue diseñado para permitir que el tráfico origen identifique la prioridad de entrega de paquetes deseada, comparada con otros paquetes del mismo origen de tráfico.

- Cieros valores especifican la prioridad de tráfico para el cual la fuente provee control de congestión, como por ejemplo el TCP que responde a la congestión y pérdida de paquetes
- Otros valores especifican la prioridad de tráfico que no responde a situaciones de congestión, como el tráfico de tiempo real

Pero a decir verdad, no está claro aún cuales serían los beneficios de clasificar el tráfico según el campo de Prioridad del header de IPv6. Puede llegar a tener mayores beneficios a medida que se desarrollen algoritmos más avanzados para descartar paquetes.

Pero en verdad, luego hubo consenso para cambiar el campo de Prioridad, en cuanto a su semántica interna, a un **campo de Clase** (Class field) el cual incluye los siguientes subcampos:

- Delay Sensitive Flag (D-bit) (1 bit): es un único bit que identifica paquetes que son sensibles al retardo y pueden llegar a requerir un tratamiento especial por parte de los routers intermedios, como por ejemplo forwardear paquetes con el D-bit seteado a través de enlaces de red de baja latencia.

- Priority (3 bits): es similar en contexto y función al campo de **IP Precedence en IPv4**. Cuanto mayor es el valor en el campo, mayor es la prioridad. Estas modificaciones fueron hechas en parte para facilitar el mapeado de los valores contenidos en el campo IP Precedence dentro del campo de Prioridad de IPv6 (ambos campos contienen 3 bits).

El subcampo de Prioridad de IPv6 puede ser usado para identificar y discriminar tráfico basado en estos valores mientras los paquetes viajan por

la red, de una manera similar al uso del campo IP Precedence de IPv4. La base de este modelo de CoS es que durante los períodos de congestión, se descartan paquetes según el valor de este campo. Un mecanismo para evitar la congestión como Enhanced RED (eRED) puede ser usado para descartar paquetes basado en el hop-by-hop a través de una red.

El campo de Flow Label tiene 24 bits y esta diseñado para identificar en forma única cualquier flujo de tráfico de manera tal que los nodos intermedios puedan usar esta etiqueta para identificar flujos para un tratamiento especial, como por ejemplo con un protocolo de reserva de recursos como RSVP u otro.

En síntesis, el campo de Flow Label apunta a dar soporte a IntServ (o RSVP) y el campo Clase apunta a dar soporte a DiffServ. **Ambos campos son considerados como las características de soporte QoS de IPv6.**

10.5.Conclusión

Hay un continuo desacuerdo en la comunidad de las redes acerca de la necesidad del uso de IPv6. De hecho, la mayoría de las mejoras diseñadas dentro de la especificación del protocolo IPv6 han sido retroalimentadas por IPv4. Por ejemplo, lo que se gana migrando a IPv6 en cuanto al beneficio que proveen las características de “dynamic neighbor-discover” y la autoconfiguración de direcciones de red, pueden ser rechazadas por la disponibilidad y subsecuente despliegue en gran escala del protocolo DHCP (Dynamic Host Configuration Protocol) [75] para IPv4.

De una manera similar, el campo de Prioridad de IPv6 no ofrece ninguna mejora sustancial sobre la utilidad del campo IP Precedence del header de IPv4. El campo de Prioridad de IPv6 ofrece ocho niveles adicionales de distinción (16 posibles valores) que los que ofrece el campo IP Precedence de IPv4 (8 posibles valores). Sin embargo esto no prueba que tenga un beneficio considerable. La mayoría sugiere que como nadie hasta ahora ha implementado en forma comercial un simple mecanismo de distinción para diferenciar clases de servicios de dos niveles, el hecho de incrementar el posible número de clases de servicios no es una buena razón para preferir IPv6 sobre IPv4. Y además algunos importantes ISP’s consultados sobre el interés de parte de ellos en ofrecer clases de servicios a sus clientes, han dicho que no estarían interesados en ofrecer más de tres niveles porque implicaría problemas de complejidad en la configuración y administración.

La utilidad del campo Flow Label de IPv6, por otro lado, puede ser muy beneficiosa. Sin embargo, el único uso actual que se ve es el Flow Label en

conjunto con un protocolo de reservación de recursos, tal el caso de RSVP. El Flow Label puede posiblemente ser usado para asociar un flujo particular con una reserva de recursos específica. Fuera de esto, su posible uso no está muy bien determinado.

Por lo tanto, IPv6 no ofrece ningún beneficio sustancial de QoS más allá de lo que hoy ofrece IPv4.

10.6.Evaluación para el caso en estudio

La principal razón por la cual me aboqué al estudio de IPv6 era la necesidad de analizar sus ventajas respecto a QoS, pero evidentemente hoy por hoy no representa ninguna sustancial. La principal razón por la cual me aboqué al estudio de IPv6 era la necesidad de analizar sus ventajas respecto a QoS, pero evidentemente aún no se cuentan con los mecanismos necesarios fuera de los experimentales como para incorporarlos al presente trabajo. Y menos aún si el esquema de qutrabajo implica el uso de redes IP “best-effort” en lugar de redes IP con mecanismos de QoS.

Por lo tanto, en el esquema de trabajo que propongo no voy a hacer uso de IPv6 dado que no me brinda ningún beneficio extra al que obtengo con el uso de IPv4, más aún si no voy a hacer uso de esquemas de calidad de servicio a nivel de red y superior.

11.Calidad de Servicio

11.1.Descripción

El concepto de calidad de servicio (QoS) es un término ambiguo con muchas interpretaciones y confunde a veces a muchas personas [18]. Básicamente se define como calidad de servicio a la situación en que la red provee un tratamiento diferencial a ciertos tipos de tráfico pero que no tiene mucho ingenio en describir en forma exacta cuáles mecanismos son usados para proveer estos tipos de servicios o qué estos servicios actualmente proveen.

En contraste es más adecuado hablar de Clases de Servicios (CoS) diferenciadas porque al menos esto implica que los distintos tipos de tráficos pueden ser separados en clases, cada uno de las cuales es tratada en forma diferente en su viaje a través de la red. Por supuesto, hay diferentes métodos disponibles para entregar CoS diferenciadas.

La diferenciación en cuanto a las clases de servicios se puede hacer en alguna manera particular, pero algunos de los métodos más comunes de implementarla consiste en identificar y clasificar el tráfico según:

Protocolo: protocolos de red y transporte como IP, TCP, UDP, IPX, etc.

Puerto de protocolo fuente: protocolos de aplicaciones específicas tales como FTP, SSH, etc. dependientes de sus propias direcciones de host fuentes

Puerto de protocolo destino: protocolos de aplicaciones específicas tales como FTP, SSH, etc. dependientes de sus propias direcciones de host destino

Dirección de host fuente: direcciones de host dependientes del protocolo indicando el origen del tráfico
Dirección de host destino: direcciones de host dependientes del protocolo indicando el destino del tráfico

Interface de dispositivo fuente: interface sobre la cual el tráfico entra a un determinado dispositivo, también conocida como “interface de ingreso”

Flujo: una combinación de las direcciones de host fuente y destino, como así también los puertos fuente y destino

Para hacer una diferenciación del tráfico de la red hace falta “marcar” al tráfico antes para luego agruparlo. Esta marca del tráfico puede tomar diferentes formas como puede ser una reservación RSVP (Resource Reservation Protocol) o una precedencia IP para indicar preferencias o un bit ATM CLP (cell loss priority) configurado para indicar la preferencia de tráfico en una dada situación de congestión de la red.

Se puede decir que **la identificación, clasificación y marca del tráfico es el concepto fundamental en la provisión de clases de servicios diferenciadas**. Sin estos bloques de diseño, cualquier esfuerzo para proveer algún tipo de QoS no proveerá el comportamiento deseado en la red.

11.2.Calidad de servicio e IPv4

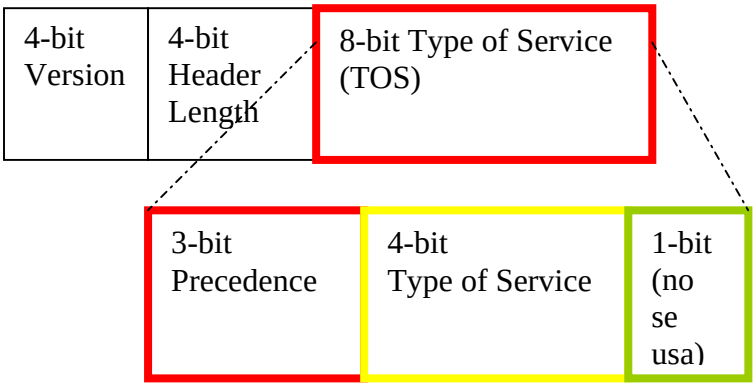
Hay algunas aproximaciones para proveer clases de servicios (CoS) dentro de IPv4.

Diferenciación por Flujo (Per-Flow Differentiation): Uno puede pensar a un flujo en este contexto como una secuencia de paquetes que comparten la misma y única información. Esta información consiste de una 4-tupla (dirección IP origen, puerto TCP o UDP origen, dirección IP destino y puerto TCP o UDP destino) que puede identificar de forma única un flujo definido para una aplicación extremo a extremo (e2e). El objetivo es extraer información del header de un paquete IP y tener cierta capacidad de asociarla con paquetes previos. El resultado pretendido es identificar el stream de la aplicación e2e del cual el paquete forma parte. Una vez que el paquete es asignado a un flujo, el mismo puede ser forwardeado con una clase de servicio asociada definida.

El propósito general de la diferenciación por flujo es definir características similares de CoS para sesiones IP e2e particulares, permitiendo por ejemplo enviar flujos de tiempo real con parámetros de CoS diferentes a los de los flujos que no son de tiempo real. Sin embargo, dado el alto número de flujos que están activos en el core de Internet en un momento dado (puede ser del orden de 260.000 flujos activos o más), esta aproximación puede ser considerada no práctica. Para mantener información del estado y manipulación de todos los flujos se requiere un gran overhead computacional que no es deseable. Este es en principio la aproximación que toma RSVP, por lo que mucha gente se opone dado que no puede escalarse a grandes redes.

IP Precedence para Clases de Servicios: Algunas implementaciones actuales usan los bits de IP Precedence en el campo TOS (Type of Service) del header IPv4 como un método para marcar y distinguir paquetes, basado en el ajuste de estos bits, mientras ellos viajan a través de la red hacia su destino. Esta es la segunda aproximación a CoS dentro de IP.

El siguiente esquema muestra el formato y la posición de los campos TOS y IP Precedence dentro del header IPv4:



El campo TOS ha sido parte de la especificación IP desde el comienzo y ha sido poco usado en el pasado.

La semántica del campo TOS fue documentada en la RFC 1349 [76] la cual sugiere que los valores especificados en dicho campo pueden ser usados para determinar cómo son tratados los paquetes con consideraciones monetarias. En la actualidad, al menos dos protocolos de ruteo –incluyendo OSPF (Open Shortest Path First) y Integrated IS-IS- pueden ser configurados para computar paths en forma separada para cada valor de TOS especificado.

Por decir un ejemplo, el firewall Netfilter/Iptables para el sistema operativo Linux, brinda la posibilidad de tratar el campo ToS del header IP (la sintaxis sería así: iptables -A INPUT -p tcp -m tos --tos 0x16). Pero la manera en que los diferentes routers y administradores trabajan con estos valores es muy variable, y la gran mayoría no los tiene en cuenta.

Conclusiones: un gran número de routers del mercado soportan TOS y IP Precedence como una primera ayuda a la solución para las necesidades de QoS y servicios de valor agregado. Los ISPs también están usando TOS y IP Precedence para proveer QoS a sus clientes de una manera simple y fácil. Pero la razón principal por la cual no se considera a TOS y IP Precedence como una solución viable para soportar QoS en Internet es la ausencia de reglas estrictas para procesar el campo TOS del header IP en los routers IP. Además existieron incompatibilidades y carencia de soporte que obstaculizaron su desarrollo. En definitiva, se puede decir que TOS es obsoleto y que ese espacio del header IP es reclamado por DSCP (DS Code Point) impuesto como estándar y ECN (Explicit Congestion Notification) usado a modo experimental, ambos pertenecientes a la arquitectura DiffServ.

11.3.Calidad de servicio en Internet para aplicaciones en tiempo real

11.3.1.Introducción

Las aplicaciones de red en tiempo real dependen de parámetros de red tales como ancho de banda, retardo, jitter (variación del retardo entre paquetes) y pérdida de paquetes para su correcta operación. El grado de sensibilidad y

tolerancia a estos parámetros varía de una aplicación a otra. Aplicaciones críticas como la cirugía remota requieren de una altísima confiabilidad de tales parámetros así como también garantía en la entrega. En cambio otras aplicaciones como la mayoría de la multimedia convencional requieren diferentes niveles de calidad de servicio en términos de dichos parámetros.

Las redes de conmutación de paquetes, tal el caso de Internet, no fueron diseñadas para proveer garantía de servicio por conexión como lo hacen las redes de conmutación de circuitos como la de telefonía. Como resultado de ello, en las redes IP los paquetes de un flujo particular pueden alcanzar su destino a través de diferentes caminos y experimentar diferentes cantidades de retardo, jitter y pérdida de paquetes a lo largo de su recorrido.

La arquitectura actual de Internet se basa en el modelo de servicio de "best-effort", el cual ha trabajado muy bien para aplicaciones tradicionales como e-mail, ftp, telnet y web. La mayoría de estas aplicaciones se basa en el protocolo de control de transmisión (TCP), el cual provee entrega confiable de paquetes pero no garantiza retardo ni jitter comprometidos. Por el otro lado, para soportar aplicaciones de tiempo real sobre una red WAN se requiere una clasificación del tráfico y garantías de nivel de servicio. Se han formulado muchas propuestas para proveer múltiples clases de servicios para tanto flujos individuales como flujos agregados con una calidad acorde a dichos flujos. Una vez implementados, dichas clases de servicios habilitarán a Internet a entregar el volumen creciente de datos de tiempo real y misión crítica a los usuarios finales. En este punto, l**a calidad de servicio se puede definir como la capacidad de proveer aseguramiento de recursos y diferenciación de servicios en una red.**

11.3.2.Aplicaciones de tiempo real y necesidad de soporte QoS

Siempre ha habido y continúan habiendo muchos debates entre quienes son partidarios de la implementación de QoS y sus oponentes, quienes sugieren agregar más ancho de banda a las redes (overprovissioning) para solucionar los problemas relacionados con la congestión, pérdida de paquetes y retardos dado que cada vez los precios para agregar ancho de banda son más bajos. Sin embargo algunas aplicaciones en tiempo real como Voz sobre IP (VoIP), videoconferencia y telemedicina requieren garantías no sólo de mayor ancho de banda sino también de delay, jitter y pérdida de paquetes. En presencia de otras aplicaciones como tráfico WWW, telnet y transferencia de archivos del tipo "bulk" en la misma red, los paquetes de las aplicaciones de tiempo real pueden experimentar variaciones grandes e impredecibles de delay, jitter y pérdida de paquetes especialmente en

ausencia de esquemas de prioridades de tráficos. Una de las ventajas de instalar mecanismos de QoS es proveer prioridad y protección a los streams de los tráficos seleccionados. En estos últimos tiempos se ha establecido un debate entre QoS y overprovisioning, y algunos apuestan a mecanismos mixtos entre ambos que son más económicos y no tan rígidos como mantener una red controlada por QoS que requiere la configuración de múltiples servidores de políticas y reservas así como también elementos de control de admisión en los extremos de la red. Otro argumento que se da usualmente a favor de QoS es la diferencia de anchos de banda entre los backbones y los extremos de Internet. En las redes que conforman los extremos de Internet generalmente hay más congestión que en los backbones. Además hay una clara necesidad de contar con mecanismos de prioridades y protecciones para los paquetes de aplicaciones de tiempo real y misión crítica en los routers extremos.

Las **aplicaciones multimedia** usualmente son sensibles al delay extremo a extremo y a las variaciones de delay (jitter), pero muchas de ellas pueden tolerar ciertos niveles de pérdida de paquetes. Y por otro lado TCP está diseñado para brindar la mayor seguridad en la entrega confiable de datos sin tener en cuenta al delay ni jitter. Otras **aplicaciones de tiempo real** como transacciones o administración remota requieren garantías en delay y pérdida de paquetes. Por ejemplo, las llamadas VoIP requieren un máximo delay extremo a extremo de 150-300 ms para una apropiada comprensión de la voz.

Dentro del modelo actual de “best-effort” de Internet, se han desarrollado algunos protocolos para soportar las aplicaciones multimedia, tales como Real Time Protocol (RTP), Real Time Control Protocol (RTCP) y Real Time Streaming Protocol (RTSP), como así también el estándar H.323 para aplicaciones de videoconferencia. Muchas aplicaciones multimedia también emplean técnicas de adaptación para lidiar con las condiciones de red inesperadas.

11.3.3.Características del tráfico de Streaming

El caso en estudio es un streaming en vivo de HDTV sin comprimir, con transporte de audio y video. Las aplicaciones de audio/video streaming pueden retardar su punto de reproducción (playback) de acuerdo al máximo delay de la red y así superar el inconveniente del jitter, pudiéndose ocasionar buffer overflows y pérdida de datos.

En general, las principales características del tráfico de audio/video streaming como tráfico del tipo real-time, son las siguientes:

- Son aplicaciones **interactivas**. Envuelven una secuencia de interacciones y transferencias de información entre los puntos extremos de la aplicación. Un caso típico es el uso de los comandos Rev/Fwd/Pause/Stop en las aplicaciones de video. Estos tipos de aplicaciones pueden depender de parámetros de QoS como ancho de banda, delay, jitter y pérdidas. Como otro ejemplo, las llamadas de telefonía IP necesitan un delay extremo a extremo menor a 300 mseg, a pesar que en realidad se produce una degradación con delays mayores a 150 mseg. En las conversaciones de voz el jitter es aceptable hasta un valor de 75 mseg. Para compensar el jitter se usan jitter buffers pero los mismos incrementan el delay extremo a extremo y deben ser usados con mucho cuidado. Por otro lado, las aplicaciones no interactivas no interactúan entre los puntos extremos. Ejemplo de este tipo de aplicaciones son el backup de datos y las transferencias de datos masivas. Estas aplicaciones típicamente tienen requerimientos de ancho de banda para proveer una performance razonable.

- Son aplicaciones **inelásticas**. El tráfico multimedia es del tipo “soft real-time” y puede tolerar alguna degradación en QoS por un breve período de tiempo y operar con una calidad más baja, en contraposición con las aplicaciones del tipo “hard real time” como la videoconferencia que no pueden funcionar para nada si sus necesidades de QoS no son encontradas en la red durante todo el tiempo.

- Son aplicaciones **tolerantes**. No se debe confundir las aplicaciones elásticas con tolerantes: mientras que las primeras no imponen ningún requerimiento de QoS, las segundas imponen requerimientos de QoS pero con rangos o niveles que permiten que las aplicaciones corran incluso si los niveles óptimos de QoS no son alcanzados. Las aplicaciones tolerantes son generalmente inelásticas con rangos de requerimientos de QoS, pero si sus límites de QoS son violados entonces no pueden ejecutarse en forma correcta. Otro ejemplo de tales aplicaciones es la telefonía IP. Así, las aplicaciones “hard real time” son un ejemplo de aplicaciones intolerantes, porque son aplicaciones que especifican valores fijos para sus requerimientos de QoS y no pueden ejecutarse si estos valores no son obtenidos. Un ejemplo de aplicaciones tolerantes es VoD: la aplicación puede tolerar pérdidas y una cierta cantidad de delay y aún así ser capaz de correr ininterrumpidamente. Cabe destacar que algunas técnicas de compresión de video como MPEG pueden tolerar pérdidas en frames

y usan relaciones entre frames sucesivos para predecir los frames pérdidos.

- Son aplicaciones **adaptivas**. Estas aplicaciones tratan de mantener la calidad percibida en niveles aceptables, incluso bajo condiciones pobres de la red. Esto puede ser alcanzado por medio de la disminución de la velocidad de transmisión o de la resolución de la transmisión, o incluso por medio del uso de técnicas de compresión y corrección de errores. Las aplicaciones adaptivas pueden usar también buffers extra para compensar las condiciones transitorias desfavorables de la red y permitir una degradación elegante en la performance. La mayoría de las aplicaciones de audio y video en Internet son adaptivas. Por otro lado las aplicaiones no adaptivas no tienen la capacidad de lidiar con condiciones transitorias desfavorables de la red: ellas pueden tolerar cierta degradación de QoS pero esto afecta directamente la calidad percibida por el usuario final. En la mayoría de los casos, las aplicaciones adaptivas son tolerantes y las aplicaciones no adaptivas son intolerantes, pero también se pueden encontrar otras combinaciones.

La calidad de una transmisión de video es sensible a la pérdida de datos dado que la mayoría de las técnicas de compresión buscan reducir las redundancias entre frames sucesivos. El efecto de las pérdidas en la calidad de video es también influenciado por parámetros tales como las técnicas de codificación usadas, velocidad de pérdidas, patrón de pérdidas y tamaño del paquete de transmisión. Los efectos de las pérdidas de paquetes grandes a menudo son más pronunciados que los efectos de los paquetes más pequeños. La calidad de video del usuario final también depende del tipo de frame que se pierde, por ejemplo en la codificación MPEG la pérdida de los frames I o P es siempre más severa que la pérdida de los frames B. En el caso de los streams multimedia consistentes de flujos de audio y video, la sincronización entre flujos es importante para la calidad total. Además se debe garantizar calidad para el audio y video en estos casos.

Un estándar común usado para medir la calidad de la voz y el video es el llamado “mean opinion score” (MOS, recomendación ITU-T P.800), el cual envuelve un gran número de personas y toma el promedio estadístico de sus opiniones acerca de la calidad de la multimedia transferida.

11.4.Requerimientos de QoS

11.4.1.Parámetros de QoS

La especificación de los parámetros de QoS generalmente depende del contexto de la aplicación involucrada. Diferentes aplicaciones o sistemas finales pueden tener diferentes interpretaciones de sus requerimientos de QoS.

Pero los siguientes parámetros son considerados como básicos:

- Throughput

Es el número efectivo de unidades de datos transportados por unidad de tiempo (por ej., bits por segundo). Este parámetro usualmente es especificado como "garantía de ancho de banda". El throughput contempla la capacidad del enlace como así también la capacidad de procesamiento de los nodos intermedios. Un cuello de botella en el ancho de banda puede poner en peligro la garantía de ancho de banda para un camino total extremo-a-extremo.

- Delay

Es el intervalo de tiempo entre la salida de los datos desde la fuente hasta la llegada en el destino. Esto es usualmente conocido como "e2e delay". La fuente y el destino pueden abarcar múltiples capas tales como las de aplicación, transporte, red, enlace e inclusive física, y por ende hay diferentes delays asociados a estas diferentes capas. La performance requerida está expresada en términos del máximo delay. Generalmente la medición del "one-way delay" se torna dificultosa debido a los problemas de sincronismo de reloj. Entonces a veces se usa el "round-trip delay" como una representación del delay, pero debido a disimilitudes típicas entre los caminos de ida y vuelta sobre Internet, esto puede no ser una buena indicación de parámetro de delay.

- Jitter

Usualmente se refiere a la "variación del delay". Como el delay, el jitter está especificado por el jitter máximo. Hay muchas definiciones acerca de cómo puede cuantificarse el jitter, y las principales son estas:

1) El jitter puede ser calculado como la diferencia entre los tiempos entre salidas Ii y los tiempos entre llegadas Ai de la i^o y la $(i-1)^o$ unidades de datos: $Ji = Ii - Ai$
2) El jitter puede ser calculado como la diferencia entre los delays de la i^o y la $(i+1)^o$ unidades de datos: $Ji = Di+1 – Di$

3) En el estándar RTP, el jitter es medido acumulativamente a través de la siguiente ecuación: J = J + (|D(i-1,i)| - J) / 16

- Pérdidas

Es el porcentaje de unidades de datos que no alcanzaron el destino en un intervalo de tiempo específico. Usualmente es representada como la "probabilidad de pérdida". La retransmisión, sin embargo, no cambia el valor de la probabilidad de pérdida de la red, pero sí es un método para la recuperación de las pérdidas.

- Confiabilidad

No debe ser confundido con pérdida, aunque algunas veces las abarca. Por confiabilidad se entiende a la entrega correcta de unidades de datos en sus destinos. Por ejemplo, un protocolo confiable puede usar retransmisiones para recuperarse de las pérdidas y entregar así un servicio confiable a las capas superiores. Los errores, duplicaciones y desordenes de paquetes son ejemplos de carencia de confiabilidad.

11.4.2.Acuerdo de servicio

Usualmente existen dos tipos de compromisos de servicios ofrecidos por los ISP's: garantizado y previsto. El primero especifica a priori límites en los parámetros de servicios y el segundo especifica el servicio esperado por la red dado el estado actual.

También existe otra clasificación de servicios: cuantitativos y cualitativos. Mientras que los primeros especifican números y cantidades para los niveles de servicios, el segundo especifica niveles relativos como "mejor que" o "más bajo que".

Para poder brindar un acuerdo de servicio, el ISP debe conocer acerca del tráfico asociado a este servicio. Esto es conocido como "perfil de tráfico" y es una sdescripción del tráfico entrante en términos de velocidad promedio, burstiness, distribución, tamaño de paquete, etc.

11.4.3.Control de admisión

Las redes tienen una cantidad finita de recursos en términos de enlaces, capacidad ded buffering en los nodos intermedios, capacidad de procesamiento, etc. La clave del modelo de QoS es cómo distribuir estos recursos de manera apropiada entre los diferentes clientes o usuarios para brindarles los requerimientos de servicios necesarios.El control de admisión se usa como protección contra la sobresuscripción de los recursos disponibles. Usualmente se compara los requerimientos de servicios y los recursos disponibles y luego se decide aceptar o rechazar los requerimientos de servicios planteados.

11.5.Provisión de QoS

La provisión de QoS a niveles de red requiere funcionalidades extra por parte de los dispositivos de red más allá del forwarding y routing de paquetes. Estas funcionalidades incluyen clasificación, queuing y scheduling, policing y shaping, y buffer management, comentadas abajo.

11.5.1.Scheduling

Existen algoritmos de scheduling que permiten asignar ancho de banda entre las diferentes clases de tráfico de una manera justa, y proveer estadísticas y garantías por flujo o por tráfico agregado sobre parámetros de red como delay, jitter y pérdida de paquetes. Los más conocidos son:

- First In First Out (FIFO) Scheduling
- Priority Scheduling
- Generalized Processor Sharing (GPS)
- Weighted Fair Queuing (WFQ)
- Class-based Queuing (CBQ)

11.6.Buffer Management

Se emplean algoritmos para evitar congestión de paquetes en las redes. Uno de los más conocidos es el algoritmo RED (Random Early Detection).

11.7.Policing

La política de tráfico es aplicada típicamente en los extremos de una red y/o cerca del origen. Cuando un paquete llega, el algoritmo de política primero determina si el paquete concuerda con el SLA (service level agreement) negociado entre el origen del tráfico y la red. En el caso que no lo esté, el algoritmo puede descartar el paquete o bien decrementar su

prioridad basado en la política y en la actual prioridad del paquete. La política de tráfico puede estar basada en un simple parámetro negociado, como la velocidad pico, o en una combinación de parámetros, como velocidad pico, tamaño del burst, hora del día, etc. Los mecanismos más usados como política de tráfico, solamente por citar sus nombres, son Token Buckets y Leaky Buckets).
Como se mencionó antes, cualquier mecanismo de QoS necesita una combinación de schedulers, clases (o colas) y mecanismos de políticas para proveer garantías de clases por flujo o por tráfico.

11.8.Reseña de QoS en ATM

ATM fue uno de los primeros marcos de trabajo para soportar QoS en forma explícita. La tecnología ATM fue desarrollada para optimizar la utilización del ancho de banda mientras se aseguran diferentes niveles de QoS por medio del uso del control y management del tráfico. Un punto clave de la operación de la tecnología ATM es el uso de unidades de datos de tamaño fijo (llamadas celdas) para hacer scheduling, queueing y buffer management de manera más fácil que si se usara tamaños de paquetes variables y así se puede estimar un comportamiento más predecible. Más aún, el establecimiento de “virtual connections” (VCs) y “virtual paths” (VPs) para transportar las celdas provee facilidades de QoS más robustas.

QoS en ATM está especificado por la velocidad de pérdida de celdas (CLR), el máximo delay de transferencia de celdas (Max-CTD), variación del delay de celdas pico-a-pico (P2P-CDV), la relación de bloques de celdas con errores severos (SECBR), la tasa de celdas fuera de orden (CMR) y la relación de error en las celdas (CER).

Las principales razones para que ATM esté confinada a los backbones de Internet son la dificultad de despliegue de las interfaces ATM en las aplicaciones de los hosts extremos de la red y el gran overhead (por ej., el tamaño fijo de las celdas y el establecimiento de los circuitos virtuales).

11.9.IP Precedence y TOS

Los campos IP Precedence y TOS fueron inroducidos por primera vez en el protocolo Internet [43], y son considerados como el primer soporte QoS en redes IP. Aunque ToS ha sido poco usado en el pasado, su uso por parte de los hosts es ahora mandatorio por los requerimientos para Internet. Muchos de los vendedores de routers ahora soportan IP Precedence y TOS como una primera solución de ayuda para QoS y servicios de valor agregado. Los

ISP’s también usan TOS para proveer QoS a sus clientes de una manera simple y fácil.

La noción de precedencia fue definida ampliamente como “una medida independiente de la importancia del datagrama”. Cabe destacar que hoy no todos los valores del campo IP Precedence tienen significado.

El campo TOS del datagrama IP provee una indicación para la QoS requerida para dicho datagrama. Se usa para seleccionar los parámetros de servicio apropiados en los elementos de red. La principal elección es el trueque entre bajo delay, alta disponibilidad y alto throughput.

Aquí se detallan los bits usados en el campo ToS:

Bit 0 Precedence	Bit 1 Precedence	Bit 2 Precedence	Bit 3 D	Bit 4 T	Bit 5 R	Bit 6 Uso futuro	Bit 7 Uso futuro

Donde:

D=delay, T=throughput y R=confiabilidad.

Aquí se ven algunos valores del campo IP Precedence en la tabla:

Bits	**Precedence**
111	Network control
110	Internetwork control
101	Critical
100	Flash override
011	Flash
010	Inmediate
001	Pririty
000	Routine

Los bits 0 a 2 son usados para la precedencia. Y la precedencia puede tomar uno de los ocho valores mostrados en la tabla. El bit 3 se usa para especificar el delay (D). D=0 indica un delay normal y D=1 indica un bajo delay. El bit 4 es usado para el throughput (T). T=0 indica un throughput

normal y T=1 indica un alto throughput. El bit 5 es usado para la confiabilidad (R). R=0 indica confiabilidad normal y R=1 indica confiabilidad alta. Los bits 6 y 7 están reservados para uso futuro.

La red puede elegir un mapeo de servicio para mapear estos valores a clases de servicios apropiadas e implementadas en la red. La técnica de mapeo y las clases de servicios no son parte de la especificación IP ToS. Es importante destacar que en IPv6, el campo IP ToS cambia su nombre a **class field**, y se agrega otro campo llamado **flow label**. El campo flow label sirve para trabajar en IntServ, mientars que el campo class field se usa en DiffServ.

La principal razón por la cual IP Precedence y IP ToS no trabajan como una solución viable para soportar QoS en Internet es la ausencia de reglas estrictas para procesar el IP ToS en los routers IP. Además existen incompatibilidades y carencia de soporte que han obstaculizado su desarrollo.

11.10.IntServ y RSVP

11.10.1.Conceptos generales

La diferencia principal entre redes ATM y IP es que las primeras son orientadas a conexión, mientras que las segundas no. La característica de que IP no está orientado a conexión (connectionless), y por ende Internet, es requerida para escalabilidad y tolerancia a fallos. IP es asociado naturalmente a “datagrama” y “per hop” routing, y por su naturaleza no soporta circuitos virtuales. Internet está basada en la entrega de datagramas según el modelo “best-effort”. No hay garantía en la entrega de datagramas. Los datagramas pueden sufrir delay, jitter, ser manipulados (mangling) o incluso perdidos (dropped) en sus caminos hacia los destinos.

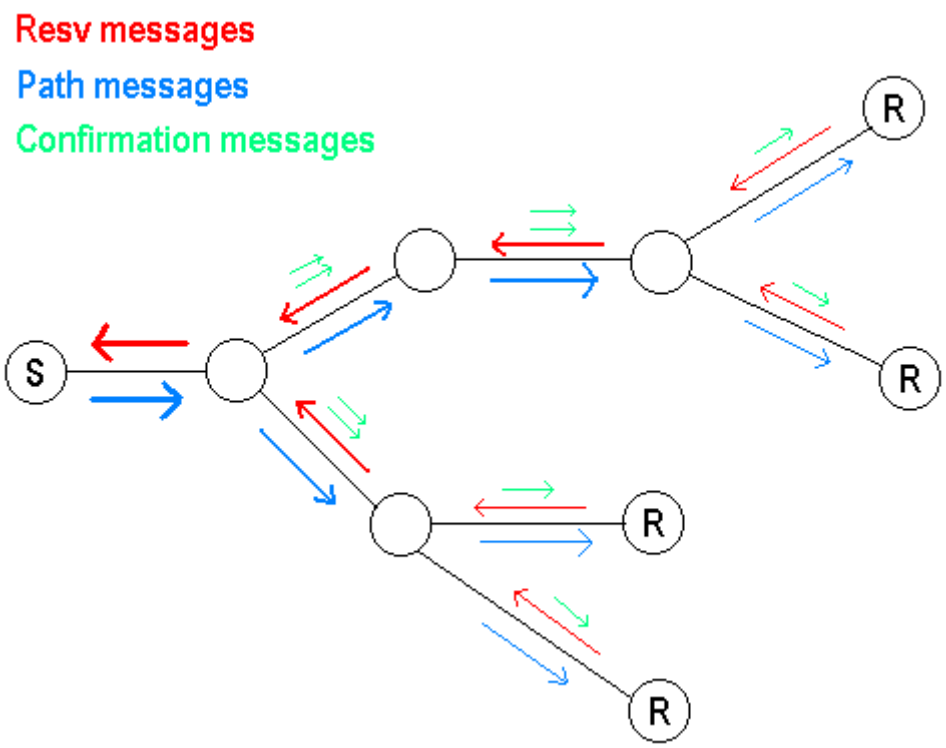
El modelo IntServ fue diseñado para proveer QoS a flujos individuales, o a una sesión individual en el caso de aplicaciones multicast. Se requiere de una sesión que requiera una garantía específica de QoS para iniciar un procedimiento de configuración usando RSVP. RSVP establece “soft states” en cada router a lo largo del camino entre origen y destino especificando los requerimientos de clase y recursos de la sesión iniciada. La reserva permanece válida siempre y cuando la sesión esté activa, pero expira si no se refresca periódicamente. Usando este modelo de servicios, si hay N sesiones individuales pasando a través de un router en un dado instante, el router necesitará mantener la información de estado necesaria para las N sesiones.

Un aspecto importante de este modelo es que la reserva de recursos se extiende a lo largo de toda la ruta (e2e path) de los paquetes de datos. Si la ruta cambia, entonces la reserva se rehace automáticamente para seguir la nueva ruta. Esto es también permitido por medio del uso de la característica “soft-state”. El modelo de servicios es unidireccional, vale decir, si una sesión de comunicaciones en ambos sentidos necesita QoS deberá reservar recursos en ambas direcciones.

A modo de comentario, es interesante decir que IntServ agrega dos clases de servicios adicionales al modelo “best-effort” usado en Internet: “guaranteed service” y “controlled load service”.

RSVP es un protocolo de reserva orientado al receptor que es usado dentro de una red IntServ para señalizar los requerimientos de QoS de una sesión de aplicación a lo largo del camino entre origen y destino.

El proceso de reserva se describe en la RFC 2210 [77] y se grafica así:



El proceso de reserva comienza con el emisor enviando un mensaje PATH hacia el receptor, que atraviesa todos los routers a lo largo del camino. Este mensaje graba todos los routers intermedios como parte del forwawrding path y calcula los parámetros de red e2e. Cada mensaje PATH transporta un objeto de especificación de tráfico llamado Tspec que describe el perfil de tráfico generado por el origen, y tiene los siguientes parámetros: rate, bucket, peak rate, minimum policed unit y maximum packet size. El mensaje PAT además transporta un objeto de especificación de divulgación denominado Adspec que se usa en el cómputo de los parámetros de QoS acumulados a lo largo del e2e path.

Cuando el mensaje PATH alcanza el receptor, el mismo responde al emisor con un mensaje RESV que transporta la especificación de reserva (Rspec) contenida en un objeto FlowSpec. Este mensaje, si es aceptado por todos los routers a lo largo del camino (mensaje RESV de confirmación), establece la reserva actual y “filter on” estos routers intermedios. A este proceso se lo conoce como ***control de admisión***.

Si un error llegara a ocurrir o no existen suficientes recursos para ser asignados, entonces se genera o bien un mensaje PATHerr o RESVerr por el router correspondiente. Este mensaje regresa al emisor y así cualquier reserva hecha en los routers intermedios se cancela.

Finalmente, una vez que una sesión de aplicación culmina, se envían mensajes Patear y RESVtear para remover los estados de reservas en todos los routers a lo largo del camino.

11.10.2.Evaluación del modelo IntServ

IntServ con RSVP tiene las siguientes características favorables:

- Flexibilidad para encontrar las necesidades de QoS: la reserva de recursos se hace por flujo y entonces se satisfacen los requerimientos de recursos de flujos individuales
- QoS asegurado y determinístico: los mensajes RSVP atraviesan el mismo camino e2e que el tráfico de datos de la aplicación desde el origen al destino y establece estados de reserva en cada router a lo largo del camino. Esto hace que el proceso de reserva sea exacto en términos de la provisión del QoS requerido.
- Ajustes a los cambios de rutas: las reservas RSVP son estados blandos (decía “soft”) y necesitan ser refrescados periódicamente. El proceso de refresco cualquier cambio en las rutas durante el tiempo de vida de una sesión y así ajusta el camino de reservas de acuerdo a ello.
- Soporte de comunicaciones multicast: dado que RSVP es un protocolo orientado al receptor, ya que múltiples receptores se unen a un multicast y responden al origen con mensajes de RESV. Los recursos requeridos son así acordemente reservados.

