Protección de contenidos

Toni Comerma Paré

PID_00198489

Universitat Oberta de Catalunya

Índice

Introducción

El objetivo de este módulo es explicar cuáles son los diferentes mecanismos disponibles para proteger el contenido audiovisual que se distribuye por internet hacia los diferentes dispositivos. Por proteger entendemos controlar el uso que los usuarios pueden hacer de este contenido (reproducirlo, copiarlo, cuántas veces, desde qué dispositivos, etc.). La seguridad es un concepto muy amplio, e incluso si lo centramos en contenido audiovisual, nos encontraremos con muchas casuísticas distintas.

Objetivos

El objetivo general de este módulo consiste en tratar las casuísticas que afectan a la seguridad en la distribución de contenidos por internet abierta a los diferentes dispositivos (móviles, ordenadores, televisores, etc.), dejando de lado otros campos, como por ejemplo, la protección del contenido *offline* (DVD, Blu-ray) o los canales de televisión de pago.

En concreto, con el estudio de este módulo alcanzaréis los objetivos siguientes:

1. Adquirir una visión lo más práctica posible, orientada a entender la tecnología subyacente a la protección de contenidos, pero especialmente qué productos encontramos hoy en el mercado y qué posibilidades ofrece cada uno.

2. Aprender a valorar cuáles son las medidas de protección necesarias en cada caso y evitar caer en una sobreprotección de los materiales que puede representar un coste y una complejidad extras.

1. Algunos aspectos de la protección de contenidos

1.1. La necesidad de seguridad

¿Por qué esta necesidad de seguridad? Probablemente la pregunta parezca ingenua hoy día, pero también sería muy extraño empezar un módulo que habla de seguridad sin analizar el porqué. El contenido que gestionamos presenta las características siguientes:

a) Su coste de producción puede ser bajo (como en un vídeo doméstico) o increíblemente alto (como en una superproducción de Hollywood). Esto significa que alguien ha invertido esfuerzos, tiempo y/o dinero para crear este contenido, y puede tener la intención de preservar su propiedad.

b) Se trata de un producto que no es material, que es simplemente un conjunto de bits, y por lo tanto el coste de su copia resulta insignificante y la facilidad de distribución es muy elevada. Esto significa que si alguien es capaz de obtener una sola copia y la publica en internet, detener su difusión es prácticamente imposible.

Dificultad de controlar una filtración e impacto de la misma

Un ejemplo de la dificultad de controlar una filtración y del impacto de la misma se puede encontrar en el 2009, cuando una versión no acabada de la película *X-men wolverine* se publicó en una web un mes antes del estreno y, a pesar de los esfuerzos de la productora y de la MPAA, no se consiguió detener su distribución.

Queda por valorar el impacto que esta filtración puede haber tenido en la recaudación de la película. Es decir, el perjuicio real que esta filtración causó a los ingresos, aparte de al prestigio. Este es un tema de eterna discusión entre partes enfrentadas.

c) Se trata de un producto de consumo puntual. Se reproduce una, quizá dos veces y, a partir de aquí, en la mayoría de los casos pierde el interés para el usuario. Esta característica se puede observar claramente en los modelos de negocio que se han desarrollado a su alrededor: los cines, los videoclubes y el consumo por internet han triunfado mucho más que la compra de contenido (vídeos antes, DVD/Blu-ray en la actualidad). Esto requiere centrar la protección en la reproducción más que en la posesión del contenido.

d) Es un producto que puede cambiar de formato. Podemos encontrar desde formatos de alta calidad para reproducción en cines hasta copias de baja calidad para móviles (o copias piratas grabadas en un cine de alguna ex república soviética con una cámara de calidad infumable). El valor de cada una de estas es distinto, pero en el fondo se trata del mismo contenido. Los formatos de mayor calidad requieren una protección más alta, puesto que de estos se

MPAA

La Motion Picture Association of America (MPAA) es la asociación que une las seis principales productoras de películas de EE. UU. Una de sus principales actividades es la lucha contra la piratería.

pueden generar fácilmente otras copias de la misma o menor calidad, pero el usuario se adapta fácilmente a calidades menores si el coste es más bajo (o gratis).

e) Es un producto que no se quiere conservar dentro de una caja fuerte, donde sería fácil de proteger; lo queremos poner al alcance del usuario, pero solo para los usos y con las condiciones que nosotros deseamos imponer. Aquí radica el problema.

Podemos concluir que hay una necesidad objetiva de proteger contenidos y controlar su difusión.

1.2. Niveles de protección

No todos los contenidos tienen el mismo valor, y se puede asegurar que proteger tiene un coste, que aumenta a medida que queremos incrementar la seguridad.

Tenemos que partir de la premisa de que la seguridad absoluta no existe. En informática, igual que en el mundo real, es posible invertir esfuerzos para conseguir niveles de seguridad superiores, pero sería ingenuo (y no haber mirado nunca atrás en el tiempo para encontrar ejemplos de que esto ha pasado) asegurar que "el sistema de seguridad X es inviolable".

Veamos qué sucede en el mundo real. Está claro que robar un banco es más complicado que robar en mi casa. ¿Por qué? Porque el banco ha invertido mucho más dinero en medidas de seguridad que yo: cámaras, agentes de seguridad, cámaras acorazadas, alarmas, etc.

- ¿Significa esto que yo tendría que implantar las mismas medidas en casa? No. ¿Por qué? Pues porque el valor del contenido que me pueden robar (y, por lo tanto, la motivación de los posibles ladrones) es menor y hay que ponderar riesgos con costes. Las medidas de seguridad que hay que implantar en casa tienen que estar en consonancia con las medidas implantadas por el resto de casas similares (para no ser un objetivo más fácil que el resto), y debemos asumir que siempre hay unos riesgos que deben valorarse. Si intentamos aplicar las mismas medidas que un banco, no nos robarán, pero no nos quedará dinero para comer.

- ¿¿Significa esto que el banco es seguro y no puede ser robado? No. ¿Significa que los esfuerzos que debe hacer alguien para robarlo tienen que ser mucho mayores. El banco también hace un cálculo de riesgos entre lo que le cuesta la seguridad y el valor del contenido, y toma las medidas que considera adecuadas. Sin embargo, al final, algunos bancos acaban siendo robados.

- ¿Siempre es tan difícil robar? A veces no se trata de que alguien consiga superar las medidas de seguridad implantadas para prevenir un robo, sino que se aprovecha una mala implantación o uso de estas medidas. ¿Por ejemplo? A veces nos hemos encontrado con que, al salir con prisas de casa, hemos dejado la puerta abierta; entonces, toda la inversión en puerta blindada y cerradura de tres puntos no sirve de nada. O alguien puede decir el código para desactivar la alarma a un amigo o en voz demasiado alta en un bar, y arruinar el sistema.

Estos hechos del mundo real tienen traslación directa a la seguridad informática:

- El nivel de protección del contenido tiene que ser proporcional al valor del contenido y estar alineado con los niveles de protección que implantan servicios similares. Veremos que existen diferentes medidas de seguridad y que todas (o prácticamente todas) pueden ser vulneradas. No obstante, el esfuerzo y la complejidad para conseguirlo son diferentes. Aunque hay medidas en las que no se han encontrado vulnerabilidades, muy probablemente se les acabarán encontrando.

- Las medidas tienen que ser implantadas de una manera técnicamente apropiada, y es preciso proteger todo el ciclo de vida con el mismo nivel de seguridad. Hay que evitar invertir enormes esfuerzos en distribuir un contenido de manera segura y descubrir que alguien se ha introducido por una vía alternativa en los sistemas informáticos y lo está robando en el origen, donde no está tan protegido.

Estudiaremos los diferentes mecanismos de seguridad en orden de menor a mayor seguridad.

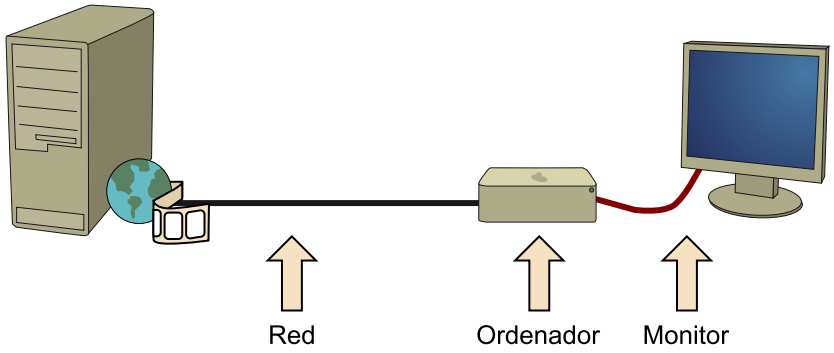
1.3. Qué se puede proteger

Básicamente, hay dos aspectos que debemos proteger: las **copias** y las **reproducciones**.

> Las **copias no controladas** son la principal fuente de piratería de contenidos y la principal amenaza que hay que evitar.

Para copiar el contenido, se tiene que encontrar un punto de la cadena entre el origen y la pantalla del ordenador en el que se pueda interceptar el contenido y copiarlo. Los puntos donde esto es posible son los siguientes:

Posibles puntos de interceptación



a) En la transmisión: si la información circula desprotegida durante la transmisión, es posible capturarla y extraer el contenido.

b) En el receptor: es el punto donde el contenido puede ser capturado alterando el dispositivo. Hay muchas posibilidades para esto. Si el receptor es un ordenador, la interceptación en este punto resulta más fácil, dado que se trata de un equipo donde se puede instalar software específico para la función. Si por el contrario se trata de un *smartphone* o de una televisión conectada, las posibilidades son más limitadas. La mayoría de los servicios se basan en la reproducción directa del contenido, pero algunos permiten la descarga del contenido al equipo para la posterior reproducción. Estos representan un reto extra, puesto que se tiene que proteger el contenido mientras está en el equipo.

Posibilidades de captura de contenido

- Solicitar la descarga desde otro aplicativo que permita grabar el contenido (por ejemplo, encontramos multitud de programas que permiten descargar vídeos de Youtube, aunque Youtube no proporciona la posibilidad de hacerlo).
- Capturar el contenido cuando entra en el ordenador, interceptando el tráfico de red.

c) De camino hacia el monitor: es posible, finalmente, interceptar el contenido cuando circula desde el reproductor hasta el monitor. Es complejo, porque se necesitan dispositivos especiales, pero resulta viable. Para intentar evitarlo, los estándares de cableado de conexión digital incorporan medidas de seguridad (el mecanismo más difundido es el protocolo HDCP, que implementan las conexiones por HDMI).

Existen medidas para proteger cada uno de estos puntos.

Las **reproducciones** son otro aspecto sobre el que se quiere establecer un control.

Incluso si se evita la copia, en algunos modelos de negocio puede interesar controlar:

- La cantidad de veces que el contenido se reproduce.
- En qué franja temporal se puede reproducir.
- Desde qué dispositivos.

Los modelos de negocio del tipo videoclub *online*[1] son los que están más preocupados por aplicar estas limitaciones.

(1)En español, *en línea*.

Hay otro control sobre la reproducción, más común y extendido, que es la reproducción **fuera del contexto original**. Este control lo aplican sitios web o servicios en general que ofrecen el contenido de manera gratuita, pero que han desarrollado un negocio en torno a este contenido, habitualmente basado en publicidad; o bien quieren preservar la imagen de marca o el control sobre cómo y dónde se consume el contenido. Estos sitios web o servicios se pueden encontrar con que un tercer lugar hace accesible este contenido desde un servicio no relacionado con el titular del contenido, lo que puede provocar pérdidas de ingresos por publicidad, mientras que se asumen los costes de distribución. Con unos ejemplos se verá más claro.

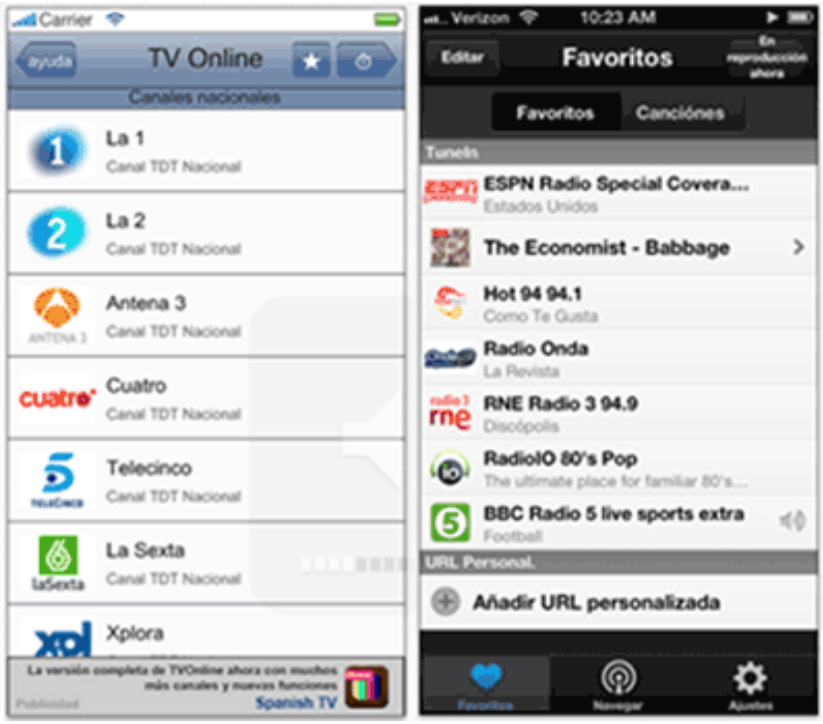
Una retransmisión de Fórmula 1

Una televisión compra los derechos para emitir la Fórmula 1 por televisión y también por internet. Aparte de los costes, la Formula One Group pone unas condiciones estrictas sobre todos los aspectos de la distribución, incluida la presencia de los patrocinadores, el aspecto visual, etc. La televisión hace la emisión en abierto y se financia con la publicidad que ha puesto en el sitio web.

Sin embargo, alguien toma la URL del vídeo en directo (o de los diferidos publicados) y la pone en una página web hecha por él con publicidad que también cobrará él, y ya tenemos el problema planteado. La televisión no desea impedir que los usuarios consuman el contenido, pero quiere asegurarse de que lo consumen en el lugar esperado.

Otro ejemplo son las aplicaciones agregadoras de canales de televisión o radio. Estas utilizan los flujos de datos (*streams*) publicados por las emisoras y los incluyen en su aplicación (de la cual obtienen ingresos por publicidad o venta), mientras que los costes de distribución los soportan las emisoras.

Aplicaciones para iOS TVOnline y tunein radio



En algunos casos hay acuerdos entre emisoras y publicadores de aplicaciones, pero en otros muchos, no.

Es posible implantar medidas de seguridad orientadas a controlar desde dónde se hace la reproducción.

No querríamos acabar esta introducción a los aspectos de seguridad sin mencionar que los temas de la protección del contenido audiovisual, la piratería y la adecuación de los modelos de negocio de las empresas titulares de los derechos a la realidad tecnológica y social están de plena actualidad, y es muy difícil mantenerse ajeno a ello o no tener una posición personal al respecto. Sin embargo, estas opiniones no influyen en la técnica y evitaremos entrar en este tipo de discusiones en este material. No obstante, pueden ser motivo de un interesante debate a lo largo del curso.

Para hacer el tema más práctico, en lugar de efectuar una explicación teórica y después ver cómo se aplica, lo haremos al revés. Repasaremos cuáles son las tecnologías presentes en el mercado para proteger contenido, y sobre la marcha, a medida que surja la necesidad, iremos introduciendo los conceptos teóricos necesarios.

Reflexión

Paradojas de la vida, la mayoría de los sitios web de *video sharing* –así se denominan los sitios web donde los usuarios suben las series de TV y películas para que las podamos ver sin pagar–, como nowvideo.eu, allmyvideos.net, etc., implantan mecanismos de seguridad para limitar la descarga de contenido que no han creado ni comprado. Lo hacen porque su negocio se basa en ingresos de publicidad y en usuarios que pagan para descargar contenido. Por lo tanto, quieren limitar lo que un usuario puede descargar de manera gratuita. Las medidas de seguridad que tienen son vulnerables (y buscando por internet, se puede encontrar cómo salvarlas), pero suficientes para una mayoría de los usuarios.

