01-02-2010

Las explicaciones del desarrollador de *Adobe Flash* para Linux y el soporte de video

Franco Catrin
FayerWayer

Un intenso debate se está llevando a cabo en el sitio Phoronix y en el blog de Mike Melanson - desarrollador del plug-in de Adobe Flash para Linux - sobre los problemas que debe sortear para implementar la reproducción de video con un rendimiento aceptable. Mike inicialmente publicó un breve post en donde culpaba a la falta de estandarización de las API□s de Video en Linux como el motivo para no soportar ningún tipo de aceleración, recibiendo una contundente respuesta de la comunidad, que mostró como ejemplo otras aplicaciones de video que muy por el contrario, se ven beneficiadas por estas API□s.

Para nadie es un misterio que independiente del sistema operativo, reproducir videos con el plug-in de Flash en sitios como YouTube es lento, algo que se nota sobre todo en computadores con procesadores antiguos o al tratar de abrir varias pestañas con videos. Cualquier usuario que intente reproducir un video de YouTube con cualquier otro reproductor de video como por ejemplo Minitube , puede darse cuenta de que los los reproductores nativos consumen muy pocos recursos en comparación al plug-in de Adobe.

Hasta hace poco, se desconocían los motivos por los que el plug-in de Adobe se comportaba tan lento, y aunque Mike partió mal quejándose sin mayor fundamento en su blog, poco a poco se ha ido entendiendo en donde está el problema, y en este artículo intentaremos explicar de qué se trata el asunto.

Las tres etapas

La reproducción de video se puede dividir en tres etapas : decodificación, postproceso y render final. En la decodificación, se toman los datos comprimidos directamente del archivo o del stream de video para obtener un sólo cuadro o imagen, se puede ver como el acto de tomar unos pocos bytes y convertirlo en un pixmap sin compresión. El postproceso puede ser la aplicación de filtros (desentrelazado, eliminación de ruido, etc) o cualquier otra operación que se desee realizar sobre el cuadro obtenido, por ejemplo el dibujado de los subtítulos o un logotipo. Finalmente el render se encarga de volcar esa imagen final en memoria de video para que quede a la vista del usuario, esto incluye pasar por el sistema gráfico y en el caso de Flash en particular, dibujar el cuadro en un área rectangular de la página web renderizada en un browser.

Mike dice que no se puede comparar el rendimiento del plug-in de Flash con un reproductor de video normal porque se trata de resolver problemas que muy diferentes: En un reproductor de video normal las tres etapas son baratas en términos de uso de recursos porque no hay mayor transformación desde el cuadro obtenido en la primera etapa y el render final, por ejemplo el video decodificado puede ser enviado directamente a render usando aceleración por hardware, a menos que se requiera postproceso y aún así, los reproductores de video pueden usar métodos muy económicos para esta etapa.

El problema principal radica en que Flash Player necesita trabajar sobre imágenes que operan en el espacio de colores RGB, es decir, cada pixel tiene un componente de rojo, verde y azul, mientras que el procesamiento de video usualmente opera en el espacio de colores YUV (o derivado), en donde la representación de los pixeles se realiza con componentes más adecuados a la percepción humana e ideales para la compresión. Esta diferencia hace que mientras un reproductor de video puede pasar YUV directamente a la tarjeta de video, Flash tiene que convertir el cuadro completo a RGB para recién comenzar el postproceso que incluye el dibujo de otros componentes de Flash.

En la actualidad no hay forma de convertir YUV a RGB a través del hardware de video (GPU) y es una tarea que se hace "por software" en la CPU.

Seguramente Ustedes han visto una opción para habilitar la aceleración de video por hadware en el plug-in de Flash, Mike aclara que esta opción es sólo para el render final y lo que hace es aprovechar la GPU para escalar el cuadro al tamaño definitivo (ver bonus track). Tarea no menos importante, ya puede producir un mundo de diferencia entre reproducir el video en su tamaño original y llevarlo a pantalla completa, aún así la decodificación y conversión YUV->RGB sigue siendo sin ayuda del hardware.

En la versión 10.1 lo que se hizo fue aprovechar la aceleración por hardware para decodificación de video (primera etapa) en Windows: Se trata de DirectX Video Acceleration o DXVA, y es una API especialmente diseñada para este propósito y que permite a una aplicación usar la aceleración de video provista por el fabricante de hardware, como PureVideo de NVIDIA o Unified Video Decoder de ATI en forma independiente del hardware.

Lo que indicó Mike en su primer post es que en Linux no existía una API de decodificación de video que se pudiera usar en este momento, lo que provocó la ira de usuarios y desarrolladores que conocen de la existencia de VDPAU de NVIDIA y XvBA de ATI, justamente los equivalentes en Linux de PureVideo y Unified Video Decoder respectivamente. Por otra parte, Intel desarrolló Video Acceleration API (VA-API) para Linux, se trata de un mecanismo independiente del hardware para la decodificación de video, se puede decir que es el equivalente a DXVA de Windows para Linux, y permite usar la aceleración de decodificación de video disponible en algunos chips de Intel, NVIDIA a través de VDPAU, S3 y ATI a través de XvBA.