Sin embargo, al día de hoy, el despliegue de IntServ ha estado limitado por las siguientes razones:

- El modelo IntServ carece de escalabilidad. Esto es una consecuencia directa de la reserva de recursos por flujo y el tratamiento del tráfico en los routers intermedios y así no escala en los routers del centro o en los backbones de las redes.
- Las aplicaciones deben esperar hasta que la reservación usando RSVP se complete. Esto puede retardar el tiempo de arranque y puede ser inaceptable para ciertas aplicaciones de tiempo real que requieren una respuesta inmediata de acuerdo a límites estrictos. Esto se hace más problemático para sesiones con un tiempo de vida corto como http, donde el tiempo de establecimiento de la sesión es mucho más largo que el tiempo de transferencia de datos y se torna así mucho más significativo.
- Dado que la reserva de recursos y los cálculos de QoS son hechos a priori y tan cerca como sea posible del perfil de tráfico especificado

dado por las aplicaciones, cualquier cambio imprevisto o importante en el tráfico de datos a causa de una actualización o cambio de requerimientos de la aplicación pueden tener efectos impredecibles a menos que se haga de nuevo una negociación o reserva usando los nuevos perfiles de tráfico.

- IntServ no es compatible con el protocolo de seguridad IPSec, debido a la clasificación multicampo requerida en cada router del camino para identificar flujos individuales. Dado que IPSec encripta los headers de la capa de transporte de un paquete, los routers no tienen acceso a estos headers para poder poder llevar a cabo la clasificación. Sin embargo, este problema fue resuelto con la introducción del flow label de IPv6, el cual identifica el flujo mirando sólo en el header de la capa de red sin necesidad de la información del header de la capa de transporte.

11.11.Servicios Diferenciados

11.11.1.Introducción

El modelo IntServ no ha sido exitoso en las redes centrales de Internet por su falta de respuesta a los cambios de tráfico y carencia de escalabilidad, pero sí se usa en redes pequeñas y redes privadas. Para solucionar los problemas presentados por IntServ, el IETF decidió mirar hacia un alternativa más escalable y así desarrolló la arquitectura DiffServ. Por otro lado, el Internet2 Consortium ha adoptado el modelo DiffServ para proveer QoS en la red QBone.

Las principales diferencias entre DiffServ y su predecesor IntServ son las siguientes:

- DifServ se basa en la provisión de recursos en lugar de la señalización y reservación de recursos como IntServ.
- DiffServ maneja tráficos agregados (un grupo de flujos de tráfico manejados de manera similar a través de una parte de la red) en lugar de microflujos. Como resultado, en un fundamento por flujo, DiffServ provee garantías de QoS cualitativas en lugar de cuantitativas como IntServ.
- Los estándares DiffServ definen tratamientos de forwarding y no servicios e2e como los servicios de carga controlada y garantizada en IntServ.
- DiffeServ pone énfasis en los “Service Level Agreements” (SLA’s) entre dominios en lugar de ponerlo en una señalización dinámica e2e como en IntServ.

11.11.2.Antecedentes

La idea básica de la provisión de servicios diferenciados a paquetes de datos nació de la RFC 2638 [78], donde se introdujo la idea de usar dos bits en el header IP para marcar paquetes con el objeto de brindarles un tratamiento diferencial en los dispositivos de red.

La arquitectura propuesta usaba 2 bits: “P-bit” para marcar los paquetes para un servicio premium y “A-bit” para marcar paquetes para el servicio asegurado. Los paquetes son marcados en los routers los cuales emplean clasificadores multicampo para diferenciar flujos basado en esos dos bits. Los paquetes marcados con el servicio premium son medidos contra un perfil específico usando un medidor de balde de token (token bucket meter) con un muy pequeño tamaño de balde, mientras que los paquetes del servicio asegurado también son medidos usando un medidor de balde de token con un tamaño de balde igual al tamaño de la ráfaga de tráfico (traffic burst). Los paquetes fuera de perfil no son marcados y son tratados como “best-effort”. Estos dos bits juntos determinan la prioridad del manejo del paquete en los routers. Los paquetes premium toman una prioridad más alta que los paquetes asegurados a la hora de forwardearlos.

Además, esta arquitectura de dos bits incluye una sección con el uso de Bandwidth Brokers (BB) para asignación de servicios y establecimiento de servicios e2e.

Los conceptos anteriormente descritos son el fundamento general de la arquitectura DiffServ, como se describe a continuación.

11.11.3.Arquitectura DiffServ

Los principales componentes de la arquitectura DiffServ para proveer QoS en Internet son:

- Campo DS: la diferenciación de servicios requiere el marcado de bits seleccionados en los headers de los paquetes IP. Esto es llamado el campo DS (“DS field”) en el contexto de DiffServ. La RFC 2474 “Definition of te differentiated services field in los headers IPv4 and IPv6” define el campo DS coincidente con el byte ToS en el header IP. Pero sólo seis bits son usados para transportar los códigos DS (DSCP: DS codepoint) y los dos bits que restan son usados actualmente para notificaciones de congestión explícita (ECN). DSCP es usado para determinar el comportamiento de forwarding

que experimenta un paquete en un típico nodo DiffServ. Este comportamiento de forwarding es llamado “per-hop behavior” (PHB. DiffServ se basa en definir un número pequeño de PHB’s que implementan la diferenciación de servicios en los routers participantes y marcan los bits DSCP para asignar paquetes entrantes a uno de estos PHB’s. Hay un PHB (class selector PHB) definido para tener compatibilidad con los bits IP Precedence del byte ToS, y los otros dos PHB’s son definidos para implementar los servicios premium (expedited forwarding EF) y asegurado (assured forwarding AF).

- CS PHB: como se mencionó antes, sirve para mantener la compatibilidad con los bits IP Precedence del byte ToS. Puede ser usado para crear ocho diferentes niveles de prioridad con el mayor valor que indica una más alta prioridad de forwarding.

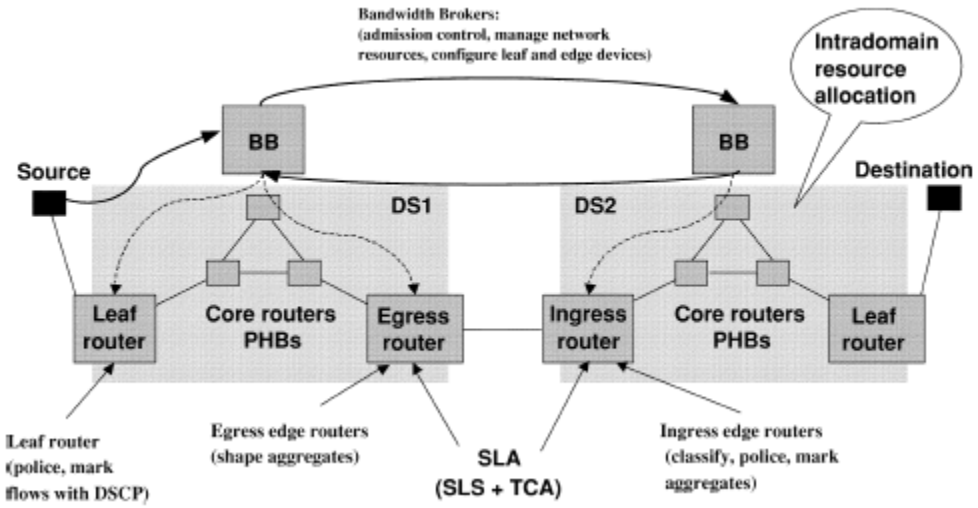
- EF PHB: está diseñado para implementar un servicio con bajo delay, jitter y pérdidas en añadidura al ancho de banda asegurado, como se especifica en la RFC 3246 [79]. La idea atrás de EF PHB es que los paquetes marcados con un EF DSCP se encuentren con colas de tamaño muy pequeño en los nodos de forwarding. Esto a menudo es alcanzado por medio de la asignación de recursos de forwarding con un velocidad mayor que la velocidad de llegada de los paquetes EF. EF se usa en servicios que tienen requerimientos estrictos de delay y jitter tale como las aplicaciones de tiempo crítico y multimedia.

- AF PHB: se usa para diagramar servicios con una pérdida controlada y un ancho de banda asegurado. Tales servicios no tienen garantías de delay o jitter. El grupo AF PHB consiste de tres comportamientos de forwarding con diferentes precedencias de descarte de paquetes (drop). El AF PHB define dos velocidades: una velocidad de información comprometida (CIR: commited information rate) que es el mínimo ancho de banda de red para ser asegurado, y una velocidad de información pico (PIR: peak information rate) para una velocidad por encima del CIR para acomodar las ráfagas (burst). AF PHB se hace importante en losd casos de congestión de red cuando se deben descartar paquetes.
- Per-Domain Behavior (PDB): son diseñados para que sean instalados en nodos individuales o routers. Se denomina “behavior agrégate” (BA) a un grupo de paquetes que experimentan el mismo comprtamiento en cada nodo mientras atraviesan un dominio. El término BA leugo se hará sinónimo e PDB, que es uno de los principales bloques para construir una red DiffServ. Un PDB se usa

para definir un conjunto de servicios edge-to-edge que tienen parámetros de red medibles como los que experimentan un conjunto de paquetes con el mismo DSCP cuando atraviesan un dominio DiffServ. Se entiende por dominio DiffServ a parte de la red bajo una única administración y compatible con los estándares DiffServ. Entonces, los PDB's se usan para construir servicios entre puntos de ingreso y egreso de un dominio. Los atributos de los PDB's (throughput, tasa de pérdidas, etc.) se publican como especificaciones de niveles de servicios en los límites del dominio. Más de un PDB puede estar basado en el mismo PHB, y por otro lado un PDB se basa en un único PHB dentro de un único dominio.

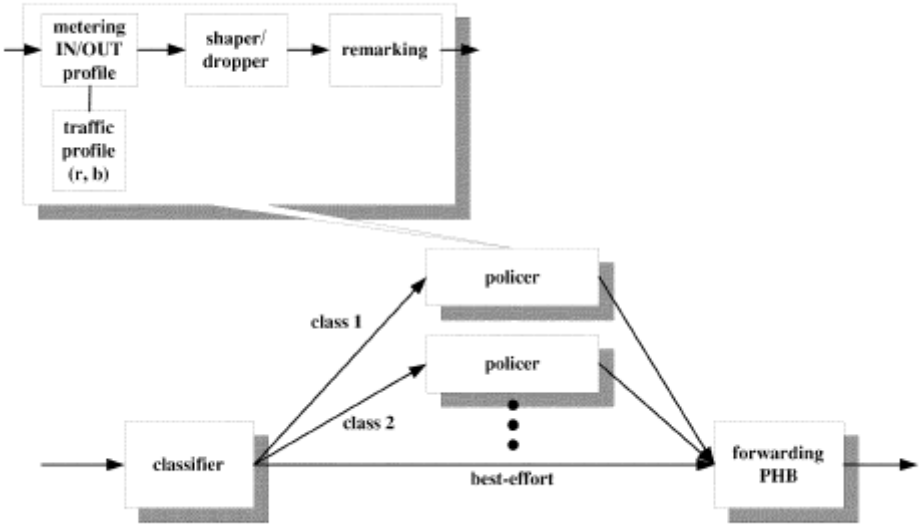
- Arquitectura DiffServ: una red DiffServ consiste de múltiples dominios DiffServ (DS's) que pueden ser vistos como sistemas autónomos (AS's). Los routers de cada extremo del dominio hacen el acondicionamiento del tráfico necesario en los extremos. Cada dominio DS hace dos acuerdos con cada uno de sus dominios vecinos: un SLA especificando los servicios que este dominio proverá y un acuerdo de acondicionamiento de tráfico (TCA) al cual estará sujeto el tráfico entrante a este dominio. Los dominios adyacentes negocian SLA's ente ellos y con loc clientes que acceden a sus redes. Cada dominio DS configura y provee sus nodos internos para que los SLA's puedan ser cumplidos. Esta distribución de las responsabilidades de configuración agrega flexibilidad a la arquitectura Diffserv. Es importante remarcar que aquí, a diferencia del moldelo IntServ, DiffServ emplea "provisión de recursos" con la ayuda de los SLA's y no usa "reservación de recursos" para establecer diferentes servicios.

El siguiente es un esquema usual de una red DiffServ:

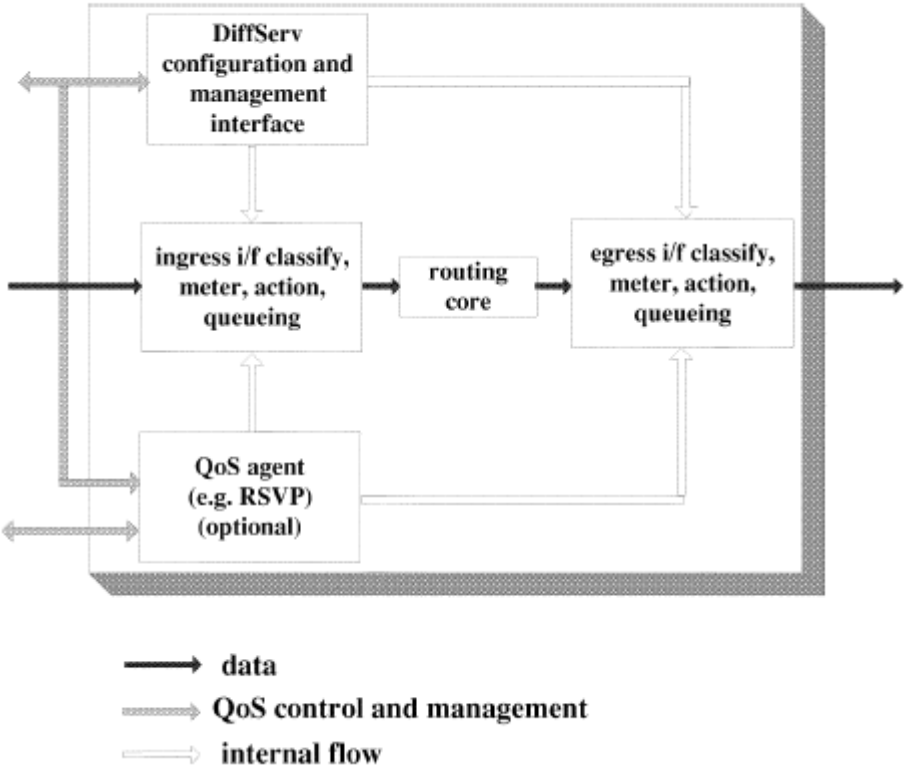


Ahora seguiremos el trayecto completo que recorre un paquete típico en una red DiffServ, desde que sale del origen hasta que llega al destino. Primero el paquete es medido contra un cierto perfil de tráfico negociado entre el cliente y el proveedor de servicios de red. Luego el paquete es marcado con un DSCP apropiado para encontrar un cierto nivel de servicios en la red del ISP. El paquete puede ser marcado ya sea en el host (origen) o en el “firts-hop” router (conocido como “leaf router”) o incluso hasta en los routers de borde. De hecho, un paquete puede ser remarcado sucesivamente desde el momento que deja las premisas del cliente y entra al dominio del ISP:

Una vez que el paquete ha sido marcado, pasa a formar parte de un comportamiento agregado específico con todos los otros paquetes marcados con el mismo DSCP. En el router de ingreso del dominio DS, el paquete se somete a un acondicionamiento de tráfico.



El paquete entonces es clasificado usando ya sea clasificadores MF o BA. El paquete entonces es medido contra un contrato de tráfico negociado y pasa por un policer/shaper, en el caso de ser necesario. En este punto, el paquete puede ser también remarcado con un DSCP diferente para indicar una degradación en el nivel de servicio. La siguiente figura muestra el funcionamiento de un router extremo DiffServ:



Los routers internos o del centro de la red implementan los tratamientos necesarios de forwarding de tráfico para diferentes PHB's soportados por el dominio DS.

A la salida del dominio DS, el paquete puede pasar a través de otro nivel de acondicionamiento de tráfico en el router de egreso del dominio. Este nivel de acondicionamiento de tráfico garantiza que el tráfico que abandona un dominio DS y entra al dominio adyacente sigue el contrato de tráfico acordado entre dichos dominios.

Entonces queda claro que la arquitectura DiffServ lleva la complejidad de la administración de la red hacia los extremos y deja el manejo y forwarding de los paquetes para el centro de la red tan simple y rápido como sea posible. Esta propiedad de escalabilidad es importante en DiffServ y marca la diferencia sobre otros esquemas de QoS como IntServ. Aunque por un lado el manejo del tráfico agregado reduce la flexibilidad de las garantías de provisión de QoS a los flujos individuales, por el otro mejora el total de la escalabilidad de la arquitectura.

- Bandwidth Broker y marco de políticas: se necesita una infraestructura de administración para montar servicios e2e a lo largo de múltiples dominios DS. La arquitectura DiffServ de dos bits propuso el uso de BB's para el mencionado propósito. Un BB es un agente que reside ya sea en cada dominio DS o entre dominios. Los BB's se comunican entre ellos para establecer los servicios e2e y mantener el estado de la información necesaria en lugar de mantenerlo en los routers de los dominios participantes. Los BB's son parte de lo que se llama marco de políticas. Las políticas se usan para regular el acceso a los recursos y servicios de red, basadas en un criterio administrativo. El marco de políticas es usualmente responsable por el control de admisión y la provisión de recursos a través de dos entidades principales: un punto de decisión de políticas (PDP) y un punto de refuerzo de políticas (PEP).

11.11.4.Evaluación de la arquitectura DiffServ

La arquitectura DiffServ tiene las siguientes ventajas sobre IntServ:

- Escalabilidad: DiffServ provee diferentes niveles de servicios a tráficos agregados en lugar de flujos individuales, no haciendo usao de esta manera de la necesidad de mantener los estados por flujo en cada router del camino e2e. Esto hace a DiffServ más atractiva para Internet.

- No tiene un tiempo de establecimiento: no hay necesidad de señalización en DiffServ, y generalmente los servicios se construyen con SLA's y forwarding y acondicionamiento del tráfico a través de los nodos y dominios de la red.

- Control de admisión "on-the-fly": esto se hace a través de políticas y remarcado de tráfico en los routers de borde, y así no hay necesidad de tomar una decisión de control de admisión en cada nodo del camino e2e.

- Compatibilidad con IPSec: dado que el tratamiento de paquetes en DiffServ requiere sólo el IP header, puede ser usado conjuntamente con IPSec contrariamente a lo que sucede con IntServ. Sin embargo existen algunos temas de la arquitectura acerca del manejo de los campos DS en los túneles IPSec que imponen algunas trabas.

Y por otro lado, existen en DiffServ algunos problemas operativos que necesitan ser resueltos antes que la arquitectura DiffServ pueda ser puesta en práctica:

- El estándar DiffServ propuesto de esta manera no provee aseguramiento de e2e QoS para el tráfico de Internet. Sólo especifica cómo los dominios individuales pueden ser configurados para proveer diferenciación de servicios a diferentes clases de tráfico.

- La arquitectura DiffServ no tiene un esquema específico para un control de admisión exacto. Sí usa una política y shaping de tráfico para proveer un control de admisión al instante (on-the-fly) en los routers extremos y de borde.

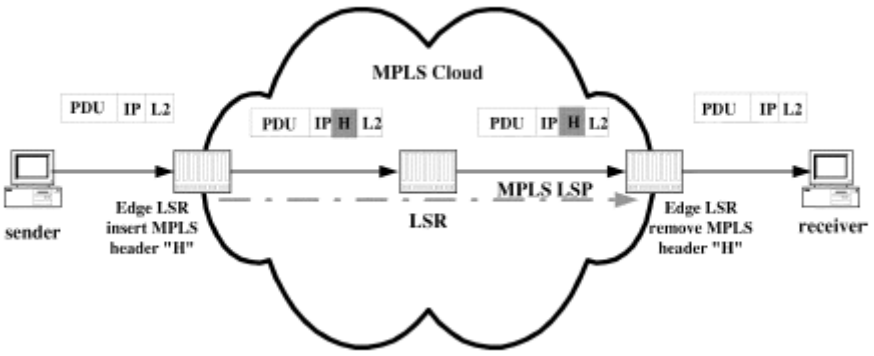
- DiffServ necesita de mapeo e interoperabilidad con otros esquemas (como MPLS y ATM) para ser implementado a través de muchos dominios. Existen muchas propuestas pero se necesita mayor investigación aún.

- Es necesario desarrollar interfaces de programación de aplicaciones (API's) apropiadas en los hosts finales para que las aplicaciones puedan sacar provecho de las capacidades de DiffServ.

11.12.MPLS y la ingeniería de tráfico

MPLS son las siglas de Multiprotocol Label Switching. Es un esquema avanzado de forwarding que trabaja entre la capa 2 y la capa 3 del modelo OSI. MPLS es similar a DiffServ, y todavía más parecido a ATM y Frame Relay. Se basa en el tagging de cada paquete con un header específico que determina su camino y forwarding en los nodos de la red. Los routers MPLS, llamados “label switching routers” (LSR’s), usan estos tags (labels) a lo largo de las tablas de forwarding para mapear un paquete a un camino específico denominado “label-switched path” (LSP). Los LSP’s corren entre dos LSR’s –un LSR de ingreso y un LSR de egreso- ambos ubicados en los extremos de la red MPLS. Un grupo de paquetes que tengan el mismo comportamiento de forwarding sobre un mismo camino es denominado “forwarding equivalent class” (FEC). El término “traffic trunk” se define como un agregado de flujos con la misma clase de servicio, o FEC, que puede ser colocado en un LSP. La configuración de LSR’s para mapear labels específicos a LSP’s específicos se hace por medio de un protocolo de distribución de labels (LDP).

A continuación se muestra el esquema de una red MPLS:



La habilidad de MPLS de soportar un ruteo explícito lo transforma en un buen candidato para ser usado en “traffic engeneering”. Traffic engeneering se ha transformado en una herramienta importante usada por los proveedores de servicios de red para el control de los recursos y la optimización de la performance.

Con respecto a DiffServ y MPLS, hay muchas propuestas para soportar DiffServ sobre MPLS. Los DSCP’s pueden ser mapeados a diferentes labels en los headers MPLS, y los LSP’s son usados para construir PDB’s dentro de los dominios DS.

Resumiendo, el IETF ha propuesto dos modelos de servicios –IntServ y DiffServ- para proveer QoS en una red basada en IP. El estándar DiffServ es más escalable y apto para Internet y además está siendo cada vez más implementado por los vendedores de equipamientos de redes. Se usa DiffServ y MPLS para soportar QoS sobre múltiples dominios, y se usa Traffic Engeneering en la optimización de performance y control de recursos en redes QoS. Pero como desventaja, la arquitectura DiffServ actual no provee soporte e2e QoS para las aplicaciones de usuarios finales. Hay muchos temas sin resolver como el agregado de tráfico, el control de admisión y el mapeo QoS a nivel aplicación, los cuales deben ser resueltos antes que DiffServ se ponga en práctica.

11.13.Evaluación para el caso en estudio

Los requerimientos de las aplicaciones multimedia pueden variar con el algoritmo de codificación/decodificación y el protocolo de tiempo real usado, tamaños de paquetes y el uso de un canal simple o canales separados para cada componente del stream multimedia. Para nuestro caso de transmitir un video streaming de alta capacidad, hay elementos clave que introducen un piso determinante de latencia y jitter. Estos son principalmente los routers intermedios de la red en su camino extremo a extremo, que por su lógica de funcionamiento (trabajan con matrices de rutas, cache de forwarding y path discovery) hasta el momento es imposible anular su efecto nocivo desde el punto de vista del retardo.

Hay dos escuelas de pensamiento en cuanto a garantía de calidad en los servicios prestados. Una de ellas sostiene el uso de **QoS** y la otra sostiene el uso de C**ontrol de Congestión** (esta última para garantizar seguridad y justicia a los otros flujos que usan la misma red). Pero también existe la posibilidad de emplear **Overprovisioning**, que es ampliar la capacidad de las redes por sobre los picos de tráfico esperados, dado que es más económico y fácil de administrar que QoS, bajo un monitoreo exhaustivo. En realidad, uno se puede preguntar qué es hacer un overprovisioning de la red y hay varias respuestas. Algunos dirán que es aumentar el ancho de banda de la red en un 10-15%, otros dirán que es aumentar el ancho de banda de la red hasta que no haya colas en las interfaces de los dispositivos, y otros dirán que es aumentar el ancho de banda de la red hasta que no haya más pérdidas de paquetes. Pero en definitiva, es aumentar la capacidad de la red para garantizar un servicio confiable.

En mi caso descarto overprovisioning porque me interesa estudiar la transmisión HDTV con algún mecanismo explícito que me permita que la aplicación sea capaz de obtener la suficiente información de la red acerca

de las condiciones actuales de congestión y así pueda adaptarse en forma apropiada (tal el caso de los mecanismos de control de congestión). Además, en el caso de Abilene como backbone, el porcentaje de uso promedio es de aproximadamente el 18% por lo cual la capacidad es más que buena para soportar el tráfico ofrecido.

Se pueden ver algunas mediciones de velocidad y utilización de los backbones de Abilene [80]:

```
Mbps   %Util Xferred Gb  Network
   ------  ----- ----------  -------
 1 1653.1   17.3    17434.8  Backbone circuit to Chicago (OC-192)
 2 1638.7   17.2    17283.3  Backbone circuit to Indianapolis (OC-192)
 3 1519.2   15.9    16022.3  Backbone circuit to Kansas City (OC-192)
 4 1489.1   15.6    15705.7  Backbone circuit to Kansas City (OC-192)
 5 1458.8   15.3    15385.6  Backbone circuit to Denver (OC-192)
 6 1348.2   14.1    14219.2  Backbone circuit to Indianapolis (OC-192)
 7 1073.2   11.3    11319.0  Backbone circuit to Sunnyvale (OC-192)
 8 1000.1   10.5    10548.2  Backbone circuit to Denver (OC-192)
 9  961.7   10.1    10142.5  Backbone circuit to New York (OC-192)
10  834.2    8.7     8798.6  Backbone circuit to Washington (OC-192)
11  738.2    7.7     7786.0  Backbone circuit to Chicago (OC-192)
12  720.1    7.6     7594.9  Backbone circuit to New York City  (OC-
192)
13  705.8    7.4     7444.5  Backbone circuit to Sunnyvale (OC-192)
14  639.6    6.7     6745.3  Backbone circuit to Seattle (OC-192)
15  633.9    6.6     6685.2  Backbone circuit to Los Angeles (OC-192)
16  602.5   25.4     6354.7  MAX (via MAX) (OC-48)
17  499.2    5.2     5265.0  Calren (10GIGE)
```

También es difícil hacer frente a los argumentos de la gente que sostiene el overprovisioning, dado que con la actual baja utilización de los backbones IP, es muy difícil convencerla que implemente soluciones costosas de QoS. Y con overprovisioning también es difícil garantizar en forma absoluta el delay y jitter dado que **el principal problema radica en los enlaces de acceso y end-hosts.**

La opinión de la gente que trabaja con Abilene es que se podría decir que nunca ha experimentado congestión en dicho backbone, o quizás sí pero por periodos de unos pocos segundos que suceden normalmente por problemas de congestión en los links de acceso. Pero es un hecho muy raro usualmente.

Otra alternativa válida es también comenzar con la red normal usando un control de congestión, y si se quiere tener menos riesgos de congestión, se puede hacer overprovisioning de la red, aunque es una alternativa más onerosa.

Es una realidad que el crecimiento y la actualización de las redes fue más rápido que el crecimiento del apetito de los usuarios por mayor ancho de banda.

Con respecto al descarte de QoS, muchas opininiones concuerdan en que los protocolos de QoS no trabajan bien cuando son realmente requeridos. Además de sufrir de ciertas características de interoperabilidad a través de diferentes plataformas y dominios, la complejidad agregada le quita valor también. QoS se puede aplicar para trabajar en redes sin congestión, pero performances similares se pueden obtener sin QoS pero usando overprovisioning en la red. Cuando las redes se vuelven congestionadas, los esquemas de QoS usualmente tambalean debido al agotamiento de los recursos.

Un desafío interesante a futuro sería desarrollar un sistema tal que si un camino de red reservado y un ancho de banda dado están disponibles (sea empleando MPLS, DiffServ, o cualquier mecanismo que sea) la red pueda transmitir a dicha velocidad; pero si ese camino no está disponible y si el flujo de datos atraviesa una red IP “best-effort”, entonces se empleará un mecanismo de control de congestión que sea “network-friendly”.

Esta tesis estudia la transmisión de streams HDTV sin comprimir, por lo cual la latencia no es una característica de extrema importancia. En consecuencia se puede llegar a usar un jitter buffer para acomodar las variaciones de latencia en el trayecto de los datos. Se puede tener un **jitte buffer de hasta 10 segundos** siguiendo los estándares **ITU G.1010**.

Por tales motivos, se descarta un análisis de aplicación de QoS a esta investigación, dejándola como tema de otra tesis si es que se quiere ahondar en los beneficios que reditúa su uso para aplicaciones de streaming multimedia de alta velocidad.

12.Multicast y Broadcast

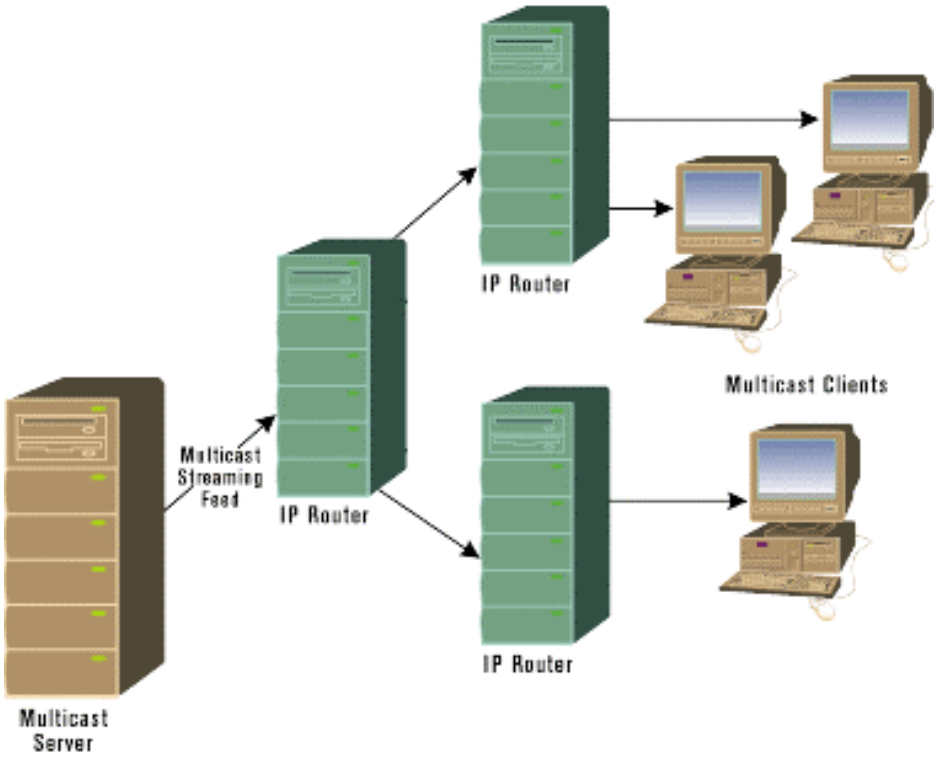
12.1.Descripción

La mayoría de las comunicaciones son unicast, o sea, punto a punto. Esto es ineficiente para tratar con muchos receptores dado que el servidor debe generar N copias de los datos a transmitir para N clientes, lo que conlleva además a un cuello de botella significativo en determinados casos. El multicast es una forma eficiente de distribuir información hacia muchos hosts en una red como puede ser Internet y sin la necesidad de tener que

crear una conexión cliente/servidor para cada host. O sea, es una difusión punto-multipunto donde un host fuente envía un mensaje a un grupo de hosts destinos. La principal ventaja del multicast es que reduce el ancho de banda consumido enviando un solo mensaje hasta un host cercano al grupo destino, para luego ser entregado a los miembros del mismo.

En forma muy resumida, los routers de las redes que trabajan con multicast deberán contar principalmente con el protocolo MBGP (Multiprotocol Border Gateway Protocol) [110]. MBGP es una mejora al BGP que permite el intercambio de rutas de IP multicast. BGP intercambia dos grupos de rutas, un grupo para enrutamiento unicast y otro grupo para enrutamiento multicast. Las rutas asociadas con multicast son usadas por el Protocolo de Multicast Independendiente (PIM) [111] para la construcción de los árboles de distribución, y estos dominios PIM se conectan a través del protocolo de descubrimiento de orígenes multicast (MSDP) [112]. Recordamos aquí que los core routers de Internet2 son Cisco y Juniper.

A continuación se detalla el diagrama de transmisión multicast:



Los grupos multicast tienen direcciones Internet especiales. Las direcciones multicast destino son copiadas en el campo “Destination Address” de los paquetes IP.

Para ser un cliente multicast, el host debe estar conectado a una red habilitada para multicast.

El multicast agrega complejidad a la red. El uso de redes IP con transporte “best-effort” trae acarreada heterogeneidad. El multicast inserta un order de magnitud mayor de heterogeneidad, dado que cada receptor “ve” diferentes características de pérdidas y además es difícil tener un feedback oportuno de una manera escalable.

Algunos de los productos que soportan multicast son:

Routers Multicast	3Com, Ascend, BayNetworks, Cisco Systems
Switches Multicast	3Com, PACE Switch
Sistemas Operativos Multicast	Windows 95 Windows 2000, Mac OS, todas las versiones de Unix
Stack Multicast	FTP Software, ICAST Communications, Microsoft, NetManage,
Aplicaciones Multicast	ICAST Communications, Starburst

12.2.Evaluación para el caso en estudio

Para el caso en estudio se descarta la transmisión multicast debido a que no hay mucha experiencia en el mundo y a que también se hace difícil encontrar redes con grandes anchos de banda para transmitir y experimentar con este tipo de tráfico. En el caso de Abilene, este backbone está implementado para trabajar con multicast hoy en día. Por otro lado, la infraestructura de Internet2 –backbones, conectores, gigapops e instituciones miembros- no tiene plenamente implementado el multicast en sus routers –lo que implica principalmente configurar los protocolos MBGP, PIM-Sparse Mode y MSDP-, razón por la cual no pueden funcionar con este tipo de modalidad de transmisión. En verdad no hay

ninguna limitación técnica a la hora de implementar multicast en la transmisión de HDTV sobre IP. Pero una de las razones por las cuales la comunidad internacional no ha llevado a cabo implementaciones de este tipo es el alto costo de los equipos que soportan este tipo de modalidad de difusión.

Por otro lado, en el caso de llevar a cabo la transmisión de HDTV por redes IP “best-effort”, el protocolo de control de congestión más popular para transmitir audio y video usado en la actualidad soporta únicamente la modalidad de transmisión unicast –como se estudiará más adelante en este trabajo-, descartando de esta manera la modalidad multicast en el caso de tomar este camino.

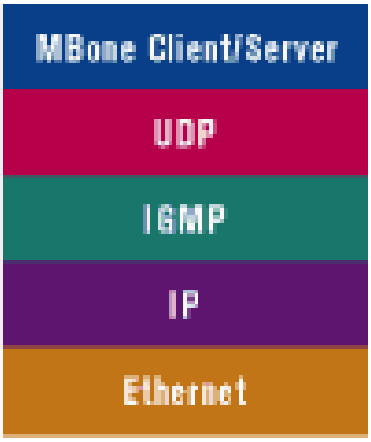
Finalmente, en esta instancia a mi me interesa investigar la problemática básica de la transmisión de HDTV sin comprimir a través de redes IP, razón por la cual la modalidad unicast sería la mejor elección por ser el escenario más simple para poder comenzar con el análisis, pudiendo así quedar el estudio de la transmisión multicast con sus dificultades inherentes de implementación como tema de otra tesis.

13.Algunas plataformas de transmisión de audio y video: MBone, 6Bone y Qbone

13.1.Descripción MBone

MBone o Multicast Backbone [81] es un backbone experimental para tráfico multicast de audio y video a través de Internet, basada sobre los protocols IP y UDP. MBone usa software especial que corre sobre estaciones de trabajo Unix para construir una red privada sobre Internet para hacer efectivo la transmisión de la información de multicast. El protocolo IP soporta multicasting –hay campos del datagrama para ello- que sería la transmisión de paquetes de audio y video a diferentes direcciones IP. Básicamente se usa el protocolo IGMP (Internet Group Multicast Protocol) [82], y para ello hay predeterminados bloques de direcciones IP destinados al Multicast en Internet.

Un esquema del stack de protocolos sería el siguiente:



Dado que la mayoría de los servers IP de Internet no soportan el multicasting, el MBone fue configurado como una red dentro de Internet a los fines de transmitir multicast. El MBone consiste de servers conocidos que manejan el protocolo multicast y que en su mayoría son estaciones de trabajo Unix. Para pasar paquetes multicast a través de routers de Internet que no manejan dicho protocolo es que se hace uso del tunneling, que es una técnica para encapsular paquetes multicast como paquetes unicast y así el router lo puede forwardear sin ningún problema hasta que el paquete alcance el siguiente server multicast que es el encargado de desencapsularlo y enviarlo en forma apropiada. Los routers trabajan con el demonio de routing multicast llamado “mrouted”.

Mbone se usa principalmente para transmisiones multimedias de los encuentros de los miembros del IETF. El ancho de banda del canal para el multicast MBone es de 500 Kbps, evidentemente muy inferior a las necesidades de transmisión de HDTV sin comprimir.

13.2.Descripción 6Bone

6Bone [83] es una red de prueba que trabaja sobre IPv6 y que asiste a la evolución y desarrollo de dicho protocolo. Es un proyecto de colaboración, operado informalmente con la supervisión del grupo de trabajo Ngtrans (IPv6 Transition) del IETF.

El 6Bone comenzó como una red virtual (usando encasulamiento de IPv6 sobre IPv4) operando sobre la Internet basada en IPv4 para soportar el transporte IPv6, y está migrando lentamente hacia enlaces nativos IPv6.

El objetivo primero de esta red fue el de testear los estándares y las implementaciones, mientras que actualmente se centra en testear los procedimientos operativos y de transición.

La red 6Bone opera bajo el IPv6 Testing Address Allocation [84].