2. Tecnologías de protección de contenidos

2.1. Orígenes y convergencia de las tecnologías de protección de contenidos

Las tecnologías de protección del contenido audiovisual tienen dos orígenes distintos, que marcan su idiosincrasia y sus puntos fuertes y débiles: las que provienen de internet y las que provienen del mundo de la televisión.

a) Tecnologías de protección que provienen del mundo de internet

El vídeo en línea por internet empezó prácticamente sin sistemas de protección del contenido. Durante años, la mayoría del contenido se ha emitido en abierto y solo una pequeña parte se ha protegido. Por este motivo, los fabricantes de tecnología se han centrado más en proporcionar herramientas para desarrolladores y crear una experiencia agradable para el usuario que en proveer herramientas para proteger su contenido. Esto no debería entenderse como un "no absoluto"; desde el principio ha habido fabricantes que han desarrollado mecanismos de seguridad, pero siempre han sido soluciones de nicho que se han aplicado a proyectos de poca repercusión para el público en general. Actualmente, sin embargo, hay soluciones que ofrecen las máximas garantías de seguridad, equivalentes a las que se pueden encontrar en cualquier dispositivo.

b) Tecnologías de protección que provienen del mundo de la televisión

En este entorno, desde hace muchos años, antes de que internet llegara al gran público, ya había canales de pago, y con estos se desarrollaron mecanismos para controlar el acceso a estos canales. Tecnológicamente eran muy distintos a los sistemas actuales, pero con la llegada de la digitalización y de los canales IPTV (Imagenio de Movistar, OrangeTV de Orange, etc.), que comparten algunas características con la internet abierta, las tecnologías son cada vez más similares. Este siempre ha sido un entorno con unas características concretas.

- Se trata de negocios que mueven unas grandes inversiones económicas. Poner en marcha un servicio como un canal de televisión requiere una inversión de capital enorme y unos proyectos de ingeniería mastodónticos.

- Se paga por el servicio, lo que significa que las empresas tienen unos ingresos importantes que deben compensar la inversión que hacen, naturalmente.

- Las empresas controlan toda la cadena de valor: desde la emisión y la distribución hasta el equipo que instalan en nuestro domicilio, lo que les permite elegir qué elementos tecnológicos quieren utilizar y garantizar la interoperabilidad de todos estos elementos.

- El contenido que se distribuye tiene un alto valor (como lo demuestra el hecho de que los usuarios están dispuestos a pagar por este contenido) y, por lo tanto, para mucha gente es tentador acceder al mismo de manera gratuita. Como decíamos, cuanto más valor tenga el contenido, más esfuerzos se harán para conseguir acceder a este de modo ilegítimo y más protección habrá que porporcionarle.

Este escenario ha desembocado en unas tecnologías fiables, que integran todos los elementos de la cadena y de un coste elevado.

Contraponemos este modelo con el de la internet abierta, en la que los dispositivos (ordenador, *smartphone*, etc.) son propiedad del usuario, con potencias de cálculo, software, sistemas operativos instalados, formatos, tamaños de pantalla, etc., muy distintos. Se trata de un ecosistema muy variado, fuera del control de una única empresa, como es el caso de la televisión. Cuando nos damos de alta a una televisión de pago, se nos suministra un descodificador controlado por la empresa, que no cumple ninguna otra función y que no se puede manipular. Si estamos dados de alta en dos canales, tenemos dos descodificadores. Imaginemos esto en internet y que necesitáramos un ordenador para acceder a Youtube y otro distinto para acceder a Vimeo. El control extremo a extremo no es posible y puesto que han de adaptarse a un entorno más heterogéneo, compartido y controlado por el usuario, la compatibilidad es un requisito imprescindible.

También cambian los modelos de negocio (habitualmente empiezan con una inversión relativamente pequeña y unos márgenes más reducidos), lo que obliga a soluciones más incrementales y de implantación más rápida

Por último, cambia igualmente el tipo de usuario (o su actitud, puesto que la misma persona no tiene las mismas expectativas o requerimientos en diferentes circunstancias). Con su *smartphone* u ordenador, no está dispuesto a renunciar al control. Quiere controlar qué se instala en sus dipositivos y es reticente a que lo fuercen a instalar aplicaciones; tolera mal que estas nuevas aplicaciones generen problemas o que le impongan restricciones extra (el uso del navegador X, o la versión de sistema operativo Y). Estas preferencias del usuario acaban teniendo mucha importancia a la hora de seleccionar qué tecnología utilizar.

Convergencia

Estos dos orígenes –las tecnologías de protección de contenidos provenientes de internet y las provenientes del mundo de la televisión– van acercándose y convergiendo.

Aplicación de Youtube XL en un Sony Bravia

Aplicaciones disponibles en un Samsung SmartTV

Esto se debe tanto a que los fabricantes intentan ampliar su negocio hacia otros campos, como a que desde un punto de vista tecnológico, ambos mundos se acercan:

a) Las televisiones de pago se van moviendo hacia sistemas IPTV, que utilizan la misma tecnología de transmisión que internet (IP), de modo que se produce una convergencia que los fabricantes de tecnología provenientes de internet aprovechan para intentar llegar a las televisiones.

b) Las televisiones se empiezan a conectar a internet –fenómeno que se conoce como televisiones conectadas[2]– y se puede trasladar el contenido disponible en internet hacia estos dispositivos. Esto provoca los fenómenos siguientes:

- Por un lado, los fabricantes de televisores, que tradicionalmente han trabajado con los proveedores de tecnología de seguridad de su entorno, los prefieren puesto que hay unas relaciones sólidas y experiencias compartidas. Por este motivo, intentan llevar a los proveedores de contenidos hacia estas tecnologías.

- Por otro lado, los proveedores de contenidos que vienen de internet han estado trabajando con las tecnologías de este entorno y se encuentran más cómodos, de manera que intentan que estos fabricantes entren en el terreno de las televisiones y que los fabricantes de televisión implanten tecnologías provenientes de internet.

(2)En inglés, *connected TV*.

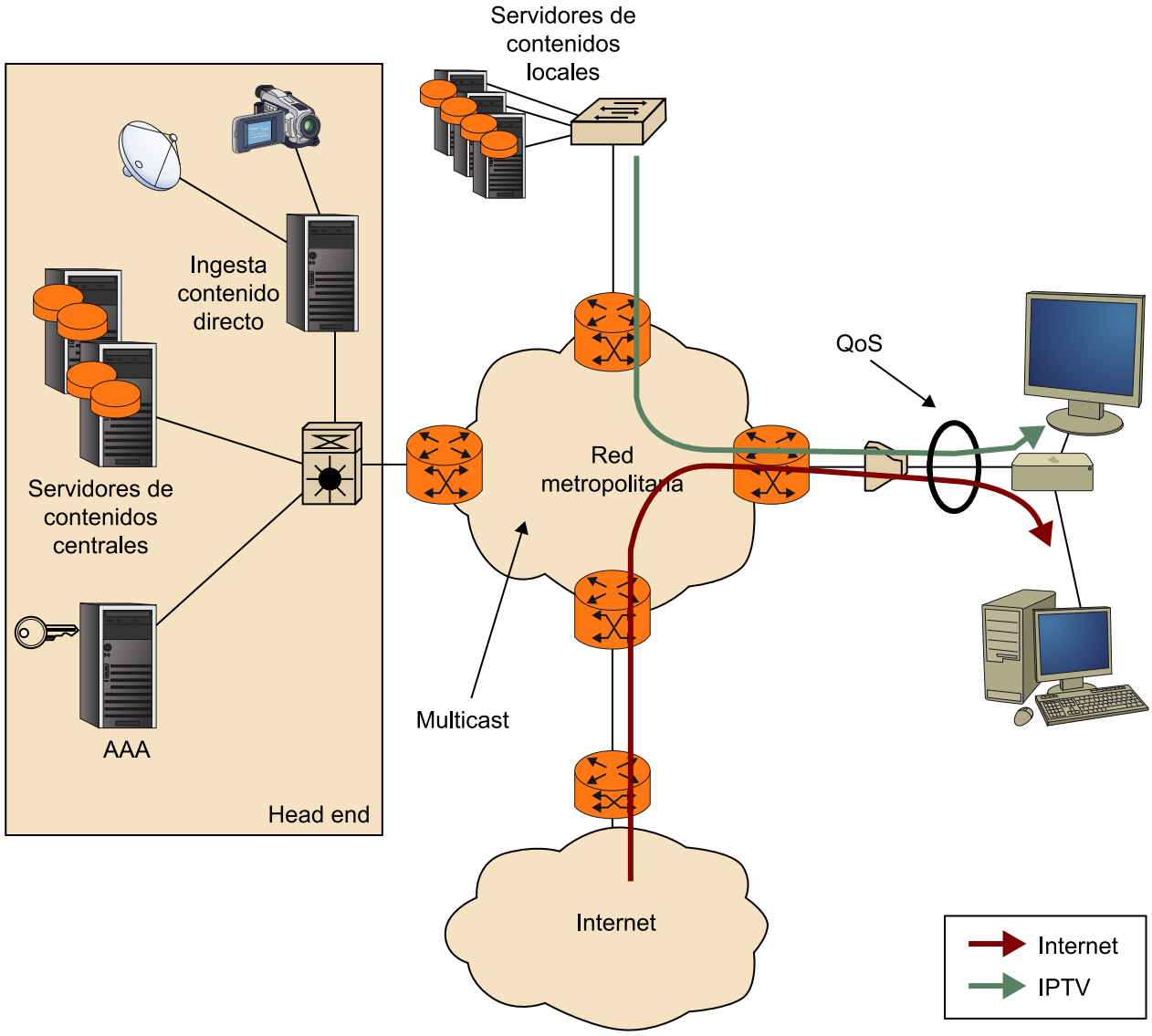
IPTV y *over the top*

Estos dos nombres definen dos maneras distintas de transmisión de contenidos hacia un televisor utilizando redes de comunicación basadas en IP. Tienen en común el hecho de utilizar IP, pero a partir de aquí todo son diferencias, tanto tecnológicamente como de modelo de negocio:

1) IPTV (***IP Television***) es un modelo en el que el operador de comunicaciones de banda ancha (el que habitualmente nos da acceso a internet) utiliza su infraestructura propia para hacer llegar a los domicilios canales de TV vía IP por una red propia. Llega al receptor por el mismo cable que internet y posiblemente también por el mismo que el teléfono, pero las infraestructuras están completamente separadas.

Un sistema de IPTV como el que se puede ver en el esquema siguiente consta de una ubicación central en la que se gestionan todos los contenidos –tanto en directo como bajo demanda– y se encuentra toda la infraestructura de gestión necesaria (monitorización, seguridad, facturación, etc.) a partir de la cual los contenidos se envían a centros metropolitanos que se encargan de su distribución a los usuarios finales. Estos centros pueden tener copias locales del contenido para descargar la infraestructura central y la red. Por último, el usuario final se conecta a esta red mediante el enrutador (*router*) que tiene instalado en casa, hacia el DSLAM de la central más cercana al domicilio.

Esquema simplificado de una red IPTV



El operador conecta su red a internet y da acceso a los usuarios autorizados, pero como hemos dicho, controla toda la red. Esto permite dos cosas fundamentales:

- Puede utilizar dentro de su red los protocolos de destino múltiple (*multicast*), lo que le permite una difusión de los canales en directo mucho más eficiente que por internet (circula un flujo de datos único por cada canal hasta cada nodo de la red, en lugar de un flujo por cada usuario). En contraposición, el destino múltiple es un protocolo que no está soportado en internet.

- Puede asignar QoS a todos los flujos de datos, lo que le permite asegurar la calidad de la recepción. Incluso en el tramo final hacia el usuario, en el que el tráfico de IPTV comparte el canal con el de internet o teléfono, es posible separar estos canales, reservar ancho de banda para cada uno y configurar circuitos virtuales de ATM para cada servicio.

La parte negativa es que si estamos conectados al proveedor A, no podemos utilizar el servicio de televisión del proveedor B porque las redes no se comunican.

2) ***Over the top*** (***OTT***), por el contrario, hace referencia a los servicios que utilizan una infraestructura de comunicación que no está bajo su control. Esto es lo que sucede con cualquier servicio que utiliza internet, desde Google hasta un blog personal, puesto que todos son usuarios de una red de comunicaciones gestionada por diferentes empresas. La transmisión de datos en estos entornos se caracteriza por el *best-effort*, sin una garantía de no pérdida de paquetes ni de ancho de banda. Es más, las condiciones de la red son muy cambiantes.

Por ejemplo, se puede estar reproduciendo correctamente un vídeo hasta que otro equipo de la misma red empieza a descargar un programa y ocupa todo el ancho de banda. Estos servicios no disfrutan de las ventajas que proporciona IPTV y tienen que lidiar con unas comunicaciones sin QoS y sin destino múltiple. Como contrapartida, no hay que hacer inversión en infraestructura dado que ya está disponible, y esto simplifica mucho la creación de negocios, minimiza su coste y facilita su profusión. Además, el hecho de que la red sea común hace que con una sola conexión se pueda acceder a múltiples servicios.

A pesar de estas dificultades, los servicios OTT funcionan y han triunfado de manera decidida, como se puede ver en el fenómeno del vídeo por internet. En lo que respecta al tráfico, algunas previsiones auguran que en el 2016 el 86% del tráfico de internet será vídeo (podéis ver *Cisco Visual Networking Index: Forecast and Methodology, 2011-2016*). Ya en la actualidad, un solo servicio de vídeo de pago como NetFlix consume más del 30% del tráfico de internet en EE. UU. (podéis ver *Sandvine Global Report*).

También encontramos el mundo de los teléfonos y las tabletas[3], en el que ha habido igualmente un choque de tecnologías, pero en este caso el resultado ha sido bastante claro. Los fabricantes tradicionales de teléfonos (Nokia, Ericsonn, Alcatel, Motorola, etc.) se han visto barridos del mercado por los productos cuyo origen es el mundo internet/ordenador. Apple (con su iPhone) y Google (con Android) han revolucionado el mercado. Ahora es más importante lo que se puede hacer con el dispositivo que las prestaciones físicas que este añade (pantalla, cámara, etc.), y esto lo marca el sistema operativo que lleva dentro y el ecosistema de creadores de aplicaciones que hay alrededor (Apple Store, Google Play). Sistemas más abiertos, menos controlados por el fabricante, han tenido éxito y han llevado contenidos audiovisuales a estos dispositivos básicamente provenientes de internet. El hecho de que tanto los contenidos como la tecnología provengan de este entorno también ha hecho que las soluciones de seguridad más implantadas provengan de internet.

(3)En inglés, *tablets*.

Android

Android no es un teléfono, sino el sistema operativo que hace funcionar el teléfono. Sin embargo, es el elemento que tienen en común toda una gama de terminales de diferentes fabricantes, y lo que ha hecho que destaquen y tengan éxito en el mercado.

2.2. Principales fabricantes de tecnologías de protección de contenidos

Hablaremos de dos tipos de fabricantes de tecnologías de protección de contenidos: los que provienen del mundo de internet y los que vienen del mundo de la televisión.

2.2.1. Provenientes del mundo de internet

Como principales fabricantes de tecnologías de protección de contenidos que provienen del mundo de internet, mencionaremos Adobe, Microsoft y Apple.

Adobe

Adobe es una reputada empresa de software orientada a la producción de contenidos audiovisuales y a internet con una gama muy amplia de productos, entre los que destacan aplicaciones como Photoshop, Premiere o Acrobat (PDF). Sin embargo, ninguna de estas nos interesa en el presente contexto.

A mediados de los noventa, al principio de la internet comercial, cuando una web era texto e imágenes, la empresa Macromedia –posteriormente adquirida por Adobe– pone en el mercado Flash Player, un complemento[4] que se incrus-

(4)En inglés, *plugin*.

ta en el navegador web y permite la creación de animaciones con un entorno de desarrollo simple y orientado a los diseñadores, los usuarios primeros. Las animaciones en movimiento triunfan y el uso del producto se extiende hasta el punto de que a estas alturas Flash está instalado en casi todos los ordenadores. En versiones posteriores, el producto evoluciona e incorpora la capacidad de reproducir vídeo y audio y un lenguaje de programación (ActionScript) orientado a objetos. Se puede decir que crea una nueva categoría de aplicaciones: las *rich internet applications*. Al mismo tiempo que evoluciona Player, Adobe desarrolla otros productos para cubrir las necesidades de distribución del contenido (Adobe Media Server) e incorpora medidas de seguridad a los productos (Adobe Access).

Lectura recomendada

Podéis encontrar un extenso artículo sobre Flash Player en la versión inglesa de Wikipedia:

Wikipedia. "Adobe Flash".

Aunque recientemente Flash Player –la aplicación que se instala en el dispositivo del usuario– ha recibido críticas especialmente en lo que respecta a dispositivos móviles, que Adobe ha anunciado que dejará de desarrollarlo para estas plataformas (es interesante una búsqueda en Google por "flash for mobile", por el debate encendido que hay) y que Apple decidió vetarlo en sus dispositivos basados en iOS, en el entorno de los ordenadores personales es omnipresente.

El veto de Apple a Flash

Podéis leer unas declaraciones de Steve Jobs sobre el soporte de flash en iOS:

Thoughts on Flash

Porcentaje de ordenadores con Flash Player instalado

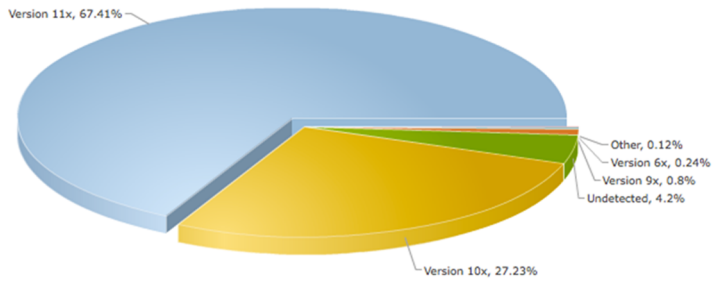
OVERALL FLASH PLAYER SUPPORT	TOTAL AVG	JAN '12	FEB '12	MAR '12	APR '12	MAY '12	JUN '12
Flash Support (all versions)	95.80%	95.98%	95.41%	95.56%	96.02%	96.00%	95.79%

Report Filters
Click here to drill-down to find the information you are looking for!