En la actualidad varios reproductores de video ya están usando estas API□s e incluso la implementación del plugin de Flash libre llamada Gnash, ya puede usar VA-API.

Para Mike esto sólo resolvería la primera etapa (decodificación) pero al menos podría quedar a la par con Windows, y el único sistema que quedaría atrás sería MacOSX, ya que según los desarrolladores de Flash no existe una API que permita obtener el cuadro decodificado tal como lo necesitan.

Aun así, aunque se cuente con decodificación por hardware se mantiene el problema de convertir los cuadros de YUV a RGB lo que hace que muchos se cuestionen qué tan adecuado es Flash para reproducir videos en la web y se enfatice la necesidad de que el famoso tag video de HTML5 tome fuerza y se convierta en un estándar para la publicación de videos.

Bonus Track : Las malas prácticas

Una de los aspectos más criticados del plug-in de Flash en Linux es que no soporta ningún tipo de aceleración y que los videos a pantalla completa pueden llegar a consumir gran parte del poder del procesador. La verdad es que Flash en Linux sí usa aceleración en la tercera etapa de render y lo hace a través de OpenGL, y también hay que decir que está mal implementado.

La idea es sencilla: El cuadro listo para ser renderizado se convierte en una textura y a través de OpenGL se le pide al chip de video que ajuste esta textura al tamaño que se necesita. Considerando que el hardware de video es especialista en transformar texturas, se trata de una operación que no le hace ni cosquillas al computador.

El problema es que Mike, quien parece ser el único desarrollador de este plug-in, no encontró una forma confiable de detectar el hardware, y en sus pruebas se encontró con que algunos drivers de video tenían problemas si se trataba de llamar a la funcionalidad requerida y ésta no se encontraba soportada. Por lo que simplemente se limitó a buscar el valor del el atributo "client glx vendor string" que arroja el comando glxinfo (*glxinfo* | *grep client*), y si este decía SGI entonces deshabilitaba la aceleración con OpenGL.

El problema es que salvo los drivers de NVIDIA, todos los demás dicen SGI porque usan una biblioteca común y no tiene nada que ver con las capacidades del hardware. Ian Romanick, un representante de Intel en Architecure Review Board de OpenGL lo fustigó por esta mala práctica:

Como uno de los representantes de Intel en el ARB de OpenGL, estoy horrorizado por este mal uso del atributo client de GLX. Este atributo no tiene ninguna información sobre la existencia o la calidad del render por hardware. Está orientado para propósitos de depuración y usarlo para cualquier otra cosa está completa e innegablemente equivocado. Por favor corrige este bug.

Mike respondió que fue el único método confiable de detección que pudo encontrar y nuevamente Ian de Intel respondió:

Si encuentras que una interfaz debería funcionar pero no funciona, por favor, reporta el bug. No te conformes usando interfaces que no fueron diseñadas para eso o que funcionan en formas no documentadas. Sé que es la forma estándar de hacer estas cosas en Windows. Nosotros preferimos corregir nuestros bugs cuando los conocemos, para que no tengas que hacer ese tipo de cosas

aquí. El hecho de que el "método más confiable de detección" sea confiable sólo en el driver de código cerrado de NVIDIA, es una clara evidencia de que no es confiable.

No es claro si este bug está o no corregido, pero al menos se incluyó un mecanismo para hacer que Flash no intente detectar el soporte de hardware por esta vía, para que el usuario avanzado habilite la aceleración por hardware explícitamente.

Conclusiones

De este debate se puede sacar una conclusión muy clara: Si el desarrollo del plug-in de Flash fuera abierto, por muy experto que sea Mike, podría haber recibido ayuda de otros expertos en video para Linux y también de fabricantes de hardware para implementar todo lo necesario con tal de habilitar la decodificación de video y además hubiera evitado el problema de la detección del hardware por medios no confiables, y que afectó a muchos usuarios. De hecho le proponen implementar la conversión a RGB que necesita Flash en los mismos drivers, porque hay hardware que lo soporta, lo que mejoraría el rendimiento del plug-in notablemente.

Se agradece que Mike haya compartido estos detalles que podrían ayudar a obtener más luces sobre el asunto, pero si trabajara más cercano a la comunidad de código abierto y los fabricantes de hardware, hace mucho tiempo que la experiencia de Flash en Linux podría haber mejorado.

http://www.fayerwayer.com/2010/01/las-explicaciones-del-desarrollador-de-adobe-flash-para-linux-y-el-soporte-de-video/