13.3.Descripción Qbone

Se ha establecido una red experimental de prueba denominada QBone [85] que es parte de Internet2, para estudiar los temas relacionados con el despliegue y la implementación de los mecanismos de QoS basados en la arquitectura DiffServ. QBone está formada por participates de universidades, empresas, organizaciones tales como vBNS, Abilene, ESNet, CA*Net2, SURFnet, NREN, y muchos otros. Qbone ofrece un servicio llamado Qbone Premium Service que se basa en la característica denominada “EF PHB” de DiffServ. Esta característica se usa para implementar un servicio con bajo delay, jitter y pérdidas en añadidura al ancho de banda asegurado y se usa en servicios que tienen requerimientos estrictos de delay y jitter tales como las aplicaciones de tiempo crítico y multimedia.

Basado en diversos estudios, el grupo de trabajo Qbone ha concluido que existen algunas dificultades en el despliegue del Qbone Premiun Service a gran escala y son necesarios estudios más amplios a futuro. Las principales limitaciones son:

- Actualización de las redes para los proveedores (por ejemplo, todas las interfaces de acceso deben hacer policing)
- Cambios dramáticos en las operaciones de las redes, arreglos de peering y modelos de negocios
- Ausencia de medios razonables para verificar el servicio (por parte de los usuarios y los proveedores)

Más aún, en los ambientes de Internet2 existieron muy pocos problemas de performance de aplicaciones que se hayan podido adjudicar a la congestión de la red. En lugar de ello, la baja de la performance e2e casi siempre es provocada por fallas en los extremos o cerca de los mismos, como pueden ser los stacks TCP que se vuelven inutilizables como por un inadecuado socket buffering, errores duplex en Ethernet y problemas con los cables.

Por lo tanto, en este entorno, sólo las formas más simples de QoS entre dominios (interdoimain QoS) pueden ser desplegada. Actualmente se están

focalizando en desarrollar variantes de servicios “best-effort” que se despliegan en forma incremental, sin necesidad de policing, accounting ni cambios significativos en las prácticas operativas.Dentro de estos nuevos servicios, el denominado Qbone Scavenger Service (QBSS) está siendo testeado y permite que los usuarios, aplicaciones y redes de campus marquen el tráfico para que pueda ser convenientemente tratado en las interfaces que experimenten congestión. QBSS es diseñado para el tráfico TCP del tipo “bulk” que actualmente corre en forma voluntaria durante los períodos de baja utilización de la red (por ejemplo, transferencias nocturnas de juegos de datos científicos o backups de redes). También es usado por algunas universidades de Internet2, como en el caso de compartir archivos de recreación por parte de los estudiantes.

Adicionalmente, se están estudiando servicios como el denominado Aklternative Best Effort (ABE) que podría proveer aplicaciones interactivas con baja latencia y clase de servicio “best-effort”.

13.4.Evaluación para el caso en estudio

En el caso de MBone, es una red que usa multicast y además no tiene la velocidad de transmisión suficiente como para entregar tráfico HDTV sin comprimir. Por lo tanto queda descartado para la presente investigación.

En el caso de la red 6Bone, descarto su uso dado que trabaja con el protocolo IPv6, el cual lo descarté antes para mi actual investigación.

Y por último, en el caso de Qbone, descarto su uso dado que la implementación óptima de QoS aún no está madura como para basarme en ella para mi desarrollo.

14.Definición final de la plataforma de trabajo

14.1.Descripción y justificación

Estas son las características e hipótesis principales del sistema propuesto:

- HDTV sin comprimir con estándar de video SMPTE-296M en principio y audio AC-3.
- Luego el throughput de HDTV se amolda a la capacidad máxima del enlace de menor ancho de banda (enlace limitante) y las NIC del emisor y receptor

- Live streaming de audio/video no interactivo, y no una aplicación de audio/video interactivo como una videoconferencia o una sesión de VoD con interactividad de las funciones de Play/Rew/Fwd/Stop
- Audio y video se transmiten en streams independientes y sincronizados en el receptor
- Backbone compartido con otros flujos de datos (no es un medio controlado)
- Abilene como backbone
- Unicast: dos extremos se comunican punto a punto, uno de ellos transmite y el otro recibe
- Gigabit Ethernet a 1 Gbps como tecnología de red en los extremos (sería el enlace limitante para el tráfico HDTV) y en las placas de red del emisor y receptor
- Red IP "best-effort"
- RTP/UDP/IP
- Uso de RTP payloads y RTP profiles
- Control de congestión aplicado al flujo de video por medio de la variación de la tasa de frames de video
- Jitter buffers independientes para el audio y video en el receptor, que serán monitoreados constantemente para controlar los estados de congestión más pronunciados. En el caso del video, la tasa de frames saltará entre los valores [60 fps, 50 fps, 40 fps, 35 fps], mientras que en el caso del audio la tasa de muestreo saltará entre los valores [48 KHz, 44.1 KHz, 32 KHz].

El sistema aceptará una entrada de HDTV SMPTE-296M desde una cámara HDTV o un server con contenido HDTV, mientras que el audio AC-3 ingresa aparte. El input de video tiene una velocidad de transmisión de datos activos de video que se obtiene de esta manera:

- Estándar SMPTE-296M
- HDTV 720p 60 fps
- 720 x 1280 = 921600 pixels activos/frame
- 60 fps = 1105920000 bits/seg
- El payload activo es de aprox. 1.1 Gbps

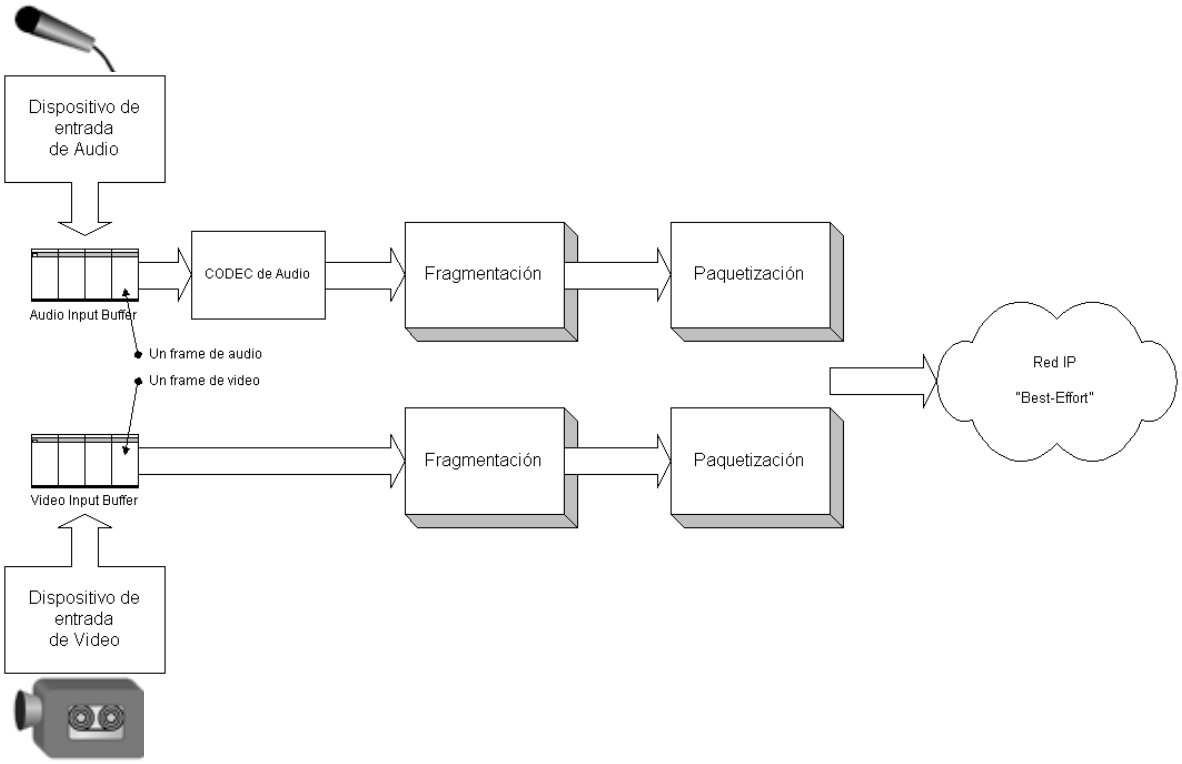
Por lo tanto, es obvio que sólo teniendo en cuenta la velocidad de datos del video, la tasa de transmisión es mayor que 1 Gbps. Por lo tanto hay que reducir algún parámetro del video en detrimento de la calidad. En este caso se elige que la profundidad de color sea de 8 bits en lugar de los 10 bits originales especificados por el estándar SMPTE-296M.

Cabe destacar que mi mayor interés está centrado en la transmisión de video más que la de audio, dado que dicho tráfico es un gran consumidor de recursos de red y me interesa analizar su comportamiento.

14.2.Esquema básico de transmisión de HDTV sin comprimir

En la transmisión de HDTV sin comprimir intervienen un emisor y un receptor, los cuales interactúan a través de una red IP.

En el **extremo emisor**, se capturan los datos multimedia sin comprimir en forma separada (audio samples y frames de video) dentro de sus respectivos buffers, desde los cuales se obtienen los frames. Los frames luego son asignados con un timestamp y un número de secuencia y así se cargan en paquetes RTP (paquetizado), estando ahora listos para ser transmitidos. Si los frames son muy grandes pueden ser fragmentados dentro de diferentes paquetes RTP. Luego que los paquetes de datos han sido enviados, los buffers de entrada son eventualmente liberados (acá no hay que dejar de tomar en cuenta la opción que el emisor no descarte los datos dado que podrían ser necesitados para llevar a cabo la corrección de errores en el caso de ser requerido, y así los almacene por un cierto tiempo dependiendo del esquema de corrección de errores usado). Además, el emisor es responsable de generar reportes de estado en forma periódica para los streams multimedia que está generando, inclusive aquellos requeridos para la “lyp-synchronization” (sincronización de audio y video). También recibe el feedback de la calidad de recepción por parte del lado receptor y usará dicha información para adaptar su transmisión (control de congestión).



Las muestras de audio en el extremo emisor son capturadas, digitalizadas y almacenadas en el buffer de audio de entrada. Generalmente el buffer de entrada se hace disponible a la aplicación luego que se han recolectado un número fijo de muestras de audio. Esto impone un cierto delay porque la primera muestra de audio no estará disponible hasta que que la última muestra haya sido almacenada.

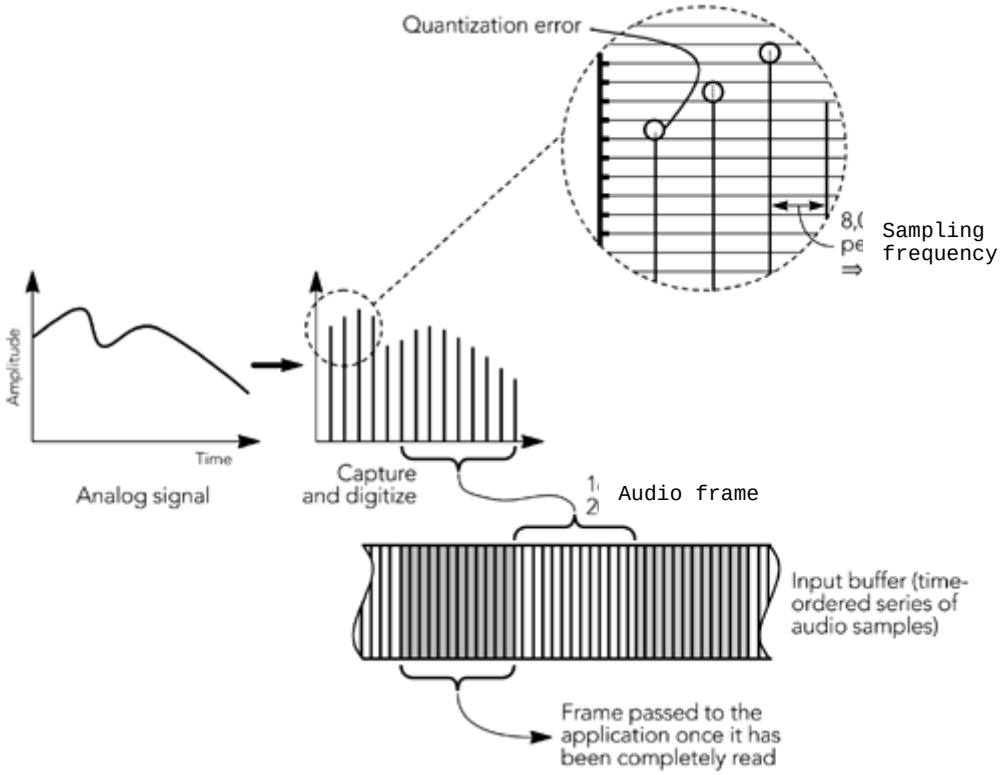


Diagrama del libro “RTP: audio and video for the Internet”
de Colin Perkins

En el caso del video, se empleará un dispositivo de captura que opere capturando los frames completos de la imágen de video entrelazado. Los dispositivos de captura deben ofrecer su salida en una variedad de formatos, espacios de color (por ejemplo YUV), profundidades y subsampling. En este punto es importante aclarar que la aplicación debe ser capaz de “parsear” (analizar) la cola de entrada (input queue) y determinar dónde comienza y dónde finaliza cada línea de escaneo del video (aunque también hubiese sido factible contar con un dispositivo de captura de video que capture directamente las líneas de escaneo en lugar de los campos o frames enteros). Luego las líneas de escaneo se asocian con el instante de captura (capture time) del frame correspondiente y son pasadas al módulo RTP de la aplicación para su correspondiente paquetización y transmisión.

Generación de paquetes RTP

Cada frame tiene asociado un timestamp, del cual se deriva el timestamp RTP. Si el formato de payload soporta fragmentación –tal el caso de los formatos de payload para video sin comprimir y audio AC-3 que se explicarán más adelante-, los frames de audio y las líneas de escaneo de

video se fragmentan según el MTU del path de la red. Finalmente se generan uno o más paquetes RTP para cada frame o línea de escaneo, cada uno de los cuales incluye los datos multimedia y cualquier payload header requerido. El formato del paquete multimedia y el payload header se definen de acuerdo a la especificación del formato de paylaod usado. Las partes críticas en la generación de los paquetes RTP son la asignación de timestamps a los frames, la fragmentación de largos frames y la generación del paylaod header. Luego de la transmisión de los paquetes RTP –como ya se mencionó antes-, la cola de entrada es liberada a menos que se guarden los datos un cierto tiempo para usarlos con algún tipo de algoritmo de corrección de errores.

Timestamps y el modelo de timing RTP

El timestamp RTP representa el instante de muestreo del primer octeto de datos en el frame. Comienzo con un valor inicial aleatorio y se incrementa a una velocidad dependiente de la multimedia.

Durante la captura de multimedia en vivo –tal el caso en estudio-, el instante de muestreo es simplemente el instante en el cual se captura la multimedia desde el video frame grabber (placa capturadora de HDTV de la PC) o el dispositivo de audio (placa capturadora de sonido de la PC). Como el audio y video van a ser sincronizados, se deberá tener sumo cuidado en tener en cuenta el delay en los diferentes dispositivos de captura.

En el **extremo receptor**, se recolectarán los paquetes RTP que provienen de la red, se recuperará el timing y se llevará a cabo la sincronización entre los flujos multimedia para luego llevar a cabo la reproducción o playback. Además enviará al emisor el feedback de calidad de recepción para permitir que éste adapte su transmisión.

El primer paso del proceso receptor es recolectar los paquetes de la red e insertarlos dentro de una cola de entrada (input queue) para cada stream (audio y video). Luego los paquetes son recolectados desde la cola de entrada y así son insertados en un jitter buffer (playout buffer). A la cola de entrada llegan los paquetes en el orden impuesto por la red y luego al ingresar al jitter buffer se ordenan según los timestamps, y así el proceso de insertar paquetes dentro del citado buffer corrige cualquier reordenamiento inducido durante el transporte. Los paquetes permanecen en el buffer hasta que se reciban los frames completos y para remover cualquier variación del timing inter-paquete causado por la red (jitter). El cálculo de la cantidad de delay a agregar es uno de los aspectos más críticos del diseño de la

implementación RTP. Luego cada paquete es marcado (tagged) con el tiempo de reproducción deseado para el frame correspondiente.

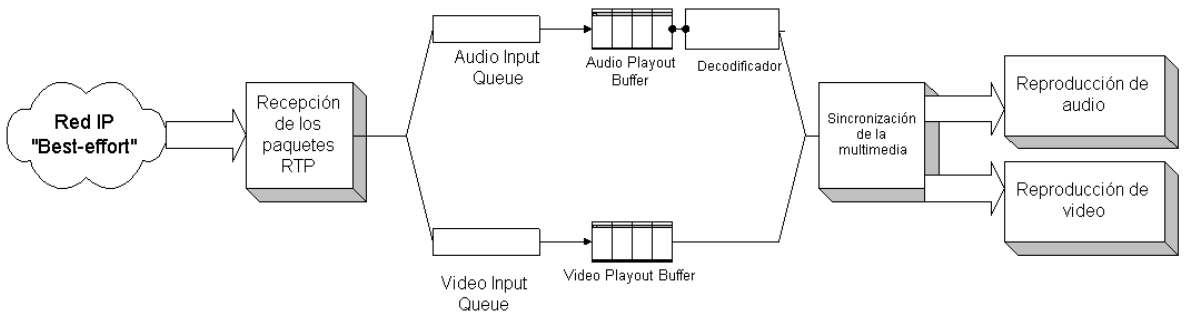
Luego de que es alcanzado el tiempo de playout, los paquetes son agrupados para formar frames completos, y eventualmente se corregirán los errores en el caso de ser requerido.

En el caso del audio, que arriba codificado, luego de ser reparados los frames son decodificados. Dependiendo del CODEC especificado, el proceso de decodificación de los frames puede ser hecho antes que el proceso de reparación de los mismos.

En este punto puede haber diferencias observables en las velocidades de clock nominales del emisor y el receptor. Tales diferencias se manifiestan como una derivación (drift) del reloj multimedia RTP relativo al reloj de playout. El receptor debe compensar esta desviación de reloj para evitar saltos en la reproducción.

Finalmente los datos multimedia son reproducidos en el receptor, audio y video en un dispositivo de salida por separado. Vale decir que los streams de audio y video no se mezclarán para obtener un solo stream para la reproducción en un único dispositivo de salida.

Como se ve, la operación del receptor es mucho más compleja que la del emisor, principalmente a causa de la recuperación del timing de los streams afectados por el jitter inducido por la red.



En el caso de la reproduccón de video, si la misma se la hace en un monitor faltaría entonces un bloque conversor de color YUV a RGB.

Como se mencionó antes, RTP usará dos puertos UDP para cada sesión: uno para datos y otro para el tráfico de control. Esto significa que la aplicación en el extremo receptor abrirá dos sockets para cada sesión. Dado que RTP corre sobre IP/UDP, los sockets usados son del tipo SOCK_DGRAM (estándar) como los provistos por los sockets API Berkeley en los sistemas Unix y los Winsock provistos por Microsoft.

Cuando un paquete arriba al receptor, lo primero que hace es ser procesado en el sistema operativo a nivel IP/UDP, que incluye un buffer de recepción del socket. Luego el paquete se dirige al stack RTP y de allí al módulo de recepción del paquete, com se ve en la siguiente figura:

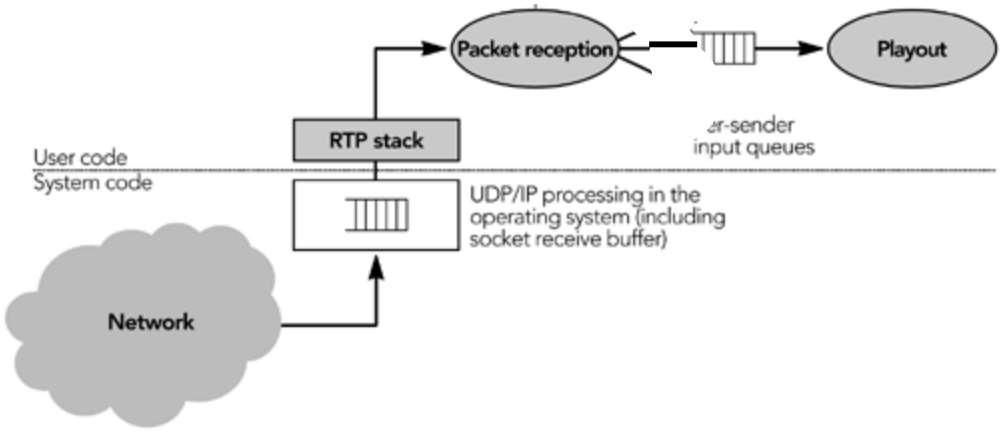


Diagrama del libro “RTP: audio and video for the Internet”
de Colin Perkins

Aquí hay que tomar en cuenta que como esta es una implementación con una transmisión de alta velocidad, habría que analizar la posibilidad de agrandar el tamaño del buffer del socket más allá del valor predeterminado con el objeto de no perder paquetes.

En este punto vale la pena aclarar que los formatos de paylaod RTP definen una velocidad nominal de reloj para el stream multimedia, pero no especifica requerimientos sobre la exactitud y estabilidad del reloj. El emisor y receptor corren comúnmente a velocidades mínimanmente diferentes, forzando a que el receptor lleve a cabo una compensación para dicha variación. Así los receptores deberán detectar la presencia del corrimiento del reloj, estimar su magnitud y ajustar el punto de compensación. Una manera de hacerlo es ajustar el reloj del receptor para que se corresponda al reloj del emisor.

Más adelante se explicarán los aspectos del diseño de los media buffers y la reproducción de la multimedia.

14.3.Opciones para transportar HDTV sobre IP en Internet2

Un sistema para transportar HDTV sin comprimir sobre IP deberá aceptar una señal de video digital SMPTE-292M y encapsularla dentro de RTP para transmisiones sobre IP. En el lado receptor, la señal puede ser regenerada o el video puede ser mostrado directamente. Hay un número de opciones en que esto puede ser hecho, dependiendo de la meta del transporte.

Si el intento es enlazar equipos ya existentes, entonces la aproximación correcta puede ser la **emulación de circuito**, donde la señal SMPTE-292M (que incluye el formato SMPTE-296M) es mapeada sobre IP en forma indistinta de su contenido.

La alternativa es un **empaquetado nativo**, donde se define un formato de payload RTP para transportar el video directamente, usando SMPTE-292M (que incluye SMPTE-296M) sólo localmente.

La emulación de circuito provee una entrega transparente del bit-stream de HDTV, adecuado para la ingresarlo a otros dispositivos. Soporta cualquier formato que soporte el estandar SMPTE-292M, sin la necesidad de ser adaptado a los detalles de ese formato. La principal desventaja es que el empaquetado se desentiende del medio, y no se puede optimizar basado en el formato de video. Esto hace a la emulación de circuito algo intolerante a las pérdidas.

El empaquetado nativo mira el contenido del stream de video, y actúa sobre los datos de video dentro de él. Así, se necesitan definir formatos nativos para todas las posibles resoluciones de video, aunque aquellos formatos pueden hacerse más óptimos. También expone el contenido a la manipulación por sistemas finales, más bien que ocultándolo dentro de otra capa (layer) del framing. Su uso estaría ligado al formato de payload RTP para video SMPTE-292M [70].

Si se desea mostrar y manipular el contenido HDTV en computadoras y no hay necesidad de regenerar la señal de salida SMPTE-292M, entonces una elección adecuada sería el uso del empaquetado nativo, empleado en conjunto con el formato de payload RTP para video sin comprimir [86], considerado como estándar propuesto en Internet a partir de septiembre de 2005.

14.4.Transporte de audio y video

El contenido de video casi siempre está asociado con información adicional como pistas de audio, etc. En las aplicaciones profesionales de video digital, estos datos comúnmente son embebidos en las porciones no activas del stream de video, conocidos como los períodos de blanking horizontal y vertical.

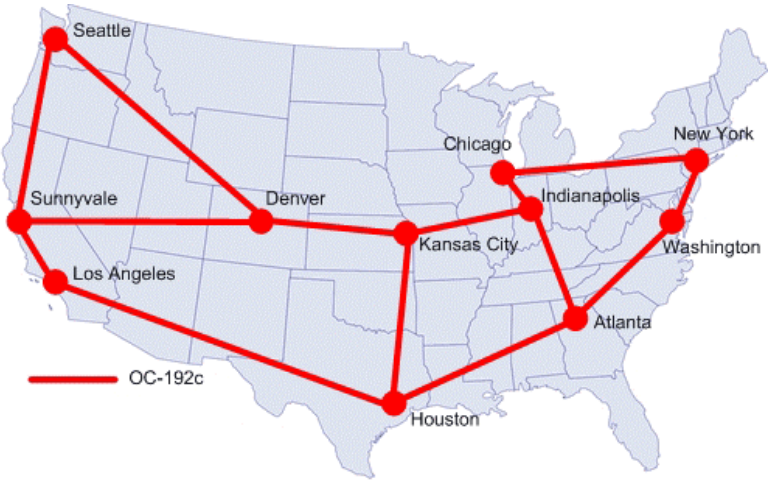
Como se dijo en el ítem anterior, en este trabajo se usará el empaquetado nativo y en consecuencia se usará para el video el "RTP Payload for Uncompressed Video", el cual especifica que el audio sincronizado con el video no se puede enviar como datos auxiliares dentro del mismo stream RTP de video, sino que se deberá enviar por intermedio de otro stream RTP independiente. Para el audio se usará el "RTP Payload Format for AC-3 Audio".

Por lo tanto, **tendremos que transmitir un stream RTP de audio y otro stream RTP de video en forma independiente**, cada uno con su correspondiente formato de payload RTP.

14.5.Nivel de red y enlace

El escenario planteado para la transmisión de HDTV es Internet2, con lo cual se hace uso de su backbone Abilene. Como se mencionó anteriormente, Internet2 está conformada por diferentes tecnologías de red a nivel físico y de enlace, que ofrecen una velocidad máxima de transmisión de 10 Gbps.

A continuación se repite el mapa de la velocidad de los troncales actuales de Abilene [7]:



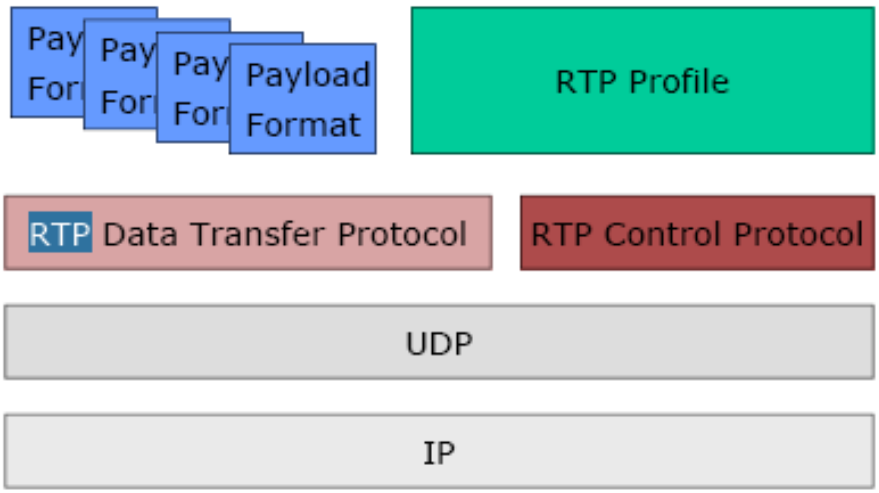
En cuanto al uso de tecnologías de red en estas capas, es lo mismo transmitir usando Packet over SONET (POS) que Gigabit Ethernet. Desde el punto de vista del delay y jitter, los partidarios de SONET dicen que debido a la característica de los paquetes TDM sobre SONET estos valores serán mas predecibles. En cambio, los partidarios de Ethernet dicen que una red bien diseñada, y quizás hasta bajo un esquema de "under provissioning", es comparable con SONET en términos de delay y jitter. Desde el punto de vista de los costos, la implementación de la tecnología Ethernet es más económica que la implementación de POS, y además Ethernet tiene mayor disponibilidad en todas partes.

En definitiva, la transmisión de HDTV sobre Internet2 puede adoptar cualquier tecnología de nivel 1 y 2 subyacentes, disponibles al día de hoy.

En verdad, el estudio en profundidad de las características implícitas de cada tecnología de nivel físico y enlace con respecto a los parámetros de QoS puede ser tema de otra tesis. En este escenario planteado, lo que importa más que nada es el uso de jumbogramas y no el paquete estándar de 1500 Bytes como se justifica más adelante, con el fin de obtener más performance de red.

14.6.Protocolos de nivel de red y transporte

El stack de protocolos de multimedia desde el nivel de red hacia arriba utilizado en esta transmisión se basa en el siguiente esquema básico de componentes de protocolos:



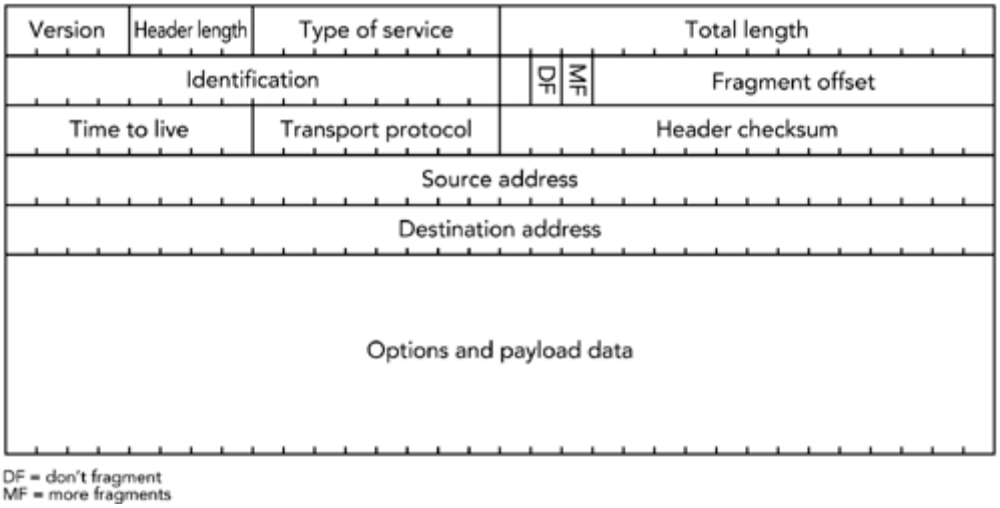
A continuación, se detallan brevemente los formatos de paquetes del nivel de red y transporte, principalmente porque son necesarios para calcular más adelante la tasa del overhead involucrado en la transmisión.

Nivel de Red

Se usa IP versión 4 [43] como protocolo de red, dado que el objeto de esta tesis es justamemente afirmar la factibilidad de transmitir un tráfico de alta velocidad y de tiempo real –como lo es HDTV- sobre redes de paquetes IP.

El protocolo IP permite hacer una abstracción de una única red, sin cambiar la naturaleza de los sistemas subyacentes.

A continuación se grafica el formato del header IPv4, que es el estándar actual de Internet [87]:



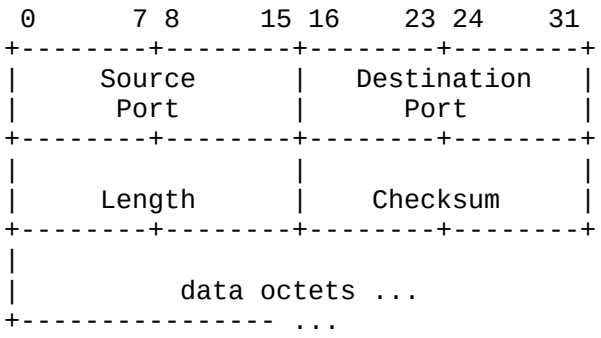
Nivel de Transporte

Se usa UDP [42] que es un protocolo no orientado a conexión y que por ende se usa en aplicaciones donde se necesita tiempo real o cerca de tiempo real como el audio y video. UDP es un protocolo de transporte no confiable que no envía ACK's al host origen de la transmisión, y por lo tanto no puede ser controlado por los mecanismos tradicionales de TCP [87].

UDP entonces debe lidiar con algunos problemas como:

- Retransmisión para entrega confiable de paquetes
- Paquetización y reensamble
- Control de flujo
- Control de congestión (congestion avoidance)

Formato del header de UDP:



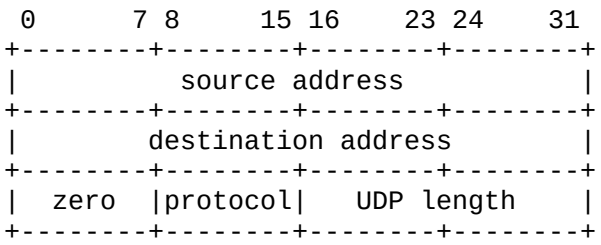
Source Port: 16 bits – El número de puerto del emisor, contiene un cero si no se usa.

Destination Port: 16 bits – El puerto al cual se destina el paquete

Length: 16 bits – La longitud en bytes del header UDP y los datos encapsulados; el mínimo valor para este campo es 8.

Checksum: 16 bits – Campo para chequeo de errores

El pseudo header, conceptualmente prefijado para el header UDP, contiene los siguientes campos:



Protocolos que corren sobre UDP

Se usará RTP y RTCP [40] como los protocolos para transmitir los paquetes multimedia de audio y video. Los mismos ya fueron detallados en la sección 6 del presente trabajo.

Pero en breves palabras, vale destacar básicamente que RTP -originalmente creado para ser usado en conferencias multicast- fue desarrollado por el Audio/Video Transport Working Group de la IETF y provee funciones de transporte de red extremo a extremo, tales como audio, video o simulación de datos, sobre servicios de redes unicast o multicast. RTP no direcciona reservación de recursos y no garantiza calidad de servicio para servicios de tiempo real. RTP provee detección de pérdidas y recepción de reportes de calidad de servicio, recuperación del timing y sincronismo, identificación del payload y origen (source) y marcado de eventos significativos dentro del stream multimedia.

El transporte de datos está aumentado por un protocolo de control (RTCP) para permitir el monitoreo de la entrega de datos de manera escalable a grandes redes multicast, y proveer una mínima funcionalidad de control e identificación. RTP y RTCP están diseñados para ser independientes de las capas subyacentes de transporte y red. Además, el protocolo soporta el uso de traductores y mezcladores a nivel RTP.

RTP es la clave para el transporte de audio y video en redes IP, junto con sus **perfiles** y **formatos de payload** [47] asociados. RTP por el momento necesita de un perfil para tener un uso en particular, antes que el protocolo RTP sea finalmente completado (su primera versión fue finalizada en 1996 y siempre está en continua revisión). Los perfiles son acompañados por diferentes especificaciones de formatos de payload que describen el transporte de un formato multimedia en particular.

Vale destacar aquí que RTP basa su filosofía en los conceptos de “Application Level Framing” y “End-to-end Principle”. El primer concepto hace mención a que sólo la aplicación tiene el suficiente conocimiento de para tomar una acertada decisión de cómo los datos deben ser transportados, mientras que el segundo hace mención a que los extremos son los responsables por los datos asegurando así confiabilidad extremo a extremo inclusive si los saltos (hops) intermedios no son confiables (o sea, los extremos son inteligentes y la red es “muda”).

14.7.Transporte de Audio

El audio de origen de la transmisión es del tipo AC-3. Por lo tanto para su debida transferencia se usará un formato de payload RTP para el transporte de audio codificado AC-3, documentado en la RFC 4184 [88], la cual fue aprobada en octubre de 2005 con carácter de "Estándar de Protocolo Propuesto". AC-3 es un sistema de codificación de audio multicanal, alta calidad y baja velocidad usado en HDTV en USA, en DVD, televisón por cable y satélite y otros medios. Este documento incluye soporte para fragmentación de datos.

AC-3 es el algoritmo de compresión digital para audio adoptado por la ATSC. Un stream de audio AC-3 es una secuencia de "sync frames" de igual tamaño. Cada frame de sync es una unidad independiente de datos. El decodificador AC-3 deberá ser presentado con el frame de sync entero.

Hay una necesidad creciente de usar audio AC-3 sobre redes IP. El protocolo RTP provee un mecanismo de sincronización de streams y así se convierte en la mejor solución de transporte para AC-3. Las aplicaciones de streaming de audio AC-3 incluyen el streaming desde un servidor multimedia hogareño hacia un display, video en demanda y radio multicanal sobre Internet.

AC-3 puede transmitir 5.1 canales de audio a velocidades aproximadas a la mitad de un canal PCM. AC-3 fue diseñado para señales muestreadas a velocidades de 32, 44.1 y 48 KHz. El codificador AC-3 acepta audio PCM en forma de "palabras" como entrada de un tamaño de 24 bits como máximo y luego produce un bit stream codificado consistente con este estándar. La velocidad de datos puede entonces variar entre 32 Kbps y 640 Kbps dependiendo del número de canales y la calidad deseada .

A continuación se detalla una tabla obtenida del estándar AC-3 correspondiente a los diferentes códigos de tamaño de frame para una palabra de 16 bits:

Table 5.18 Frame Size Code Table (1 word = 16 bits)

frmsizecod	Nominal Bit Rate	fs = 32 kHz words/syncframe	fs = 44.1 kHz words/syncframe	fs = 48 kHz words/syncframe
'000000' (0)	32 kbps	96	69	64
'000001' (0)	32 kbps	96	70	64
'000010' (1)	40 kbps	120	87	80
'000011' (1)	40 kbps	120	88	80
'000100' (2)	48 kbps	144	104	96
'000101' (2)	48 kbps	144	105	96
'000110' (3)	56 kbps	168	121	112
'000111' (3)	56 kbps	168	122	112
'001000' (4)	64 kbps	192	139	128
'001001' (4)	64 kbps	192	140	128
'001010' (5)	80 kbps	240	174	160
'001011' (5)	80 kbps	240	175	160
'001100' (6)	96 kbps	288	208	192
'001101' (6)	96 kbps	288	209	192
'001110' (7)	112 kbps	336	243	224
'001111' (7)	112 kbps	336	244	224
'010000' (8)	128 kbps	384	278	256
'010001' (8)	128 kbps	384	279	256
'010010' (9)	160 kbps	480	348	320
'010011' (9)	160 kbps	480	349	320
'010100' (10)	192 kbps	576	417	384
'010101' (10)	192 kbps	576	418	384
'010110' (11)	224 kbps	672	487	448
'010111' (11)	224 kbps	672	488	448
'011000' (12)	256 kbps	768	557	512
'011001' (12)	256 kbps	768	558	512
'011010' (13)	320 kbps	960	696	640
'011011' (13)	320 kbps	960	697	640
'011100' (14)	384 kbps	1152	835	768
'011101' (14)	384 kbps	1152	836	768
'011110' (15)	448 kbps	1344	975	896
'011111' (15)	448 kbps	1344	976	896
'100000' (16)	512 kbps	1536	1114	1024
'100001' (16)	512 kbps	1536	1115	1024
'100010' (17)	576 kbps	1728	1253	1152
'100011' (17)	576 kbps	1728	1254	1152
'100100' (18)	640 kbps	1920	1393	1280
'100101' (18)	640 kbps	1920	1394	1280

Aquí se puede observar que para una misma velocidad de transmisión del bit stream de audio AC-3, hacen falta menos cantidad de palabras por syncframes (frames de sincronización detallados más abajo) para una tasa de muestreo de 48 KHz, y más cantidad de palabras por syncframes para tasas de muestreo de 44.1 KHz y 32 KHz.