DATE RANGE	INTERVAL	CHART TYPE	BROWSER	OPERATING SYSTEM	SEARCH ENGINE	CONNECTION
Jan '12 - Jun '12	Monthly	3D Pie	All Browsers	All Operating Systems	All Search Engines	All Connections

TIMEFRAME » INTERVAL » CHART TYPE » EXPORT » SHARE »

Flash Plugin Version Support



Fuente: statowl.com

Los dos factores que más han contribuido a esta omnipresencia han sido los siguientes.

- Fue de los primeros. Apareció en un momento en el que los usuarios reclamaban una solución para mejorar la interactividad de la web y hacerla más atractiva, y su uso se universalizó.

- La manera de programar, muy simple y adaptada a los diseñadores, contribuyó a crear una gran comunidad de desarrolladores.

Estos dos factores se realimentaron hasta ponerlo al frente, sin prácticamente competencia.

Adobe ha sabido evolucionar, mantener el liderazgo, incorporar nuevas funcionalidades y desarrollar el lenguaje de programación hacia un modelo orientado a objetos, y de este modo atraer a los desarrolladores.

Reflexión

Actualmente, Flash Player va perdiendo su posición de privilegio. Hay discusiones respecto al porqué, pero podríamos apuntar hacia tres grandes causas:

1) Encontramos alternativas viables para conseguir el mismo efecto. Con la aparición y evolución del lenguaje Javascript y CSS, se pueden hacer webs con la misma interactividad y espectacularidad que con Flash. Continúa siendo necesario para reproducir vídeo, pero recientemente la versión 5 del estándar de HTML ha incorporado la etiqueta (*tag*) <video> (de la que hablaremos más adelante), que pretende resolver esta carencia.

2) Adobe ha fracasado en los dispositivos móviles. No ha conseguido trasladar el producto de los ordenadores personales a los teléfonos móviles / tabletas. Ha hecho versiones con funcionalidad limitada, pero que no han acabado de satisfacer. Esto ha llevado a Adobe a anunciar que deja de intentar llevar Flash Player a los teléfonos móviles.

3) No olvidemos la competencia del mercado. Argumentos técnicos objetivos aparte, Adobe ha ocupado una posición de casi monopolio en este segmento de mercado y esto ha hecho que el resto de los competidores busquen formas de desbancarlo.

Microsoft

Microsoft es un veterano en el sector de vídeo por internet. Mucho antes de que Adobe desarrollara sus productos, Microsoft era –junto con una empresa casi desaparecida, que también tenía una buena tecnología, denominada RealNetworks– líder en el mercado, con un conjunto de productos que cubrían todas las necesidades, incluida la securización del contenido. Sin embargo, fue relegada a una posición de comparsa con la aparición de Flash Player.

No obstante, Microsoft es una gran compañía, con capacidad financiera y de innovación, y ha vuelto a la lucha con una nueva versión de su solución. Hay que decir que tanto a principios del 2000 como en la actualidad, la tecnología de Microsoft es igual o superior en prestaciones a la de Adobe. Todo lo que se puede hacer con la segunda se puede hacer con la primera. Lo que marcó la diferencia y decantó la balanza en favor de Adobe fue Flash Player: la posibilidad de personalizar la reproducción y llegar a la audiencia con una experiencia de usuario más atractiva e integrada con el contenido de la página web.

En la imagen siguiente, se puede ver una captura de una antigua página web con el reproductor incrustado (enmarcado con un recuadro verde). El aspecto del reproductor era poco configurable, y la barra de controles de la parte infe-

rior era siempre la misma. Por no hablar de las dificultades de reproducción en Apple o Linux, o simplemente con otros navegadores que no fueran Internet Explorer.

Reflexión

Esta situación que experimentó Microsoft de tener un producto tecnológicamente avanzado, pero que no coincide con el que el usuario pide, es una constante en el mundo de internet. Seguramente Microsoft tenía unos objetivos estratégicos (potenciar su sistema operativo, diferenciarse de la competencia, quién sabe), pero sin duda no tuvo éxito. A menor escala, esto nos puede suceder al poner en marcha cualquier proyecto. Sea cual sea nuestra idea, la tecnología que utilicemos o la visión que tengamos de las necesidades de los usuarios, si estas no encajan con la realidad, las posibilidades de tener éxito son exiguas.

Microsoft cambió de estrategia incorporando una tecnología equivalente a Flash Player, denominada Silverlight, y apoyando el códec h.264 para ganar compatibilidad, de modo que actualmente va ganando presencia. También ha hecho un cambio en la tecnología de distribución para evolucionar hacia el *streaming* adaptativo y, al mismo tiempo, extenderse fuera del ámbito de los ordenadores (hacia teléfonos, tabletas y televisores). Aparte del reproductor, dispone de piezas de software para toda la cadena de creación y distribución.

Apple

Apple es el tercer fabricante con un peso importante en el vídeo por internet. A pesar de tener unas herramientas mucho más limitadas que los anteriores, dispone de un nicho de mercado importante en los dispositivos iOS, en los que el consumo de vídeo es muy importante, y ello le da relevancia.

Apple fue de los primeros, junto con Microsoft, en tener productos para la distribución de vídeo por internet, y también fue de los que más se ajustó a los estándares existentes (utilizaba el códec MPEG-2 y posteriormente fue de los primeros en adoptar h.264 y RTSP como protocolo de transmisión), pero estos productos no consiguieron una posición predominante en el mercado, básicamente por el mismo problema que Microsoft: requerían un reproductor –Quicktime– poco personalizable, que no permitía la personalización de Flash Player, hasta que, junto con el lanzamiento de iPhone e iPod, desarrolló un nuevo protocolo de transmisión: HTTP Live Streaming (HLS).

Las opciones de securización son más limitadas que en los anteriores. Las veremos más adelante.

2.2.2. Provenientes del entorno *broadcast*

Los principales fabricantes de tecnologías de protección de contenidos que provienen del entorno *broadcast*[5] que mencionaremos son Marlin y Widevine.

(5)En español, 'difusión amplia'.

Marlin

Marlin no es propiamente una empresa, sino una agrupación de fabricantes de televisores (Sony, Philips, Samsung y Panasonic), más una empresa de I+D (Intertrust), que se unen para impulsar de manera conjunta una solución tecnológica estandarizada y segura para la distribución de contenido audiovisual en línea a dispositivos de electrónica de consumo. Con este fin, se plantean los objetivos siguientes:

- Especificar un conjunto de protocolos y especificaciones que definen cómo deben ser las interacciones entre las distintas partes del sistema (por ejemplo, cómo se tiene que comunicar el reproductor con el servidor de contenidos, proteger y codificar el contenido, etc.).

- Crear los elementos de software que implementan el estándar anterior y comercializarlos (de esta parte se encarga Intertrust). En lo que respecta a productos, dispone de reproductor de contenidos (Wasabi Marlin, la pieza de software que se incorpora a los dispositivos y reproduce el contenido), software para la distribución del contenido (Bluewhale Marlin), para la securización (Octopus DRM), etc.

El hecho de que separen la especificación de la implementación permite que desarrollos de terceros puedan interoperar con los suyos, y está consiguiendo ganar presencia en el mundo de las televisiones conectadas.

Recientemente, la Asociación Española de Empresas de TV Interactiva (AEDETI) eligió Marlin y Microsoft como productos para implantar la seguridad en las televisiones conectadas (DRM[6]).

(6)*DRM* es la sigla de *digital rights management*.

Lectura complementaria

Podéis leer el anuncio de AEDETI en el artículo siguiente:

"La industria propone la adopción de los DRM de Marlin y Microsoft"

Widevine

Widevine es un fabricante con una larga experiencia en la creación de soluciones para la industria de la electrónica de consumo. Como Intertrust, tiene productos para todas las necesidades: desde el reproductor, que adaptan y licencian a cada dispositivo, hasta servidores para la distribución y soluciones de seguridad (DRM).

Ved también

El DRM se trata en el apartado 4 de este módulo didáctico.

Recientemente, Widevine ha sido adquirida por Google y el sector está a la expectativa de qué puede significar este movimiento, tanto de manera técnica como estratégica.

Reflexión

Una diferencia entre el entorno internet y el de la electrónica de consumo es que en el primero, el usuario tiene el control de su dispositivo y la libertad de decidir qué software instala; los ordenadores, las tabletas y los teléfonos dan la flexibilidad de instalar software.

Por el contrario, los dispositivos de electrónica de consumo son más cerrados, y habitualmente el usuario no puede instalar nada. Esto implica también una diferente forma de relación entre los fabricantes de tecnología y sus clientes en función del entorno.

En el entorno internet, los fabricantes deben intentar convencer al usuario o al creador de un servicio de la utilidad de su producto. Esto da una mayor flexibilidad tanto a usuarios como a desarrolladores. Aquí el cliente es el creador de un servicio o el usuario final.

En cambio, en el entorno de electrónica de consumo, los fabricantes como Marlin o Widevine tienen que convencer al fabricante del producto (Sony, LG, Philips, etc.) para que incorporen su tecnología. Son contratos con el fabricante, más estratégicos, grandes y de mayor importe.

3. Tipos de protección de contenidos

En este apartado, que podemos considerar el principal, analizaremos las diferentes opciones que tenemos para proteger el contenido, y valoraremos los pros y los contras en cada caso.

3.1. Prevención del *hotlinking*

El *hotlinking* es una práctica que consiste en enlazar desde una web contenido de otra sin su permiso. Un ejemplo de esto es el caso de la Fórmula 1 que hemos presentado antes, pero aparte de vídeo, también se puede hacer con imágenes o cualquier otro recurso. El sitio web de origen carga con los costes de distribución (puesto que cada usuario accede a sus servidores), pero no obtiene su crédito. Es diferente de hacer una incrustación[7] de un contenido, lo cual constituye una práctica legítima puesto que hay lugares que lo permiten y alientan (se puede incrustar el reproductor de YouTube en cualquier página, pero lo que no se puede hacer es obtener el acceso al vídeo y publicarlo en nuestra web con un reproductor distinto).

[7] En inglés, *embed*.

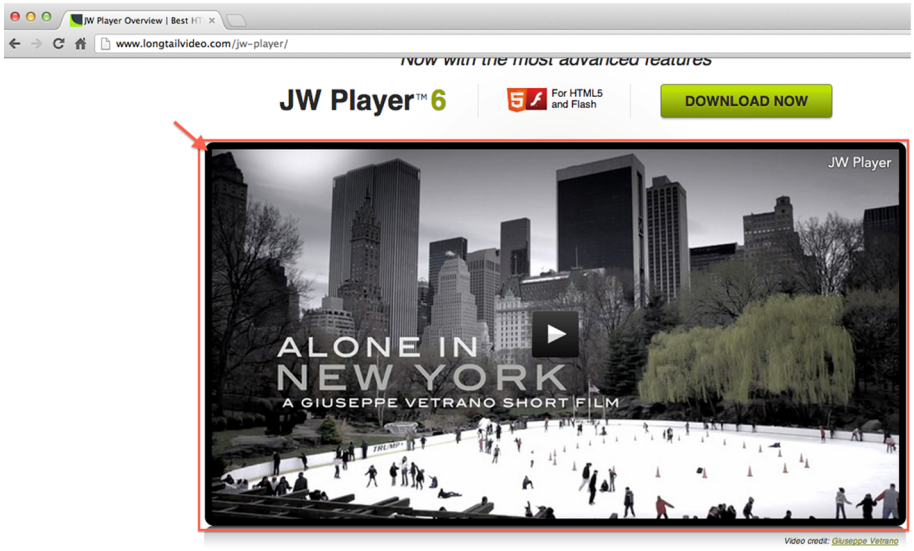
Ved también

Podéis ver el caso de la Fórmula 1 en el subapartado 1.3 de este módulo didáctico.

Todos los contenidos que se publican en línea, tanto directos como diferidos, se acaban presentando como una URL que tendrá una forma u otra en función del protocolo que utilice. Veamos un ejemplo de esto.

Un sitio web como http://www.longtailvideo.com/jw-player/ pertenece a los autores de un reconocido reproductor de vídeo. En esta página, tienen un reproductor para mostrar un vídeo de ejemplo; en este caso no es tan importante el contenido como el reproductor en sí, puesto que es lo que ellos quieren mostrar.

Captura de la página con el reproductor incrustado



Si se observa el código fuente de la página (con el botón derecho, casi todos los navegadores tienen una opción que permite verlo), a la mitad del código HTML se puede encontrar el código Javascript siguiente:

Código Javascript para incrustar el reproductor

```
<script type='text/javascript'>
jwplayer('player_1').setup({
   file: "http://content.bitsontherun.com/videos/lWMJeVvV-364767.mp4",
   width: "876",
   height: "470",
   image: "/content/images/jw-player/lWMJeVvV-876.jpg",
   logo: {
          file: "http://p.jwpcdn.com/6/0/logo.png",
          link: "http://www.longtailvideo.com/jwpabout/?a=l&v=" +
               jwplayer.version + "&m=f&e=a"
   },
   abouttext: "JW Player " + jwplayer.version,
   aboutlink: "http://www.longtailvideo.com/jwpabout/?a=r&v=" +
              jwplayer.version + "&m=f&e=a",
   sharing: {
         code: encodeURI("<iframe
     src='http://content.bitsontherun.com/videos/lWMJeVvV-364767.mp4' />"),
         link: "http://www.longtailvideo.com/jw-player/"
       }
});
</script>
```

Este código, que se ejecuta al cargar la página, provoca que se muestre el reproductor. No es importante entenderlo; simplemente, observemos la URL resaltada en rojo:

http://content.bitsontherun.com/videos/lwmjevvv-364767.mp4

Esta URL indica el fichero del vídeo que el reproductor invocará para descargarlo y reproducirlo. Al ser el código visible en las páginas HTML, resulta muy fácil inspeccionar su contenido. Si se pone esta URL en un navegador, el vídeo se reproducirá en el reproductor por defecto (y no el de longtailvideo.com), y se puede utilizar cualquier utilidad para descargarlo en nuestro ordenador.

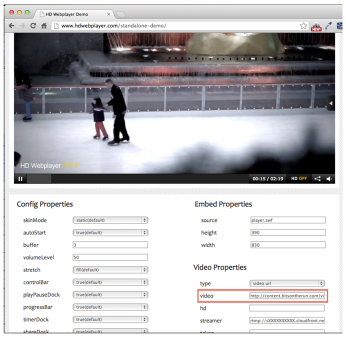
Descarga del fichero utilizando wget

```
$ wget http://content.bitsontherun.com/videos/lWMJeVvV-364767.mp4
--2013-01-25 23:45:35--  http://content.bitsontherun.com/videos/lWMJeVvV-
364767.mp4
Resolving content.bitsontherun.com (content.bitsontherun.com)..q. 66.132.221.169
Connecting to content.bitsontherun.com
(content.bitsontherun.com)|66.132.221.169|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 24377712 (23M) [video/mp4]
Saving to: 'lWMJeVvV-364767.mp4.3'

100%[=============================================================================
====================>] 24,377,712   627KB/s   in 38s

2013-01-25 23:46:15 (627 KB/s) - 'lWMJeVvV-364767.mp4.3' saved
[24377712/24377712]
```

Veamos cómo se puede hacer *hotlinking* fácilmente. Vamos a la web de otro desarrollador de reproductores, HDwebplayer; tienen una página que permite probar las características del reproductor:

http://www.hdwebplayer.com/standalone-demo/

Si en esta página introducimos la URL anterior en la casilla “Vídeo”, podemos reproducir el vídeo en este reproductor.



Player reproduciendo un vídeo de otro sitio web

Podéis intentar incrustar el reproductor en una página web con el código que proporciona la misma página:

Código para incrustar HDwebplayer en una página web

```
<object id="player" classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
name="player" width="830" height="390">
<param name="movie" value="http://www.hdwebplayer.com/standalone-
demo/components/com_webplayer/player.swf"/>
<param name="wmode" value="opaque" />
<param name="allowfullscreen" value="true" />
<param name="flashvars" value="&skinMode=static&autoStart=true&volumeDock=true&
playPauseDock=true&timerDock=true&controlBar=true&shareDock=true&
stretch=fill&logoPosition=center&
video=http://content.bitsontherun.com/videos/lWMJeVvV-
364767.mp4&title=undefined&buffer=3&volumeLevel=50" />
<param name="allowscriptaccess" value="always" />
<object type="application/x-shockwave-flash"
data="http://www.hdwebplayer.com/standalone-
demo/components/com_webplayer/player.swf"  width="830"  height="390">
<param name="movie" value="http://www.hdwebplayer.com/standalone-
demo/components/com_webplayer/player.swf"  />
<param name="wmode" value="opaque" />
<param name="flashvars" value="&skinMode=static&autoStart=true&volumeDock=true&
playPauseDock=true&timerDock=true&controlBar=true&shareDock=true&
stretch=fill&logoPosition=center&
video=http://content.bitsontherun.com/videos/lWMJeVvV-
364767.mp4&title=undefined&buffer=3&volumeLevel=50" />
<param name="allowscriptaccess" value="always" />
</object></object>
```

En un dispositivo móvil o en un televisor conectado no es tan fácil, puesto que probablemente no se pueda ver el código fuente, pero hay muchas herramientas para inspeccionar el tráfico HTTP entre el equipo y los servidores (por ejemplo, configurando un servidor intermediario[8] para la inspección de tráfico como Paros o ZAProxy).