Típicamente las implementaciones de CODECs AC-3 ofrecen al usuario diferentes alternativas de tasa de bits, y estas tasas son traducidas por el CODEC en tamaños de frames. Por lo tanto, **la forma de cambiar la tasa**

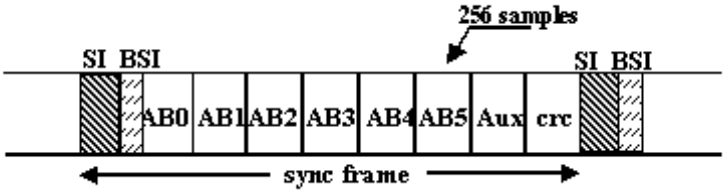
de bits del audio AC-3 es cambiando la cantidad de palabras por frame de sincronización.

También al bajar la tasa de muestreo del audio se obtiene un efecto de reducción de la velocidad del audio AC-3, pero esto no es recomendaddo a causa de que el cambio de frecuencias de muestreo repercute en la calidad de sonido del usuario final.

Por otro lado, si se reduce la cantidad de bits por muestra de datos de audio agregaría una indeaseable distorsión por cuantización, por lo tanto tampoco es recomendado.

En la presente implementación, el audio AC-3 comenzará a ser transmitido con una tasa de muestreo de 48 KHz en principio, y a una velocidad de 640 Kbps.

Los streams de bits de AC-3 están organizados dentro de frames de sincronización. Cada frame de sincronización AC-3 contiene los siguientes campos:



- AB: audio block (cada uno representa 256 muestras PCM para cada canal)
- SI: synchronization information header (contiene información necesaria para adquirir y mantener la sincronización del CODEC)
- BSI: bit stream information header
- CRC: cyclic error check

El frame entero representa el tiempo de duración de 1536 muestras PCM a través de todos los canales codificados (ATSC).

La duración de un frame AC-3 varía de acuerdo a la velocidad de muestreo de esta manera:

Velocidad de muestreo	Duración del frame
48 kHz	32 ms
44.1 kHz	aprox. 34.83 ms
32 kHz	48 ms

Campos del header RTP

Payload Type (PT): está especificado por un RTP profile bajo el cual este formato de payload es usado.

Marker bit (M): se establece en 1 para indicar que el payload del paquete RTP contiene como mínimo un frame completo AC-3 o contiene el fragmento final de un frame AC-3.

Extensión bit (X): definido por el RTP profile usado.

Timestamp: una palabra de 32 bits que corresponde al instante de muestreo para el primer frame AC-3 en el paquete RTP. Los paquetes que contienen fragmentos del mismo frame deben contener el mismo timestamp. El timestamp del primer paquete RTP debe ser elegido en forma aleatoria, y luego se incrementará linealmente de acuerdo a los números de muestras incluidas en cada frame.

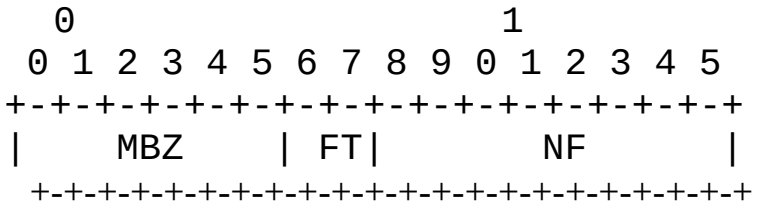
RTP payload format para audio AC-3

De acuerdo a la RFC 2736 [89], los formatos de payload deben contener un número integral de unidades de datos de aplicación (ADU: application data unit). Un ADU será equivalente a un frame AC-3. Cada RTP payload deberá comenzar con el payload header de 2 bytes seguido por un número integral de frames AC-3 completos o un fragmento único de un frame AC-3. Si el tamaño de un frame AC-3 excede el MTU de la red, deberá ser fragmentado para ser transmitido dentro de un paquete RTP.

A)Payload-specific header (header específico de payload)

Hay un header de payload (payload header) de dos octetos en el comienzo de cada payload.

A1)Payload header(header de payload): cada RTP payload de AC-3 debe comenzar con el siguiente payload header

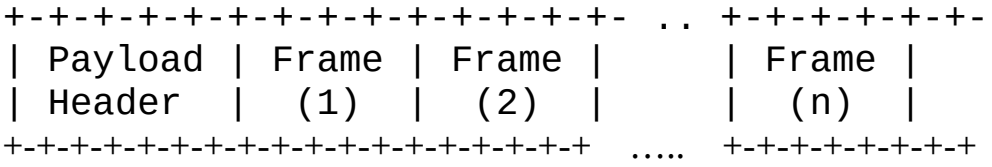


Must Be Zero (MBZ): los bits marcados con MBZ deberán setearse con valor cero y serán ignorados por los receptores; estos bits son reservados para futuras extensiones.

Frame Type (FT): este campo de dos bits indica el/los tipo/s de frame/s presentes en el payload, que en general se refieren a frames completos o fragmentos de frames.

Number of frames/fragments (NF): un campo de 8 bits cuyo significado depende del Frame Type (FT). Indica el número de frames completos AC-3 en el RTP payload o indican el número de fragmentos (y en consecuencia paquetes) que constituyen el actual frame. El valor de NF es igual para los paquetes que contienen fragmentos del mismo frame.

La siguiente figura ilustra el RTP payload completo de AC-3:



B) Fragmentación de los frames AC-3

El tamaño de un paquete AC-3 depende de la velocidad de muestreo del audio y de la velocidad de datos del codificador (indicadas en el “Synchronization Information header del frame AC-3). Dijimos ya que si el tamaño del frame AC-3 excede el tamaño del MTU de la red, se deberá realizar la fragmentación a nivel RTP. La fragmentación puede ser hecha en cualquier byte del frame. Los paquetes RTP que contienen fragmentos del mismo frame AC-3 serán enviados en orden consecutivo desde el

primer hasta el último fragmento. Esto permite que el receptor ensamble los fragmentos en el orden correcto.

Registración del tipo de multimedia

Nombre del tipo: audio

Nombre del subtipo: ac3

Parámetros requeridos: velocidad --- la velocidad del clock del timestamp de RTP es igual a la velocidad de muestreo de audio. Las velocidades permitidas son 32000, 44100 y 48000.

Parámetros opcionales: canales --- desde el lado emisor es el número de canales presentes en el stream AC-3; desde el lado receptor es el máximo número de canales de salida que el receptor entregará. Será un número entre 1 y 6, dado que los canales AC-3 son 5.1.

Consideraciones de codificación: este tipo de multimedia se transmite en frames y contiene datos binarios.

Consideraciones de seguridad: el formato de payload descrito está sujeto a las consideraciones de seguridad de RTP y cualquier RTP profile que se aplique.

Consideraciones de interoperabilidad: ninguna

Aplicaciones que usan este tipo de multimedia: compresión de audio multicanal de audio y audio para video.

Restricciones de uso: este tipo de multimedia depende del framing RTP y de esta manera sólo se define para transferencias vía RTP [40]. El transporte con otros protocolos de framing no está definido.

Uso SDP

La información transportada en el tipo de multimedia MIME tiene un mapeo específico a los campos en el Session Description Protocol (SDP) [90], el cual es usado comúnmente para describir sesiones RTP. Un ejemplo podría ser:

m=audio 49111 RTP/AVP 100
a=rtpmap:100 ac3/48000/6

14.8.Transporte de video

14.8.1.Formato de payload para transporte RTP

Para el caso de transportar video HDTV sin comprimir con RTP se necesita un formato de payload RTP específico, detallado en la RFC 4175 [86], el cual tiene carácter de estándar propuesto desde septiembre de 2005. El citado RFC especifica un esquema de paquetización para encapsular video sin comprimir en un formato de payload para el protocolo de transporte en tiempo real (RTP). Este draft soporta un rango de formatos de video standard y high definition, incluyendo formatos comunes de televisión tales como ITU BT.601 y formatos estándares de la Society of Motion Picture and Television Engineers (SMPTE) tales como SMPTE 274M y SMPTE 296M.

Para el caso del presente trabajo, como ya se mencionó antes, se usará el estándar SMPTE 296M que define un sistema con 720x1280 pixels y escaneo progresivo a 60 fps. Además este formato define la representación digital en los componentes de color YCbCr, donde los componentes Cb y Cr son submuestreados horizontalmente por un factor de 2, o sea con codificación 4:2:2, y 10 bits por muestra. Por lo tanto, en este RFC se especifica un formato de payload para encapsular este formato de video para ser transportado por RTP.

14.8.2.Diseño del payload

Cada línea de escaneo del video digital es paquetizada dentro de uno o más paquetes RTP. Si los datos para un completo escaneo de las líneas exceden el MTU de la red, la línea escaneada deberá ser fragmentada en múltiples paquetes RTP, cada uno de los cuales más pequeño que el MTU. **Un paquete RTP puede contener datos de más de una línea de escaneo.** Sólo las muestras activas son incluidas en el RTP payload; las muestras inactivas y los contenidos del blanking horizontal y vertical no se deben transportar. Los números de líneas de escaneo son incluidos en el RTP payload header (junto con un identificador de campo para el caso de video entrelazado). Para el caso del formato de video SMPTE 296M, el rango del número de líneas escaneadas válidas va del 26 al 745.

El payload header contiene una extensión de 16 bits al número de secuencia estándar de RTP, extendiendo así el número de secuencia a 32 bits y habilitando el formato de payload para altas velocidades de datos sin ambigüedad. Esto es necesario dado que el número de secuencia de 16 bits de RTP se agota de inmediato para altas velocidades de datos, y

produciendo un roll-over o sea, vuelve a comenzar. Por ejemplo, para el caso de un video stream de 1 Gbps con tamaños de paquetes de al menos 1000 Bytes, se comprobó que el número de secuencia para paquetes de RTP hará un roll-over a los 0.5 seg., y esto puede traer consecuencias negativas al detectar pérdidas y paquetes fuera de orden, en particular cuando el RTT es mayor que 0.5 seg. El número de secuencia extendido de 32 bits permitiría en este caso llegar a las 9 horas sin que se produzca el roll-over de los números de secuencia d elos paquetes RTP.

Cada línea de escaneo comprende un número entero de pixels. Cada píxel está representado por un número de muestras, las cuales serán codificadas en 8 bits (dado que la profundidad de color fue previamente downsampleada de 10 a 8 bits). Una muestra puede representar un componente de color o un componente de luminancia del video. Las muestras de color pueden ser compartidas entre pixels adyacentes, y esto es conocido como submuestreo (subsampling), y el propósito es reducir el tamaño de los datos de la imagen.

Los pixels que comparten valores de muestras deberán ser transportados en forma conjunta como un “píxel group”. Como en este caso se usan 10 bits para las muestras, cada píxel puede comprender un número no entero de octeto. En este caso, se deberán combinar varios pixels en un grupo de pixels alineados por octetos para la transmisión. Estas restricciones simplifican la operación de los receptores dado que se asegura que un payload completo esté alineado en octetos, y que las muestras relativas a un único píxel no se fragmenten a lo largo de múltiple paquetes.

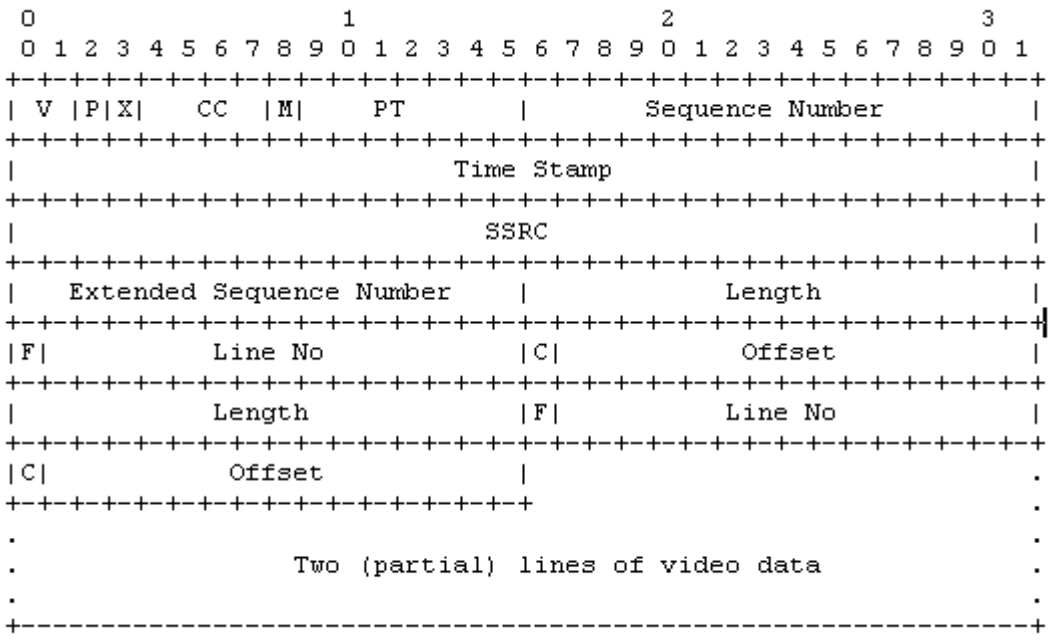
El parámetro “pgroup” es el tamaño en octetos del grupo más pequeño de pixels de manera que: (1) el grupo comprenda un número entero de octetos; y (2) si se usa submuestreo de color entonces las muestras sólo se comparten dentro del grupo. Cuando se transportan líneas activas de video digital, los datos de video no deberán ser fragmentados dentro de un pgroup.

El contenido de video casi siempre está asociado con información adicional como pistas de audio, etc. En las aplicaciones profesionales de video digital, estos datos comúnmente están embebidos en las porciones no activas del video stream (períodos de blanking horizontal y vertical) y por lo tanto se precisa una precisa sincronización. El formato de payload RTP para video sin comprimir requiere que las aplicaciones que usan tales datos auxiliares sincronizados deberán entregarlos en sesiones RTP separadas que operan en forma concurrente con la sesión de video, y así entonces se usará el mecanismo normal de RTP para sincronizar la multimedia.

14.8.3.Paquetización RTP

El RTP header estándar es seguido por 2 octetos de payload header que extienden el número de secuencia RTP, y por 6 octetos de paylaod header para cada línea (o línea parcial) de video incluida. Luego siguen una o más líneas, o líneas parciales, de datos de video. Este formato hace que el payload header de 32 bits esté alineado en un caso común, donde una línea de escaneo (o fragmento) de video es incluida en cada paquete RTP.

Por ejemplo, si se encapsulan dos líneas de video, el formato de payload será de esta manera:



RTP Payload Format con dos líneas de video (parciales)

14.8.3.1.RTP Header

Los campos del RTP header fijo tiene su significado habitual, con los siguientes agregados:

Payload Type (PT): 7 bits – Es un campo de tipo de payload colocado en forma dinámica que define al payload como “video sin comprimir”.

Timestamp: 32 bits – Para el escaneo de video progresivo, el timestamp denota el instante de muestreo del frame al cual pertenece el paquete RTP. Los paquetes no deben incluir datos de múltiples frames, y todos los paquetes pertenecientes al mismo frame deben tener el mismo timestamp.

Se deberá usar un timestamp con reloj de 90 KHz, y si el instante de muestreo no corresponde con un valor entero del reloj (como puede ocurrir en el caso de interleaving) el valor deberá truncarse al siguiente entero más bajo, sin ambigüedad.

Marker bit (M): 1 bit – Si se transmite el video escaneado en modo progresivo, el Marker bit denota el fin de un frame de video. Este bit deberá ser establecido en 1 para el último paquete del frame de video y 0 para otros paquetes.

Sequence Number: 16 bits – Ese incrementan los 16 bits del campo estándar de RTP en 16 bits más en el payload header para evitar que los números de secuencia no sufran un roll-over.

14.8.3.2.Payload Header

A continuación se detallan los campos del payload header:

Extended Sequence Number: 16 bits – Son los bits de alto orden del campo de números de secuencia extendido de 32 bits, con el orden de bytes que llegan desde la red.

Length: 16 bits – Es el número de octetos de datos incluidos por la presente línea escaneada, con el orden de bytes que llegan desde la red.

Line Number: 15 bits – Es el número de línea escaneada de los datos encapsulados, ordenados por bytes con el orden que llegan desde la red. Los paquetes RTP sucesivos pueden contener parte de la misma línea escaneada (con el número de secuencia RTP incrementado, pero con el mismo timestamp) en el caso que sea necesario fragmentar una línea.

Offset: 15 bits – Es el offset del primer píxel de los datos del payload dentro de la línea escaneada. En el presente caso que se transporta datos de vidoe con formato YCbCr, este valor es el offset del píxel de la muestra de luminancia. El valor se brinda con el orden de bytes que llegan desde la red. El offset tiene un valor de cero si la primera muestra en el payload corresponde al comienzo de la línea, y luego se incrementa por uno para cada píxel.

*Field Identification (F):*1 bit – Identifica el campo al que pertenece la línea escaneada, para el escaneo entrelazado. En el caso del presente trabajo, como usamos escaneo progresivo, F siempre está establecido en 0.

Continuation (C): 1 bit – Determina si un header adicinal de una línea escaneada sigue al actual header de la línea escaneada en el paquete RTP. Se establece en 1 si sigue un header adicional, implicando así que el paquete RTP está transportando datos para más de una línea escaneada. En le caso contrario, se establece en 0. En un único paquete se pueden llegar a incluir diferentes líneas escaneadas, hasta llegar al límite del MTU del camino de red. La única manera de determinar la cantidad de líneas escaneadas incluidas por paquetes es analizando el payload header.

14.8.3.3.Datos del Payload

Dependiendo del formato de video, cada paquete RTP puede incluir ya sea una sola línea de escaneo completa, un solo fragmento de una línea de escaneo, o una o más líneas de escaneo completas y fragmentos de líneas de escaneo. La longitud de cada línea de escaneo o fragmento de línea de escaneo debe ser un múltiplo entero del tamaño en octetos del pgroup. Las líneas de escaneo deberán fragmentarse para que el paquete RTP resultante sea de tamaño menor o igual al MTU de la red.

Es posible que la longitud de la línea de escaneo no sea divisible por el número de píxeles en un pgroup, y así el dato de píxel final de una línea de escaneo no se alinea ni con un octeto ni con el límite del pgroup. Sin embargo el payload debe contener unnúmero entero de de pgroups; el emisor debe llenar los bits restantes del final del pgroup con 0 y el receptor deberá ignorar los datos así completados.

En el caso de vido con formato YCbCr, el empaquetado de las muestras dentro del payload depende del submuestreo de color usado. Para el formato de video YCbCr 4:2:2, los componentes Cb y Cr están horizontalmente submuestreados por un factor de 2 (cada muestra Cb y Cr corresponde a dos componentes Y). Las muestras son empaquetadas en el orden Cb0-Y0-Cr0-Y1, tanto para el escaneo de líneas entrelazado como progresivo. Para el caso como el presente de muestras con 8 bits (luego de haber sufrido un downsampling desde los 10 bits iniciales) el pgroup se forma con dos pixels adyacentes (4 octetos).

14.8.4.Consideraciones con respecto a RTCP

RTCP se deberá usar según lo especificado en la RFC 3550. Se hace notar que el conteo de octetos del emisor en los paquetes SR y el número acumulado de paquetes perdidos se agotan rápidamente para los streams de alta velocidad. Esto significa que que estos dos campos puede que no representen de manera exacta el conteo de octetos y el número de paquetes

perdidos desde el comienzo de la transmisión, como se define en la RFC 3550.

14.8.5.Consideraciones IANA

Se requiere que el IANA registre un nuevo subtipo MIME.

Nombre del tipo de video MIME: video

Nombre del subtipo MIME: raw

Parámetros requeridos:

Rate: es la velocidad del clock del timestamp RTP. Las aplicaciones que usan este formato de payload deberán usar un valor de 9000.

Muestreo: determina el modo de muestreo (o submuestreo) de colordel video stream. Los valores definidos actualmente son RGB, RGBA, BGR, BGRA, YCbCr-4:4:4, YCbCr-4:2:2, YCbCr-4:2:0 y YCbCr-4:1:1. También se podrán registrar nuevos valores.

Ancho: determina el número de pixels por línea. Este número es un entero entre 1 y 32767.

Alto: determina el número de líneas por frames. Este es un entero entre 1 y 32767.

Profundidad: determina el número de bits por muestra. Este es un entero con valores típicos que incluyen 8, 10, 12 y 16.

Colorimetría: este parámetro define un conjunto de especificaciones de colorimetría y otras características de transferencia para el video fuente. Lops valores válidos y sus especificaciones son: BT601-5, BT709-2 y SMPTE-240M. Además se pueden llegar a registrar nuevos valolres en el futuro.

Consideraciones de codificación: el video sin comprimir puede ser transmitido con RTP.

Aplicaciones que usan este tipo de multimedia: comunicación de video.

14.8.6.Mapeo de parámetros MIME dentro de SDP

La información transportada por la especificación MIME tiene un mapeo específico hacia los campos en el Protocolo de Decripción de sesión (SDP), el cual es comúnmente usado para describir sesiones RTP. Cuando SDP es usado para especificar sesiones que transportan vidoe sin comprimir, el mapeo es el siguiente:

- El tipo MIME (“video”) va en el campo “m=” de SDP como un nombre de medio.
- El subtipo MIME (nombre de formato de payload) va en el campo “a=rtpmap” de SDP como el nombre de codificación.
- Los restantes parámetros van en el campo “a=fmtp” de SDP.

14.9.Esquema de control de congestión

14.9.1.Introducción

Dada la proliferación de las redes de alta velocidad y aplicaciones multimedia, cada día se está tornando más importante la consideración de esquemas de control de congestión. Esto es especialmente crítico para aplicaciones con requerimientos exigentes de ancho de banda, debido al potencial riesgo de interrupción del tráfico de red existente, tal el caso de la transmisión de HDTV sin comprimir.

La mayoría de los tests realizados con transmisiones de HDTV de alta velocidad, tuvieron una severa limitación: debido a la falta de esquemas de control de congestión y QoS, estos tests sólo pudieron haberse realizado con el permiso y el cuidadoso monitoreo del staff de operaciones de las redes sobre las cuales se desarrollaron para asegurar que este tráfico no afecte al resto del tráfico transportado, como fue el caso del experimento de Tektronix.

El objetivo del control de congestión es evitar el colapso de la congestión, que sería la situación por la cual unincremento de la carga de la red resulta en undecremento en el número de paquetes entregados en forma eficiente por la red, que puede ocurrir por una congestión de los nodos intermedios del camino de red.

El colapso de la congestión no es un problema teórico. En los comienzos de Internet allá por mediados de los años 80’s, TCP no tenía desarrollado un esquema de congestión y se produjeron colapsos. Así el protocolo TCP fue

mejorado por el mecanismo de control de congestión desarrollado por Van Jacobson [91].

Además del problema del colapso de la congestión, existe otra razón para el uso de un mecanismo de control de congetión y es la equidad o justicia en el consumo del ancho de banda. El modelo adoptado por TCP es el de compartir la capacidad disponible aproximadamente en forma uniforme entre todos los flujos. Por lo tanto si los flujos multimedia usan esquemas de control de congestión con un algoritmo diferente, no habrá equidad con los flujos TCP que comparten el mismo canal de comunicaciones. Por lo tanto, para que el tráfico multimedia y los flujos TCP/IP puedan coexistir y compartir en forma justa el ancho de banda disponible, el tráfico no-TCP deberá ser “amigable” o “equitativo”con TCP.

14.9.2.Control de congestión en flujos TCP

En principio TCP tiene un amplio conjunto de mecanismos para el control de congestión [38] [91] [92].

Comencemos por decir que TCP usa el protocolo de ventana deslizante, mediante el cual el origen de la transmisión incluye números de secuencias en los paquetes y así estos son “echoed” mediante paquetes ACK transmitidos por el receptor. Si se usa una ventana en lugar de un paquete se pueden transmitir más de un paquete antes que un ACK sea recibido. Cada paquete TCP ACK incluye una ventana de recepción por medio de la cual le indica al origen cuántos octetos pueden ser aceptados en cualquier momento, además de mandarle el número de secuencia continuo más alto de los paquetes de datos recibidos. Entonces el origen envía la suficiente cantidad de paquetes para llenar la ventana de recepción, antes de recibir un ACK. A medida que legan los ACK’s, la ventana de recepción se desliza y así se envían más datos.

Además de una ventana de recepción, los emisores TCP incorporan una ventana de congestión dimensionada de acuerdo a una estimación de la capacidad de la red y de esta forma previene a los emisores de saturar la red. La ventana de congestión comienza con el envío de un paquete y se incrementa de acuerdo ya sea a un algoritmo de “slow-start” (comienzo lento) o a un algoritmo de “congestion avoidance” (evitación de la congestión) mientras que no haya pérdidas de paquetes.

En el modo “Slow-start”, la ventana de congestión se incrementa por el tamaño de un paquete con cada ACK recibido. Así el emisor alcanza en

forma gradual la velocidad completa que le permite la red, ilustrado de la siguiente manera:

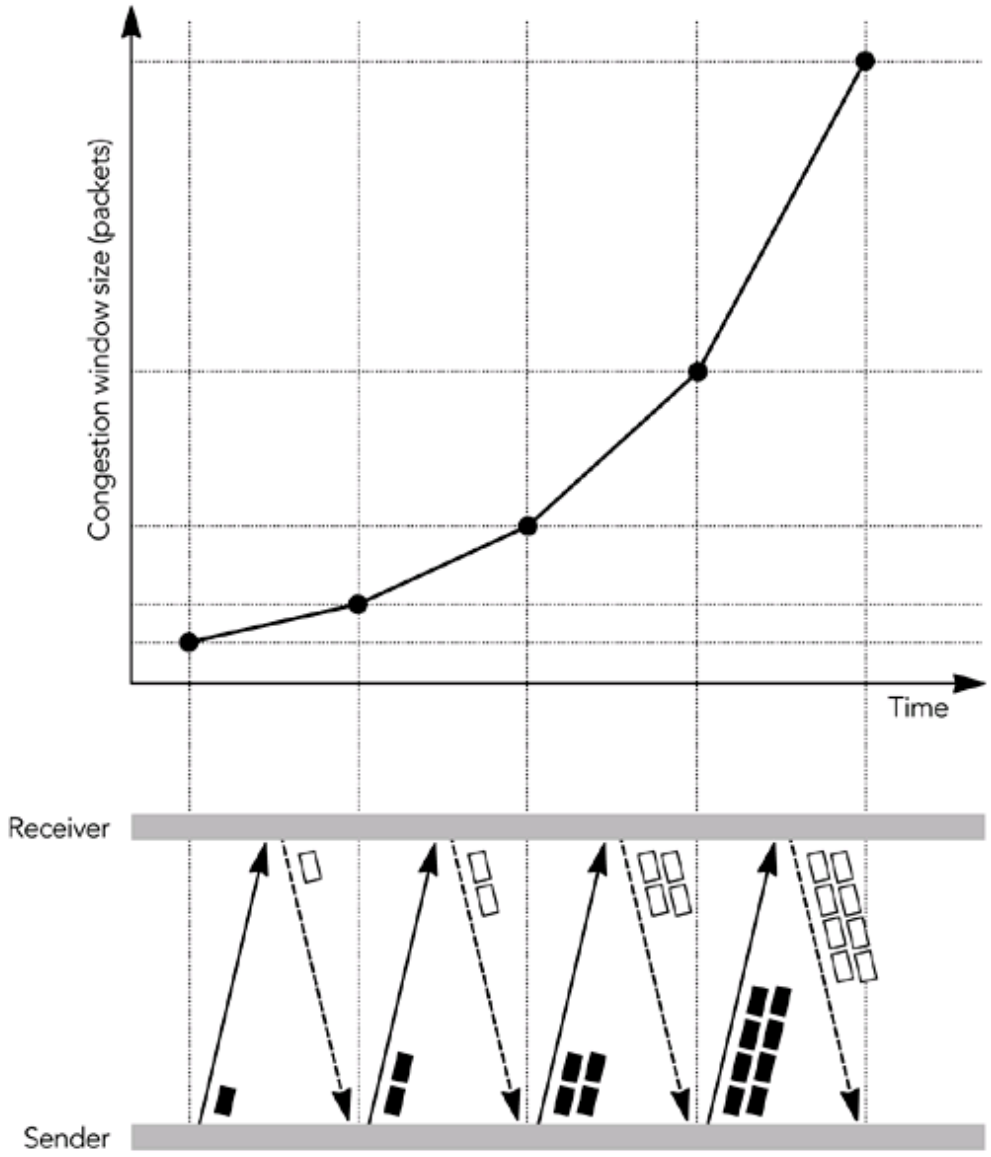


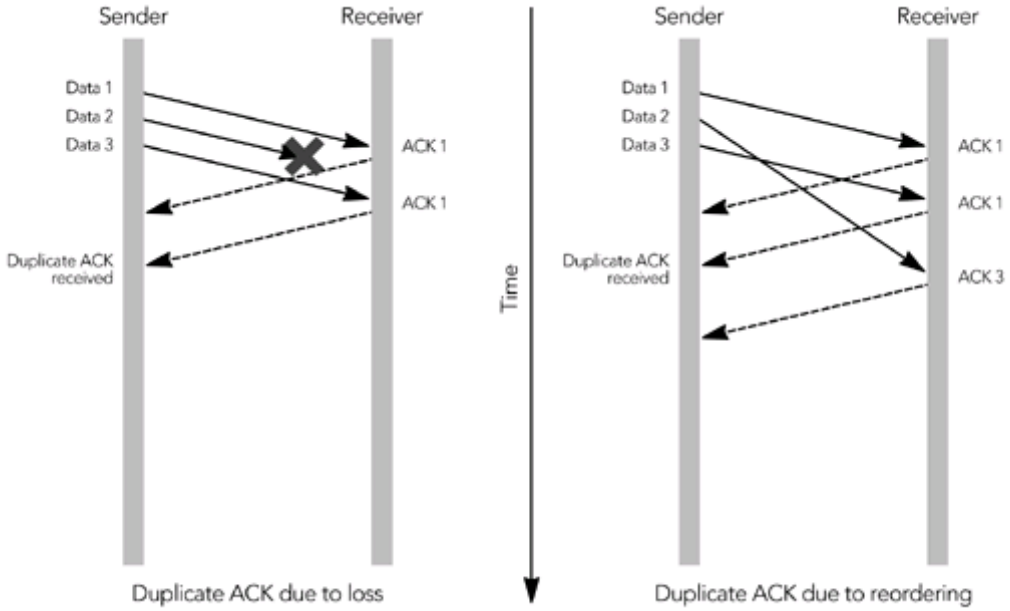
Diagrama del modo “Slow-Start” de TCP

La fase de slow-start continúa hasta que se pierde un paquete, o hasta que se excede el umbral de slow-start.

En el modo de “congestion avoidance”, la ventana de congestión se incrementa por el tamaño de un paquete cada RTT, en lugar de incrementar un paquete por cada ACK recibido. El resultado es un crecimiento lineal de la ventana de congestión, un crecimiento aditivo en la velocidad de envío.

Las conexiones TCP incrementan su velocidad de envío de acuerdo a uno de los dos algoritmos descriptos recién, hasta que ocurre una pérdida de paquete. La pérdida de un paquete indica que se alcanzó la capacidad de la red y ha habido una congestión momentánea. El emisor puede detectar congestión de dos maneras: por un timeout o por medio de la recepción de un triple ACK duplicado. Cuando ocurre un timeout, el emisor establece el umbral de slow-start a la mitad del número de paquetes actuales o dos paquetes, depende de cuál valor es mayor. Entonces establece la ventana de congestión al tamaño de un paquete y entra en show-start. El proceso de slow-start continúa hasta que el umbral de slow-start sea excedido, que es cuando el emisor ingresa al modo de evitación de la congestión. Un timeout prolongado causará que el emisor desista y se desconecte.

La otra manera que el emisor puede detectar una congestión es por la presencia de paquetes ACK duplicados, lo que puede suceder porque un paquete se pierde o es reordenado como se detalla en la siguiente figura:



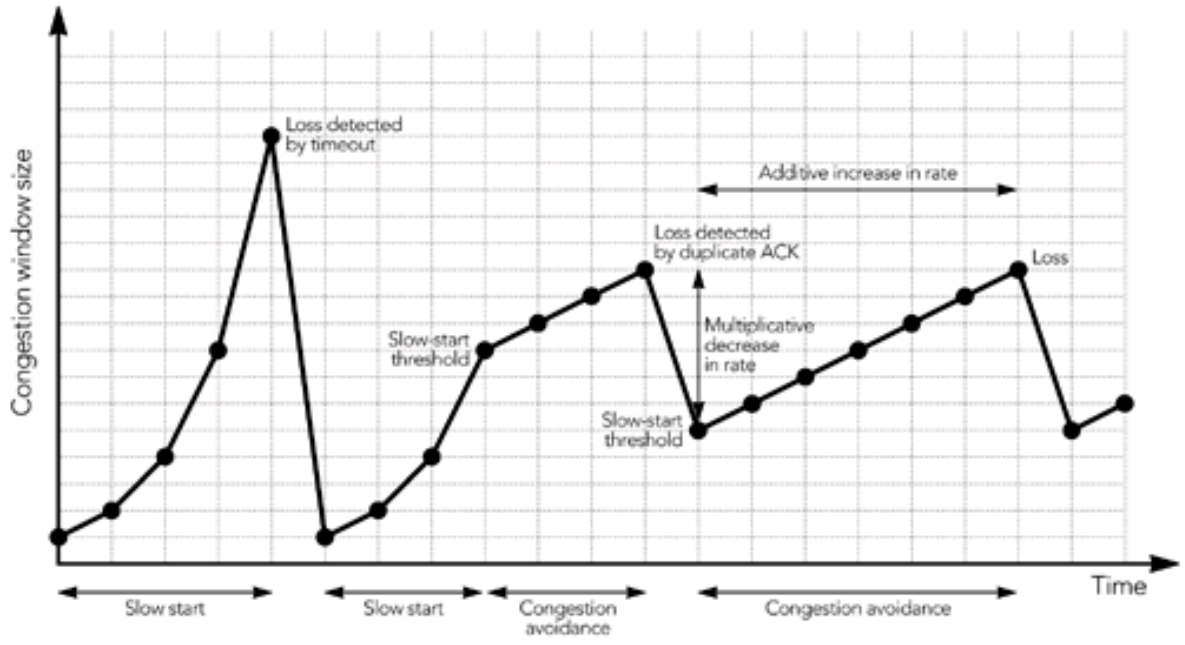
Generación de paquetes ACK duplicados

Los paquetes ACK, como ya se mencionó, contienen el número de secuencia continuo más alto recibido, y por lo tanto si un paquete de datos se pierde los siguientes paquetes ACK contendrán el número de secuencia antes de la pérdida hasta que el paquete perdido sea retransmitido. Si un paquete es reordenado también se generará un ACK duplicado, pero en este

caso la secuencia de ACK's vuelve a su normalidad cuando finalmente arriba ekl paquete reordenado.

Si el emisor recibe tres paquetes ACK duplicados, asume que el paquete se perdió a causa de la congestión. Entonces el emisor establece su ventana de congestión y el umbral de slow-start a la mitad de los paquetes actuales o a dos paquetes, según sea el mayor valor. Luego retransmite el paquete perdido e ingresa en el modo de "congestion avoidance".

La combinación de estos algoritmos da como resultado un throughput TCP con la siguiente característica:



Variación en la velocidad de un flujo TCP a lo largo del tiempo

La característica de esta curva de velocidad en el tiempo tiene como patrón un **incremento aditivo ("additive-increase") y un decremento multiplicativo ("multiplicative-decrease"), con una gran variación en el throughput en cortos períodos de tiempo**.

La rápida variación del throughput es lo que hace que un sistema que usa TCP sea estable. La característica de **"multiplicative-decrease"** asegura una respuesta rápida ante la congestión, previniendo así el colapso. La

característica de **“additive-increase”** -que prueba el máximo throughput posible- asegura que la capacidad de la red esté utilizada a pleno.

Existen diferentes estudios acerca de la variación del throughput TCP y de los demás flujos TCP que compiten a la vez [93] [94] [95] [96]. Estos estudios demuestran que los flujos que compiten a la vez obtienen en promedio aproximadamente iguales “pedazos” de la capacidad, a pesar que el perfil del throughput muestre que sus “pedazos” compartidos de ancho de banda probablemente no sean equitativos.

En TCP hay una inequidad sistemática: dado que el algoritmo responde al feedback, las conexiones con menor tiempo de RTT responden con sus feedbacks más rápido y así trabajan mejor. De esta manera, las conexiones con RTT más largos reciben sistemáticamente en promedio un “pedazo” de la capacidad de la red menor.

14.9.3.Características del tráfico multimedia

Es deseable que el tráfico multimedia coexista en forma armoniosa con otros tipos de tráfico en la red. Para ello el tráfico multimedia deberá emplear un algoritmo de control de congestión que sea equitativo con TCP.

Como ya se mencionó antes, TCP adapta su congestión de red en cortos períodos de tiempo y brinda “pedazos” compartidos de ancho de banda mayores a las conexiones que tienen un RTT corto. Y cuanto más estudios se realizan, más claro está que TCP no es del todo equitativo. Si se considera un breve período de tiempo, siempre un flujo TCP ganará sobre otro en cuanto a la adquisición de ancho de banda. Y en el largo plazo, el resultado sí es más justo. Las diferentes variante de TCP también tiene su efecto, por ejemplo: SACK-TCP toma un mejor throughput con ciertos patrones de pérdida.

Para el tráfico multimedia es deseable que su comportamiento en cuanto a la congestión sea equitativo con los flujos TCP, a pesar que TCP no es equitativo con él mismo exactamente.

Un ejemplo claro sería el uso de un browser de web para escuchar un streaming de audio. Algunos uusarios preferirán que el audio sea más agresivo que las aplicaciones como e-mail o web, pero otros usuarios dirán que necesitan que su aplicación web sea más agresivas que el tráfico de audio porque su uso es para transacciones online.

Se puede decir que si una aplicación necesita una prioridad mayor que la normal, este requerimiento deberá ser hecho a la red enn lugar de ser hecho a los protocolos de transporte; sería el caso del uso de protocolos como RSVP y DiffServ que proveen calidad de servicio. El soporte a estos servicios está muy limitado en la actualidad, estando disponibles en algunas redes privadas pero en en Internet. Las aplicaciones que intentan correr sobre Internet entonces deberían considerar alguna forma de control de congestión amigable con los flujos TCP.

En la actualidad no existe ningún estándar de control de congestión para el tráfico multimedia en Internet. El IETF está trabajando para definir un estándar para el control de la velocidad amigable con TCP (TCP-Friendly Rate Control) que es apto para aplicaciones unicast.

14.9.4.Descripción de TFRC

El principal inconveniente que tiene el algoritmo de control de congestión de TCP para ser usado por streams de audio y video es que presenta grandes cambios de velocidad en breves períodos de tiempo. En general, los CODECs de audio operan a una velocidad fija o pueden adaptar su velocidad entre un conjunto de velocidades fijas, y los CODECs de video tienen una mayor capacidad de adaptación porque pueden variar la tasa de frames y la relación de compresión. Los estudios han demostrado que los usuarios finales prefieren una calidad a velocidad estable, inclusive si el stream de calidad variable tiene una calidad promedio mayor.

Varios algoritmos de control de velocidad amigables con TCP [97] [98] se percataron que si se suaviza la variación de la velocidad de envío en el corto plazo, se obtiene un algoritmo acorde para las aplicaciones de audio y video. Estos algoritmos alcanzan equidad con los flujos TCP cuando son promediados sobre intervalos de tiempo de varios segundos pero no son equitativos en breves períodos de tiempo. Son potencialmente aptos para ser usados con aplicaciones de audio y video unicast, y como ya se mencionó antes, la IETF está trabajando para definir el estándar.

En el presente trabajo voy a hacer uso del mecanismo de control de congestión denominado TCP Friendy Rate Control (TFRC) especificado en la RFC 3448 [99] con categoría de “estándar propuesto”, basado en una ecuación por la que se obtiene el throughput máximo, y se aplica en flujos unicast que operan en ambientes Internet con “best-effort”. Vale decir que es un mecanismo razonablemente justo cuando compite por ancho de banda con otros flujos TCP (se considera “razonablemente justo” cuando su velocidad de transmisión está dentro de un factor del doble de la velocidad

de transmisión de un flujo TCP bajo las mismas condiciones) y tiene una mucho menor variación del throughput con el tiempo comparado con TCP siendo de esta manera más acorde para aplicaciones como telefonía o streaming multimedia donde es importante una velocidad de envío de datos relativamente constante.

Por otro lado, la penalidad de tener un throughput más parejo que TCP mientras compite en forma justa por ancho de banda es que TFRC responde de manera más lenta que TCP a los cambios en el ancho de banda disponible. De esta manera TFRC sólo debería ser usado cuando la aplicación tiene un requerimiento de throughput parejo, en particular evitando la característica propia de TCP que disminuye su throughput a la mitad en respuesta a la pérdida de un paquete. Para las aplicaciones que sólo necesitan transmitir tantos datos como sea posible en el tiempo más corto que sea posible se recomienda el uso de TCP, y si la confiabilidad no es un requerimiento se puede llegar a usar un esquema de control de congestión AIMD (Additive-Increase, Multiplicative-Decrease) con parámetros similares a los usados por TCP.

TFRC está diseñado para aplicaciones que usan un tamaño fijo de paquete, y varían su velocidad de envío en paquetes por segundo en respuesta a la congestión. Como comentario, existe el mecanismo denominado TFRC-PS (TFRC-PacketSize) que es una variante de TFRC para aplicaciones que tienen una velocidad de envío fija pero que varían su tamaño de paquete en respuesta a la congestión, como es el caso de algunas aplicaciones de audio.

TFRC es un mecanismo basado en el receptor, con el cálculo de la información de control de congestión (por ejemplo, la tasa de pérdidas) basada en los datos del receptor y no en los datos del transmisor.

Es importante aclarar **que TFRC no tiene conocimientos de las necesidades de la aplicación que hace uso de su mecanismo, se puede decir que sólo conoce las condiciones de la red.** Una aplicación puede adaptar su velocidad al límite establecido por TFRC –lo cual implicará cierta pérdida de la calidad- o parar de enviar los datos. Y además vale la pena repetir que el objetivo de TFRC es compartir de manera equitativa la capacidad de la red., y no está diseñada de ninguna manera para que algunas aplicaciones tomen más ancho de banda que otras.

Por último, se debe decir que hubo una experiencia del Information Sciences Institute from the USC [100] que ha experimentado con TFRC con datos “mudos” (dummy data) que simularon los datos de video y que ha

tenido gran reconocimiento en la comunidad de investigadores internacionales del área de audio y video.

14.9.5.Implementación del control de congestión

14.9.5.1.Mecanismo del protocolo

Para implementar el mecanismo de control de congestión, TFRC usa directamente una ecuación del throughput para la velocidad de envío permitida como función de la tasa de eventos de pérdidas y el RTT (Round Trip Time). A los fines de competir en forma justa con los flujos TCP, TFRC usa la ecuación de throughput de TCP que describe la velocidad de envío de TCP como función de la tasa de eventos de pérdidas, el RTT y el tamaño de paquete. Se define como un **evento de pérdida** a uno o más paquetes perdidos o marcados de una ventana de datos, considerando dicha ventana como un RTT, y donde un paquete marcado se refiere a una notificación de congestión de ECN (Explicit Congestion Notification, es un mecanismo para informar a los routers que puede llegar a ocurrir congestión, y así si un router recibe un paquete marcado con ECN deberá tomar acciones al respecto como droppear los paquetes pertenecientes a los tráficos de baja prioridad).

De manera general, el mecanismo de control de congestión de TFRC actúa de esta manera:

- El receptor mide la tasa de eventos de pérdidas y envía esa información al emisor
- El emisor también usa este mensaje de feedback para medir el RTT
- La tasa de eventos de pérdidas y el RTT alimentan entonces la ecuación del throughput de TFRC, dando como resultado una aceptable velocidad de transmisión.
- El emisor entonces ajusta su velocidad de transmisión para acomodarla a la velocidad calculada por medio de la citada ecuación.

Vale destacar entonces que la dinámica de TFRC es sensible a cómo se toman y se aplican las mediciones de los parámetros arriba descritos.

14.9.5.2.Ecuación de throughput de TCP

Cualquier ecuación realista que exprese el throughput TCP como función de la tasa de eventos de pérdidas y el RTT debería ser acorde para ser usada en TFRC. Sin embargo se hace notar que la ecuación del thoughput de TCP deberá reflejar el comportamiento del timeout de retransmisiones de TCP, dado que ello es lo que domina el throughput TCP en las mayores tasas de pérdidas. Se hace notar que la asunción implícita en la ecuación del throughput acerca del parámetro de la tasa de eventos de pérdidas debe estar de acuerdo en forma razonable a cómo se mide actualmente la tasa de pérdidas o tasa de eventos de pérdidas. Dado que esta correspondencia no es perfecta para la ecuación del throughput y los mecanismos de medición de la tasa de pérdidas detallados más abajo, en la práctica las asunciones están muy cercanas y se consideran eficaces.

TFRC se basa en la emulación de la función de respuesta en estado de equilibrio para TCP, derivada por Padhye y sus colaboradores [101]. La función de respuesta es un modelo matemático para el throughput de una conexión TCP, una predicción del throughput promedio dados la tasa de pérdida y el RTT de la red. La derivación de la función de respuesta es algo compleja pero Padhye demostró que el throughput promedio de una conexión TCP (X) bajo condiciones de equilibrio puede ser modelado de esta manera:

$$X = \frac{s}{R\sqrt{2bp/3} + t_{RTO}\left(3\sqrt{3bp/8}\, p\,(1+32p^2)\right)}$$

Donde:

X es la velocidad de transmisión en bytes/segundos

s es el tamaño de paquete en bytes

R es el RTT (Round Trip Time) en segundos

p es la tasa de eventos de pérdidas, entre 0 y 1, del número de eventos de pérdidas dado como una fracción del número de paquetes transmitidos (no es lo mismo que la fracción de paquetes perdidos)

t_RTO es el valor del timeout de las retransmisiones TCP en segundos

b es el número de paquetes confirmados (acknowledged) por un único acknowlegement de TCP
Luego, esta ecuación se ve simplificada al hacer que **t_RTO = 4R**, quedando de esta manera:

$$\mathbf{X} = \frac{S}{R\sqrt{\frac{2p}{3}} + t_{RTO}(3\sqrt{\frac{3p}{8}})p(1+32p^2)}$$

Un cálculo de un valor más exacto de t_RTO es posible, pero los experimentos han demostrado que usando esta simplificación se obtiene un resultado razonablemente justo con respecto a las implementaciones TCP existentes. Otra posibilidad podría ser establecer el valor de t_RTO como el max(4R,1seg) para estar de acuerdo con el valor mínimo recomendado de 1 seg en el RTO. Pero en definitiva, usando la fórmula anterior es suficiente para que el emisor pueda regular el throughput.

Muchas de las actuales conexiones TCP usan acknowledgements retardados, enviando un acknowledgement por cada dos paquetes de datos recibidos, y así tener una velocidad de envío modelada por b=2. Sin embargo, TCP también tiene permitido enviar un acknowledgement para todos los paquetes de datos, y así sería b=1. A causa de que muchas implementaciones TCP no usan acknowledgement retardados, se recomienda usar b=1.

En el futuro, diferentes ecuaciones TCP pueden ser sustituidas por esta ecuación. El requerimiento es que la ecuación de throughput sea una aproximación razonable de la velocidad de envío de TCP para el control de congestión TCP.

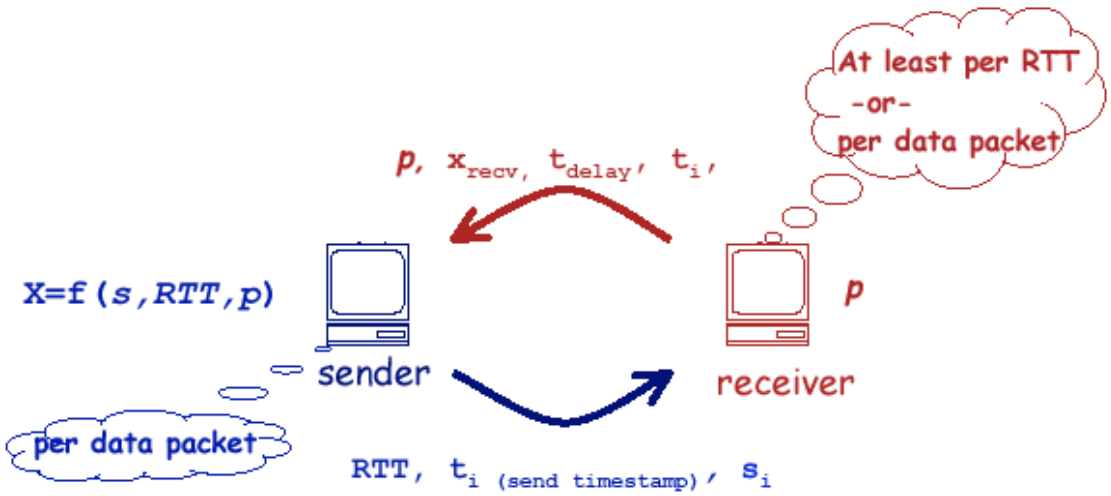
Los parámetros s (tamaño de paquete), p (tasa de eventos de pérdidas) y R (RTT) necesitan ser medidos o calculados por una implementación TFRC, y se especifican más adelante. Todas las velocidades de datos aquí detalladas se especifican en bytes/segundos.

Se puede decir que aunque esta ecuación parece compleja, los parámetros son relativamente fáciles de medir. **Una aplicación que usa RTP conoce**

el tamaño de paquete usado para transmitir, el RTT se puede obtener de la información contenida en los paquetes RTCP SR y RR, y una aproximación de la tasa de eventos de pérdida es reportada en los paquetes RTCP RR. Y como ya se mencionó arriba, el tiemout de retransmisión TCP se lo aproxima satisfactoriamente al cuádruple del RTT.

Una vez que estos parámetros han sido medidos, el emisor puede calcular el throughput promedio que tiene una conexión TCP sobre un camino de red similar en un estado de equilibrio (el throughput promediado sobre varios segundos, asumiendo que la tasa de pérdidas es constante). Estos datos pueden entonces ser usados como parte de un esquema de control de congestión. Si la aplicación está transmitiendo a una velocidad mayor que la calculada para TCP, entonces debería reducir su tasa de trsnasmisión para corresponderse con el valor calculado o de lo contrario corre riego de congestionar la red. Si la aplicación está transmitiendo a una velocidad menor que la calculada para TCP, entonces debería inctrementar su velocidad para corresponderse con la calculada. **La aplicación opera con información obtenida con feedbacks: cambia la tasa de transmisión, mide la tasa de eventos de pérdida, cambia la tasa de transmisión para corresponderse, mide la tasa de eventos de pérdida y así sucesivamente.** Las aplicaciones que usan RTP se pueden beneficiar de los reportes dprovenientes de los paquetes RTCP RR; estos reportes causan que la aplicación reevalúe y cambie su tasa de transmisión.

A continuación se representa el funcionamiento de TFRC entre emisor y receptor:



14.9.5.3.Contenidos de los paquetes

Como TFRC se usa sobre un protocolo de transporte, los formatos de los paquetes dependen del protocolo de transporte utilizado, en mi caso RTP.

14.9.5.3.1.Paquetes de datos

Cada paquete de datos enviado por el emisor de datos contiene la siguiente información:

- Un número de secuencia, que se incrementa por uno por cada paquete de datos transmitido. El campo debe ser lo suficientemente grande para que no haya dos diferentes paquetes con el mismo número de secuencia en el historial de paquetes recibidos del receptor al mismo tiempo.

- Un timestamp que indica el instante en que el paquete es enviado. Se denota como ts_i al timestamp del paquete con secuencia i. Generalmente se mide en mseg. Este timestamp es usado por el receptor para determinar cuáles pérdidas pertenecen al mismo evento de pérdidas. El receptor hace “echo” del timestamp para habilitar que el emisor estime el RTT, para aquellos emisores que no guardan los timestamps de los paquetes de datos transmitidos. En el presente caso los emisores guardan los timestamps de los paquetes que mandaron, por lo cua no hace falta que el receptor los envie de vuelta al emisor.

- La estimación actual del RTT llevada a cabo por el emisor. La estimación reportada en el paquete i se denota como R_i. El RTT estimado es usado por el receptor, a lo largo del timestamp, para determinar cuándo las diferentes pérdidas pertenecen a un mismo evento de pérdida.

14.9.5.3.2.Paquetes de feedback

Cada paquete de feedback enviado por el receptor contiene la siguiente información:

- El timestamp del último paquete recibido, denotado por t_recvdata. Si el último paquete recibido en el receptor tiene un número de secuencia i, entonces t_recvdata = ts_i. Este timestamp es usado por el emisor para estimar el RTT, y sólo se necesita si el emisor no guarda los timestamps de los paquetes transmitidos. En este caso no

haría falta porque, como ya lo mencioné, el emisor guarda los timestamps de los paquetes marcados.

- La cantidad de tiempo transcurrido entre la recepción del último paquete de datos en el receptor y la generación de este reporte de feedback. Se denota con t_delay.

- La velocidad a la cual el receptor estima que los datos fueron recibidos desde que el último reporte de feedback fue enviado. Se denota con X_recv.

- La actual estimación de la tasa de evento de pérdidas (p) por parte del receptor.

14.9.6.Protocolo de envío de datos del lado del emisor

El emisor de datos envía un stream de paquetes de datos al receptor a una velocidad controlada. Cuando el emisor recibe un paquete de feedback desde el receptor, el emisor cambia su velocidad de envío, basado en la información contenida en el reporte de feedback. Si el emisor no recibe un reporte de feedback en un tiempo igual a 4 RTT, baja la velocidad de envío a la mitad. Esto se logra por medio de un timer llamado “nofeedback timer”.

Se especifica el protocolo del lado del emisor por medio de los siguientes pasos:

- Medición del tamaño medio de paquete que se está enviando
- Comportamiento del emisor cuando se recibe un paquete de feedback
- Comportamiento del emisor cuando expira el nofeedback timer
- Programación de la transmisión en sistemas operativos de no-tiempo real

14.9.6.1.Medición del tamaño de paquete

El tamaño de paquete (s) normalmente es conocido para una aplicación, y en el caso del presente trabajo es así. Simplemente como comentario, el tamaño de paquete puede ser desconocido en los siguientes casos:

- El tamaño de los datos varía en forma natural dependiendo de los datos

- La aplicación necesita cambiar el tamaño de los paquetes en lugar del número de paquetes por segundo para llevar a cabo el control de congestión. Es el caso de algunas aplicaciones de audio donde cada cada paquete debe representar un intervalo de tiempo fijo.

14.9.6.2.Inicialización del emisor

Para inicializar el emisor, se establece el valor del throughput (X) a 1 paquete/segundo y se configura al nofeedback timer para que expire después de 2 segundos. Los valores iniciales para el RTT (R) y t_RTO permanecen indefinidos hasta que sean establecidos como se describe más abajo. El valor inicial de tld, para el Time Last Doubled durante el slow-start, se establece en –1.

14.9.6.3.Comportamiento del emisor cuando recibe un paquete de feedback

El emisor conoce su velocidad actual de transmisión T y mantiene una estimación de su actual RTT (R) y de su intervalo de timeout (t_RTO).

Cuando el emisor recibe un paquete de feedback en el instante t_now, se ejecutan las siguientes acciones:

1) Se calcula un nuevo RTT de muestra

R_sample = (t_now - t_recvdata) - t_delay

2) Se actualiza el RTT estimado

Si no se han recibido paquetes de feedback antes:

R = R_sample

De lo contrario:

R = q*R + (1-q)*R_sample

TFRC no es sensible al valor preciso para la constante de filtro q, y se recomienda un valor de q=0.9.

3) Se actualiza el intervalo de timeout

t_RTO = 4R

4) Se actualiza la velocidad de transmisión

Si la tasa de eventos de pérdida es mayor que cero (p>0), entonces se calcula T_calc usando la ecuación del throughput TCP:

T = max(min(T_calc, 2*T_recv), s/t_mbi)

De lo contrario, si (t_now – tld) >= R:

X = max(min(2*X, 2*X_recv), s/R)
tld = t_now

Nótese que si p=0 entonces el emisor está en la fase de slow-start, donde dobla aproximadamente la velocidad de transmisión cada RTT hasta que ocurra una pérdida. El término s/R asegura una velocidad de transmisión mínima de 1 paquete/segundo durante el slow-start. El parámetro t_mbi vale 64 seg y representa el máximo intervalo de backoff entre paquetes en ausencia de feedbacks. Así, si p > 0 el emisor envía como mínimo 1 paquete cada 64 seg.

Como comentario importante, **se hace notar que la velocidad de envío con TFRC está limitada a la velocidad máxima que la aplicación puede enviar los datos en ausencia de pérdidas y en un estado de equilibrio o a lo sumo al doble de la velocidad a la cual el receptor estima que los datos fueron recibidos desde que el último reporte de feedback fue enviado (2*X_recv).**

5) Reseteo del nofeedback timer para que expire después de max(4R,2s/X) seg.

14.9.6.4.Expiración del nofeedback timer

Si expira el nofeedback timer (en la práctica se espera por 4 RTT’s) por ejemplo a causa de congestión en el camino de retorno de los paquetes de feedback, el emisor debería desarrollar las siguientes acciones:

1) Bajar la velocidad de envío a la mitad

2) Recalcular el valor de X como se describió en el punto 4 del ítem anterior

3) Re-arrancar el nofeedback timer para que expire después de max(4R,2s/X) seg

14.9.7.Cálculo de la tasa de eventos de pérdida

Es de vital importancia para TFRC que se obtenga una medida exacta y estable de la tasa de eventos de pérdida. La medición de la tasa de pérdida es llevada a cabo en el receptor, y se basa en la detección de paquetes perdidos o marcados a través del número de secuencia de los paquetes que llegan.

14.9.7.1.Detección de paquetes perdidos o marcados

TFRC asume que todos los paquetes contienen un número de secuencia que es incrementado por uno para cada paquete que se envía. A los fines de esta especificación, si un paquete es retransmitido se lo hace con un número de secuencia que es el siguiente al último en la secuencia de transmisión, y no el mismo número de secuencia con que el paquete fue transmitido originalmente. Sólo a modo de comentario, si el protocolo de transporte tiene el requerimiento que el paquete retransmitido tenga el número de secuencia original, entonces ese mismo protocolo de transporte deberá ser el encargado de distinguir los paquetes retardados (delayed) de los retransmitidos y cómo detectar las retransmisiones perdidas.

El receptor mantiene una estructura de datos para conocer qué paquetes han arribado y qué paquetes se han perdido. Aquí se asume que esta estructura de datos conciste de una lista de paquetes que han arribado y a los que el receptor les aplicó un timestamp cuando cada paquete fue recibido.

La pérdida de un paquete es detectada mediante el arribo de al menos tres paquetes con número de secuencia mayor que el del paquete perdido. El requerimiento para tres paquetes subsecuentes es el mismo que en TCP, y hace que TFRC sea más robusto ante la presencia de reordenamiento. Por otro lado, en contraste con TCP, si en TFRC un paquete arriba tarde (depués de tres paquetes subsecuentes arribados), dicho paquete puede llenar el hueco en el registro de recepción de TFRC y así el receptor puede recalcular la tasa de eventos de pérdidas. En la práctica, esta estructura de datos es específica a la implementación.

En el presente trabajo, no se trabaja con conexiones habilitadas para ECN (Explicit Congestion Notification) y por ende no se tiene en cuenta a los paquetes marcados a la hora de calcular la tasa de eventos de pérdidas. Pero como comentario, para el caso de una conexión habilitada para ECN, un paquete marcado es detectado como un evento de congestión tan rápido

como el mismo arriba, sin necesidad de tener que esperar el arribo de los paquetes siguientes.

14.9.7.2.Traducción de historia de pérdidas a eventos de pérdidas

TFRC requiere que la fracción de pérdidas sea robusta ante varios paquetes perdidos consecutivos que forman parte del mismo evento de pérdidas. Esto es similar a TCP, el cual típicamente reduce a la mitad la ventana de congestión durante cualquier RTT. De esta manera, el receptor necesita mapear la historia de pérdidas de paquetes dentro de un registro de eventos de pérdidas donde como ya lo dije, **un evento de pérdidas es uno o más paquetes perdidos dentro de un mismo RTT**. Para realizar este mapeo, el receptor necesita conocer el RTT que va a usar, el cual usualmente es suministrado por el emisor como piggy-back de informacion de control en un paquete de datos (piggybacking en general es la técnica mediante la cual se envían paquetes de respuesta que tienen anexadas las confirmaciones de los paquetes recibidos). En TFRC se recomienda usar como valor de RTT al calculado por el emisor según lo descrito en el punto 14.5.3.3.

Para determinar si un paquete perdido (o marcado, pero ya dije que aquí descarto el uso de ECN) debería comenzar un nuevo evento de pérdida, o ser considerado parte de un evento de pérdida existente, se necesitan comparar los números de secuencia y timestamps que arriban al receptor. Para ello existen un juego de fórmulas que están detalladas en la RFC 3448, y que toman en cuenta la secuencia y el tiempo de recepción de un paquete que arriba, el número de secuencia del paquete perdido, el número de secuencia del último paquete que arribó con número de secuencia anterior al paquete perdido y su correspondiente tiempo de recepción, y el número de secuencia del primer paquete que arribó luego del paquete perdido y su correspondiente tiempo de recepción.

14.9.8.Protocolo de recepción de datos

El receptor envía mensajes de feedback al emisor en forma periódica. **Normalmente se debería enviar al menos un paquete de feedback por RTT**, a menos que el emisor esté transmitiendo a una velocidad menor que un paquete por RTT en cuyo caso se debería enviar un paquete de feedback por cada paquete de datos recibido. Un paquete de feedback además debería ser enviado cuando se detecta un nuevo evento de pérdidas sin tener que esperar hasta el final del RTT, y también cuando se reciba un paquete de datos fuera de orden que remueve un evento de pérdidas de la historia.

Si el emisor está transmitiendo a una alta velocidad (muchos paquetes por RTT) puede llegar a haber algunas ventajas al enviar mensajes de feedback periódicos con frecuencia mayor a uno por RTT dado que esto permite una respuesta más rápida para cambiar las mediciones del RTT y mayor resiliencia para el feedback de los paquetes perdidos. Sin embargo, no se gana mucho más si se envía un gran número de mensajes de feedback por RTT.

14.9.8.1.Comportamiento del receptor ante la recepción de un paquete de datos

Cuando arriba un paquete de datos, el receptor lleva a cabo los siguientes pasos:

1. Agrega el paquete al historial de paquetes
2. Toma el valor de p como p_prev y calcula el nuevo valor de p como se describe en la RFC 3448
3. Si p > p_prev, hace que el feedback timer expire y desarrolla las siguientes acciones: calcula el p promedio, calcula la velocidad de recepción medida (X_recv), prepara y envía paquetes de feedback que contienen la información detallada en 14.5.2.3.2. y restartea el feedback timer para que expire después de R_m seg (es la medición del RTT incluida en el paquete S_m que es el máximo número de secuencia de un paquete en el receptor). En el caso que p<= p_prev no se toma ninguna acción.

Si no se recibieron paquetes de datos desde que el último feedback fue enviado, entonces no se envía ningún paquete de feedback y el feedback timer se re-arranca para expirar luego de R-m seg.

14.9.8.2.Inicialización del receptor

El receptor es inicializado por el primer paquete que arriba a él. Si el número de secuencia de este paquete es i, entonces cuando llega este primer paquete:

- Establece p = 0
- Establece X_recv = 0
- Prepara y envía un paquete de feedback
- Establece el feedback timer para que expire después de R_i seg

Además se lleva a cabo la inicialización de la historia de las pérdidas después del primer evento de pérdidas. El número de paquetes hasta la primera pérdida no puede ser usado para computar directamente la velocidad de envío, dado que esta velocidad cambia rápidamente durante este tiempo. TFRC asume que la velocidad de datos correcta después de la primera pérdida es la mitad de la velocidad de envío al momento que la pérdida ocurrió. TFRC aproxima esta velocidad target por medio de X_recv, la velocidad de recepción sobre el RTT más reciente. Después de la primera pérdida, en lugar de inicializar el primer intervalo de pérdidas al número de paquetes enviados hasta la primera pérdida, el TFRC del lado receptor calcula el intervalo de pérdida que podría ser requerido para producir la velocidad X_recv, y usa este intervalo de pérdida sintético para alimentar el mecanismo de historial de pérdidas. TFRC hace esto por medio de buscar algún valor p para el cual la ecuación de throughput del punto 14.5.2.2. dé una velocidad de envío dentro del 5% de X_recv, dado el actual tamaño de paquete (s) y el RTT (R). Entonces, el primer intervalo de pédidas se establece en 1/p. El 5% de tolerancia se introduce porque la ecuación de throughput es difícil de invertir, y lo que se quiere es reducir los costos de calcular p numéricamente.

14.9.9.TFRC aplicado sólo al video y no al audio

En el caso del presente trabajo, **asumo que no conozco el estado de ocupación del enlace por el que transporto el tráfico HDTV** de audio y video.

Se envían dos streams RTP separados y sólo se aplica control de congestión TFRC al flujo de video y no al de audio. La justificación es la siguiente:

- El video sin comprimir originalmente se lo puede enviar a una tasa de frames de 60 fps, pero se puede variar esta tasa y así variar la velocidad de video. Con lo cual el video sin comprimir sería apropiado para ser usado con el mecanismo de control de congestión TFRC dado que el video podría ir acomodando su velocidad a lo largo del tiempo para corresponderse al throughput calculado por la ecuación usada por TFRC, según los valores de tasa de eventos de pérdidas, RTT y tamaño de paquete.

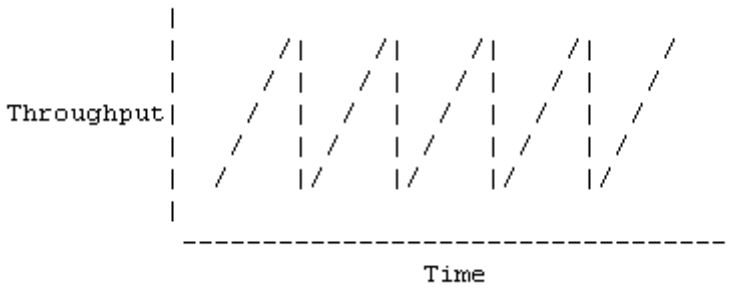
- El audio es codificado en el formato AC-3 y sólo soporta tasas de muestreo de 48 KHz, 44.1 KHz y 32 KHz. Este tipo de CODEC no soporta valores intermedios de tasas de muestreo con lo cual no se puede variar a lo largo del tiempo dicho parámetro. En consecuencia,

no se puede aplicar el mecanismo de control de congestión TFRC con este CODEC de audio.

Con lo antes dicho, los streams se comportan en forma independiente, pero sólo se aplica control de congestión al video HDTV.

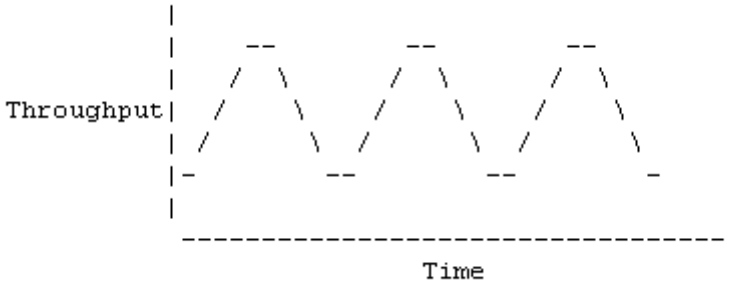
14.9.10.Representación gráfica de la velocidad de TCP y el control de congestión TFRC

Una vez que TCP pasa la fase de slow-start ingresa al control de congestión denominado “congestion avoidance”, cuya representación de la velocidad en función del tiempo tiene aproximadamente la forma de una “diente de sierra”:



Throughput con el control de congestión AIMD

En el caso del control de congestión TFRC, la respuesta inmediata a la pérdida de paquetes es menos dramática. A manera de compensar este hecho, TFRC es menos agresivo para buscar nuevo ancho de banda posteriormente a una pérdida. La representación gráfica de la velocidad en función del tiempo tiene una forma aproximada a una sinusoide:



Throughput característico para TFRC

14.10.Perfil RTP para TFRC

14.10.1.Introducción

TFRC es un esquema de control de congestión para flujos unicast en redes "best-effort" basado en una ecuación de throughput TCP, y necesita algún mecanismo para ser implementado. El perfil RTP para TFRC [102] –cuya última revisión data de 2005- es el encargado de proveer a los flujos RTP el mecanismo adecuado para usar el control de congestión en redes IP "best-effort".

El citado Internet Draft define un perfil llamado "RTP/AVPCC" para ser usado por el protocolo de transporte de tiempo real (RTP) y su protocolo de control asociado con el control de congestión TFRC.

Debido al número de características inherentes de TFRC, el perfil RTP/AVPCC difiere de otros perfiles RTP (AVP) en lo siguiente:

- TFRC es un esquema de control de congestión unicast, por lo tanto por extensión, RTP/AVPCC puede ser usado solamente por flujos unicast RTP.

- Un emisor TFRC confía en recibir feedback desde el receptor una vez por RTT o por paquete de datos. Para ciertos flujos, dependiendo de los tiempos RTT y de las velocidades de datos, estos requerimientos de TFRC pueden resultar en un tráfico de control que puede exceder las recomendaciones para ancho de banda y timing para el tráfico de control (o sea, tráfico RTCP) dadas en la RFC 3550, la cual recomienda que el ancho de banda para el tráfico de control sea una fracción pequeña y conocida del ancho de banda de sesión. En dicha RFC se recomienda que el ancho de banda para el tráfico RTCP se fije en el **5%** del ancho de banda de la sesión, y 5 segundos como el intervalo mínimo de transmisión de paquetes RTCP. **El perfil RTP/AVPCC recomienda realizar modificaciones a estas recomendaciones para así satisfacer las necesidades de timing para el tráfico de control de una manera segura**.

El Internet Draft de TFRC trata principalmente de brindar los medios para soportar el intercambio de información de control de congestión de TFRC entre el emisor y el receptor a través de las siguientes modificaciones hechas a los protocolos RTP y RTCP: **(1)** Agregados al RTP Data Header;

(2) Extensiones a los RTP Receiver Reports; y **(3)** Modificaciones a los intervalos de tiempo RTCP recomendados.

14.10.2.Principales características

Hay un beneficio destacado a la hora de aplicar el perfil RTP para TFRC, que se resume de la siguiente manera:

- Las aplicaciones multimedia que carecen de control de congestión pueden incorporar transporte con control de congestión sin delay, usando el perfil RTP/AVPCC.

- El uso del perfil RTP/AVPCC no requiere de cambios a nivel de sistema operativo, sino que su implementación se hace en la capa de aplicación.

- Los flujos AVPCC/RTP/UDP enfrentan las mismas restricciones al atravesar un firewall que las que enfrentan los flujos UDP y no requieren de modificaciones de NAT o de configuraciones de firewall especiales.

- El uso del perfil RTP/AVPCC con varias aplicaciones multimedia le dará a los investigadores, desarrolladores e implementadores un mejor entendimiento de la complicada relación entre la calidad de la multimedia y el control de congestión basado en una ecuación.

Además hay ciertas características técnicas acerca de cómo la información de control de congestión es intercambiada:

- Un emisor RTP/AVPCC transmite un timestamp al receptor RTP/AVPCC con cada paquete de datos que envía. En suma al control de congestión, el timestamp enviado puede ser usado por el receptor para calcular el jitter.

- Un receptor RTP/AVPCC sólo le provee al emisor RTP/AVPCC la tasa de eventos de pérdidas computada por el receptor.

- RTP soporta sólo un número de secuencia (sequence number) de 16 bits, y esto hace que RTP sea susceptible a ataques de inyección de datos.

14.10.3.Estructuras de los paquetes RTP y RTCP y comportamiento del protocolo

La sección “RTP Profiles and Payload Format Specifications” de la RFC 3550 enumera los ítems que pueden ser especificados o modificados en un perfil. A continuación se detallan las modificaciones para el caso del perfil RTP/AVPCC:

- RTP Data Header: el formato estándar del RTP data header es modificado como se especifica más adelante.
- Payload Types: el payload type en el RTP data header se reduce a 6 bits.
- Agregados al RTP Data Header: se agregan dos campos fijos de 32 bits, el timestamp de envío y el RTT. El timestamp de envío siempre está presente y el campo RTT está presente si el bit R está establecido.
- Extensiones del RTP Data Header: no se definen extensiones del RTP Data Header, pero las aplicaciones que operan bajo este perfil pueden usar tales extensiones.
- RTCP Packet Types: no se definen RTCP packet types adicionales en este perfil.
- RTCP Report Interval: este perfil está restringido a flujos unicast, y además que en todo momento haya un emisor y un receptor activos. **Las sesiones que operan bajo este perfil pueden especificar un parámetro separado para el ancho de banda del tráfico RTCP en lugar de usar la fracción por defecto del ancho de banda de sesión. En particular esto puede ser necesario para flujos de datos donde el intervalo mínimo recomendado para RTCP es aún mayor que el RTT.**
- Extensión SR/RR: se definen 16 octetos de extensión RR para el paquete RR de RTCP.
- Uso de SDES: las aplicaciones pueden usar cualquiera de los ítems SDES descritos en la especificación RTP.
- Seguridad: este perfil no especifica ninguna consideración adicional de seguridad.
- Congestión: este perfil especifica cómo usar RTP/RTCP con el control de congestión RTCP.
- Protocolos subyacentes: el perfil especifica el uso de RTP sólo sobre flujos UDP unicast, multicast no deberá ser usado.
- Encapsulamiento: este perfil se define para encapsulamiento solamente sobre UDP.

14.10.4.Implementación del loop de feedback de TFRC

TFRC depende del intercambio de información de control de congestión entre un emisor y un receptor. En esta sección se describe la manera en que el perfil RTP/AVPCC acomoda estos intercambios:

Paquetes de datos

Un emisor TFRC transmite la siguiente información en cada paquete de datos que envía al receptor:

- Un número de secuencia, incrementado de a uno por cada paquete de datos transmitido.
- Un timestamp indicando el tiempo de envío del paquete y la estimación actual del RTT por parte del emisor. Luego esta información es usada por el receptor para computar los intervalos de pérdida TFRC.

En este sentido, la numeración de secuencia estándar de RTP es suficiente para la funcionalidad de TFRC. En cambio, para el cómputo de los intervalos de pérdida, el perfil RTP/AVPCC extiende el RTP Data Header de la manera siguiente: un campo de 32 bits para transmitir el timestamp de envío y un campo adicional de 32 bits sólo presente cuando cambia el RTT para transmitir el RTT. La presencia del RTT es indicada por el bit R en el RTP Header.

Paquetes de feedback

El receptor TFRC le provee un feedback al emisor como mínimo una vez por RTT o por paquete de datos recibido (según cual sea el intervalo de tiempo más largo), con lo siguiente:

- El timestamp del último paquete de datos recibido, t_i.
- La cantidad de tiempo transcurrido entre la recepción del último paquete de datos en el receptor y la generación de este reporte de feedback, t_delay. Esto es usado por el emisor para el cómputo del RTT.
- La velocidad a la cual el receptor estima que los datos fueron recibidos desde que el último reporte de feedback fue enviado, x_recv.
- La actual estimación del receptor de la tasa de eventos de pérdidas, p.