[8] En inglés, *proxy*.

Como conclusión, es muy fácil que alguien pueda utilizar el contenido que se envía desde los equipos de un servicio para reproducirlo en otro. Con esta técnica no se quiere impedir la reproducción, sino asegurar que solo se hace desde el sitio web o la aplicación del emisor, y evitar el pirateo del contenido por otras webs.

3.1.1. *Swf verification*

Swf verification es una técnica creada por Adobe que intenta asegurar que el contenido solo se reproduce desde un reproductor concreto. El funcionamiento de la verificación del reproductor[9] es simple:

[9] En inglés, *swf verification*.

- El desarrollador que crea el reproductor genera una "firma" utilizando una herramienta suministrada por el proveedor. La firma consiste en una función resumen[10] (SHA256) del reproductor, y la deposita en el servidor que distribuye los vídeos (1). El reproductor se instala en el servidor web (1).

Más detalles

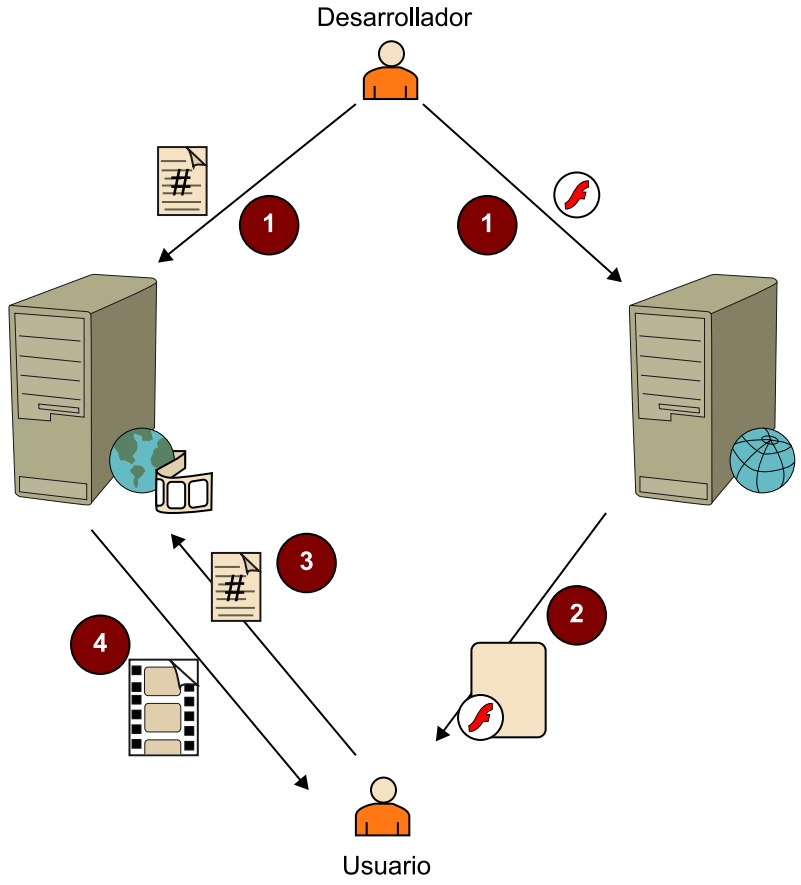
Encontraréis más detalles de cómo se configura una verificación de reproductor en un artículo de Adobe Developer Connection:

"*SWF verification for Protected HTTP Dynamic Streaming*".

- Cuando el usuario que ha navegado en el sitio web (2) utiliza el Player para acceder al vídeo, este envía al servidor del contenido una "firma" generada a partir del Player (3).

- Si esta coincide con una de las que hay almacenadas en el servidor, se autoriza la conexión (4). De lo contrario, se deniega el acceso.

Ciclo de uso de la verificación del reproductor



Esta técnica requiere el uso de:

- Un reproductor basado en Flash.
- Un servidor de *streaming* Adobe Flash Media Server.
- Un protocolo de transmisión propietario de Adobe RTMP (o alguna de sus variantes).

Pros:

- Fácil de implementar. No requiere cambios en el desarrollo ni implica limitaciones adicionales.

Contras:

- Está limitado al uso de servidores de Adobe.

(10)En inglés, función *hash*.

3.1.2. *Secure token*

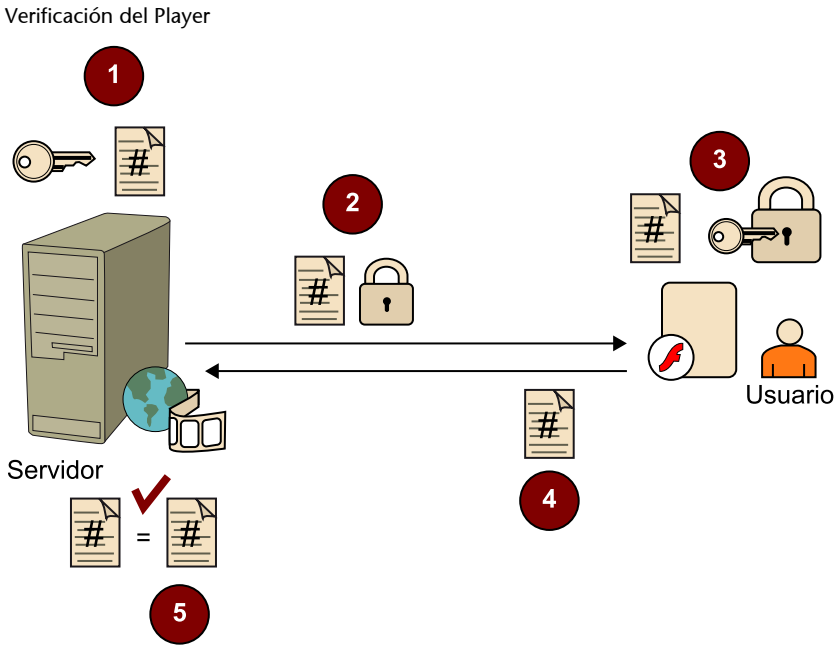
Secure token es una solución similar a la anterior, pero desarrollada por Wowza[11]. Se basa en el intercambio de una información *token* cifrada entre servidor y reproductor mediante una clave compartida que tienen tanto el servidor como el reproductor.

[11]Wowza Media Server es un servidor de *streaming* compatible con Microsoft, Adobe y Apple.

Más detalles

Encontraréis más detalles de cómo se puede implementar la técnica del *secure token* en:

"How to protect RTMP streaming using SecureToken (ModuleSecureToken)"



El servidor genera un código que es cifrado con una clave (1) y se envía al cliente al iniciar la conexión (2). El reproductor, utilizando la misma clave, extrae el código (3) y lo devuelve al servidor (4). Este comprueba si es igual a la que él ha enviado (5) y autoriza o no la conexión.

Esta técnica requiere el uso de:

- Un reproductor basado en Flash.
- Servidor de *streaming* Wowza Media Server.
- Protocolo de transmisión propietario de Adobe RTMP (o alguna de sus variantes).

Pros:

- Impide el *hotlinking*.

Contras:

- Hay que modificar el reproductor para implantar esta técnica.

- El código Flash es fácil de descompilar y, por lo tanto, también lo es acceder al código fuente y encontrar la clave utilizada para descifrar. Con esto se puede violar la seguridad. Es posible mejorar la seguridad utilizando ofuscadores de código, que hacen más difícil el proceso de descompilar, pero no imposible.

Implementación de la descompilación

Una búsqueda de Google por "swf decompiler" devuelve todo lo necesario para saltarse la protección.

- Está limitado al uso de servidores de Wowza.

3.1.3. Desarrollos a medida

Para tecnologías que no implementan estas opciones, se pueden desarrollar soluciones como *secure token* sin demasiada dificultad, añadiendo código desarrollado a medida en servidor y cliente, pero con unos niveles de protección bajos.

Como conclusión, el *hotlinking* se puede prevenir hasta un nivel, pero las técnicas expuestas, sin combinarlas con otras, no ofrecen un nivel de seguridad importante. Estos desarrollos a medida están pensados para evitar el *hotlinking*, no la copia del contenido.

3.2. Restricción de dominio

La restricción de dominio es una problemática similar a la anterior. Pretende evitar el caso de que alguien pueda tomar el código que permite incrustar el reproductor en una página web y copiarlo en otro lugar. En este caso, las técnicas de prevención del *hotlinking* no funcionarían (puesto que no se intenta evitar el reproductor; se utiliza el reproductor legítimo, pero desde otro sitio web). Aquí aparecen las restricciones de dominio, que se basan en especificar una lista de dominios "de confianza" desde los que se permite acceder al contenido. Esta técnica se implementa de distintas maneras en función de la tecnología utilizada.

3.2.1. Flash

Una aplicación basada en Flash, antes de establecer una conexión en un servidor diferente del que se ha descargado el código, hace una petición HTTP solicitando un fichero denominado crossdomain.xml, que tiene que existir en la raíz de servidor. En este fichero se indica la lista de dominios autorizados. El complemento de Flash hace la comprobación y, si no aparece el dominio de la página en el que está incrustado, rechaza establecer la conexión. Esta protección es aplicable no solo a conexiones para acceder a contenido audiovisual, sino también a cualquier tipo de conexión.

Más detalles

Los detalles de la sintaxis del fichero crossdomain.xml se pueden encontrar en la dirección siguiente:
"*Cross-domain policy for Flash movies*"

3.2.2. Microsoft Silverlight

El complemento de Microsoft incorpora una solución similar, pero en este caso el fichero se denomina clientaccesspolicy.xml.

Más detalles

Encontraréis más detalles sobre el fichero clientaccesspolicy.xml en la dirección siguiente:

"Making a Service Available Across Domain Boundaries"

Estas dos soluciones se basan en que el complemento (la pieza de software instalada en el equipo del usuario y suministrada por el fabricante) implementa esta seguridad y rechaza hacer acciones no autorizadas. Este hecho resulta muy habitual (por ejemplo, los navegadores tienen restricciones similares respecto a qué conexiones pueden hacer los programas en Javascript que ejecutan), pero tiene las limitaciones que ya hemos visto: si se accede al contenido sin utilizar los complementos correspondientes, estos mecanismos no protegen. Herramientas de descarga como rtmpdump o wget pueden bastar para saltarse la protección. Son un primer nivel de protección, pero si se quiere seguridad hay que combinarlos con otras protecciones.

3.2.3. Comprobación del Referer

Hay otra solución para los casos siguientes.

- El reproductor no está basado en uno de los complementos anteriores (como por ejemplo Javascript, una aplicación de móvil, una consola o un televisor conectado).

- El protocolo de transmisión es HTTP y, por lo tanto, el servidor es un servidor web genérico (Apache, nignx, etc.).

Para entender bien la base de esta técnica, hay que comprender un aspecto determinado del protocolo HTTP, que es el que utilizan los navegadores. El navegador almacena la URL de la página que presenta al usuario (y pone esta información a disposición del código que se ejecuta dentro de la página, como hemos visto en el caso anterior). No obstante, en todas las llamadas HTTP que hace mientras se está en la página, esta información se enviará al servidor en la cabecera del mensaje, dentro del campo Referer.

Ejemplo de cabecera HTTP con Referer

```
GET /dd361754.MVP_DKurata.jpg HTTP/1.1
Host: i.msdn.microsoft.com
Connection: keep-alive
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_8_2) AppleWebKit/537.17
(KHTML, like Gecko) Chrome/24.0.1312.56 Safari/537.17
Accept: */*
Referer: http://msdn.microsoft.com/en-US/aa497440
Accept-Encoding: gzip,deflate,sdch
Accept-Language: en-US,en;q=0.8
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.3
```

En el ejemplo anterior se puede ver una llamada para obtener una imagen jpg, y entre otra información aparece el campo Referer con una URL que corresponde a la página activa.

Con esta información, el servidor utiliza reglas de configuración (que la práctica totalidad de servidores permiten) o desarrolla una pequeña aplicación para identificar de dónde llega la petición, y la acepta o no, de modo que obtiene un resultado como el anterior.

El siguiente ejemplo muestra cómo se limitaría el acceso a los ficheros con extensiones .mp4 y .mp3 de un servidor solo a peticiones que lleguen de www.midominio.com.

Reglas para limitar el acceso por Referer a Apache

```
RewriteCond %{HTTP_REFERER} !www.elmeudomini.com [NC]
RewriteRule \.(mp4|mp3)$     -    [F,NC]
```

Esta protección funciona correctamente mientras los clientes tengan el comportamiento esperado. Esto es así habitualmente, pero resulta muy fácil, utilizando herramientas disponibles para todo el mundo, enviar peticiones a un servidor y alterar el campo Referer a voluntad y saltarse así la protección. Como las anteriores, es una protección para el usuario habitual, pero no se puede considerar que ofrezca protección contra un usuario con unos conocimientos técnicos mínimos. Por ejemplo, utilizando curl, una utilidad muy popular y disponible para casi todos los sistemas, se puede hacer una petición para cambiar el Referer como sigue:

Petición con un campo Referer inventado

```
curl --referer http://www.elmeudomini.com/inventada.html -o video.mp4
http://www.elmeudomini.com/video.mp4
```

3.3. Cifrado

Otro punto de vulnerabilidad es el camino entre el servidor y el reproductor. Los bytes que circulan pasan por múltiples redes, ninguna de las cuales está bajo el control del emisor, de modo que en estos puntos el contenido puede ser interceptado y copiado. Para evitar este problema, la solución consiste en cifrar la comunicación entre estos dos puntos.

El proceso de cifrado varía en función del protocolo utilizado, pero generalmente hay que cifrar:

- RTMP para reproducción en tiempo real[12] con tecnología Flash.
- HTTP para HLS, Smooth Streaming, descarga progresiva o Adobe HDS.

(12) En inglés, *streaming*.

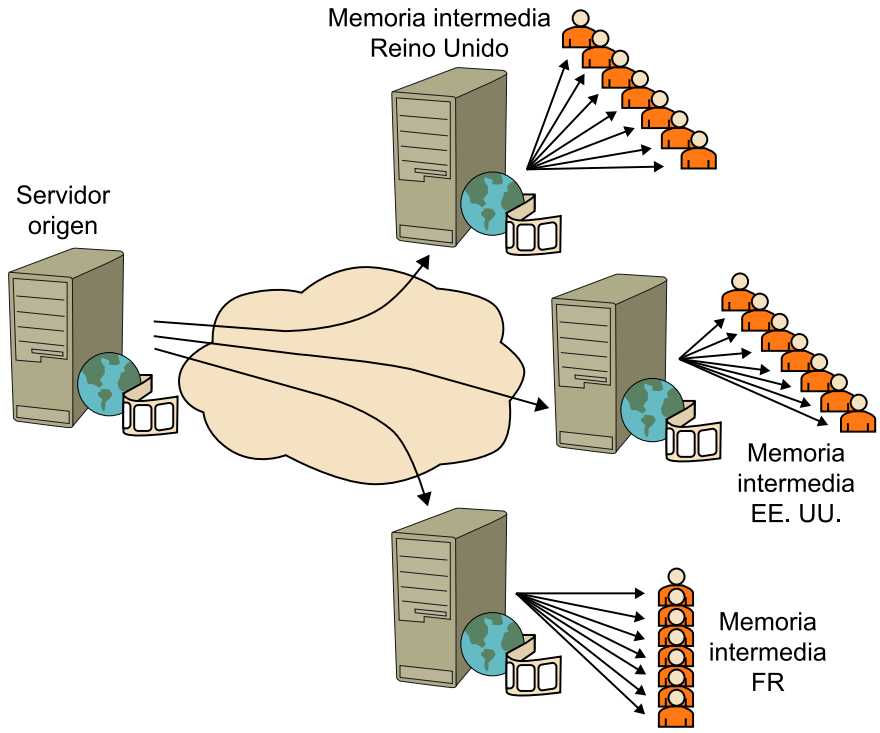
La tendencia actual del mercado

Actualmente, hay una tendencia en el mercado a evolucionar hacia la descarga en tiempo real por internet basada en HTTP y abandonar protocolos diseñados nativamente para descarga en tiempo real (como RTSP, RTMP –Adobe– o MMS –Microsoft–). Esto se produce porque HTTP es el protocolo más utilizado en internet y hay una infraestructura de servidores gigantesca que puede distribuir contenido utilizando este protocolo. Si los protocolos de reproducción en tiempo real utilizan HTTP, se puede usar esta infraestructura común y no es necesario instalar y mantener equipamiento distinto para cada tecnología. Esto es un gran ahorro para empresas como las CDN (*content distribution networks*), que tienen que invertir mucho dinero para dar servicios de reproducción en tiempo real.