Para acomodar el feedback de estos valores, el perfil RTP/AVPCC define una extensión de 16 octetos a los Receiver Reports de RTCP.

14.10.5.Agregados al RTP Data Header

El perfil RTP/AVPCC cambia el RTP Data Header de la siguiente manera:

- Define un tamaño de 6 bits para el campo de Payload Type (PT).
- Define un campo R de 1 bit en el segundo octeto.
- Define dos campos fijos adicionales de 32 bits para: (1) el timestamp de envío de los paquetes y (2) el RTT de la manera en que lo mide el emisor (este campo está presente si el campo R tiene el bit establecido en 1.

Los campos adicionales del RTP Header son definidos y usados de la siguiente manera:

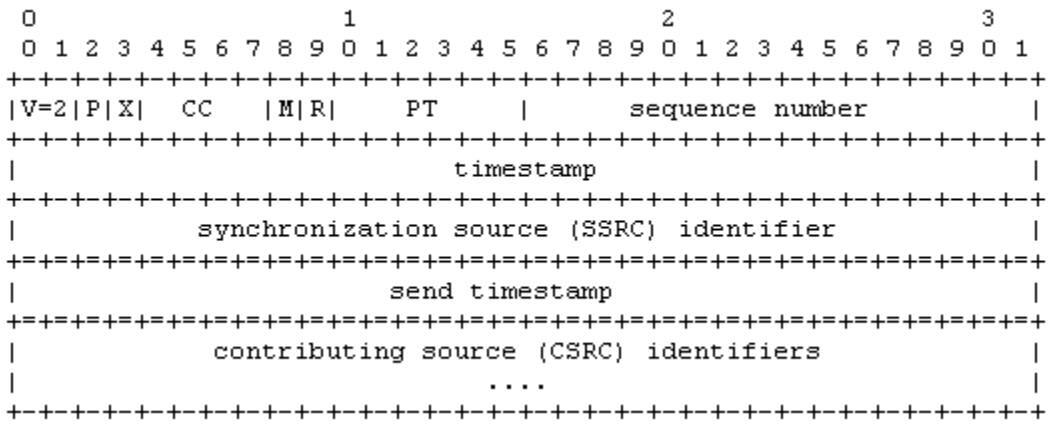
Round Trip Time Indicator (R): 1 bit – Este campo indica la existencia del campo RTT adicional. Si el bit R está establecido, entonces el RTP Payload Header incluye un campo de 32 bits que representa el actual RTT.

Payload Type (PT): 6 bits – El perfil RTP/AVPCC usa un campo de 6 bits para el tipo de payload en lugar de usar 7 bits.

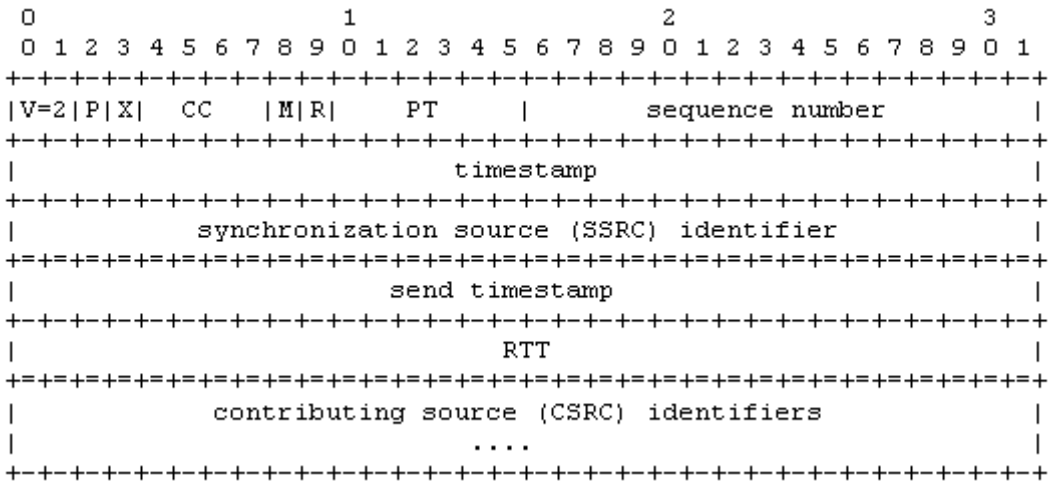
Timestamp de envío: 32 bits – Este timestamp indica el momento en que el paquete es enviado.

Round Trip Time (RTT): 32 bits – El RTT es medido por el emisor RTP/AVPCC en milisegundos. Este campo está presente sólo si el bit R está establecido.

A continuación se presentan los gráficos relativos a RTP Header y sus agregados:



RTP header y agregados con R=0, el RTT no está incluido



RTP header y agregados con R=1, el RTT está incluido

14.10.6.Extensiones del Receiver Report de RTCP

El perfil RTP/AVPCC define cuatro campos fijos de 32 bits (16 octetos) como extensiones específicas al perfil para los Receiver Reports. Estos cuatro campos son:

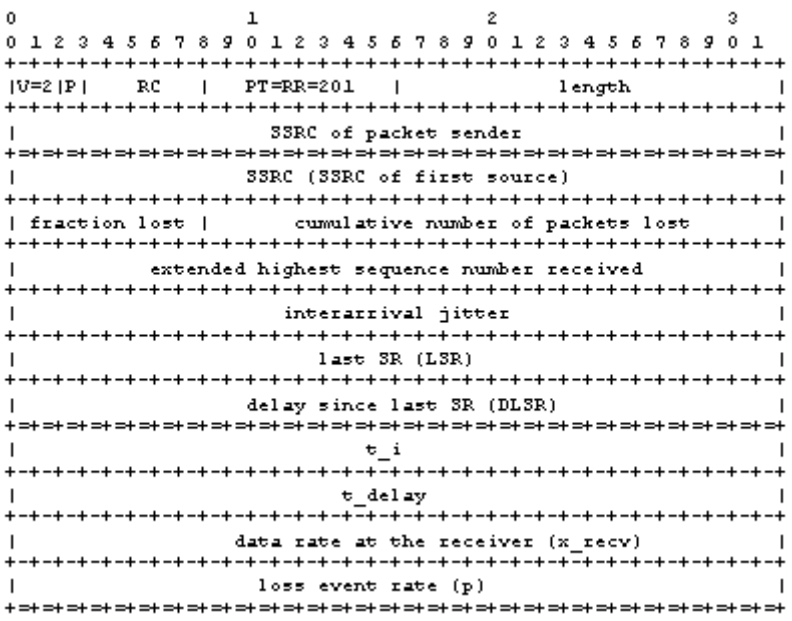
Timestamp (t_i): 32 bits – Es el timestamp del último paquete de datos recibido, t_i, medido en milisegundos.

Delay (t_delay): 32 bits – Es la cantidad de tiempo transcurrido entre la recepción del último paquete de datos en el receptor y la generación de este reporte de feedback, t_delay. Esto es usado por el emisor para calcular el RTT.

Velocidad de Datos(x_recv): 32 bits – Es la velocidad a la cual el receptor estima que los datos fueron recibidos desde que el último reporte de feedback fue enviado, medido en bits por segundo.

Tasa de eventos de pérdidas (p): 32 bits – Es laestimación actual del receptor de la tasa de eventos de pérdidas, p.

En el siguiente gráfico se detalla lo arriba dicho:



Extensiones del RTCP Receiver Report

14.10.7.Intervalos de tiempo de RTCP

El perfil RTP/AVPCC recomienda el uso de los requerimientos de intervalos de feedback de TFRC para los intervalos de tiempo de RTCP, sólo cuando el ancho de banda del tráfico de control no exceda el 5% del tráfico de datos recomendado por la RFC 3550.

Un emisor TFRC requiere feedback del receptor al menos una vez por RTT o por paquete de datos recibido (basado en el intervalo de tiempo más largo). **Estos requerimientos son para asegurar una reacción a tiempo ante la congestión.**

En ciertos casos, los requerimientos de timing de TFRC pueden resultar en intervalos de tiempo para el tráfico RTCP que son más pequeños que el intervalo de tiempo mínimo reducido recomendado por la RFC 3550, que sería de 360 dividido por el ancho de banda de sesión en Kbps, es decir:

$$T(seg) = 360/X(Kbps)$$

Pueden existir casos donde el requerimiento de feedback TFRC en segundos es menor que el intervalo de tiempo mínimo reducido recomendado por la RFC 3550. Y en estos casos habría también que evaluar si el tráfico de feedback es menor al 5% del tráfico de datos recomendado por la misma RFC. Si esto es así, no hay ningún problema en adoptar dicho intervalo como intervalo de feedback.

Hay que tener en cuenta que un mayor tráfico de control o feedback implica mayor capacidad de procesamiento por parte del emisor y receptor, y eso hay que tenerlo en cuenta a la hora de la definición del sistema.

14.10.8.Consideraciones de IANA

El perfil RTP para TFRC extiende el perfil para las conferencias audiovisuales con mínimo control y necesita ser registrado para el Protocolo de Descripción de Sesión (SDP) como “RTP/AVPCC”.

Protocolo SDP:

Nombre:	RTP/AVPCC
Forma larga:	RTP Profile for TCP Friendly Rate Control
Tipo de nombre:	proto
Tipo de atributo:	Media level only
Aclaración:	Todavía no es una RFC

14.10.9.Conclusión acerca del uso del perfil RTP/AVPCC

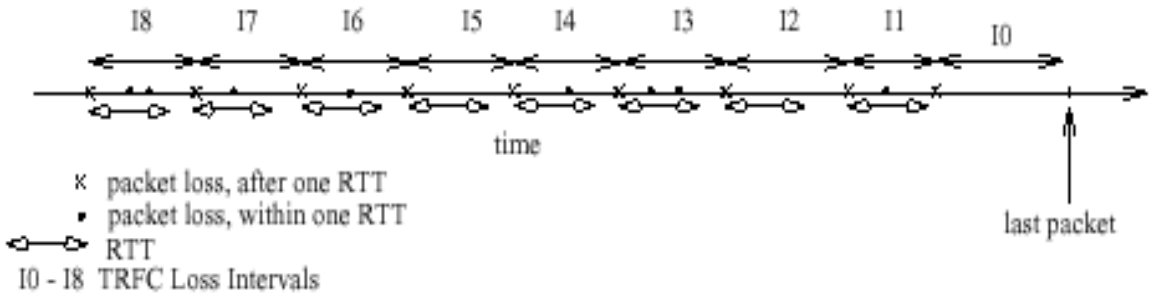
En el caso del presente trabajo, como ya se mencionó, se envía el audio y video en streams RTP separados, cada uno con su formato de payload RTP respectivo como ya se describió, pero sólo el video lleva asociado el perfil RTP/AVPCC para su adaptación al control de congestión TFRC en forma independiente.

Vale decir que la velocidad del stream de video varía en forma independiente a lo largo del tiempo, según la ecuación del throughput aplicada por TFRC.
Obviamente que en ningún caso el throughput video adaptado con TFRC superará la velocidad máxima de transmisión de video de HDTV.

14.11.Implementación práctica de TFRC

Como ya lo expresé antes, TFRC se implementa por medio del perfil RTP/AVPCC aplicado al flujo de video, y se basa en la habilidad del emisor de ajustar la velocidad de transmisión de acuerdo a la cantidad de pérdidas de paquetes que el flujo está experimentando.

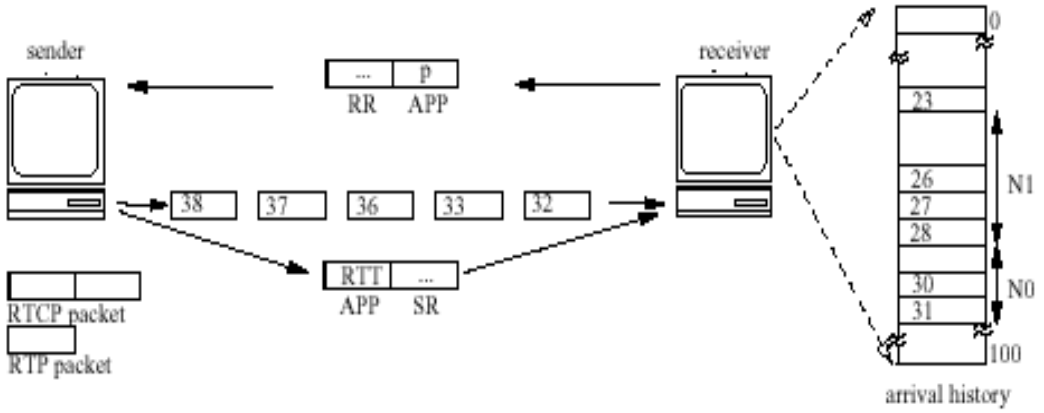
Para medir un intervalo de pérdida en TFRC [106], una vez que ocurre una pérdida inicial, cualquier otra pérdida subsiguiente dentro de un RTT es ignorada. La figura siguiente lo detalla:



Con el objeto de llevar a cabo la implementación de TFRC, se necesitan los siguientes loops de feedback implementados con RTCP:

- Primero, en forma periódica el emisor debe enviar el RTT percibido hacia el receptor, y de esta manera permite que el receptor compute la tasa de eventos de pérdidas (p).
- Segundo, el receptor debe enviar la tasa de eventos de pérdidas computada (p) de vuelta hacia el emisor.

En la siguiente figura se detalla este proceso:



Donde:

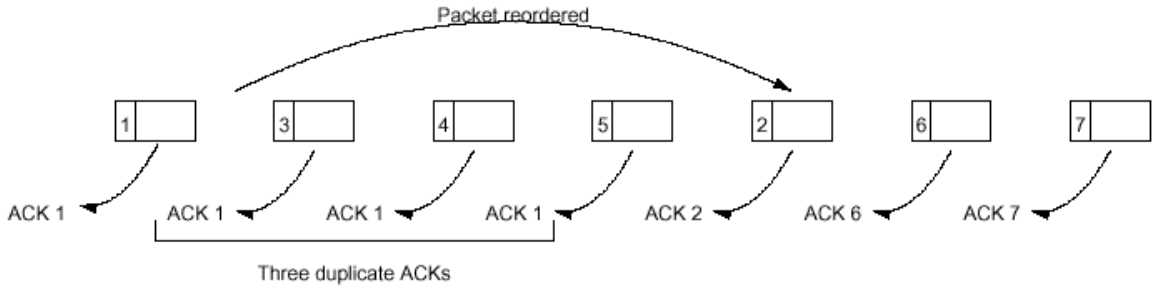
N0, N1, etc., son los intervalos de pérdidas que se guarda en la memoria del receptor.

La implementación contempla el uso de RTP sobre UDP/IP. RTP provee feedback usando el RTP Control Protocol (RTCP). A intervalos regulares de tiempo, la implementación genera Receiver Reports (RR) por parte del receptor o Sender Reports (SR) por parte del emisor que proveen el feedback de la calidad de recepción y soporte para “lip-synchronization” (es la sincronización de la señal de audio y la correspondiente señal de video para que no haya una falta notable de simultaneidad entre ellas). Además hay paquetes RTCP definidos para la aplicación (APP), los cuales son “piggy-backed” a intervalos de tiempo regulares sobre paquetes RR o SR de RTCP. Entonces, una implementación posible de este mecanismo puede ser la de que cada vez que el emisor genera un Sender Report también envía el RTT al receptor en un paquete APP. Y a su vez, cuando el receptor envía un Receiver Report también incluye un paquete APP con el último cómputo de la tasa de eventos de pérdidas (p). Todo esto se puede apreciar en forma detallada en el gráfico anterior.

Experiencias previas con TFRC demuestran que en los casos que no hay pérdidas en la red y que por ende la velocidad de transmisión se debería mantener constante, esto no ocurre así debido a la presencia de reordenamiento de paquetes. Por lo tanto, a causa del reordenamiento de paquetes uno obtiene velocidades menores que las esperadas, y la conclusión es que TFRC trata al reordenamiento de paquetes como

pérdidas. Vale decir que un evento de reordenamiento de paquetes puede disparar un nuevo intervalo de pérdidas, como si fuese una pérdida real de un paquete.

Por ejemplo, los paquetes que arriban como mínimo cuatro lugares fuera de orden pueden causar que TCP envíe un triple ACK duplicado, dando la apariencia de pérdida. Aquí se ve un esquema de este comportamiento, donde se dan tres ACKs duplicados del paquete número 1 (es duplicado dado que en principio se dio un ACK del paquete número 1 y luego sobrevienen tres ACK más de ese mismo paquete):



El análisis detrás de la ecuación de control TFRC refleja este comportamiento, y por lo tanto se espera que TFRC trate al reordenamiento de paquetes como pérdidas.

Una gráfica aproximada de la curva de velocidad TFRC a lo largo del tiempo sería como la siguiente:

Curva de velocidad TFRC

14.12.Uso de Jitter Buffers de Audio y Video

14.12.1.Descripción y definición de los buffers

A este tipo de buffers también se los conoce como “media buffers” (buffers para la multimedia) o playout buffers (buffers para la reproducción) [47]. Un media buffer es una cola FIFO (firts-in-firts-out) de datos multimedia, que es llenada por una fuente de datos como un codificador o la red, y drena hacia la red o hacia algún dispositivo de reproducción, con el objeto de ocultar las variaciones de la velocidad de transmisión de los datos multimedia y así tener siempre una reproducción continua del audio y video.

Los media buffers se miden en segundos, y no en paquetes o bytes. Se construyen completamente a nivel aplicación y están separados de las colas de transmisión y recepción de la capa de transporte.

En esta implementación se asume que no ningun jitter en el emisor y que el buffer de reproducción se encuentra en el receptor y a está implementado a nivel aplicación.

Luego que los paquetes de datos son extraídos de la cola de entrada y se depositan en el playout buffer clasificados según su timestamp, los frames se mantienen allí por un determinado período de tiempo para suavizar las variaciones del timing causadas por la red. Además el hecho de mantener los datos en un buffer también permite recibir y agrupar los fragmentos de los frames y llevar a cabo algún tipo de corrección de error de los datos. Luego se ocultan los errores que permanecen sin corregir y finalmente se reproduce la multimedia.

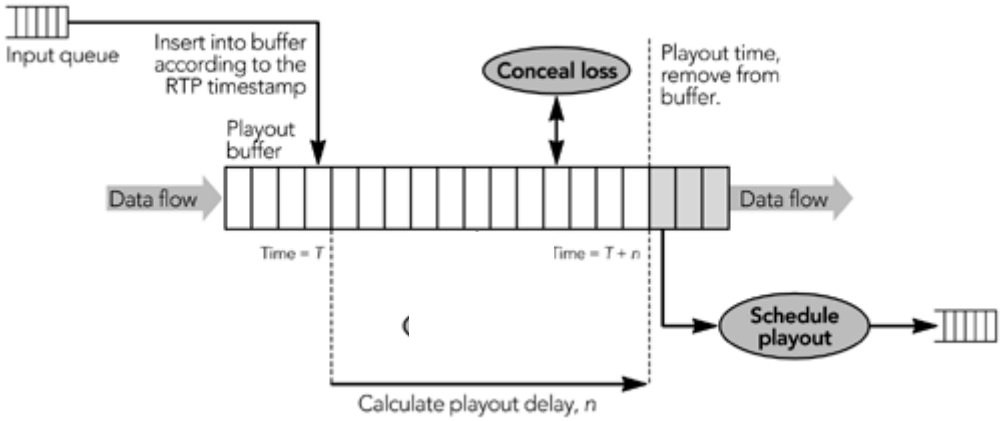


Diagrama del libro “RTP: audio and video for the Internet”
de Colin Perkins

En el caso de esta implementación en particular no considero en principio el empleo de ningun método de corrección de errores ni ocultamiento de los mismos.

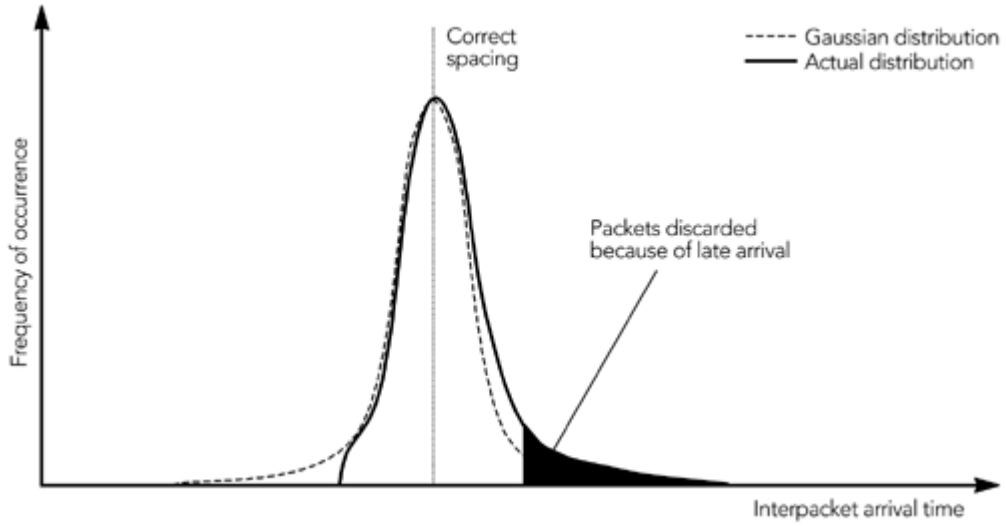
Al momento que el paquete de datos arriba al buffer, se calcula un tiempo de reproducción (playout time) que es un aspecto crítico de la implementación. Para ello se deberá tener en cuenta lo siguiente:

- Delay entre la recepción del primer y último paquete de un frame
- La variación del timing inter-paquetes causada por el jitter de las colas de los dispositivos de red y los cambios de rutas
- El delay que la aplicación esta dispuesta a soportar en base a la calidad y latencia de recepción
- El delay antes de aplicar un método de corrección de errores, en el caso que se decida su uso

Los paquetes que arriban al buffer en forma tardía son descartados para la reproducción y cuanto mejor se calcule el tamaño del buffer menos ocurrirá este hecho. En el caso de las aplicaciones no interactivas –tal el caso del streaming en vivo de HDTV sin comprimir- se puede elegir un tamaño de buffer significativamente más grande que el jitter esperado, llegando al orden de varios segundos. Por lo tanto, para minimizar el delay de la reproduccón –o sea, el tamaño del buffer- es necesario estudiar con detenimiento las propiedades del jitter.

Se puede asumir que la distribución del jitter se aproxima a una distribución normal Gaussiana, y así entonces se puede calcular más

fácilmente un delay para la reproducción de la multimedia. Se calcula la desviación estándar del jitter, y de la teoría de la probabilidad se sabe que más del 99% de una distribución normal se encuentra dentro de tres veces la desviación estándar del valor medio. La distribución del jitter depende del camino (path) que el tráfico toma a través de la red y del otro tráfico que comparte ese camino. La causa más importante del jitter es la competencia con otro tráfico, que resulta en la variación de delays de las colas de los routers intermedios. Además, se hace notar que un cambio en la tasa de frames –tal el mecanismo que se sugerirá en esta implementación y que se explica más adelante- provoca una variación del timing entre paquetes.



Distribución del jitter de red

Una aplicación para un escenario no interactivo podría elegir un delay de reproducción que sea igual a tres veces la desviación estándar en los tiempos entre arribos de paquetes. Pero en la práctica se puede aproximar la desviación estándar al valor de la estimación actual del jitter, por lo cual:

Playout Delay = 3 * Jitter

Y por ende el tiempo de reproducción de un frame se calcularía de esta manera:

Tiempo de Reproducción = Tiempo Actual + 3 * Jitter

Para el caso del streaming en vivo de HDTV sin comprimir, se puede contar con un buffer de algunos segundos para audio y video, de acuerdo a las recomendaciones **ITU G.1010 "End-user multimedia QoS categories"**. En este documento, que define el modelo para diferentes categorías de QoS para multimedia desde el punto de vista del usuario, se menciona que el video "one-way" no posee elementos de conversación y por ende el requerimiento de delay no es tan estricto, sugiriendo un valor de delay menor o igual a 10 seg. Con el audio streaming ocurre lo mismo, que al no haber ningún elemento conversacional (como e el caso de la mensajería de voz también) los requerimientos de delay son más relajados; la misma tabla citada antes sugiere valors de buffers de audio de hasta 10 seg. Además en la tabla de requerimientos de QoS del documento ITU G.1010 se considera al streaming de audio y video con una pérdida de paquetes del orden del 1%, valor por encima del que se experimenta en la red Abilene. Además, se menciona allí que el streaming de audio y video es tolerante a errores. Por último, se puede decir que la elección de un buffer de 10 segundos es favorable para el uso de un mecanismo de corrección de errores como por ejemplo las retransmisiones ARQ, dado que los paquetes con error permanecerán en el buffer hasta que sean corregidos.

En la figura se ve la tabla de la ITU G.1010 al respecto, con la recomendación de los 10 seg de buffer:

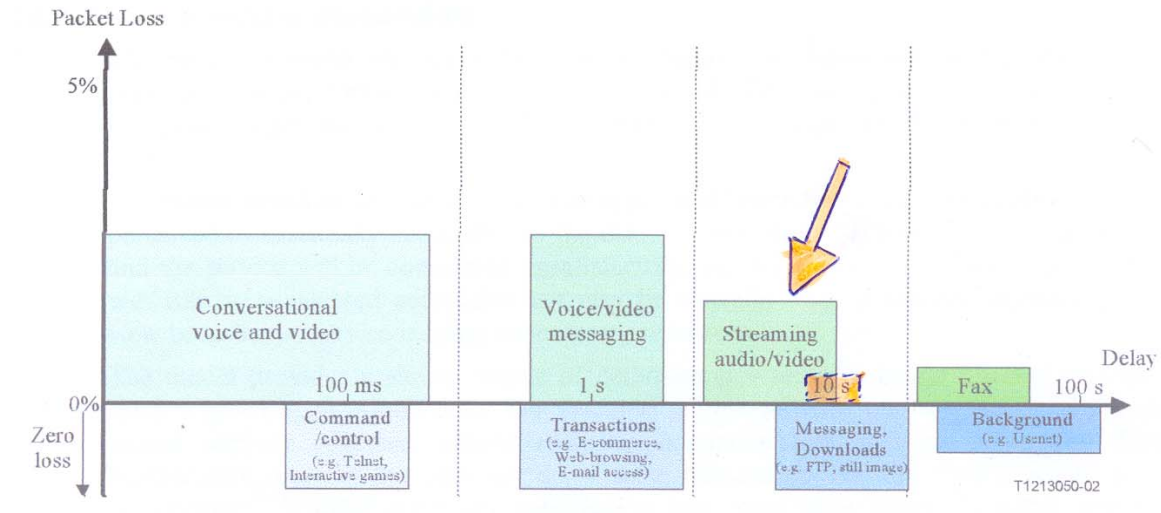


Figure 1/G.1010 – Mapping of user-centric QoS requirements

Por lo tanto, en el presente trabajo **voy a considerar un jitter buffer fijo (no dinámico) de video de 10 seg**. Esto quiere decir que del lado receptor habrá que esperar un tiempo de 10 seg hasta que se llene el buffer para así luego hacer el playback del audio/video sin tener problemas con las fluctuaciones de la red.

En la práctica, también se han realizado experiencias de transmisión de audio y video con buffers de más de 10 seg. Como consecuencia de estos valores de buffer, los resultados del streaming de audio y video son aceptables. Este buffer en el lado receptor ayuda a ocultarle a la aplicación los efectos del "slow-start" de TFRC y las variaciones de la velocidad de transmisión. En verdad, los efectos de TFRC son más notables para el caso de streams interactivos de HDTV que para el caso de streams "one-way" de HDTV, tal como lo es el caso tratado en este trabajo.

La ITU G.1010 dice al respecto de sus recomendaciones y desde el punto de vista del usuario final que si se excede un límite superior, ya sea en pérdidas o en delay, el servicio será considerado insatisfactorio; y que si se excede un límite inferior, a pesar de que el servicio aún podría considerarse aceptable, puede ser muy devorador de recursos de red de manera totalmente innecesaria y esto no sería adecuado.

Con respecto al buffer de audio, el mismo deberá diseñarse con una longitud tal que el audio y el video correspondientes a un mismo instante de muestreo en el emisor sean reproducidos al mismo tiempo en el receptor. Se recuerda nuevamente que las longitudes de buffers para audio y video streaming pueden ser de varios segundos, tal lo especificado en la recomendación ITU G.1010.

14.12.2.Estructura del buffer de video para ajustar la tasa de frames del video

El streaming de video será regulado principalmente por el mecanismo de control de congestión denominado TFRC. En el caso del streaming de audio, al ser de mucho menor velocidad que el video, el tiempo entre paquetes (interpacket timing) será mucho mayor y eso favorece a evitar los estados de congestión que duran un tiempo muy pequeño. Además se recuerda que esta implementación se hace bajo la hipótesis que el backbone es Internet2, y en esa red verdaderamente no se presenta congestión. Diferentes miembros de la lista del Working Group de QoS de Internet2 [103] me han confiado que han padecido congestión sólo una vez o nunca trabajando en Abilene. Por lo tanto, la congestión es más factible que aparezca en los puntos extremos.

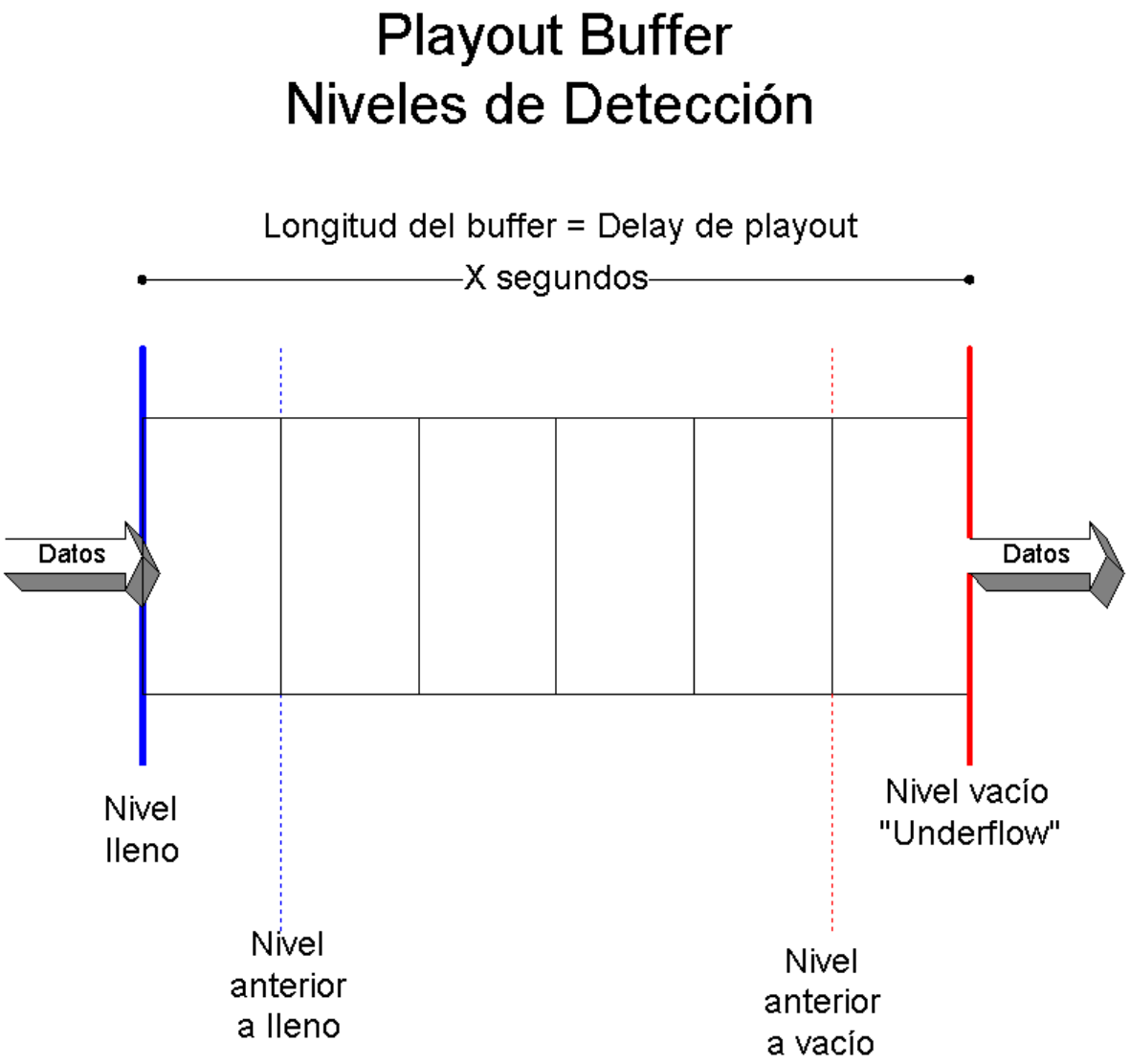
Como un adicional para el video, y para tener cierto control de la congestión del audio, se puede usar un **mecanismo de monitoreo del estado de ocupación de los buffers del extremo receptor para ordenar al emisor que incremente o decremente algún parámetro por medio del cual se pueda controlar la velocidad del stream correspondiente.**

En el caso del video, se analiza el nivel de ocupación del buffer de video para bajar la tasa de frames de video en caso de que exista congestión en la red y de esta manera decrementar la velocidad, y también para subir la tasa de frames de video en caso que la congestión haya desaparecido para intentar obtener mayor ancho de banda. El ajuste de la tasa de frames se hace a nivel aplicación, y el estudio de su lógica esta excento de esta tesis. Se aclara que este esquema de control de congestión se realiza en paralelo al uso de TFRC.

En el caso del audio, dado que los codificadores AC-3 pueden usar un rango diferente de tasas de bits para codificar datos de audio, se analiza el nivel de ocupación del buffer de audio para ajustar la tasa de bits del codificador (extremo emisor) en tiempo real. Por lo tanto, el codificador reducirá la tasa de bits en caso de que exista congestión en la red e incrementará la tasa de bits en caso que la congestión haya desaparecido. Se aclara nuevamente que este esquema de control de congestión es el único empleado para el audio, ya que no se emplea TFRC como en el caso del video.

Cabe señalar que, cuando el nivel de ocupación de uno de estos buffers está lleno o vacío, el receptor deberá indicarle mediante un feedback al emisor que incremente o decremente la velocidad de transmisión. Ese feedback obviamente atravesará la red en sentido contrario al streaming proveniente del emisor, y en caso de congestión, el feedback igualmente alcanzará al emisor porque supongo que **la congestión es unidireccional**. Vale decir, si en el path emisor-receptor ocurre un evento de congestión en la interface de un router intermedio, la otra interface que está unida al camino del receptor no tendrá motivos en principio para estar congestionada y así el feedback del receptor podrá alcanzar al emisor para que adapte su velocidad.

A continuación se detalla un esquema del diseño de los buffers de audio y video y sus niveles de detección:



Tanto en el caso del audio como del video, si el buffer está vacío (condición de “underflow”) significa que hay congestión en la red. Cuando la congestión desaparece, el buffer se va llenando nuevamente hasta su nivel prestablecido. La aplicación deberá monitorear los buffers de audio y video constantemente para saber si debe bajar o subir la tasa de frames del video o la tasa de muestreo de la señal de audio, de acuerdo al nivel de ocupación de los buffers.

En el punto 14.16 de este documento se detallará el funcionamiento del sistema en base al monitoreo de los buffers de playout de audio y video.

14.13.Sincronización del audio y el video

Una sesión multimedia como la de la presente implementación comprende un stream de audio y otro de video transportados por sesiones RTP separadas y sincronizadas en el receptor.

En general, los streams multimedia se transmiten en forma separada -en lugar de hacerlo juntos en el mismo stream y presincronizados- básicamente porque hay heterogeneidad en las redes, CODECs y requerimientos de la aplicación. En este caso particular se estudia la transmisión de audio y video en forma separada usando dos formatos de payload RTP diferentes para cada tipo de medio, y forzando luego a que el sincronismo sea llevado a cabo por el receptor.

En el proceso de sincronización intervienen dos partes [47]: el emisor debe asignar un clock de referencia a los streams y el receptor necesita resincronizar la multimedia deshaciendo las interrupciones de timing causadas por la red.

Comportamiento del emisor

En el emisor van a existir un reloj de referencia común, una línea de tiempo de audio y una línea de tiempo de video. El emisor habilita la sincronización de los streams de audio y video haciendo correr un reloj de referencia común y anunciando periódicamente -a través de RTCP- la relación entre el reloj de referencia y los streams multimedia, permitiendo que el receptor trabaje sobre la relación del timing relativo entre los dos diferentes streams.

A continuación se puede apreciar las las líneas de tiempo que maneja el emisor:

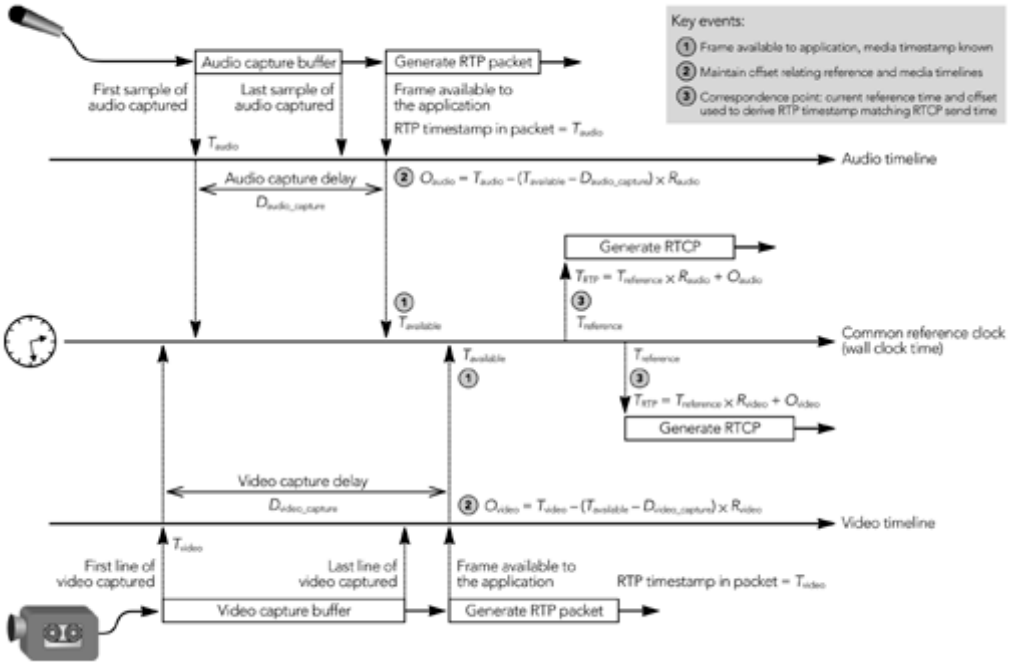


Diagrama del libro “RTP: audio and video for the Internet”
de Colin Perkins

La referencia entre el reloj de referencia y el reloj de la multimedia se hace notar bien en el esquema de arriba cuando se genera cada paquete RTCP. En el paquete RTCP se incluye una muestra del clock de referencia junto con el timestamp RTP calculado.

Cada aplicación en el emisor que está transmitiendo streams RTP necesita acceso al reloj de referencia (Treference en el diagrama) y debe identificarcel tipo de medio con referencia a un identificador canónico de origen.

El reloj de referencia común es el “reloj de pared” (wall clock) usado por RTCP. El mismo toma la forma de un timestamp con formato NTP, contando segundos y fracciones de segundos desde la medianoche UTC (Coordinated Universal Time) desde el 1º de Enero de 1900. El emisor establece en forma periódica una corresponencia entre el reloj multimedia para cada stream y el reloj de referencia común; esto es comunicado a los receptores por medio de paquetes RTCP SR.

En un escenario típico como el presente, no hace falta sincronizar al emisor y receptor a un reloj externo, como por ejemplo usando una fuente de tiempo NTP. Por lo tanto, los relojes del emisor y receptor nunca deberán sincronizarse el uno con el otro. A los receptores no les interesa conocer el valor absoluto del timestamp con formato NTP en los paquetes RTCP SR, sólo les basta con saber que el reloj es común entre los diferentes tipos de medios, y que tiene la suficiente exactitud y estabilidad para permitir llevar a cabo la sincronización.

A manera de comentario, se necesita tener relojes sincronizados sólo en los casos que los streams multimedia son generados por diferentes hosts y deben ser sincronizados. Por ejemplo, sería el caso de diferentes cámaras que dan diferentes puntos de vista de una escena, y cad acámara conectada a un host diferente. En este caso los hosts emisores deberían usar NTP o algún otro protocolo de tiempo para alinear sus relojes de referencia a una base de tiempo común. RTP no dice nada acerca de qué protocolo de tiempo usar en estos casos.

El otro requerimiento para la sincronización es identificar los orígenes que van a ser sincronizados. RTP hace esto asignándole a un nombre en común a los orígenes que deben ser sincronizados y de esta manera el receptor puede identificar los flujos que deben ser sincronizados. Cada paquete RTP contiene un campo identificador de origen de sincronización (SSRC) para asociar los orígenes con una base de tiempo del medio. El mapeo de los identificadores SSRC a un nombre canónico persistente (CNAME) es provisto por los paquetes RTCP descriptores de origen (paquetes SDES). Un emisor debe asegurar que las sesiones RTP a ser sincronizadas en la reproducción tengan el mismo CNAME para que así los receptores puedan alinear la multimedia. El nombre canónico se elige algorítmicamente de acuerdo al nombre de usuario y dirección de red del host emisor.

Comportamiento del receptor

El receptor deberá determinar cuáles streams serán sincronizados y alinear su presentación sobre la base de la información contenida en los paquetes RTCP.

La determinación de los strams a sincronizar se hace por medio de los CNAME de los paquetes descriptores de origen RTCP (paquetes SDES). Dado que los paquetes RTCP son transmitidos cada cierrto tiempo, entonces puede llegar a existir un delay entre la recepción del primer paquete de datos y la recepción del paquete RTCP que indica los streams que deben ser sincronizados, el receptor puede reproducir los datos

multimedia durante este tiempo pero no puede sincronizarlos porque no posee la información requerida.

La operación de sincronización en la cual el receptor alínea audio y video para su correcta presentación, es iniciada por la llegada al receptor de los paquetes RTCP SR que contienen el mapeo entre el reloj del medio y un reloj de referencia común a ambos medios. Una vez que este mapeo ha sido determinado para los streams de audio y video, entonces el receptor tiene la información necesaria para sincronizar la reproducción.

La primera etapa de la “sincronización de labios” es determinar, para cada stream a ser sincronizado, cuándo los datos correspondientes a una referencia de tiempo particular serán presentados al usuario final. El tiempo de reproducción (playout time) para los datos de un stream deberán ser ajustados para corresponderse al otro stream. Este ajuste se traduce en un **offset** que se agrega al delay del playout buffer para un stream y así la multimedia se puede reproducir de manera alineada.

A continuación se grafica el diagrama de “sincronización de labios” en el receptor:

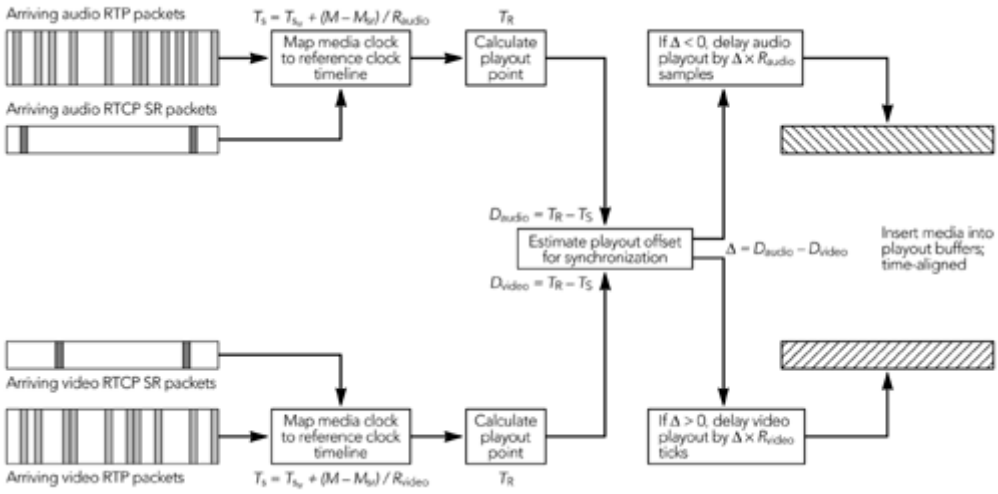


Diagrama del libro “RTP: audio and video for the Internet”
de Colin Perkins

A continuación se presenta un gráfico con el mapeo entre las líneas de tiempo para lograr la sincronización del audio y video en el receptor:

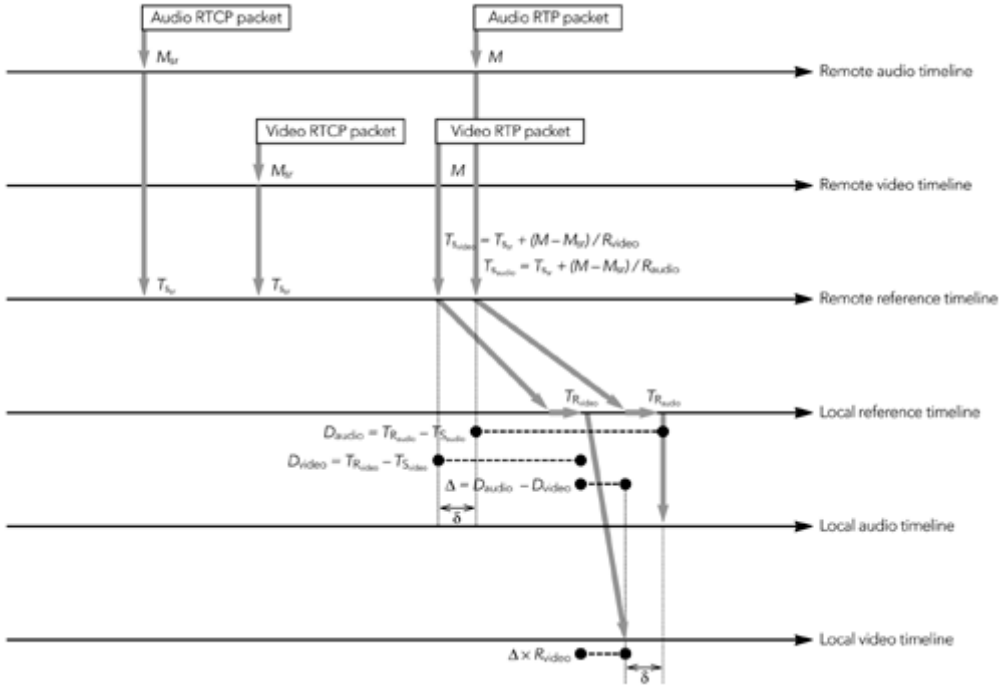


Diagrama del libro “RTP: audio and video for the Internet”
de Colin Perkins

Primero, el receptor observa el mapeo entre el reloj del medio y el reloj de referencia tal como fuer asignado por el emisor, para cada stream que va a sincronizar. Este mapeo le llega al receptor en forma de paquetes RTCP SR, y como la velocidad nominal del reloj del medio es conocida a través del payload format, entonces el receptor puede calcular el tiempo de captura del reloj de referencia para cualquier paquete de datos una vez que ha recibido el RTCP SR de aquel origen.

El receptor también calcula el tiempo de presentación para cualquier paquete en particular, de acuerdo a su reloj de referencia local. Esto es igual al timestamp RTP de cada paquete, mapeado a la línea de tiempo del reloj de referencia más un delay de buffering de playout en segundos y cualquier otro delay debido a los procesos de decodificación y rendering.

Una vez que que se conocen los tiempos de captura y playout de acuerdo a la línea de tiempo común de referencia, el receptor puede estimar el delay relativo entre la captura del medio y su correspondiente reproducción para cada stream. Si los datos muestreados en el instante Ts de acuerdo al reloj

de referencia del emisor son presentados en el instante Tr de acuerdo al reloj de referencia del receptor, la diferencia entre ambos:

$$D = Tr – Ts$$

D es el delay relativo captura-reprodución en segundos. Dado que los relojes de referencia en el emisor y receptor no están sincronizados, este delay (D) incluye un offset que es desconocido pero puede ser ignorado porque es común a través de todos los streams y aquí el interés es conocer el delay relativo entre streams.

Por lo tanto, una vez que se han determinado los delays relativos captura-reproduccón para los streams de audio y video, se puede obtener el **delay de sincronización** de la siguiente manera:

$$Dsynch = Daudio – Dvideo$$

Si el delay de sincronización es cero, entonces los streams están sincronizados; si no es igual a cero entonces significa que un stream está siendo reproducido antes que otro, y así el delay de sincronización provee el **offset relativo** en segundos.

Para el stream que está adelantado, el delay de sincronización en segundos se multiplica por la velocidad nominal del reloj del medio para convertirlo en unidades de timestamps del medio y luego aplicarlo como un offset constante al cálculo de la reproducción para ese stream, retardando la reprodución para que se corresponda al otro stream. Vale decir que el resultado es que los paquetes para un stream permanecen más tiempo en el playout buffer del receptor, para compensar ya sea el procesamiento más rápido en otras partes del sistema o el delay de red reducido, y son así presentados al mismo tiempo que los paquetes del otro stream.

El delay de sincronización debería ser recalculado cuando el delay de reproducción para cualquiera de los streams es ajustado, dado que cualquier cambio en el delay de reproducción afectará el delay relativo de los dos streams. Además debería recalcularse el offset cuando ocurra un nuevo mapeo entre el tiempo del medio y el tiempo de referencia, cuando se reciban paquetes RTCP SR.

Exactitud de la sincronización

Cuando se lleva a cabo la sincronización, hayq eu tener en cuenta la exactitud, y sobreviene la pregunta: cuál es el offset aceptable entre streams

que deban ser sincronizados ??? La respuesta no es simple dado que la percepción humana de la sincronización depende de qué es lo que se está sincronizando y de la tarea que se está nevando a cabo.

Por ejemplo, en general los requerimientos para la sincronización de los labios ente audio y video es relativamente flexible y varía con la calidad de video y la tasa de frames. Pero cuando se presentan imágenes de más alta calidad y vide con tasas de frames más altas, se hace más notable la ausencia de sincronización porque es más fácil observar el movimiento de los labios.

14.14.Tamaño de paquete

Ya en el punto 4 del presente trabajo se estudió con detalle los "jumbogramas" o "jumbo frames", que son paquetes de datos con tamaños mayores a los clásicos 1500 Bytes usados hoy en día en la mayoría de las redes e Internet para aumentar su performance. Cuando hablo de paquetes de datos me estoy refiriendo específicaente al nivel IP.

Recordemos que la recomendación de Internet2 es trabajar hacia un IP MTU "extremo-a-extremo" de por lo menos 9000 bytes por ahora, y eventualmente más grande. Con respecto a esto, Abilene actualmente trabaja con valores de IP MTU de 9180 Bytes y 9000 Bytes. Estos valores de MTU se usan tanto para IPv4 como para IPv6.

Los jumbo frames proveen casi 50% más de throughput con casi el 50% menos de carga de CPU que utilizando frames de 1500 Bytes. A pesar de que este overhead local puede reducirse en parte con diseños de sistemas mejorados, descarga de trabajo hacia las interfaces NIC, etc., lo que más importa es el enlace WAN.

En el presente caso de estudio, **adopto un tamaño de paquete IP de 9000 Bytes**, dado que con esto mejoro la performance de la transmisión de HDTV, y esto es deseado dado que lo que yo estoy proponiendo es enviar la multimedia a través de redes locales con tecnología Gigabit Ethernet y esto pone obviamente un límite en la velocidad máxima total de transmisión (datos más headers de los diferentes protocolos). No obstante los backbones de Abilene son de capacidad de 10 Gbps, el límite de la capacidad de transmisión lo impone aquí la tecnología de red de los extremos que está acotada a 1 Gbps.

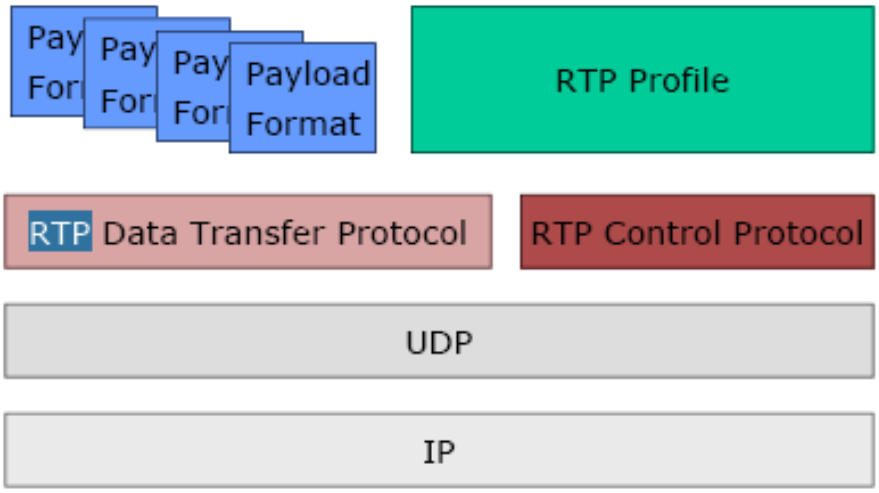
Al adoptar un tamaño de paquete IP de 9000 bytes, estoy aumentando la performance de la red, con lo cual quiero decir que voy a transmitir mayor

cantidad de datos en igual unidad de tiempo. En consecuencia, podré así enviar más información de HDTV sin necesidad de recortar la cantidad de muestras de componente de color o la tasa de frames por segundo, por ejemplo.

14.15.Velocidad de transmisión total requerida

Ya se mencionó anteriormente que tanto el emisor como el receptor se encuentran en redes extremas de tecnología Gigabit Ethernet, y también usan NICs de 1 Gbps. Por lo cual esta velocidad es el cuello de botella de la transmisión del streaming en vivo de HDTV sin comprimir.

En esta instancia hace falta recordar nuevamente el esquema de componentes de protocolos que entran en juego en la transmisión:



El **tamaño de paquete IP es de 9000 Bytes**, y por lo tanto incluye a los headers IP, UDP, RTP, perfil RTP y formato de payload RTP para cada stream de audio y video por separado. Los paquetes de control RTP se transmiten como un stream aparte para cada flujo.
Header IP = 20 Bytes

Header UDP = 8 Bytes

En el caso del video, el formato de payload RTP agrega 6 bytes por línea de video y el perfil TFRC agrega 8 Bytes u 4 Bytes (según esté presente o no el RTT) por paquete de datos.

Overhead máximo = 6 Bytes + 8 Bytes

En el caso del audio, el formato de payload RTP agrega 2 bytes y el perfil TFRC agrega 8 o 4 Bytes (según esté presente o no el RTT) por paquete de datos.

Overhead máximo = 2 Bytes + 8 Bytes

En el caso de RTP, el header tiene 12 Bytes (tendría 16 Bytes en el caso que usara el campo CSRC pero este no es el caso).

Header RTP = 12 Bytes

Como también se lo mencionó antes, el video es el tráfico con mayores requerimientos de ancho de banda. Originalmente se captura video SMPTE-292M a 1.485 Gbps, pero obviamente esta velocidad es imposible de transmitirla por enlaces de 1 Gbps. Por tal motivo es que el emisor al comienzo pasa de 10 bits de profundidad de color original a 8 bits, y así se obtiene una velocidad adecuada para la transmisión.

Cálculo de la velocidad de video SMPTE-296M:

720x1280 / 60 fps / 10 bits / 4:2:2

720*1280*60*20 = 1105920000 bps = 1.10 Gbps

Cálculo de la velocidad de video luego de reducir 10 bits a 8 bits la profundidad de color:

720*1280*60*16 = 884736000 bps = 884.7 Mbps

La velocidad máxima del audio AC-3 a 48 KHz es:

640 Kbps

La velocidad total de la transmisión será entonces:

Vel(total de la transmisión) = Vel(video) + Vel(audio)

Vel(video) = Vel(datos de video) + Vel(IP/UDP overhead) + Vel(RTP overhead) + Vel(overhead del formato de payload RTP y el perfil TFRC) + Vel(tráfico de control)

Vel(audio) = Vel(datos de audio) + Vel(IP/UDP overhead) + Vel(RTP overhead) + Vel(overhead del formato de payload RTP y el perfil TFRC) + Vel(tráfico de control)

Cálculo de los paquetes por segundo necesarios para transportar los datos de video:

El tamaño de paquete IP usado dijimos es de 9000 Bytes.

A dicho tamaño de paquete hay que restarle el overhead de IP, UDP, RTP, formato de payload RTP y perfil TFRC.

Por lo tanto, el tamaño del paquete usado para transportar sólo datos sería:

9000 Bytes – 20 Bytes – 8 Bytes – 12 Bytes - 6 Bytes - 8 Bytes = 8946 Bytes

El cálculo de los paquetes de datos de video por segundo sería así:

(720*1280*60*2) Bytes por paquete / 8946 Bytes = 12362.17 paquetes por segundo

Cálculo de la velocidad del IP/UDP overhead para el video:

En base a la cantidad de paquetes por segundo necesarios para transmitir los datos de video se calcula el IP/UDP overhead de la siguiente manera:

12362.17*28*8 = 2769126.08 bps = 2.76 Mbps

Cálculo de la velocidad del RTP overhead para el video:

Overhead RTP = 12 Bytes

12362.17*12*8 = 1186768.32 bps = 1.18 Mbps

Cálculo de la velocidad del overhead del formato de payload RTP y el perfil TFRC para el video:

Overhead máximo = 6 Bytes + 8 Bytes

12362.17*14*8 = 1384563.04 bps = 1.38 Mbps

Cálculo del ancho de banda de sesión para el video:

El ancho de banda de sesión comprende los datos y los protocolos de transporte y red, por lo tanto se calcula de esta manera:

Ancho de banda de sesión = 884.7 Mbps + 2.76 Mbps + 1.18 Mbps + 1.38 Mbps

Ancho de banda de sesión = 890 Mbps

Cálculo de la velocidad del tráfico de control para el video:

En el protocolo RTP especificado en la RFC 3550 se recomienda que el tráfico de control tenga una velocidad del 5% del ancho de banda de sesión, por lo tanto sería así:

$$5\% \text{ de } 890 \text{ Mbps} = \frac{5*890}{100} = 44.5 \text{ Mbps}$$

Cálculo de la velocidad total de video:

Vel(video) = Vel(datos de video) + Vel(IP/UDP overhead) + Vel(RTP overhead) + Vel(overhead del formato de payload RTP y el perfil TFRC) + Vel(tráfico de control)

Vel(video) = 890 Mbps + 2.76 Mbps + 1.18 Mbps + 1.38 Mbps + 44.5 Mbps

Vel(video) = 939.82 Mbps

Cálculo de la velocidad de audio:

En el caso del audio, hay que tener en cuenta la velocidad máxima de transmisión de los datos (640 Kbps), la velocidad del IP/UDP overhead, la velocidad del RTP overhead, la velocidad del formato de payload RTP (agrega 2 bytes) y el perfil TFRC (agrega 8 o 4 bits según esté presente o no el RTT) y la velocidad del tráfico de control. Siempre se toma en cuenta que el tamaño del paquete IP es de 9000 Bytes.

Todo este cálculo está en el orden de 1 Mbps, un tráfico muy pequeño comparado con el tráfico de video HDTV.

Cálculo de la velocidad total de la transmisión:

Vel(total de la transmisión) = Vel(video) + Vel(audio)

Vel(total de la transmisión) = 939.82 Mbps + 1 Mbps = 940.82 Mbps

Vel(total de la transmisión) = 941 Mbps

Por lo tanto, se puede concluir que la velocidad total de la transmisión del streaming en vivo de HDTV sin comprimir es menor a 1 Gbps, que es el valor del cuello de botella de los enlaces y las NIC usadas en esta implementación. Por lo tanto, encuanto a la velocidad de transmisión, la implementación es factible de ser llevada a cabo.

14.16.Corrección de errores

Según las experiencias en el mundo con TFRC, si se utiliza este mecanismo de control de congestión en redes IP de última generación como lo es Internet2, las pérdidas van a ser mínimas ya que inmediatamente después de una pérdida, la tasa de transmisión disminuye de manera importante, reduciendo rápidamente la congestión y normalmente no hay más pérdidas.

Por lo tanto, se podrían usar retransmisiones ARQ (Automatic Retransmissions reQuest) para recuperar esos paquetes perdidos, dado que es importante decir que los paquetes perdidos pueden ser recuparados si el receptor le pide al emisor que los retransmita. La retransmisón es una aproximación natural a la corrección de errores y trabaja bien en determinados escenarios.

En el esquema de retransmisiones ARQ, el receptor detecta errores y pide al transmisor que lleve a cabo las retransmisiones correspondientes. Este método no es muy recomendado para aplicaciones en tiempo real dado que el tiempo que se pierde en el renvío puede ser considerable, pero dado que aquí se usa streaming en vivo con un jitter buffer de varios segundos, entonces no habría problema en hacer uso del mismo.

La retransmisión no es parte del estándar RTP, pero sin embargo existe un draft del perfil RTP para las retransmisiones [104] que provee un marco de trabajo basado en RTCP para los pedidos de retransmisiones y otros tipos de feedbacks.

Otra alternativa de método de corrección de errores evaluada es el FEC (Forward Error Correction) que consiste en corregir los errores (incluído la

pérdida de paquetes) en el extremo receptor. Antes de la transmisión, los datos son procesados con un algoritmo que agrega bits de información extras para que el receptor pueda recuperar los paquetes recibidos con error. Este método controla los errores en un canal de comunicación “one-way”, dado que el receptor no le pide nada al transmisor cuando encuentra errores en los paquetes que recibe, sino que simplemente con la información extra recupera dichos paquetes. Si por ejemplo tengo una transmisión de HDTV a 1 Gbps, usando FEC es muy probable –en forma aproximada- que ese throughput se eleve a 1,150 Gbps, con lo cual se necesitaría un ancho de banda bastante mayor y eso no es adecuado si se trata de usar tecnología Gigabit Ethernet en los extremos con velocidades máximas de 1 Gbps, y justamente el objetivo de este trabajo es acomodar el stream de HDTV a dicha velocidad.

Como conclusión del uso de técnicas de control de errores para el presente trabajo, en principio no voy a hacer uso de retransmisiones ARQ porque tomo como hipótesis válida que la tasa de error en la transmisión es muy baja y además el streaming de audio/video es tolerante a un error muy bajo, tal como ocurre en Abilene. Pero desde ya no se descarta su uso, más aún si la transmisión de HDTV se realiza a mayor velocidad, usando por ejemplo tecnología OC-48c o superior extremo a extremo.

14.17.Descripción del funcionamiento de la implementación

A continuación se describe el funcionamiento del sistema de transmisión de streaming en vivo de HDTV sin comprimir sobre redes IP “best-effort”, desde un emisor hacia un receptor a través de Internet2 y con redes en los extremos de 1 Gbps de velocidad. En esta instancia intercambié diferentes ideas con los miembros del grupo de trabajo del Audio Video Transport Working Group del IETF a través de la lista de discusión abierta al público [105], los cuales han compartido este mecanismo en principio.

Desde el emisor se transmiten dos streams independientes de audio y video, ambos con su correspondiente formato de payload RTP y su perfil TFRC, y además en el receptor de monitorean constantemente el estado de ocupación de los playout buffers de audio y video para ajustar en forma discreta la tasa de muestreo del audio y la tasa de frames del video.

Teniendo en cuenta que a causa del monitoreo del buffer de video la velocidad de la aplicación saltará entre los valores [60 KHz, 50 KHz, 40 KHz, 30 KHz], es interesante hacer notar de qué velocidades estamos hablando. Para ello haremos el cálculo correspondiente a cada tasa de frames de video.

Velocidad de los datos con 8 bits de profundidad de color y a:

60 fps ----- 720*1280*16*60 = 884.7 Mbps
50 fps ----- 720*1280*16*50 = 737.2 Mbps
40 fps ----- 720*1280*16*40 = 598.8 Mbps
30 fps ----- 720*1280*16*30 = 442.3 Mbps

El sistema funciona de la siguiente manera:

1.- Comienza la transmisión de HDTV, se recibe el video desde una cámara de video HDTV a 1.485 Gbps (full rate) 720p con 10 bits de color y 60 fps, y se recibe al mismo tiempo el audio AC-3 desde su correspondiente dispositivo muestrado a 48 KHz.

2.- El extremo emisor la aplicación convierte el video de 10 bits a 8 bits en forma irreversible para acomodar la velocidad de transmisión a menos de 1 Gbps, y aplica inmediatamente control de congestión TFRC, enviando paquetes de datos por el enlace IP con 8 bits de color de video. La velocidad máxima de la aplicación se dará con un valor de 60 fps en el caso que las no haya pérdidas.

3.- Al aplicar TFRC, comenzará la fase de slow-start hasta que ocurra la primera pérdida y luego la aplicación adaptará su velocidad de transmisión a la velocidad calculada por la ecuación del throughput TCP usada por TFRC, comenzando desde la velocidad registrada en el instante anterior a dicha pérdida. La aplicación adaptará su velocidad variando la tasa de frames del video únicamente.

4.- Luego de 10 segundos y teniendo en cuenta el offset de sincronización de audio y video, comienza a reproducirse la multimedia en el receptor.

5.- En el extremo receptor se monitorea en forma continua el estado de ocupación de los buffers de audio y video.

6.- En el caso del video se especifica que, a consecuencia del monitoreo del correspondiente buffer, la tasa de frames variará entre 60 fps (máximo) y 30 fps (mínimo) en escalones discretos de 10 fps. No se toman valores menores para no bajar mucho la calidad de video que percibe el usuario final.

7.- Inicialmente si el nivel del buffer de video no llega a vaciarse, la aplicación seguirá transmitiendo a la velocidad fijada por TFRC.

8.- En el caso del video defino que, según sea la velocidad actual de transmisión, al ocurrir un buffer underflow la tasa de frames descenderá al valor de fps inmediatamente por debajo tomado entre los valores [60, 50, 40, 30]. Y por otro lado, defino que si luego el buffer se llena nuevamemente, la tasa de frames ascenderá al valor de fps inmediatamente por encima tomado también entre los valores [60, 50, 40, 30].

9.- Suponiendo que luego del slow-start impuesto por TFRC la aplicación está transmitiendo a 48 fps, si ocurre un "underflow" (el buffer está vació o muy cerca de estar vacío), aplico reducción de frames por segundo al video a 40 fps, que sería el escalon de fps definido inmediatamente por debajo del valor actual.

10.- Si el buffer alcanza luego el nivel de lleno, entonces dependiendo de la velocidad de transmisión actual impuesta por TFRC, incremento a 60 fps (en el caso que la velocidad actual estuviese entre 55 fps y 60 fps) o incremento a 50 fps (en el caso que la velocidad actual estuviese entre 45 fps y 50 fps).

11.- En el caso que el buffer hubiera sobrepasado el nivel anterior a vacío sin llegar a alcanzar el nivel de lleno, y luego se hubiera vaciado nuevamente, entonces la tasa de frames por segundo hubiese bajado al valor por debajo. Si por ejemplo la velocidad de transmisión actual hubiese sido de 53 fps, la aplicación transmitirá ahora a 50 fps.

12.- Si ahora ocurre lo mismo que en el paso anterior y suponiendo que la velocidad actual sea de 48 fps, la tasa de frames por segundo baja esta vez a 40 fps.

13.- Si ahora el buffer de video se va llenando poco a poco hasta alcanzar el nivel de buffer lleno, entonces la tasa de frames por segundo aumenta al valor superior a la velocidad actual.

14.- Si el buffer de video varía su nivel de ocupación desde lleno, bajando más allá del nivel anterior a lleno y luego alcanzando nuevamente el nivel de lleno, la tasa de frames por segundo aumenta nuevamente al valor predefinido por encima de la velocidad actual.

15.- La variación de la tasa de frames de video sigue esta lógica a lo largo de la transmisión de HDTV.

16.- En el caso del audio, dado que AC-3 es un CODEC que maneja diferentes tasas de frames, se comienza transmitiendo a la máxima

velocidad que es de 640 Kbps. Voy a tomar tres tasas de bits diferentes del CODEC para ajustar durante la transmisión.

17.- Si el nivel del buffer no llega a vaciarse, se sigue transmitiendo a la misma tasa de bits.

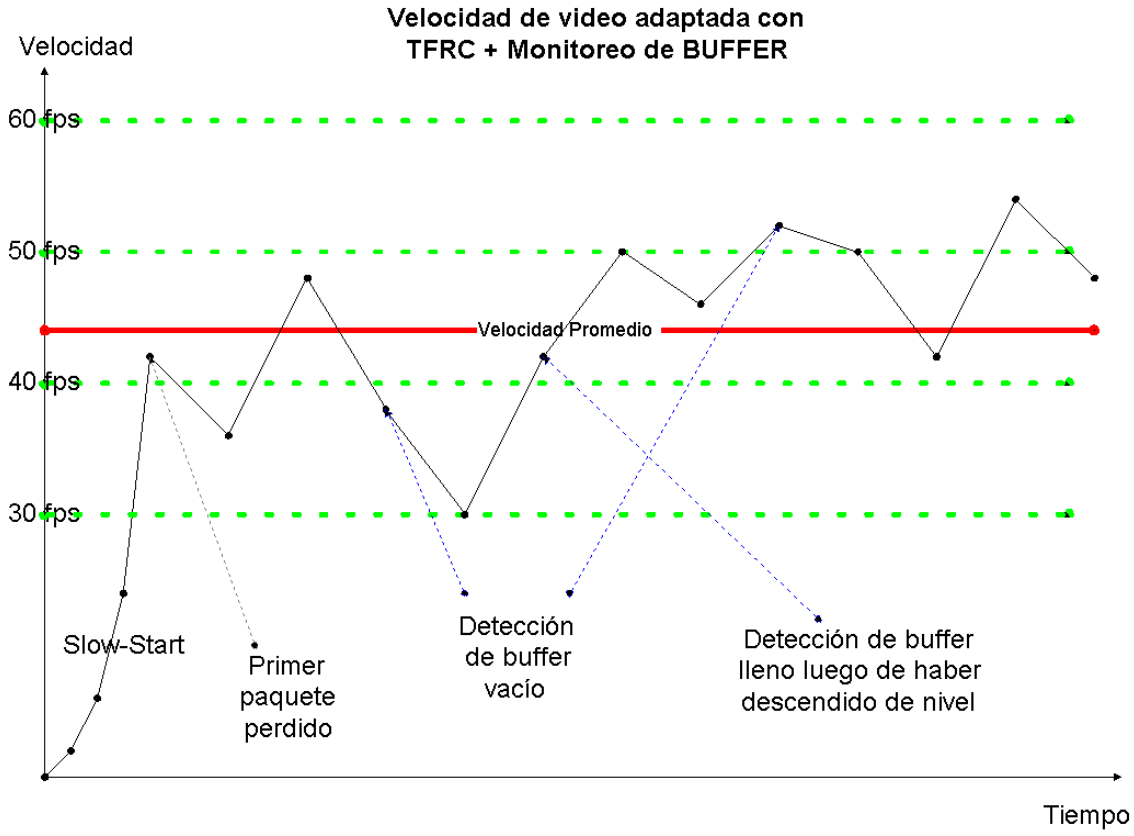
18.- Si ocurre un "underflow" (el buffer de audio está vació o muy cerca de estar vacío), el extremo receptor le dará la orden al emisor que aplique reducción de la tasa de bits

19.- En el caso que el buffer hubiera sobrepasado el nivel anterior a vacío sin llegar a alcanzar el nivel de lleno, y luego se hubiera vaciado nuevamente, entonces la tasa bits hubiera bajado a valor menor.

20.- Si el buffer de audio alcanza luego el nivel de lleno, entonces incremento la tasa de bits al valor superior.

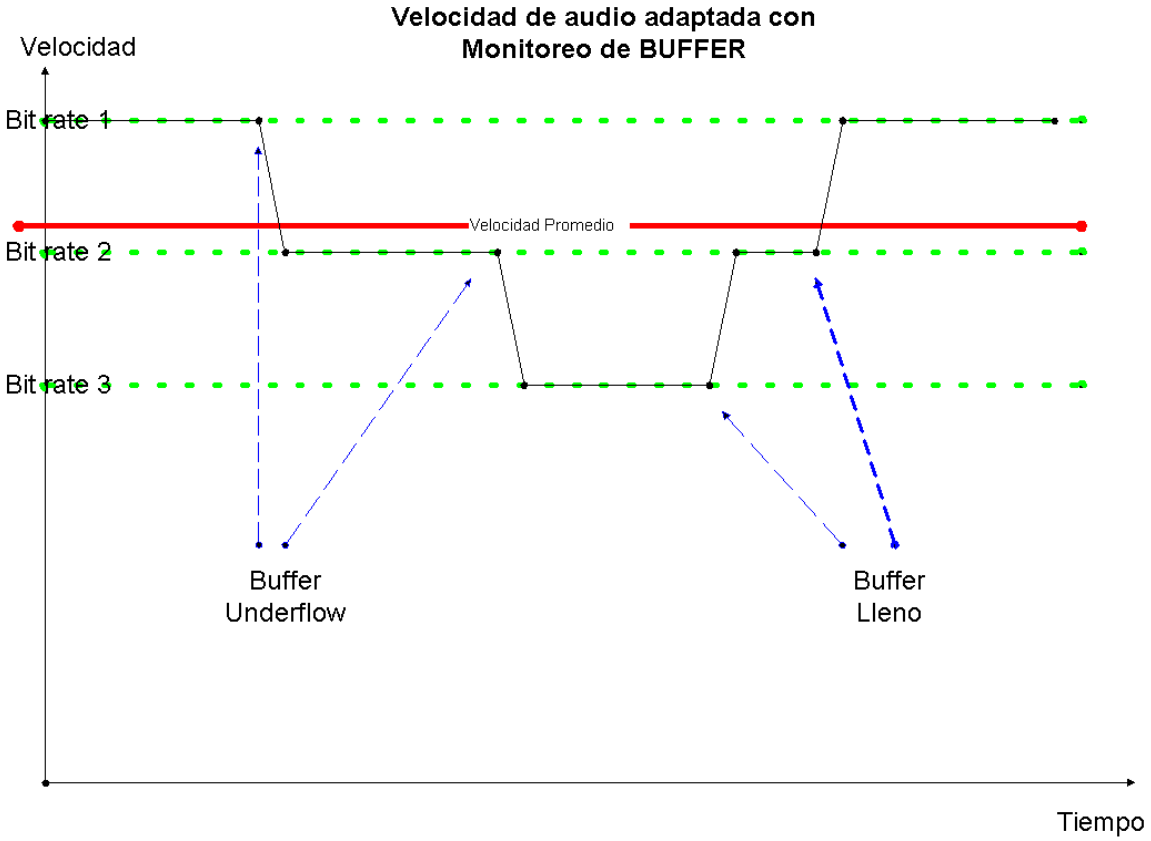
21.- La variación de la tasa de muestreo de audio sigue esta lógica a lo largo de la transmisión de HDTV.

Finalmente, las curvas de velocidades serían las siguientes:



En la fase de “slow-start” se incrementa la velocidad comenzando por un paquete por segundo y luego se va duplicando cada vez que recibe un paquete de feedback cada RTT (la escala de tiempo en el gráfico no se corresponde en esta fase). Luego de que tiene lugar la ocurrencia del primer paquete perdido, la velocidad es adaptada con la ecuación del throughput de TFRC. En esta fase, la velocidad cambia de acuerdo al ritmo de recepción de los paquetes de feedback, cuyo ideal es de al menos una vez por RTT o por paquete de datos recibido (basado en el intervalo de tiempo más largo, en este caso claramente el RTT de la red es mayor que tiempo entre paquetes que es del orden de 10e(-9)). Estos requerimientos son para asegurar una reacción a tiempo ante la congestión.

Es importante recordar aquí que los RTT’s mínimos en promedio en Internet2 están en el orden de los 14-60 mseg, definidos principalmente por el tiempo de propagación de la luz por la fibra óptica.