El hecho de que diferentes tecnologías utilicen HTTP no presupone que sean iguales o compatibles. Cada fabricante desarrolla una solución distinta, pero todas las soluciones tienen en común el hecho de que pueden compartir una base de servidores HTTP estándar.

La otra gran ventaja de HTTP es que soporta el *caching* de contenidos, es decir, es posible almacenar copias de los vídeos (completos o partes) en servidores cerca del usuario final y así minimizar los costes de transmisión de datos y la carga de la red. Esto permite una reducción de costes y una mejora de la calidad.

Uso de memoria intermedia (*cache*) para optimizar la distribución



Esto se entiende mejor con un ejemplo. Supongamos que queremos distribuir un contenido desde un servidor situado en Barcelona, pero una gran parte de nuestros usuarios está en otros países. Esto significaría que para cada usuario concurrente, tenemos un flujo de datos que atraviesa varios países. Con 1.000 usuarios concurrentes desde Francia, Gran Bretaña y Estados Unidos, y si el vídeo tiene una velocidad de transmisión (*bitrate*) de 1.000 kbps, el resultado es 1 Gbps de tráfico internacional. Si podemos tener servidores de memoria intermedia (*cache*) cerca del usuario final, en el mejor de los casos dispondremos de tres flujos de tráfico internacional entre el servidor origen y los tres servidores de memoria intermedia. El resto del tráfico será local dentro de cada país.

Obviamente:

- El coste es más bajo (local frente a internacional).
- El servidor origen no tiene que soportar toda la carga, sino solo una pequeña parte, lo que facilita su escalabilidad.

- La calidad del servicio será mejor, puesto que una transmisión internacional tiene más riesgo de presentar problemas (pérdida de paquetes, latencia, retraso –*jitter*–) que una local.

En los dos casos, el cifrado se hace utilizando el protocolo de seguridad TLS. En el caso de HTTP, esto significa el ya conocido HTTPS; y en el caso de RTMP, RTMPS (Secure RTMP).

Este protocolo permite garantizar lo siguiente.

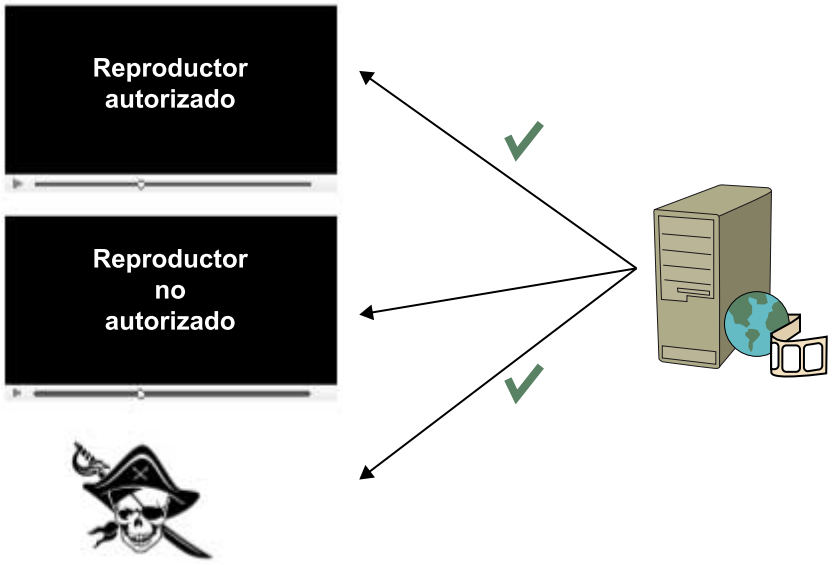
- La comunicación no es espiada.
- La comunicación no es alterada por el camino.
- El origen es realmente el que afirma ser (no hay usurpación de identidad).

Sin embargo, a pesar de esto, el cifrado por sí solo no protege de una descarga indebida. El motivo es que las mismas aplicaciones/reproductores, legítimos o no, pueden hacer llamadas tanto a URL seguras como no seguras. Hagamos un símil para entenderlo: desde un navegador, podemos hacer una llamada a http://www.google.com. Si Google decide cambiar y utilizar un protocolo cifrado (para evitar que se puedan espiar las consultas que los usuarios hacen en el buscador y garantizar su confidencialidad[13]), y pasa a **https**://www.google.com, no hay nada que impida acceder al servicio. Lo que el cifrado garantiza es que uno no espíe al otro, pero no limita el acceso.

(13)De hecho, esto ya es así. Encontraréis una explicación de los motivos en "*Making search more secure*".

3.3.1. Autentificación

Las técnicas y aplicaciones concretas de técnicas que hemos visto hasta ahora trataban igual a todos los usuarios; o mejor dicho, no trataban un aspecto básico como el control de "quién" intenta utilizar el servicio. El resultado es que podíamos controlar algunos accesos mientras se utilizara un reproductor de confianza o el complemento (Flash o Silverlight) suministrado por el fabricante. Sin embargo, si se utilizaba alguna otra aplicación, saltarse la protección resultaba fácil



La introducción de una identificación del usuario –aparte de ser una necesidad de muchos servicios– nos permite mejorar el control del acceso al contenido, puesto que además de controlar el dispositivo que se conecta, podemos validar quién lo hace.

La identificación del usuario se utiliza en las situaciones siguientes:

a) Para la práctica totalidad de sistemas que requieren pago. La mayoría lo hacen por una modalidad de suscripción, en la que el usuario se da de alta y se tiene que identificar cada vez que accede al servicio.

b) Una variante del caso anterior son los sistemas que funcionan por micropagos; es decir, en lugar de una suscripción se hace un pago puntual cada vez que se quiere acceder al servicio (por ejemplo, pagar para alquilar una película), lo que por lo tanto resulta más anónimo. Esta modalidad se popularizó hace años con micropagos por SMS y después ha evolucionado hacia pagos por tarjeta de crédito o PayPal. El hecho de no conocer el nombre del usuario no tiene un impacto técnico, puesto que en realidad se puede considerar que tenemos un usuario válido solo para una acción.

c) Finalmente, hay servicios que no exigen pago pero sí piden registrarse como usuario, con el objetivo de a) proteger el contenido y b) hacer un perfil del usuario y poder ofrecerle contenido (y publicidad, que en el fondo es lo que sustenta económicamente el servicio) personalizado.

Autentificación en HTTP

El fundamento de la autentificación en HTTP, que es un protocolo sin estado, se basa en los elementos siguientes:

- El cliente envía una petición con los datos para identificar al usuario (normalmente un nombre de usuario[14] y una contraseña[15]) al servidor.

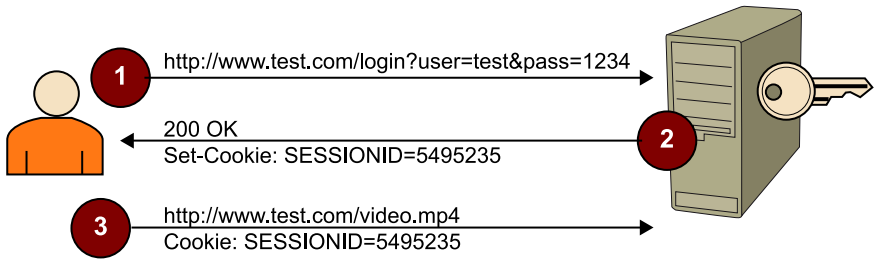
- Este valida que sean correctos (según la lógica de negocio propia de cada caso), contesta al navegador cliente y adjunta una cookie que este almacenará y enviará en todas las peticiones HTTP siguientes que haga a este servidor.

- Una vez validado el usuario, el servidor almacena una información de estado para recordar que el usuario está conectado y esta información está enlazada con el valor de la cookie de modo que, por cada petición, podrá saber de qué usuario se trata. La cantidad de información de estado que tenga dependerá de las necesidades de cada aplicación.

(14) En inglés, *login*.

(15) En inglés, *password*.

Proceso de autentificación HTTP



Este es el funcionamiento definido en el estándar de HTTP (RFC-6265), que hemos simplificado para que sea fácil de entender. Se utiliza tanto en entornos web como en móviles o televisiones conectadas, puesto que todos utilizan HTTP como base.

Más información

Una explicación más comprensible del funcionamiento de las cookies se puede encontrar en Wikipedia:

"HTTP cookie"

Una vez hecha esta autentificación, ya tenemos una seguridad de quién es el cliente, mucho más de lo que podíamos controlar con los mecanismos anteriores.

Sin embargo el escenario no suele ser tan simple, porque habitualmente el servidor que hace la validación del usuario (que contiene la aplicación) y los que proporcionan la reproducción en tiempo real no son los mismos. Esto se debe a lo siguiente:

- Los diferentes servidores pueden utilizar distintas tecnologías. Los lenguajes que se utilizan para el desarrollo web (PHP, Java, Ruby, .NET, etc.) requieren servidores web que no tienen por qué coincidir con los de los servidores de *streaming*. Además, para protocolos diferentes de HTTP, hacen falta servidores específicos.

- La tipología de tráfico de páginas web y la de reproducción de vídeo en tiempo real son muy distintas. La primera hace muchas peticiones pequeñas, con un volumen de tráfico bajo, y la segunda, pocas peticiones de

gran volumen. Por este motivo, es adecuado separar los servicios en equipos diferentes.

- Por escalabilidad, suele ser necesario instalar múltiples servidores para la distribución de *media*.

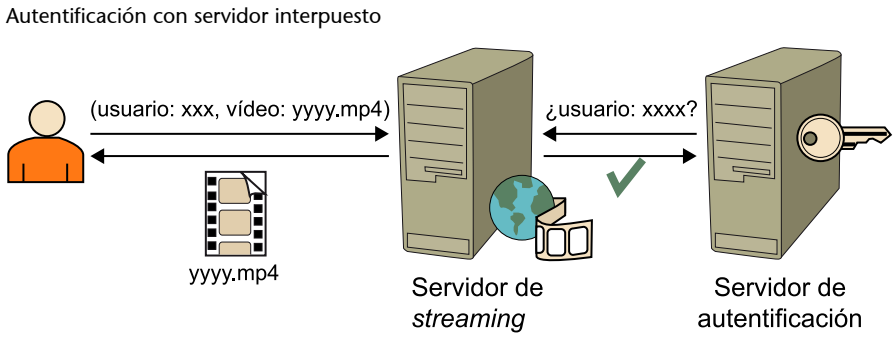
En esta situación tenemos los servidores de *streaming* separados de los de la aplicación, lo que obliga a buscar un sistema para validar el acceso al contenido.

También hay que añadir, antes de ver cómo se resuelve esta situación, un factor adicional. La distribución de contenidos es una tarea que requiere mucho ancho de banda y una gran cantidad de servidores especializados en función de la tecnología utilizada. Se trata de una gran inversión que muchas empresas prefieren evitar, de modo que alquilan el servicio de distribución de contenidos a empresas especializadas denominadas *content distribution networks* (*CDN*). Para utilizar servicios de terceros, es muy importante poder separar la lógica de la aplicación (propia de cada cliente) del acceso autorizado al contenido (que puede ser común a múltiples clientes).

Lo que se intenta hacer es implementar un proceso de validación de la manera más simple y genérica posible, añadiendo una información extra al URL del vídeo (en una consulta o en una cookie) que permita validar el acceso. Las dos opciones más habituales son las siguientes.

Opción A. El servidor de *streaming* verifica los permisos

En esta opción, el servidor de *streaming* está interpuesto entre el usuario y el servidor de autentificación.



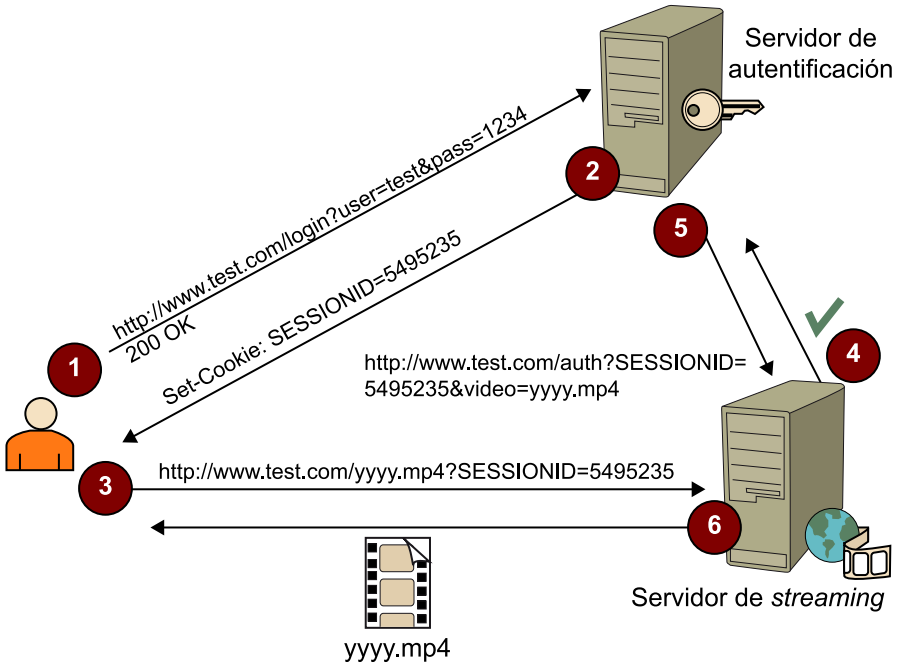
La información que envía el cliente al servidor de *streaming* puede ser un nombre de usuario más una contraseña, pero se trata de una opción poco segura. Más bien, se suele enviar alguna identificación de sesión que el navegador ha obtenido de una fase previa de autentificación. Lo importante es que el servidor de *streaming* no tiene una lógica de validación de usuarios; simplemente, recoge los parámetros recibidos del cliente y construye una llamada HTTP hacia el servidor de autentificación, el cual retorna una respuesta de OK/KO.

Más información

Un ejemplo de servidor que implementa esta técnica es erlyvideo. Se puede encontrar una descripción cuidadosa del proceso que describimos aquí en:

"Authentication and authorization".

Autentificación con servidor interpuesto, detallado



Cuando el servidor de *streaming* está proveído por una CDN, esta define una API con los parámetros que enviarán al servidor de autentificación y la respuesta que espera; al configurar el servicio, se indica cuál es la dirección del servidor de autentificación del cliente y ya se puede operar. El problema de esta opción es que si se producen problemas con el servidor de autentificación, estos se trasladan al servidor de *streaming*. Es un sistema altamente acoplado.

Opción B. Autentificación por *token*

Se trata de un método diferente para conseguir el mismo objetivo. Se basa en el hecho de que el servidor de *streaming* sea capaz de hacer una validación simple de manera autónoma. Por este motivo, el servidor de autentificación y el de *streaming* comparten una clave que permite cifrar un mensaje de manera segura:

1) El cliente solicita la URL del vídeo.

2) Se construye la URL del vídeo con una *querystring* que contiene los datos que indican el acceso que se concede al cliente. Por ejemplo (cada implementación de este método puede variar los parámetros):

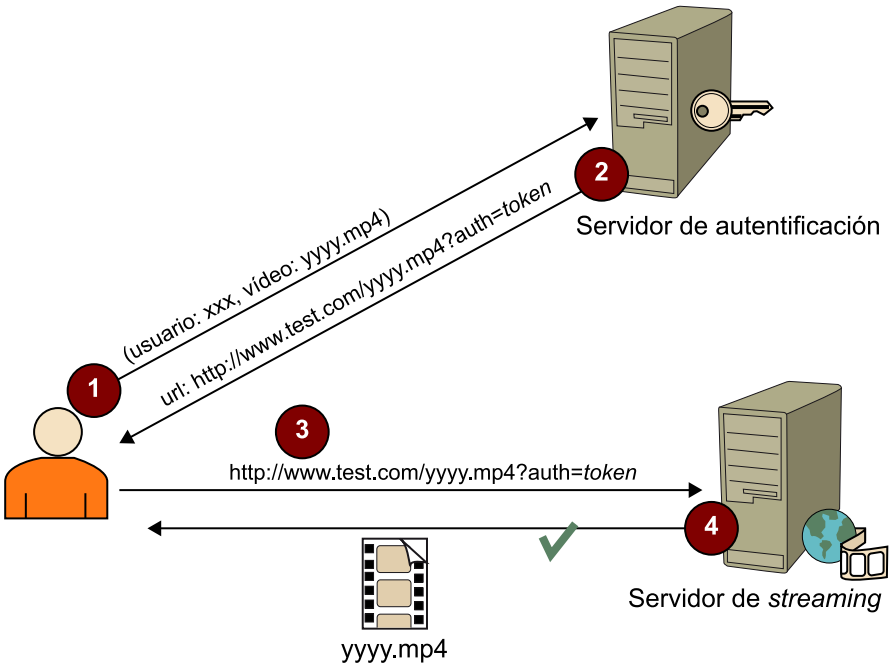
- Inicio del periodo de visualización.
- Final del periodo de visualización.
- IP autorizada.