El audio cambia de velocidad de acuerdo al feedback de la medición del estado de ocupación del buffer de playout, y esas velocidades son tienen una misma freceuncia de muestreo pero diferente cantidad de palabras por sframe de sincronización, tal cual lo ofrecen las implementaciones del CODEC AC-3.

14.18.Limitaciones de la implementación

- Una vez que hago downsampling de la profundidad de color de video de 10 bits a 8 bits en el extremo servidor de la aplicación, no voy a poder volver atrás haciendo un “upsampling”. Se transmitirá HDTV con 8 bits por muestra de componente de color.

- Baja de la velocidad del flujo de video controlado por TFRC a causa del reordenamiento de paquetes y paquetes que arriban tarde.

- El feedback RTCP al cumplir con la condición de ser un paquete por RTT, hace que el tráfico de control sea elevado y con mucho requerimiento de procesamiento de CPU.

- Existen temas relativos a la adaptación de la tasa de frames por segundo, dado que puede causar una visible pérdida de calidad de la imagen de HDTV en la pantalla del usuario final. Pero sin embargo la experiencia con otras formas de video (por ej. JPEG) demuestra que tener una baja pero estable tasa de frames por segundo es mejor que tener una tasa promedio mayor pero errática. Y además esta tasa varía sólo si el buffer se vacía, lo cual es un evento poco probable si trabajamos en Internet2.

- Con el audio ocurre lo mismo, el usuario final puede llegar a percibir los cambios de la tasa de bits debidos al disparo de la reducción o aumento de la misma dada por el monitoreo del buffer de playout.

- En cuanto a TFRC, los algoritmos para calcular la tasa de eventos de pérdida pueden no llegar a ser muy eficientes. En su lugar, las aplicaciones que hacen uso de TFRC no miden en forma directa la tasa de eventos de pérdida, y en su lugar cuentan el número de paquetes perdidos a lo largo de cada intervalo de reporte RTCP e incluyen luego ese número en los paquetes RTCP RR como una fracción de las pérdidas. Hoy en día no está muy claro en los círculos internacionales de investigadores si esta medición es correcta para un algoritmo de control de velocidad amigable con TCP, pero en principio es aceptable para probar. En verdad el objetivo del uso de TFRC no es ser exactamente equitativo con las conexiones TCP, sino serlo de alguna manera, no causar congestión y que de esta forma sea apto para ser usado por las aplicaciones de audio y video.

- La implementación del slow-start es dificultosa para una aplicación que transmite HDTV sin comprimir a muy alta velocidad, razón por

la cual su implentación es muy dificultosa. Una manera de realizar esto igualmente es, durante el slow-start, enviar “dummy data” (datos mudos) hasta que se alcance una velocidad de transmisión sustentable. Y además se impone la condición de que haya un feedback cada RTT para que el slow-start responda de manera similar a TCP que tiene una mayor frecuencia de ACK’s.

- El “slow-start” es simple de implementar para TCP porque tiene un feedback inmediato cuando ocurre alguna pérdida, pero es más difícil de implementar con un protocolo basado en RTCP que normalmente envía feedbacks a intervalos más largos de tiempo.

- Se puede llegar a establecer un cuello de botella en los hosts extremos de la aplicación, dada la alta carga de paquetes, los problemas inherentes a la velocidad de los buses PCI y la capacidad de procesamiento.

14.19.Comparación con el experimento de Tektronix

Tektronix realizó su experimento exitoso en el año 2001 desarrollando una solución propietaria para transmitir full HDTV sin comprimir a aproximadamente 1.5 Gbps.

En el caso de Tektronix, no se detalla –y tampoco en ningún otro experimento de los citados en esta tesis- si se envió también audio o no junto con el video HDTV. Si así hubiese sido, una posibilidad pudo haber sido enviar el audio embebido con el streaming de video que utilizó el formato de payload RTP para video SMPTE-292M.

En el experimento de Tektronix se empaquetó el video usando el formato de payload RTP para video SMPTE-292M, el cual requiere que la velocidad de transmisión sea constante y no esté subordinada a ningún mecanismo de control de congestión. Por lo tanto el uso de este formato de payload implica el uso de redes estrictamemente controladas o que hagan uso de algún mecanismo de reservación de recursos para garantizar la calidad de servicio. Además, el uso de este formato de payload requiere de un ancho de banda de 1.485 Gbps dedicado sólo a los datos de video; el overhead IP/UDP/RTP requiere de un ancho de banda extra. Por otro lado, el uso de este formato de payload en redes IP “best-effort” requiere que el receptor RTP deba monitorear la pérdida de paquetes para asegurar que la misma esté dentro de los parámetros aceptables (se considera que la pérdida de paquetes es aceptable si un flujo TCP a través del mismo camino de red y experimentando las mismas condiciones de red, alcanza un

throughput promedio que no es menor que el alcanzado por el flujo RTP); si la pérdida de paquetes es muy alta entonces el receptor debe abandonar la sesión.

En el caso de la presente implementación, se usa el formato de payload para video sin comprimir que incluye diversos formatos de video y extensible a otros nuevos a medida que se vayan desarrollando. Este formato de payload RTP define un esquema de paquetización del video sin comprimir para ser transportado en paquetes RTP. Si la transmisión del video HDTV sin comprimir con este formato de payload RTP se usa en una red IP “best-effort”, se requiere que el receptor deba monitorear la pérdida de paquetes para asegurar que la misma esté dentro de los parámetros aceptables; si la pérdida de paquetes es muy alta entonces el receptor o bien deberá abandonar la sesión o de otra manera deberá usar un mecanismo de control de congestión para adaptar la velocidad de transmisión. En el caso de la presnete implementación, se optó por usar el control de congestión TFRC junto con este formato de payload, y así se obtiene un marco de trabajo absolutamente compatible.

Por lo tanto, si hacemos una comparación entre los formatos de paylaod RTP usados en las implementaciones de Tektronix y en la actual se puede decir brevemente lo siguiente:

- La RFC 3497 empleada en el experimento de Tektronix define un formato de payload RTP para encapsular video SMPTE-292M. El video SMPTE-292M define una interface digital bit-serial para el transporte en área local de HDTV. SMPTE-202M utiliza palabras de 10 bits y una velocida fija de 1.485 Gbps. SMPTE-292M es usado típicamente en la industria de broadcast para transporte de otros formatos de video tales como SMPTE-260, SMPTE-295, SMPTE-274 y SMPTE-296. La RFC 3497 define una emulación de circuito para el transporte de video SMPTE-292M sobre RTP. El formato de payload es muy específico para SMPTE-292M y ha sido diseñado para ser interoperable con el equipamiento de broadcast existente empleando la velocidad constante de 1.485 Gbps.

- La RFC 4175 empleada en la actual implementación define un esquema de paquetización nativo flexible que puede paquetizar cualquier video sin comprimir a velocidades de dato variables. Además, en oposición a la RFC 3497, este formato de payload sólo transporta pixels de vidoe activo.

También se puede decir que la actual implementación sería más flexible dado que usa control de congestión para transportar el video HDTV sobre una red IP “best-effort” y así regula su velocidad de transmisión para compartir en forma equitativa el ancho de banda disponible en la red y coexistir así con los demás flujos. Sería por lo tanto una solución más realista y de alguna manera se puede proyectar a la Internet del futuro para coexistir con diferentes tipos de tráfico.

En cuanto a la transmisión unicast, tanto Tektronix como esta implementación propuesta operan bajo ese mismo concepto y en ese sentido no habría discordancia.

El experimento de Tektronix asegura la transmisión de video HDTV sin comprimir a una velocidad fija de 1.485 Gbps, lo cual es excelente para el usuario final, pero a costa de emplear hardware propietario como lo es el UNAS e interfaces de red y tecnologías de red en los extremos más rápidas que la tecnología Gigabit Ethernet. En consecuencia, es una solución muy costosa.

En cambio, la implementación actual puede operar con tecnología estándar de 1 Gbps, pero a costa de perder calidad de video y audio en el receptor de la aplicación.

En implementaciones como estas, es importante hacer notar que el video sin comprimir tiene grandes requerimientos de ancho de banda, razón más que suficiente para causar una potencial denegación de servicio en las redes que atraviesa si no tienen un ancho de banda adecuado, no tienen control,de congestión o no operan bajo monitoreo de tráfico.

Finalmente, comparando el escenario planteado por esta tesis con otras implementaciones a nivel mundial, se puede decir que en la Web hay diversa información sobre experiencias de transmisión de video HDTV sobre redes IP, pero no hay información sobre experiencias de transmisiones de audio y video HDTV en forma conjunta. La presente tesis ofrece como valor agregado a la transmisión de HDTV, la consideración de la transmisión de audio AC-3 utilizando un mecanismo de monitoreo constante del buffer de recepción para variar el throughput de audio de a saltos discretos y así llegar a controlar la congestión.

14.20.Mejoras sugeridas

A continuación se detallan algunas mejoras sugeridas para emplear en la transmisión de HDTV sin comprimir sobre Internet2:

1.- Se puede emplear el mecanismo de corrección de errores con retransmisiones ARQ, dado que al tener buffers de audio y video de varios segundos esto es posible.

2.- Estudiar algún mecanismo que posibilite separar las pérdidas de paquetes del reordenamiento de paquetes, y así la ecuación TFRC para el control de congestión respondería en forma fiel sólo a la congestión de la red.

3.- En el caso que se quiera transmitir HDTV sin comprimir con velocidad SMPTE 296-M (1.1 Gbps de video) extremo a extremo, habría que analizar las siguientes soluciones:

3.1.- Usar dos NIC Gigabit Ethernet en cada extremo y conectarlas a un switch o router a 1 Gbps. El esquema general sería así:

La aplicación en el extremo emisor separa los datos entre las interfaces de salida y transmite hacia dos diferentes IP's. Por su parte la aplicación en el extremo receptor recibe los datos desde las dos interfaces y luego los ensambla. Pero los potenciales riesgos son que cada uno de los flujos competirá contra el otro por ancho de banda y además se incrementaría el porcentaje de paquetes fuera de orden.

3.2.- Usar una NIC 10 Gigabit Ethernet en cada extremo y conectarlas a un switch o router a 10 Gbps. Obviamente esta solución es más onerosa y habría que contar con redes de gran velocidad en los extremos para que no se produzcan cuellos de botella.

4.- Implementar el sistema en una red con QoS, o un mix de "best-effort" y Qos en la cual la HDTV se transmita de una u otra manera según las características del tramo de red que atraviesa.

5.- Si se quiere lograr que la velocidad de transmisión de la aplicación de HDTV sin comprimir sea siempre cercana a la máxima permitida por la aplicación, entonces hay que pensar en algún esquema de reservación de recursos o diferenciación de servicios, para lo cual hace falta la cooperación de los routers intermedios. Recordemos que esta solución fue descartada en el presente trabajo debido principalmente a que no hay un gran consenso mundial para implementar QoS en Internet y además es una solución muy costosa, más que implementar el overprovissioning que también sería otra solución adecuada y sobre la cual se han realizado muchos experimentos en este campo de la transmisión de tráfico de ultra alta velocidad.

15.Conclusiones

La transmisión de streaming en vivo de HDTV sin comprimir es posible, siempre y cuando se la realice sobre redes de ultra alta velocidad como Internet2.

Según sean las tecnologías e interfaces de red empleadas en los extremos y el mecanismo por el cual se asegure un transporte a muy alta velocidad, se puede difundir audio y video HDTV sin comprimir con diferentes calidades de recepción.

En cuanto al mecanismo de control de congestión TFRC, todavía resta mayor experiencia mundial para llegar a conclusiones más exactas en cuanto a su implementación y operación, y así definirlo finalmente como estándar.

Dados los avances tecnológicos en lo que hace a poder de procesamiento, diseño de integrados, velocidad de enlaces y dispositivos afines, es que se espera que no pasen muchos años para poder trasladar parte de esta tecnología a las personas y empresas que necesiten de este tipo de tráfico tan interesante de transportar como lo es la televisión digital de alta definición sin comprimir y su audio asociado.

En lo que respecta a mi trabajo de investigación, el mismo fue una experiencia verdaderamente rica porque me permitió adentrarme en aspectos muy interesantes de la tecnología de redes y los protocolos de transmisión asociados para el audio y video de muy alta velocidad. Como punto importante debo destacar que este trabajo me permitió estar en contacto con expertos mundiales de varios países que están a la vanguardia de la transmisión de audio y video, pertenecientes a diferentes instituciones

como la Universidad de Washington, el Research Channel, el grupo de trabajo de Audio/Video de la IETF, el Information Sciences Institute de la Universidad del Sur de California, Abilene, el grupo de trabajo de QoS de Internet2 y diversas redes miembros de Internet2 –entre las más importantes-. Todos ellos me han transmitido principalmente su experiencia con tráficos multimedia sobre Internet2 o redes similares, y me han expresado sus diferentes puntos de vista en cuanto a la toma de decisiones a la hora de implementar sus sistemas. Específicamente, con el grupo de trabajo de Audio/Video de la IETF y los expertos del Information Sciences Institute intercambié los conceptos básicos para la implementación del control de congestión experimental TFRC sobre el tráfico de video, sobre los cuales desarrollé un eje importante en mi tesis sumándole además el transporte asociado de audio con técnicas de buffering para mantener la sincronización de los flujos.

Finalmente, y por todo lo antes dicho, espero que este trabajo de tesis sea un material de interés para todo aquel que desee trabajar o conocer el universo de la transmisión de HDTV a través de redes IP.

16.Documentación

[1] Advanced Television System Comittee - ATSC, http://www.atsc.org. Es el lugar obligado para comenzar a leer sobre los estándares de HDTV.
[2] Society of Motion Pictures and Television Engineers - SMPTE, http://www.smpte.org. De aquñi saqué la información necesaria sobre el estándar SMPTE-292M, el cual lo uso para llevar a cabo el transporte con paquetes RTP.
[3] International Telecommunication Union - ITU, http://www.itu.int
[4] Movie Picture Expert Group - MPEG, http://www.mpeg.org
[5] AC-3 Revision B – 14 June 2005, http://www.atsc.org/standards/a_52b.pdf
[6] Internet2, http://www.internet2.edu. Es el sitio web oficial de Internet2 y información sobre diferentes áreas de investigación y desarrollo, junto con un listado completo de los grupos de trabajos actuales con sus respetivas páginas web y direcciones de contacto.
[7] Abilene, http://abilene.internet2.edu. Aquí se detallan los datos relevantes del backbone Abilene, así como también mapas, topologías, interfaces web para consultar los estados actuales de diferentes dispositivos e información de contacto.
[8] Defense Research and Engineering Network, http://www.hpcmo.hpc.mil/Htdocs/DREN/index.html
[9] The Energy Sciences Network, http://www.es.net
[10] NASA Integrated Services Network, http://www.nisn.nasa.gov
[11] NASA Research and Education Network, http://www.nren.nasa.gov
[12] Very High Performance Backbone Network Service, http://www.vbns.net
[13] CANARIE, http://www.canet3.net. En esta red se han llevado a cabo otros experimentos con HDTV sobre IP, usando un software propietario que establece caminos de luz (lightpaths) extremo a extremo en forma dinámica para proveer ancho de banda dedicado y baja latencia.
[14] Asia Pacific Advanced Network, http://www.apan.net
[15] NORDUnet, http://www.nordu.net
[16] TERENA, http://www.terena.nl
[17] Géant, http://www.geant.net
[18] Paul Ferguson and Geoff Huston, “Quality of Service”, Wiley, 1998
[19] RETINA, http://www.retina.ar.
[20] Proyecto RETINA2, http://www.retina.ar/retina/retina2/index.htm. Sitio web con información sobre RETINA2, la red argentina conectada a Internet2. Se puede encontrar información sobre diferentes proyectos interesantes.
[21] AmPath – Americas Path, http://www.ampath.fiu.edu
[22] Global Crossing, http://www.globalcrossing.com

[23] Abilene Networks Operations Center, http://www.abilene.iu.edu. Este sitio ofrece alguna información complementaria al sitio web de Abilene ubicado dentro del sitio web oficial de Internet2.
[24] American Nacional Standards Institute, http://www.ansi.org
[25] IEEE P802.3ae 10 Gbps Ethernet Task Force, http://grouper.ieee.org/groups/802/3/ae/
[26] Next Generation Abilene Router Node, http://loadrunner.uits.iu.edu/upgrade/docs/rack_design-0715.doc
[27] Worst Case - Abilene One-Way Latency Statistics, http://abilene.internet2.edu/ami/owamp_worst_case.html. Este sitio es interesante porque provee información de estadísticas de latencias en Abilene –tanto con IPv4 como con IPv6-. Todas estas estadísticas toman en cuenta las "peores" latencias, es decir, las más grandes y así uno puede tener una idea de las peores condiciones de la red con respecto a este parámetro.
[28] Internet2 Net Flows – Weekly Reports, http://netflow.internet2.edu/weekly/20051024/#xputs. Este sitio muestra información de reportes semanales de los flujos que atraviesan Internet2, con distribuciones de protocolos usados, tamaños de paquetes, throughputs, etc.
[29] Abilene – Jumbo frames and Internet2, http://www.abilene.iu.edu/JumboMTU.html. Esta página de Abilene muestra información muy importante acerca de los jumbo frames, con lecturas recomendadas, ejemplos de configuración de los routers Cisco y Juniper y dos listas de mail para intercambiarc información.
[30] MTU Mailing List, mtu@psc.edu. Esta lista de mail tiene mucho movimiento y está integrada por diversos expertos delñ mundo, muchos de los cuales pertenecen a importantes empresas de EE.UU. y Europa y a diferentes universidades e instituciones miembros de Internet2. Mi participación en esta lista me sirvió para obtener información de los tamaños de jumbo frames usados y sus ventajas y desventajas.
[31] RFC 1191, "Path MTU Discovery", J.C. Mogul and S.E. Deering, Nov-01-1990, Standard Draft
[32] RFC 2923, "TCP Problems with Path MTU Discovery ", K. Lahey, September 2000, Informational
[33] "Gigabit Ethernet Jumbo Frames and why you should care", Phil Dykstra, WareOnEarth Communications Inc. http://sd.wareonearth.com/~phil/jumbo.html. Es un documento básico para entender la conveniencia del uso de los jumbo frames.
[34] Internet2 Net Flow – Contiene muestras del flujo que viaja por el backbone Abilene de Internet2, con reportes diarios y semanales, http://netflow.internet2.edu/weekly/20051024/#xputs

[35] “The macroscopic behavior of the TCP congestion avoidance algorithm”, Matthew Mathis, Jeffrey Semke, Jamshid Mahdavi, and Teunis Ott, Computer Communications Review, volume 27, number 3, July 1997.
[36] Abilene Core Node Router Proxy, http://ratt.uits.iu.edu/routerproxy/abilene/. Esta es una interface web para ejecutar comandos del tipo “show ip”, “traceroute”, “ping”,”show bgp”, etc., contra los core routers de Abilene y obtener así información actual. Me sirvió principalmente para comprobar el tamaño de los jumbo frames configurados ebn las interfaces de los routers.
[37] “Modified TCP Congestion Avoidance Algorithm”, V. Jacobson
[38] RFC 2581, “TCP Congestion Control”, M. Allman, V. Paxon and W. Stevens, Apr. 1991, Proposed Standard
[39] Iperf, http://dast.nlanr.net/Projects/Iperf. Esta herramienta tiene mucha utilidad para medir parámetros de red como latencia y jitter, importantes a la hora de diseñar las aplicaciones.
[40] RFC 3550, “RTP: A Transport Protocol for Real-Time Applications”, H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, July 2003, Standard. Esta RFC es el corazón del transporte de la multimedia, y es el estándar sobre el cual basé mi trabajo.
[41] RFC 793, “Transmission Control Protocol”, J. Postel, Sep-01-1981, Standard
[42] RFC 768, “User Datagram Protocol”, J. Postel, Aug-28-1980, Standard
[43] RFC 791, “Internet Protocol”, J. Postel , Sep-01-1981, Standard
[44] RFC 1889, “RTP: A Transport Protocol for Real-Time Applications”, Audio-Video Transport Working Group, H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, January 1996, Proposed Standard
[45] RF 1890, “RTP Profile for Audio and Video Conferences with Minimal Control”, Audio-Video Transport Working Group, H. Schulzrinne, January 1996, Proposed Standard.
[46] RFC 3551, “RTP Profile for Audio and Video Conferences with Minimal Control”, H. Schulzrinne, S. Casner , July 2003, Standard
[47] “RTP: audio and video for the Internet”, Colin Perkins, Addison Wesley, June-2003. Este libro es un referente mundial para todos los expertos que se dedican al transporte de audio y video por redes IP. El autor es miembro del IETF y posee una gran experiencia técnica.
[48] Universidad de Washington – HDTV, http://www.washington.edu/hdtv. De aquí obtuve la información inicial con la cual comencé a investigar las primeras experiencias mundiales con HDTV sobre IP. Es un sitio de lectura obligada para todo aquel se que se adentre en el mundo de HDTV/IP.
[49] Universidad de Washington, http://www.washington.edu

[50] Research Channel, http://www.researchchannel.org. Sitio con información sobre diferentes proyectos orientados al audio y video, en el cual van publicando todos los avances de sus experimentos e invitan a compartirlos en el caso de poseer conexiones de muy alta velocidad.
[51] Sony, http:// www.sony.com
[52] National Association of Broadcasters, http://www.nab.org
[53] 2NetFX, http://www.2netfx.com
[54] RFC 2250, RTP Payload Format for MPEG1/MPEG2 Video, January 1998 – D. Hoffman, G. Fernando, V. Goyal y R. Civanlar
[55] Tektronix, http://www.tektronix.com. De aquí saqué principalmente la experiencia de HDTV/IP contra la cual cotejé mi trabajo de investigación.
[56] First Uncompressed Real-time Gigabit HDTV Transmission Across Wide Area IP Network Made Possible by Tektronix, University of Washington, USC/ISI and Level 3, http://www.tek.com/Measurement/cgi-bin/framed.pl?Document=/Measurement/Products/press/hdtv/&FrameSet=optical
[57] VACANTE
[58] Pacific Northwest GigaPoP, http://www.pnw-gigapop.net
[59] Level3, http://www.level3.com
[60] NTT Uncompressed HDTV Transmission System Over the Internet, http://www.ntt.co.jp/news/news01e/0110/011026.html
[61] NTT, http://www.ntt.co.jp
[62] Fuji Televisión Network, http://www.fujitv.co.jp
[63] I-Visto, http://www.i-visto.com/
[64] “Experiments with Delivery of HDTV over IP Networks”, Colin Perkins, Ladan Gharai, Tom Lehman, Allison Mankin, USC Information Sciences Institute, http://www.east.isi.edu/projects/NMAA/publications/pv2002.pdf. Es un documento muy valioso con información muy bien descrita y diferentes valores de parámetros de red obtenidos en la implementación.
[65] UltraGrid, http://www.east.isi.edu/projects/UltraGrid
[66] NGI SuperNet, http://www.ngi-supernet.org/
[67] Robust Audio Tool, http://www-mice.cs.ucl.ac.uk/multimedia/software/rat/
[68] Internet Engineering Task Force, http://www.ietf.org
[69] RTP Payload Format for SMPTE-292M (draft-ieft-avt-smpte292-video-01.txt)
[70] RFC 3497, RTP Payload Format for Society of Motion Picture and Television Engineers (SMPTE) 292M Video , L. Gharai, C. Perkins, G. Goncher, A. Mankin, March 2003, Proposed Standard
[71] RFC 2327, “SDP: Session Description Protocol”, M. Handley, V. Jacobson, April 1998, Proposed Standard

[72] RFC 2460, Internet Protocol, Version 6 (IPv6) Specification. S. Deering, R. Hinden. December 1998, Draft Standard
[73] Cisco Systems, http://www.cisco.com
[74] 6Bone, http://www.6bone.net
[75] RFC 2131, “Dynamic Host Configuration Protocol”, R. Droms, March 1997, Draft Standard
[76] RFC 1349, “Type of Service in the Internet Protocol Suite”, P. Almquist, July 1992, Proposed Standard
[77] RFC 2210, “The Use of RSVP with IETF Integrated Services”, J. Wroclawski, September 1997, Proposed Standard
[78] RFC 2638, “A Two-bit Differentiated Services Architecture for the Internet”, K. Nichols, V. Jacobson, L. Zhang, July 1999, Informational
[79] RFC 3246, “An Expedited Forwarding PHB (Per-Hop Behavior)”, B. Davie, A. Charny, J.C.R. Bennet, K. Benson, J.Y. Le Boudec, W. Courtney, S. Davari, V. Firoiu, D. Stiliadis, March 2002, Proposed Standard
[80] Abilene Aggregate Network Statistics, http://stryper.uits.iu.edu/abilene/aggregate/html/report-2004-10-06.html
[81] MBone – Multicast Backbone, http://www.mbone.com
[82] RFC 1112, “Internet Control Message Protocol”, J. Postel, Sep-01-1981, Standard
[83] 6Bone, http://www.6bone.net
[84] RFC 2471, “IPv6 Testing Address Allocation”, R. Hinden, R. Fink, J. Postel (deceased), December 1998, Historic [pub as:Experimental]
[85] Qbone, http://qbone.internet2.edu
[86] RFC 4175, “RTP Payload Format for Uncompressed Video”, L. Gharai, C. Perkins, September 2005, Proposed Standard. Esta RFC es de importancia vital para mi trabajo dado que me brinda la manera de efectivizar el transporte de video en forma de paquetes RTP. Tiene alcance para diferentes formatos de video de alta resolución.
[87] Douglas E. Comer, “Redes Globales de Información con Internet y TCP/IP”, Tercera Edición, 1996, Prentice Hall
[88] RFC4184, “RTP Payload Format for AC-3 Audio”, B. Link, T. Hager, J. Flaks, October 2005, Proposed Standard. Esta RFC me permite paquetizar el audio AC-3 sobre RTP.
[89] RFC 2736, “Guidelines for Writers of RTP Payload Format Specifications”, M. Handley, C. Perkins, December 1999, Best Current Practice
[90] RFC 2327, “SDP: Session Description Protocol”, M. Handley, V. Jacobson, April 1998, Proposed Standard
[91] Van Jacobson, "Congestion Avoidance and Control", Proceedings of ACM SIGCOMM '88, Stanford, CA, August 1988.

[92] W. R. Stevens. TCP/IP Illustrated, Volume 1, Addison Wesley, Reading, MA, 1994.
[93] Floyd and K. Fall. "Promoting the Use of End-to-End Congestion Control in the Internet," IEEE/ACM Transactions on Networking, Volume 7, Number 4, August 1999.
[94] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. "Modeling TCP Throughput: A Simple Model and its Empirical Validation," Proceedings of ACM SIGCOMM '98, Vancouver, Canada, August 1998.
[95] V. Paxson. "End-to-End Internet Packet Dynamics," IEEE/ACM Transactions on Networking, Volume 7, Number 3, June 1999. An earlier version appeared in the Proceedings of ACM SIGCOMM '97, Cannes, France, September 1997.
[96] V. Paxson. "Measurements and Analysis of End-to-End Internet Dynamics," Ph.D. dissertation, University of California, Berkeley, 1997.
[97] S. Floyd, M. Handley, J. Padhye, and J. Widmer. "Equation-Based Congestion Control for Unicast Applications," Proceedings of ACM SIGCOMM 2000, Stockholm, August 2000.
[98] D. Sisalem, H. Schulzrinne, and F. Emanuel. "The Direct Adjustment Algorithm: A TCP-Friendly Adaptation Scheme," Technical Report, GMD-FOKUS, August 1997
[99] RFC 3448, “TCP Friendly Rate Control (TFRC): Protocol Specification”, M. Handley, S. Floyd, J. Padhye, J. Widmer, January 2003, Proposed Standard. Es el estándar propuesto por la IETF para controlar la congestión de los flujos de audio y video en redes IP con RTP y UDP. Sobre este estándar basé el desarrollo del control de congestión del video de alta resolución de mi tesis.
[100] USC Information Sciences Institute, http://www.isi.edu. En este instituto trabaja gente sobre la transmisión de HDTV en redes IP, y además con control de congestión. Es un instituto de reconocimiento internacinonal en la materia.
[101] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. "Modeling TCP Throughput: A Simple Model and its Empirical Validation," Proceedings of ACM SIGCOMM '98, Vancouver, Canada, August 1998
[102] RTP Profile for TCP Friendly Rate Control , draft-ietf-avt-tfrc-profile-05.txt
[103] QoS-WG Mailing List, wg-qos@internet2.edu. Es una lista de mail q ue me proveyó diversos puntos de vista acerca del estado actual del uso de QoS en el mundo, principalmente en las redes de muy alta velocidad. La lista está compuesta principalmente por expertos de Internet2.
[104] RTP Retransmission Payload Format, Internet Draft, draft-ietf-avt-rtp-retransmission-12.txt, September 2005
[105] Audio/Video Transport (AVT), http://www.ietf.org/html.charters/avt-charter.html

[106] “Implementing congestion control in the real world”, Ladan Gharai and Colin Perkins, University of Southern California, Information Sciences Institute, http://nmaa.east.isi.edu/publications/icme2002.pdf. Este documento tiene información sobre TFRC aplicado al transporte de HDTV en redes IP y su lectura ha sido sumamente útil para mi investigación.
[107] RFC 1119, “Network Time Protocol (version 2) specification and implementation”, D.L. Mills, Sep-01-1989, Standard
[108] RFC 2198, “RTP Payload for Redundant Audio Data”, C. Perkins, I. Kouvelas, O. Hodson, V. Hardman, M. Handley, J.C. Bolot, A. Vega-Garcia, S. Fosse-Parisis, September 1997, Proposed Standard
[109] RFC 2733, “An RTP Payload Format for Generic Forward Error Correction”, J. Rosenberg, H. Schulzrinne, December 1999, Proposed Standard
[110] RFC 2283, “Multiprotocol Extensions for BGP-4”, T. Bates, R. Chandra, D. Katz, Y. Rekhter, February 1998, Proposed Standard
[111] RFC 2362, “Protocol Independent Multicast-Sparse Mode (PIM-SM): Protocol Specification”, D. Estrin, D. Farinacci, A. Helmy, D. Thaler, S. Deering, M. Handley, V. Jacobson, C. Liu, P. Sharma, L. Wei, June 1998, Experimental
[112] RFC 3618, “Multicast Source Discovery Protocol (MSDP)”, B. Fenner, Ed., D. Meyer, Ed., October 2003, Experimental
[113] RFC 2431, “RTP Payload Format for BT.656 Video Encoding”, D. Tiñan, October 1998, Proposed Standard

17.Agradecimientos

En primer lugar agradezco profundamente al Ing. Luis Marrone, quien aceptó ser mi director de tesis para brindarme su tiempo y sus conocimientos para ayudarme a delinear el trabajo.

En segundo lugar, un especial reconocimiento a Ladan Gharai y Colin Perkins, ambos miembros del grupo de trabajo Audio/Video Transport del IETF, quienes han sabido compartir conmigo toda su experiencia en el campo de la transmisión de HDTV sobre redes IP de última generación.

También agradezco a todas las personas pertenecientes a las diversas listas de discusión internacionales de las cuales participé –y sigo participando-, con las que he intercambiado muchas ideas y me han brindado en forma desinteresada sus inestimables puntos de vista.

Por último, agradezco a Karina, Jerónimo y Lorenzo……por estar siempre a mi lado.

Tesis: “HDTV sobre IP en Internet2”
Fe de erratas y Actualizaciones

Errata: pág. 10

Donde dice “el sistema de audio de Dolby Digital Surround Sound (AC-3) de cinco canales” debe decir “el sistema de audio de Dolby Digital Surround Sound (AC-3) de seis canales (5.1)”.

Errata: pág. 65

Donde dice “un switch que soporte más de 50 puertos ATM OC12 (2.5 Mbps)” debe decir “un switch que soporte más de 50 puertos ATM OC12 (622 Mbps)”.

Errata: pág. 97

Donde dice “Mapeo de RTP y RTCP sobre protocolos de transporte de bajo” debe decir “Mapeo de RTP y RTCP sobre protocolos de transporte de capas más bajas”.

Actualización: pág. 128

Los puntos 10.4 y 10.5 están mal porque hacen referencia a los campos relativos a QoS de IPv6 pero basados en la RFC 1883 –obsoleta- que ha sido puesta fuera de vigencia por la RFC 2460 –en carácter de draft estándar actual-.

Por lo tanto los puntos citados se reemplazan en su totalidad por estos contenidos:

10.4.IPv6 y QoS

IPv6 presenta muchas mejoras con respecto a IPv4 [18] (las características de QoS de IPv4 se analizan en el apartado de QoS), por las cuales muchos han preferido migrar, pero lamentablemente la característica de QoS no es la razón más convincente para hacerlo. Es un concepto erróneo que la especificación del protocolo IPv6 incluye un soporte intrínseco de QoS. Aunque la estructura de protocolo de IPv6 es significativamente diferente a la de su predecesor IPv4, la funcionalidad básica es aún muy similar.

Dos componentes significativos de IPv6 pueden proveer un método para entregar Clases de Servicios (CoS) diferenciadas. El primer componente es el **campo de Traffic Class de 8 bits** en el header IPv6, el cual es funcionalmente equivalente al campo **Type of Service (ToS)** en IPv4. Este campo está disponible para ser usado por los nodos de origen y los routers de forwarding y puede usarse para identificar y discriminar entre diferentes clases y prioridades de tráficos basado en los contenidos de ese campo de los paquetes IPv6, y puede ser usado en DiffServ. El segundo componente es el **Flow Label de 20 bits**, el cual fue agregado para habilitar que el origen rotule los paquetes que pertenecen a flujos particulares para los cuales requiere un tratamiento especial por parte de los routers intermedios, como el servicio de tiempo real. En este último caso, los routers no necesitarán hacer una inspección a fondo del paquete para identificar el flujo, dado que esta información está disponible en el header del paquete IP. Esto es un beneficio para aquellas aplicaciones que encriptan el tráfico con IPSec, dado que el Flow Label permite que los nodos diferencien el tráfico a nivel IP.

En el caso del uso del campo Flow Label, un **flujo** es una secuencia de paquetes enviados desde un origen particular hacia un destino particular (unicast o multicast) para el que el origen desea el tratamiento especial del que se habló en el párrafo anterior y por ejemplo llevado a cabo por un protocolo de reserva de recursos como RSVP u otro. **Un flujo es identificado unívocamente por la combinación de una dirección origen y un flow label diferente de cero**. Los paquetes que no pertenecen a ningún flujo tienen un flow label igual a cero. Todos los paquetes pertenecientes a un mismo flujo deben enviarse con la misma dirección origen, dirección destino y flow label.

El campo Traffic Class del header de IPv6 fue diseñado para permitir que el tráfico origen identifique la prioridad deseada de entrega de paquetes, comparada con otros paquetes del mismo origen de tráfico.

- Ciertos valores especifican la prioridad de tráfico para el cual la fuente provee control de congestión, como por ejemplo el TCP que responde a la congestión y pérdida de paquetes
- Otros valores especifican la prioridad de tráfico que no responde a situaciones de congestión, como el tráfico de tiempo real

Pero a decir verdad, no está claro aún cuáles serían los beneficios de clasificar el tráfico según el campo de Traffic Class del header de IPv6. Puede llegar a tener mayores beneficios a medida que se desarrollen

algoritmos más avanzados para descartar paquetes. La base de este modelo de CoS es que durante los períodos de congestión, se descartan paquetes según el valor de este campo.

En síntesis, el campo de Flow Label apunta a dar soporte a IntServ (o RSVP) y el campo Traffic Class apunta a dar soporte a DiffServ. **Ambos campos son considerados como las características de soporte QoS de IPv6.**

10.5.Conclusión

Hay un continuo desacuerdo en la comunidad de las redes acerca de la necesidad del uso de IPv6. De hecho, la mayoría de las mejoras diseñadas dentro de la especificación del protocolo IPv6 han sido retroalimentadas por IPv4. Por ejemplo, lo que se gana migrando a IPv6 en cuanto al beneficio que proveen las características de “dynamic neighbor-discover” y la autoconfiguración de direcciones de red, pueden ser rechazadas por la disponibilidad y subsecuente despliegue en gran escala del protocolo DHCP (Dynamic Host Configuration Protocol) [75] para IPv4.

De una manera similar, el campo Traffic Class de IPv6 no ofrece ninguna mejora sustancial sobre la utilidad del campo ToS del header de IPv4. El campo Traffic Class de IPv6 ofrece niveles adicionales de distinción de tráfico que los que ofrecen los bits de IP Precedence del campo ToS de IPv4 (8 posibles valores). Sin embargo esto no prueba que tenga un beneficio considerable. La mayoría sugiere que como nadie hasta ahora ha implementado en forma comercial un simple mecanismo de distinción para diferenciar clases de servicios de dos niveles, el hecho de incrementar el posible número de clases de servicios no es una buena razón para preferir IPv6 sobre IPv4. Y además algunos importantes ISP’s consultados sobre el interés de parte de ellos en ofrecer clases de servicios a sus clientes, han dicho que no estarían interesados en ofrecer más de tres niveles porque implicaría problemas de complejidad en la configuración y administración.

La utilidad del campo Flow Label de IPv6, por otro lado, puede ser muy beneficiosa. Sin embargo, el único uso actual que se ve es el Flow Label en conjunto con un protocolo de reservación de recursos, tal el caso de RSVP. El Flow Label puede posiblemente ser usado para asociar un flujo particular con una reserva de recursos específica. Fuera de esto, su posible uso no está muy bien determinado.

Por lo tanto, IPv6 no ofrece ningún beneficio sustancial de QoS más allá de lo que hoy ofrece IPv4.

Actualización: pág. 185

Donde dice “Este draft soporta un rango de formatos de video estándar y high definition” debe decir “Esta RFC soporta un rango de formatos de video estándar y high definition”.

Actualización: pág. 212

Donde dice “El Internet Draft de TFRC” debe decir “El RFC 3448 de TFRC”.

Errata: pág. 214

Donde dice “este perfil especifica cómo usar RTP/RTCP con el control de congestión RTCP” debe decir “este perfil especifica cómo usar RTP/RTCP con el control de congestión TFRC”.

Errata: pág. 238

Donde dice “En el caso del audio, el formato de payload RTP agrega 2 bytes y el perfil TFRC agrega 8 o 4 bytes” debe decir “En el caso del audio, el formato de payload RTP agrega 2 bytes”.

Donde dice “Overhead máximo = 2 Bytes + 8 Bytes” debe decir “Overhead = 2 Bytes”.