Estos parámetros (a los que a veces se añade algún texto al azar para dar más solidez al proceso de cifrado) se cifran con la clave compartida y se adjuntan como *querystring*.

3) El cliente hace la llamada al servidor de *streaming* pasando la URL anterior. Este descifra la *querystring* con la clave compartida y verifica si la petición cumple los criterios indicados.

4) Si es así, inicia la reproducción.

Autentificación por *token*



En esta opción se consigue dotar de autonomía al servidor de *streaming*, que por sí solo puede verificar las peticiones. Por lo tanto, suele ser preferida sobre la primera que hemos visto.

En estos dos esquemas, el punto de vulnerabilidad reside en la validación del usuario. Si podemos garantizar que solo los usuarios válidos pueden acceder al sistema desde los reproductores válidos, el hecho de añadir autentificación al *streaming* da un buen nivel de seguridad. Hay que evitar que se pueda obtener una URL autenticada, puesto que entonces tendremos las mismas posibilidades de acceso ilegítimo al contenido.

Por este motivo, hay que combinar diferentes mecanismos para proteger el contenido: verificación del reproductor para garantizar que se utiliza el cliente adecuado y cifrado para evitar que se pueda interceptar el contenido. Si se pueden combinar las tres técnicas, entonces es posible ofrecer un buen nivel de seguridad, incluso en servicios sin identificación nominal de usuario:

- El servidor de *streaming* solo entregará el contenido si hay *token*, y al reproductor que tenemos verificado (si el reproductor no deja registrar el contenido, no habrá fugas).

- El cifrado garantiza que no se intercepta.

Los dos sistemas presentan dificultades de implementación en dos casos:

1) HTTP Streaming: en estos protocolos, no hay una sola conexión entre reproductor y servidor, sino muchas. Esto implica dos cosas:

a) Para garantizar la seguridad, hay que añadir el *token* de seguridad a cada petición, y el servidor tiene que comprobar cada una de estas peticiones, lo que supone una sobrecarga importante. Es viable, pero más complicado.

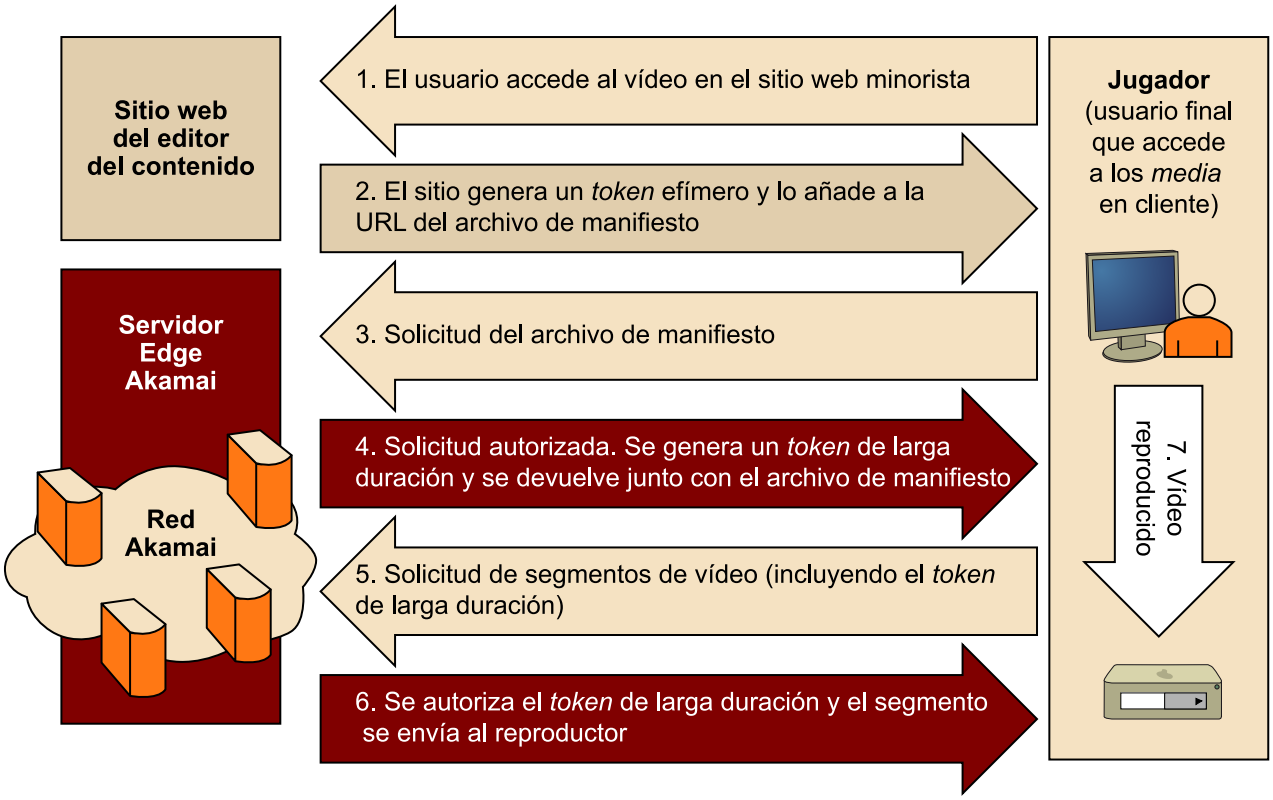
b) Aparece un problema de tiempo de vigencia del *token* de autorización. Habitualmente se acostumbra a dar un tiempo corto, para evitar que alguien pueda copiar la URL y reutilizarla o enviarla a una tercera persona. Esto funciona si solo hay una conexión al principio, pero en HTTP *streaming* se hacen múltiples conexiones, durante todo el tiempo que dura la reproducción. Esto se puede resolver si el servidor de *streaming* genera *tokens* de duración más larga para los segmentos de fichero.

Más información

En HTTP Streaming, un fichero de vídeo es segmentado en fragmentos de pocos segundos, cada uno de los cuales se solicita en una llamada HTTP separada y es el reproductor el que vuelve a unir los fragmentos. Esto permite que el cliente pueda optar por solicitar fragmentos de diferentes calidades en función de las condiciones (ancho de banda, CPU, etc.) Para más detalles, podéis consultar:

"Adaptive bitrate streaming".

HTTP Streaming autentificado. Implementación de Akamai



2) Cachés: ya hemos comentado que la gran ventaja de la descarga progresiva y del HTTP Streaming era la facilidad de almacenar el contenido en memorias caché cerca del cliente. Sin embargo, la necesidad de autentificar el acceso complica la situación, puesto que ahora se tiene que impedir que las memorias caché almacenen el contenido si no son capaces de validar las peticiones. Es viable, pero ya obliga a añadir la lógica de validación y distribuir las claves secretas a todas las memorias caché. A esto tenemos que añadir que, si se utiliza HTTPS, el uso de memorias caché se hace mucho más complicado.

Autentificación en HLS

HLS, el protocolo de HTTP Live Streaming desarrollado por Apple y posteriormente liberado como un estándar, incorpora un mecanismo de seguridad adicional que se combina con el cifrado para garantizar un nivel de seguridad elevado. Se basa en principios similares a los expuestos anteriormente, pero HLS incorpora la securización en el estándar.

Más información

Más detalles sobre el protocolo HTTP Live Streaming en "HTTP Live Streaming". HLS es una de las variantes de HTTP Streaming, como Smooth Streaming o HDS.

HLS permite cifrar cada uno de los fragmentos de vídeo utilizando un algoritmo de clave simétrica, y ofrece una manera de indicar en el fichero de manifiesto una URL que el reproductor podrá invocar para obtener la clave para descifrar el contenido.

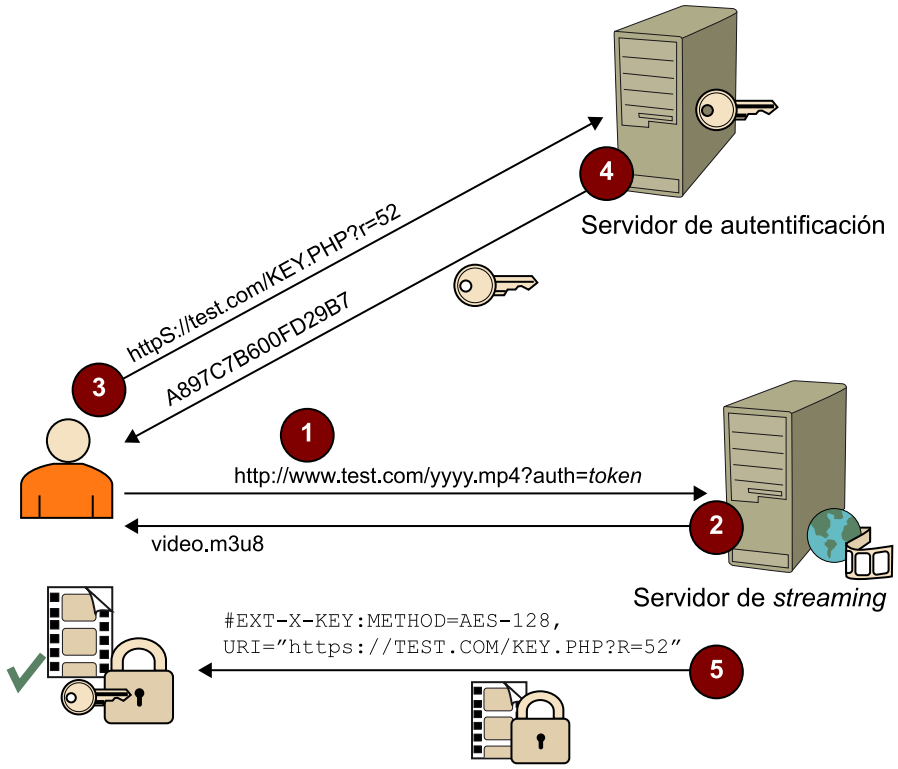
La sintaxis de la cabecera que especifica el cifrado es:

```
#EXT-X-KEY:METHOD=<method>[,URI="<URI>"][,IV=<IV>]
```

Y un ejemplo de utilización:

```
#EXT-X-KEY:METHOD=AES-128,URI=https://priv.example.com/
key.php?r=52
```

Flujo en HLS autentificado



El proceso es:

1) Se pide el fichero de manifiesto al servidor.

2) Este retorna el fichero que indica que el contenido está cifrado y la URL en el que obtener la clave de descifrado.

3) Se hace una llamada a esta URL.

4) El servidor retorna la clave para descifrar.

5) Se solicitan los fragmentos, que el reproductor descifra con la clave.

Observemos que este sistema adolece del mismo problema que los anteriores. Si podemos obtener la URL y no disponemos de validación del reproductor (y HLS no lo incorpora), podemos invocar la URL, descargar el contenido y descifrarlo. Para hacerlo de manera segura, hay que combinarlo con una técnica de autentificación y filtrado por usuarios. Sin embargo, por lo menos el problema con las memorias caché y el tiempo de vida del *token* está solventado.

De qué manera implementarlo, ya es decisión de cada cual pero, por ejemplo, el reproductor puede añadir un parámetro más a la URL que envía al servidor de autentificación para que incorpore una cookie de sesión. De este modo, el servidor no dará la clave de descifrado si el usuario no se ha identificado.

Otro problema reside en el hecho de que si la clave de cifrado es fija y se produce una filtración de esta, todo el contenido queda comprometido. Para afrontar este problema hay que optar por claves de cifrado distintas para cada fichero o por variables en el tiempo, pero a expensas de complicar el escenario, ya que esto requiere una base de datos que relacione cada fichero con la clave de descifrado correspondiente.

Una ventaja de este método es que no tiene impacto negativo en las memorias caché, dado que estas almacenan un contenido cifrado y no tienen que comprobar a quién lo entregan (es más, no requiere el uso de HTTPS puesto que el contenido ya está protegido por el cifrado). No hay que implantar ningún mecanismo de securización a las memorias caché.

Reflexión

Querríamos volver a incidir en la diferencia entre cifrar la comunicación y cifrar el contenido. El primer caso –del que HTTPS es un ejemplo– se utiliza para evitar la manipulación e interceptación de la transmisión, pero no se requiere al usuario la posesión de una clave para acceder al contenido de la transmisión. El cifrado es transparente para el usuario. Por el contrario, el cifrado del contenido exige al usuario que disponga de una clave para acceder al contenido.

Conclusiones sobre la autentificación

Pensamos que este punto es un buen momento para hacer una reflexión sobre algo que probablemente ya habréis observado:

a) Hay una gran fragmentación en el mercado. Aparte de diferentes tecnologías (HLS, RTMP, Smooth Streaming, etc.), cada una tiene distintas opciones de securización que, además, varían en función del software que se utilice (por ejemplo, con RTMP es posible elegir entre servidores de Adobe, Wowza u otros).

b) Algunas de las opciones de securización que hemos visto no son estándares, y ni siquiera son tecnologías desarrolladas por los fabricantes, sino prácticas de uso comunes que los desarrolladores suelen implementar, cada uno ajustándolas a las necesidades concretas de su caso.

c) Es un sector muy cambiante. Los fabricantes sacan nuevas versiones de los productos con mucha frecuencia, a lo cual hay que añadir la aparición de muchas implementaciones de código abierto[16] que aportan más opciones. Probablemente, cuando leáis este texto ya habrán aparecido versiones nuevas de los principales productos.

(16)En inglés, *open source*.

En resumen, nos hallamos con un rompecabezas de difícil solución. Para conseguir seguridad tenemos que combinar técnicas que dependen tanto de la tecnología utilizada como del fabricante de los servidores. Si en lugar de implantar nuestra propia solución se alquilan servicios a una CDN, también se genera una dependencia por lo que respecta a las opciones que esta ofrece.

3.3.2. Geobloqueo

En el mundo audiovisual se dice que "el contenido es el rey", y esto significa que el elemento más importante es el contenido, por encima del resto de los elementos. Este es el que atrae a la audiencia, y los derechos sobre estos contenidos se negocian casi siempre para un territorio concreto. Esto obliga a implementar un sistema que garantice que el contenido solo pueda ser accesible dentro del territorio para el que se dispone de derechos. En esto consiste el geobloqueo.

Los fundamentos del geobloqueo son bases de datos que contienen las ubicaciones geográficas de todas las direcciones IP existentes, con mayor o menor precisión según el país y la empresa que recaba la información. Estas bases de datos se pueden comprar –o bien alquilarse su acceso por un periodo de tiempo– a múltiples empresas que proveen el servicio. Las más relevantes son Neustar, MaxMind, IP2location o IPligence. De estas, MaxMind e IPLigence tienen versiones gratuitas, con información reducida, como también lo es HostIP, que se trata de una base de datos de código abierto.

Hay dos maneras de utilizar estas bases de datos:

- Descargarlas, instalarlas en un equipo y hacer un desarrollo para acceder a esta información desde las aplicaciones propias.

- Utilizar el servicio directamente, mediante servicios API REST o servicios web[17] que la mayoría proveen (por ejemplo, http://api.hostip.info/get_html.php?ip=12.215.42.19).

(17) En inglés, *web services*.

Sin embargo, el resultado es el mismo.

La implementación de la seguridad se puede hacer en dos lugares:

- **En el reproductor.** En este caso, es el reproductor el que, dentro de su lógica de negocio, comprueba que está ejecutándose dentro de la zona geográfica con derechos antes de solicitar la reproducción del contenido. Esta es la opción más flexible, pero presenta una brecha de seguridad: quizá el reproductor no reproduce el contenido, pero si alguien envía la URL a otra persona fuera del ámbito geográfico autorizado, lo podrá reproducir.

Hay que combinar esta técnica con la siguiente o con la verificación de reproductor.

- **En el servidor del contenido**: en este escenario, es el servidor mismo que envía el contenido el que hace la verificación de la IP del cliente y, por lo tanto, acepta o rechaza la petición. Es más seguro que el anterior pero presenta problemas de flexibilidad (hay que configurar las políticas de geobloqueo en el servidor de vídeo).

Si es posible, y si la configuración de las políticas de seguridad resulta lo suficientemente simple, la segunda opción es mejor puesto que proporciona un nivel de seguridad superior.

4. Los sistemas de gestión de derechos digitales

Hasta ahora, todas las soluciones de protección de contenidos que hemos visto han presentado vulnerabilidades. Algunas son muy evidentes y fáciles de explotar, y otras requieren más esfuerzo. No obstante, ninguna de estas soluciones se podía considerar segura.

¿Soluciones seguras?

Entendemos por *seguro* que no se hayan encontrado vulnerabilidades que hagan que, con una implementación correcta de la tecnología, se pueda obtener un acceso no autorizado al contenido.

Además, todas adolecen de una carencia de capacidad para expresar los tipos de acceso de los que puede disfrutar el usuario; se concede o se deniega el acceso, pero para modelizar tipos de acceso más complejos (por ejemplo, que puedan verse dos veces) hay que hacer desarrollos a medida. Y lo que es más importante, ninguna de las técnicas vistas permite proteger contenido *offline*[18]; es decir, si el contenido se descarga en el dispositivo cliente para ser visualizado más tarde (escenario que se utiliza en dispositivos móviles –que no siempre tienen conexión– o en contenido HD –que no se puede reproducir en tiempo real por limitaciones de ancho de banda–), se pierde su control. El *digital rights management* aparece para solucionar estos problemas.

(18)En español, 'fuera de línea'.

El término ***digital rights management*** (***DRM***) hace referencia a un conjunto de técnicas, herramientas y reglas para controlar el consumo de contenido digital, entendiendo como tal principalmente vídeo y audio, pero también imágenes, documentos o ficheros en general. El objetivo es entregar contenido al consumidor manteniendo el control de los derechos y posibilitar diferentes modelos de negocio.

El DRM intenta expresar las reglas que rigen distintos modelos de negocio audiovisual (alquiler, *pay-per-view*, compra, etc.) que antes se implementaban físicamente (se iba al videoclub, se alquilaba una película, se visualizaba y se devolvía), pero en el entorno digital y garantizando su seguridad. El DRM abarca mucho más que unas técnicas de securización; es un soporte tecnológico para un **tipo de negocio**, y para entender su funcionamiento, hay que empezar por explicar este tipo de negocio.

4.1. Los agentes del DRM

Los agentes implicados en el tipo de negocio mencionado son los siguientes:

- **Consumidor**: el usuario final que "consume" el producto, ya sea por descarga o reproducción en tiempo real.

- **Propietario del contenido**[19]: es la persona u organización que posee los derechos[20] sobre el contenido. Pueden ser grandes productoras, pequeños estudios, canales de televisión o usuarios particulares.

(19)En inglés, *content owner*.

(20)En inglés, *copyright*.

- **Distribuidor**: es la entidad que se encarga de la comercialización y distribución del contenido (una vez protegido) hacia el usuario final (consumidor). Adquiere los derechos pertinentes a los propietarios del contenido y esto le permite comercializar el contenido hacia el consumidor.

- **Preparador de contenido**[21]: es la entidad que se encarga de preparar los contenidos para su distribución por internet (codificación en los formatos necesarios para su distribución por los diferentes medios, cifrado, aplicación de DRM y adición de metadatos). Esta función la puede hacer una empresa dedicada, o a veces la hacen el propietario o el distribuidor.

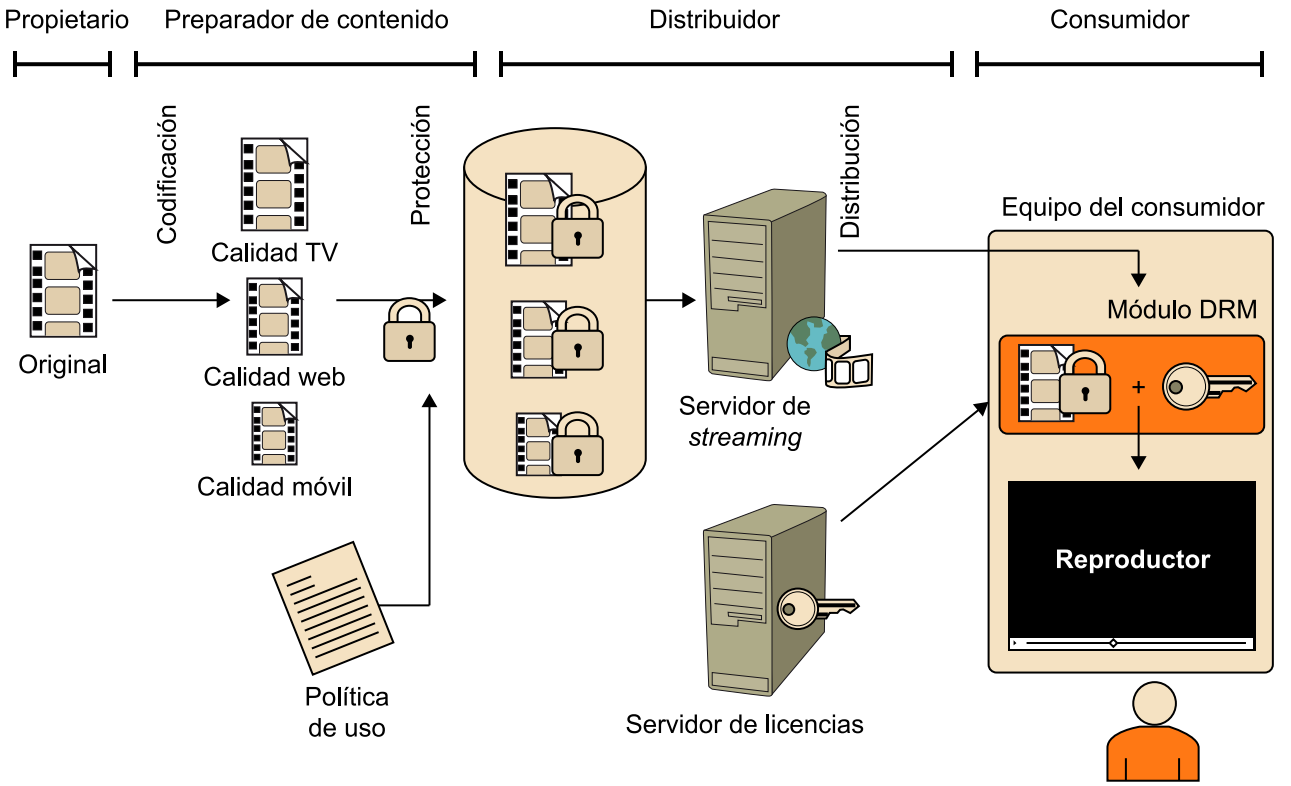
(21)En inglés, *content packager*.

4.2. El ciclo de trabajo del DRM

Los actores interaccionan unos con otros en un ciclo de trabajo[22] como el siguiente. Esta es una versión simplificada que cada empresa acaba desarrollando adaptada a los criterios propios.

(22)En inglés, *workflow*.

Ciclo de trabajo general de DRM



- El propietario dispone de un contenido en formato original, el cual quiere distribuir.

- Este contenido se recodifica a los formatos y las calidades necesarias para su distribución, según las plataformas objetivo.

- El contenido se cifra y enlaza a una o más de una políticas de uso; se obtiene así un **contenido protegido**. La **política de uso** es una definición formal de cómo los consumidores pueden utilizar el contenido, y refleja los derechos que el distribuidor ha obtenido del propietario de un modo que el reproductor es capaz de interpretar y cumplir.

- Estos ficheros se ponen a disposición de los consumidores por medio de servidores de *streaming* o de HTTP.

Hasta aquí tenemos el **ciclo de trabajo de preparación de contenido**, que suele producirse una vez por cada contenido nuevo que se incorpora al sistema.

A partir de este punto, se desencadena el **ciclo de trabajo del consumo**, que se lleva a cabo para cada acceso al contenido:

- En el equipo del cliente, el reproductor tiene un software acoplado que gestiona el DRM. Cuando el consumidor quiere reproducir un contenido, el reproductor pasa la solicitud a este contenido, que al mismo tiempo inicia la descarga del servidor.

- Puesto que el contenido está protegido, se requiere una licencia de acceso al contenido. Por este motivo, el sistema hace una solicitud al servidor de licencias con la información de los derechos de que dispone (si tiene una suscripción al servicio, su identificador, etc.). El servidor de licencias valida los datos presentados y decide si el cliente puede acceder al contenido o no. En caso afirmativo, envía la licencia al cliente. Esta licencia es una clave, específica para aquel reproductor, que descifra el contenido recibido.

- El módulo de DRM descifra el contenido y lo pone a disposición del reproductor, que lo visualiza.

Algunos ya debéis de pensar: ¿esto es seguro? ¿Y si alguien intercepta la licencia? ¿Las claves son las mismas? De momento, hemos obviado estos aspectos para dar una explicación más clara del ciclo. ¡Paciencia! Más adelante, veremos con más detalle cómo se fundamenta la seguridad.

4.3. Modelos de negocio

En el mundo físico, el mercado audiovisual ido desarrollando una serie de modelos –maneras– de hacer negocio. DRM intenta hacer posible estos modelos en el mundo digital (mayoritariamente *online*, pero también *offline*). La forma que tiene de conseguirlo pasa fundamentalmente por la licencia que hemos mencionado anteriormente.

La **licencia** contiene los derechos y las restricciones que definen cómo puede utilizarse el contenido y en qué condiciones.

Por ejemplo, una licencia puede indicar que se puede tener derecho de "reproducción" para un vídeo X, pero no derecho de "grabación en CD", así como que la reproducción es posible en el reproductor desde donde se ha descargado pero no en el reproductor del móvil, y que el derecho de reproducción expira el 31 de mayo del 2013. Si todas las condiciones se cumplen, el reproductor accederá a descodificar el vídeo y reproducirlo.

Con esta capacidad, es posible trasladar los modelos tradicionales al mundo digital. Por ejemplo, un modelo de alquiler como el de un videoclub pasa a modelizarse como una licencia para la reproducción de un vídeo por un periodo de X días, o para Y reproducciones. Un modelo de venta de contenido se modeliza con la generación de una licencia indefinida de reproducción asociada a un dispositivo concreto (para evitar la copia a otros reproductores).

Reflexión

Hablamos de DRM y modelos de negocio con casuísticas del mundo audiovisual, pero estos conceptos también son aplicables a otros sectores. Se trata de un buen ejemplo para compararlo con lo que hemos visto que es el mundo del libro digital, que también funciona con sistemas de DRM adaptados a los modelos de negocio –más simples– del sector.

En realidad, la versatilidad de los sistemas de DRM permite modelos de negocio más variados que los que eran posibles en el mundo físico. Cada fabricante ofrece su propio abanico de posibilidades, pero las más habituales son las siguientes:

- **Suscripción**: el proveedor del servicio (habitualmente el distribuidor de contenido, aunque a veces la comercialización la hace otra empresa) cobra una tarifa plana a un consumidor para acceder a un abanico de contenidos.

- **Compra**: en este modelo, el consumidor compra el derecho para un contenido determinado por un tiempo indefinido, asociado a un dispositivo o un conjunto de dispositivos propiedad del mismo consumidor.

- ***Pay-per-view***: el consumidor accede al derecho para reproducir un contenido concreto una o varias veces.

- **Regalo**: es una variante de los anteriores, en que una segunda persona puede comprar uno de los derechos anteriores en nombre de alguna otra,

que recibirá este derecho como regalo. Este modelo también puede utilizarse en casos de revendedores de contenidos.

4.4. Fundamentos criptográficos

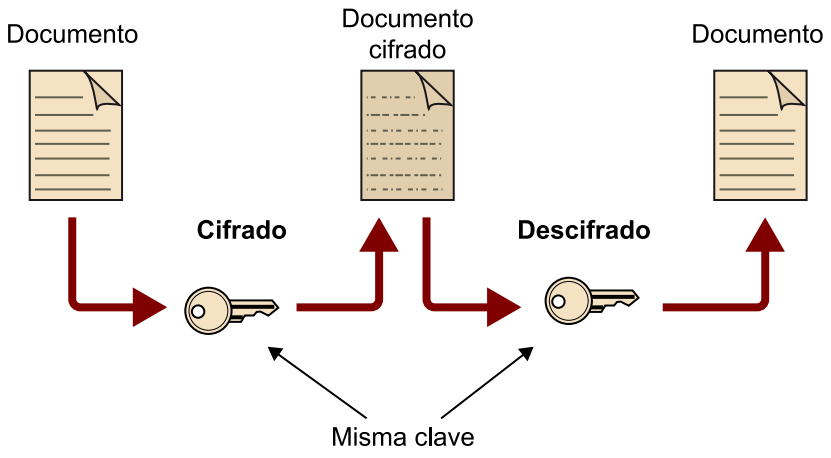
Veamos qué técnicas criptográficas se utilizan en el DRM para conseguir el grado de seguridad requerido. Aquí no explicaremos la base matemática en la que se sustentan y que les confiere la seguridad que se afirma que poseen. Para los descreídos y los inquietos, tendréis referencias complementarias para profundizar en la materia.

Asimismo identificaremos tipos de algoritmos, pero para cada tipo se pueden encontrar distintas implementaciones. Cada fabricante de soluciones de DRM utiliza los que considera más apropiados. No os preocupéis demasiado por esto; los fabricantes están lo bastante interesados en que su solución sea segura como para dedicar los esfuerzos de expertos en la materia a elegir, de entre los algoritmos posibles, un conjunto que ofrezca seguridad suficiente.

4.4.1. Cifrado de clave simétrica

El cifrado de clave simétrica es un conjunto de algoritmos que permite cifrar un contenido utilizando una clave, y descifrarlo utilizando la misma clave. Conceptualmente esto es fácil de entender, y se explica de manera visual en la figura siguiente:

Cifrado de clave simétrica



Estos algoritmos tienen la particularidad de que son computacionalmente económicos; es decir, la capacidad de cálculo necesaria para cifrar o descifrar es reducida, lo que resulta especialmente interesante cuando el volumen de información es elevado y se requiere velocidad.

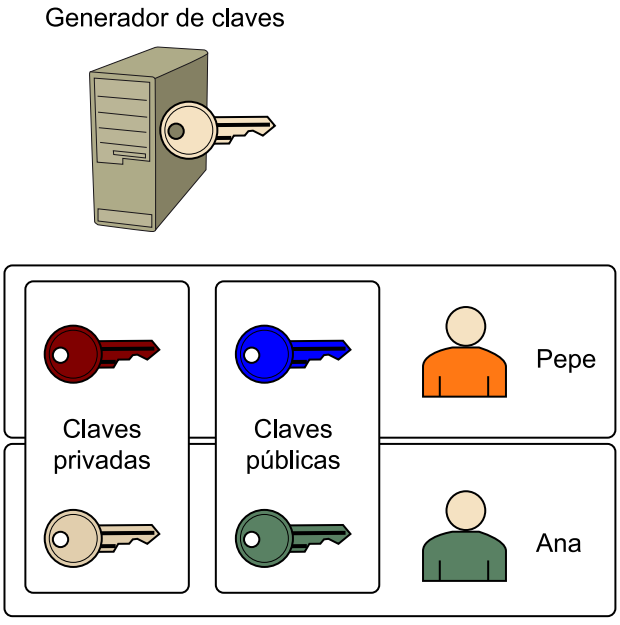
Por el contrario, presentan una dificultad clara. El emisor y el receptor deben tener conocimiento de la misma clave, y esto representa un problema. ¿Cómo se le suministra de manera segura esta clave al receptor para que no sea interceptada? Además, el receptor con conocimiento de la clave puede cifrar contenido, y podría suplantar la identidad del emisor.

Algunos algoritmos de este tipo son AES, 3DES o IDEA.

4.4.2. Cifrado de clave asimétrica

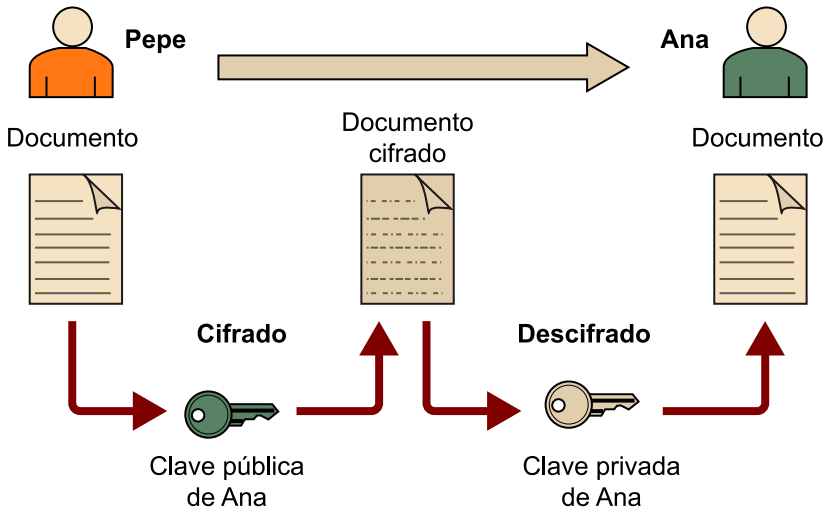
El cifrado de clave asimétrica (o clave pública, como también se le denomina) es una técnica mucho más creativa. Para evitar el problema del secreto compartido que presentaba el sistema anterior, se desarrollan estos algoritmos en los que tanto el emisor como el receptor utilizan dos claves diferentes, relacionadas de manera matemática entre sí. Una de estas se denomina **clave secreta** y la otra, **clave pública**. La primera se utiliza para descifrar (y hay que mantenerla en secreto: solo la tiene quien ha de recibir el mensaje), mientras que la otra se usa para cifrar el mensaje y se entrega a todo el mundo que debe poder enviar mensajes.

Claves en un algoritmo de clave asimétrica



Esto hace desaparecer el problema de la distribución de claves; las claves públicas pueden entregarse a cualquiera y no hace falta ningún mecanismo seguro de transmisión. Veamos un ejemplo de uso, suponiendo que Pepe quiere mandar un mensaje a Ana.

Cifrado de clave asimétrica



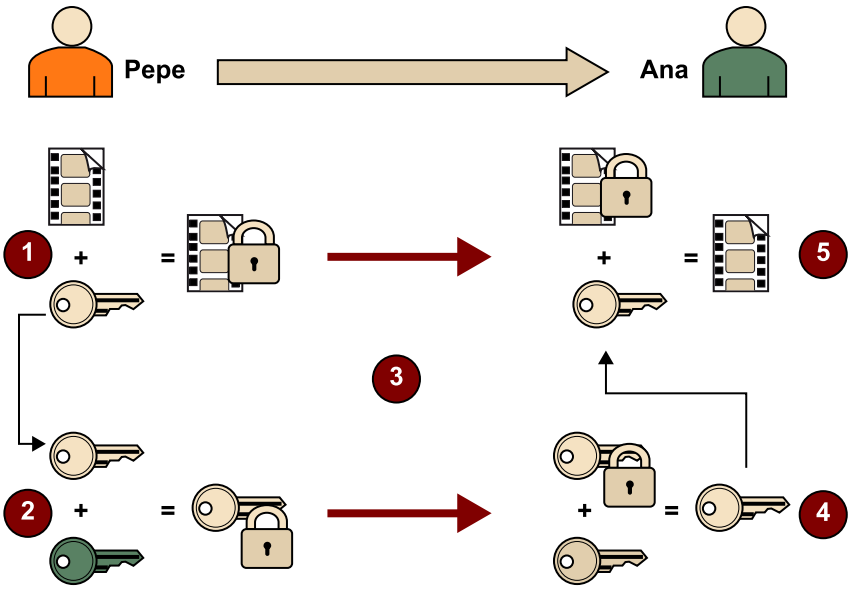
Pepe toma la clave pública de la Ana, que esta ha dado a todos sus conocidos, y la utiliza para cifrar el mensaje. Este mensaje cifrado, lo envía a Ana por un medio no protegido, puesto que solo Ana, con su clave privada, puede descifrar el mensaje.

La contrapartida es que las operaciones que utilizan estos algoritmos son computacionalmente costosas. Por este motivo, se evita cifrar grandes volúmenes de contenidos. No obstante, podemos preguntarnos: ¿qué utilidad tienen, si no se puede cifrar un contenido pesado como un vídeo? Lo que se suele hacer es utilizar cifrado simétrico para cifrar el vídeo, y cifrado asimétrico para cifrar la clave y hacerla llegar al receptor.

La base de estos algoritmos

Si hacemos una simplificación que un matemático podría fácilmente calificar de insultante, la base de estos algoritmos es que hay operaciones matemáticas fáciles y otras difíciles. Por ejemplo, multiplicar dos números es fácil (especialmente para un ordenador) mientras que encontrar los dos números originales a partir del resultado de la multiplicación (factorización) es mucho más complejo; se tienen que hacer muchas más operaciones. Los algoritmos de clave asimétrica se basan en problemas de factorización de números primos muy grandes o funciones logarítmicas discretas que tienen estas mismas propiedades, pero la dificultad de las operaciones es tan grande que se requerirían muchos años para encontrar el resultado.

Cifrado de clave asimétrica + simétrica



Volvemos al ejemplo de Pepe, que esta vez quiere enviar un fichero de vídeo a Ana. Hace lo siguiente:

1) Genera una clave simétrica nueva para esta transmisión, que se utiliza para cifrar el fichero.

2) La clave de cifrado se toma como si fuera un texto cualquiera y se cifra con la clave pública de Ana.

3) Se envían las dos informaciones a Ana. En esta transmisión no hay riesgo, puesto que los dos contenidos están cifrados.

4) Ana utiliza su clave privada para descifrar la clave simétrica.

5) Finalmente, Ana puede utilizar esta clave para descifrar el vídeo.

Ejemplos de estos algoritmos son RSA o ElGamal.

4.4.3. Firmas digitales

El cifrado de clave asimétrica permite hacer otra operación aparte de cifrar, que es la firma digital. Esta operación se utiliza de manera análoga a las firmas manuscritas del mundo físico: para probar la identidad de una persona y para validar la integridad de un documento.

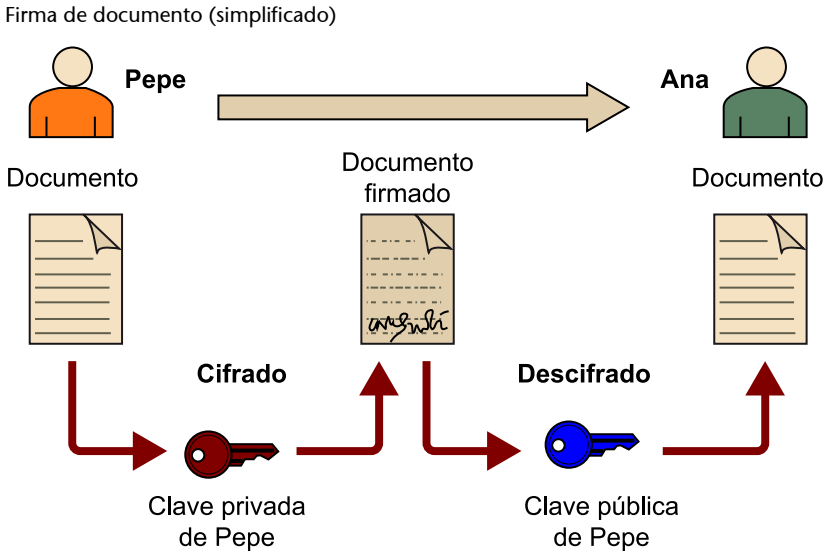
Todos estamos acostumbrados a firmar documentos; desde la aceptación de un presupuesto para arreglar una lavadora, hasta una hipoteca. El hecho de firmar implica, como decíamos, lo siguiente:

a) El documento firmado no se ha cambiado por otro puesto que en caso contrario, no tendría la firma.

b) Solo el firmante es capaz de hacer aquella firma concreta, y por lo tanto es él y solo él quien lo ha hecho.

En el mundo físico, afirmar que un documento no puede ser modificado una vez firmado es osado, y también lo es decir que no se puede reproducir una firma. No obstante, en el mundo digital el proceso es mucho más exacto y seguro.

El proceso de firmar un material se hace justamente a la inversa del proceso de cifrado. Hemos dicho que de las dos claves que teníamos, la pública se distribuía libremente a todo el mundo, mientras que la privada se mantenía en secreto y bajo la custodia del propietario. Por lo tanto, si Pepe cifra un texto con su clave privada (algo que solo puede hacer él) y lo envía, cualquier persona podrá descifrarlo (puesto que todo el mundo tiene la clave) y verificar que lo ha enviado Pepe. Ya tenemos un proceso análogo a una firma en el mundo digital.



Faltan unos detalles por resolver, pero:

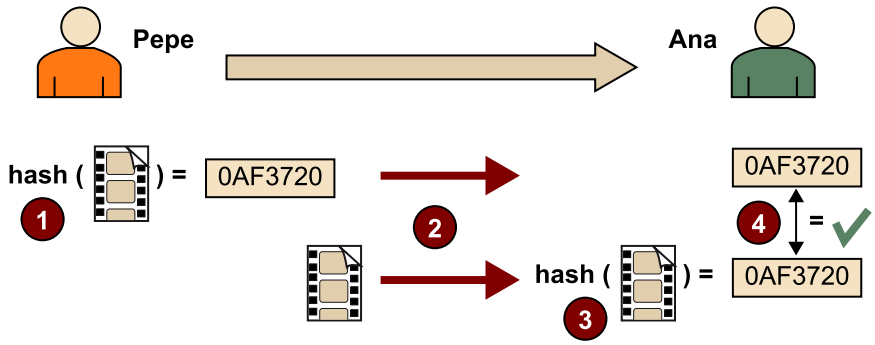
a) Habíamos comentado que cifrar grandes cantidades de contenidos con criptografía de clave asimétrica era costoso, y ahora estamos cifrando documentos solo para poderlos firmar.

b) Estamos enviando los documentos cifrados, algo que a veces no es el objetivo. Lo que se desea es acompañar el documento (no cifrado) con la firma, pero si la firma es tan grande como el documento, no parece práctico.

c) No hemos resuelto totalmente el problema de la integridad (no modificación) del documento. Si se modifica el contenido cifrado, es posible que el proceso de descifrado funcione y el original se vea alterado. Quizá no podremos controlar la modificación, pero es posible hacerla y que el receptor crea que Pepe lo ha enviado así.

Para resolver este problema, aparece otra técnica criptográfica denominada **función resumen**, que todo el mundo conoce por el nombre inglés de **función *hash***. Al aplicar este tipo de funciones sobre un contenido, retornan un resumen (*hash*) de longitud más corta (el algoritmo MD5 genera resúmenes de 128 bits, SHA-1 de 160 bits y SHA-2 de 224, 256, 384 o 512 bits), con una particularidad importante: dos contenidos distintos a la entrada, aunque solo sea por 1 bit, darán resultados de la función de resumen diferentes. Esto permite establecer una relación unívoca entre el documento original y su resumen. En el ejemplo siguiente, se puede ver el proceso de verificación de una firma:

Verificación de la función resumen



1) Pepe aplica la función resumen al fichero y obtiene el resumen.

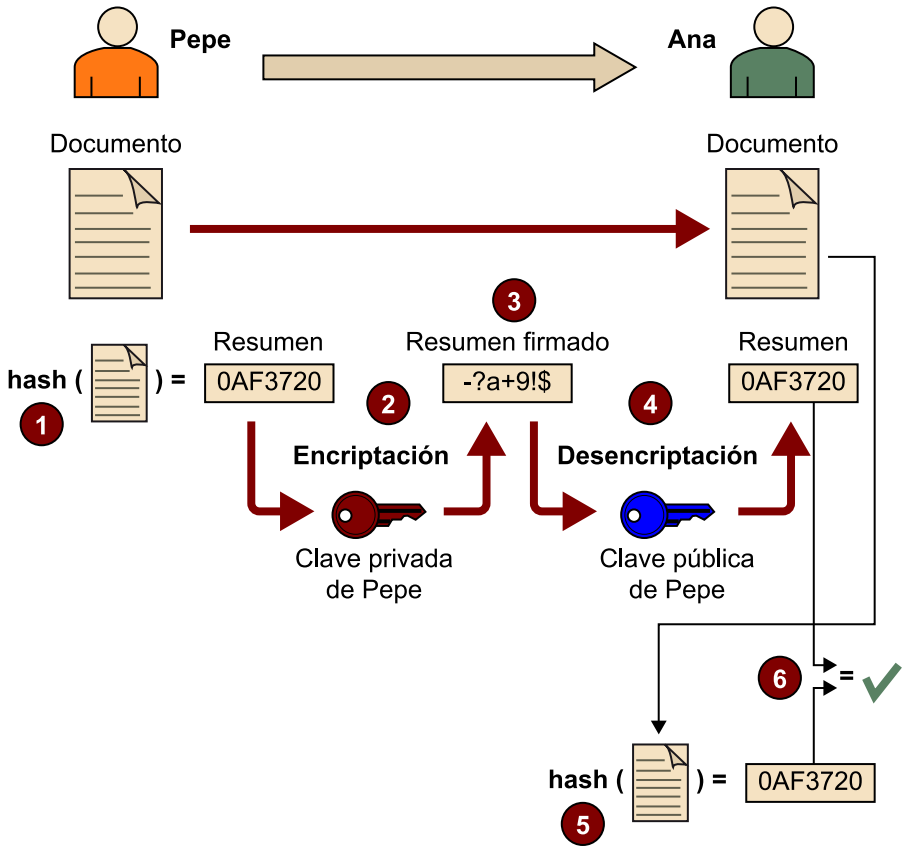
2) Envía el fichero y el resumen a Ana.

3) Ana aplica la misma función de resumen al fichero y obtiene un resumen.

4) Compara este resumen con el que ha recibido de Pepe. Si son iguales, significa que el fichero no se ha modificado.

El proceso, como lo hemos descrito en este ejemplo, no es demasiado seguro, puesto que si alguien quiere alterar el documento, solo debe tener la precaución de modificar también la firma. Sin embargo, si combinamos esta función de resumen con la firma anterior, podemos obtener una solución fiable y computacionalmente asequible. Veamos los pasos de esto en el esquema siguiente:

Firma con resumen



1) Pepe, que quiere enviar el documento, le aplica una función resumen para obtener una firma.

2) A continuación, cifra el resumen con su clave privada. Dado que el resumen son unos pocos bytes, el proceso es rápido. Esto se denomina **firma del documento**.

3) Pepe envía el documento y la firma. Esta vez no hemos duplicado la cantidad de información, sino que solo le hemos añadido una pequeña firma.

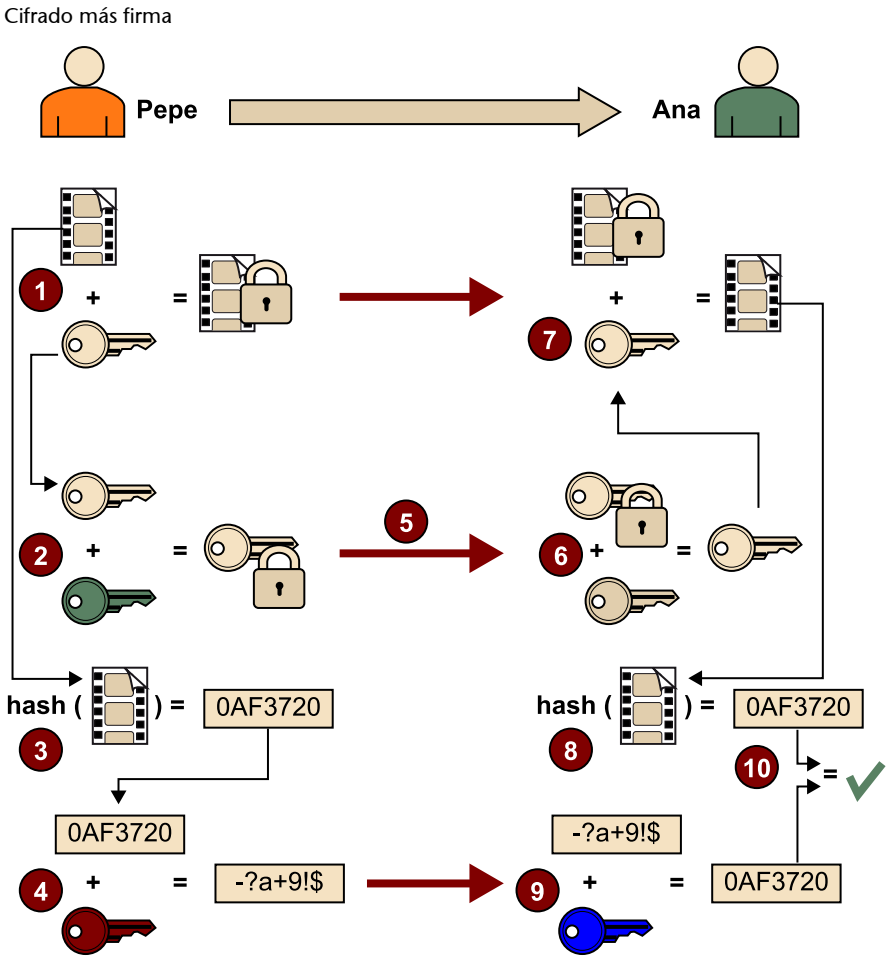
4) Ana, con la clave pública de Pepe, descifra la firma.

5) Ana aplica la función de resumen al documento recibido y calcula un segundo resumen.

6) Compara las dos firmas. Si son iguales, se puede afirmar que a) el documento no ha sido modificado y b) ha sido enviado por Pepe, puesto que la firma del resumen solo la puede haber hecho él.

4.4.4. Cifrado + firma

Si combinamos los tres últimos elementos que hemos visto, podemos lograr el objetivo de una comunicación segura: contenido secreto, no alterado y del origen esperado. Veamos cómo resultan todos los pasos juntos:



1) Pepe crea una clave compartida y la utiliza para cifrar un contenido (un vídeo, en este caso).

2) A continuación, cifra la clave compartida con la clave pública de Ana (para que solo ella la pueda descifrar).

3) Aplicando la función de resumen, genera un resumen del vídeo.

4) Cifra el resumen con su clave privada para construir la firma.

5) Se envía a Ana el vídeo cifrado (con la clave compartida), la clave compartida cifrada y la firma.

6) Cuando Ana recibe las tres piezas, empieza por descifrar la clave compartida con su clave privada (solo ella lo puede hacer).

7) Con esta clave compartida, puede descifrar el vídeo.

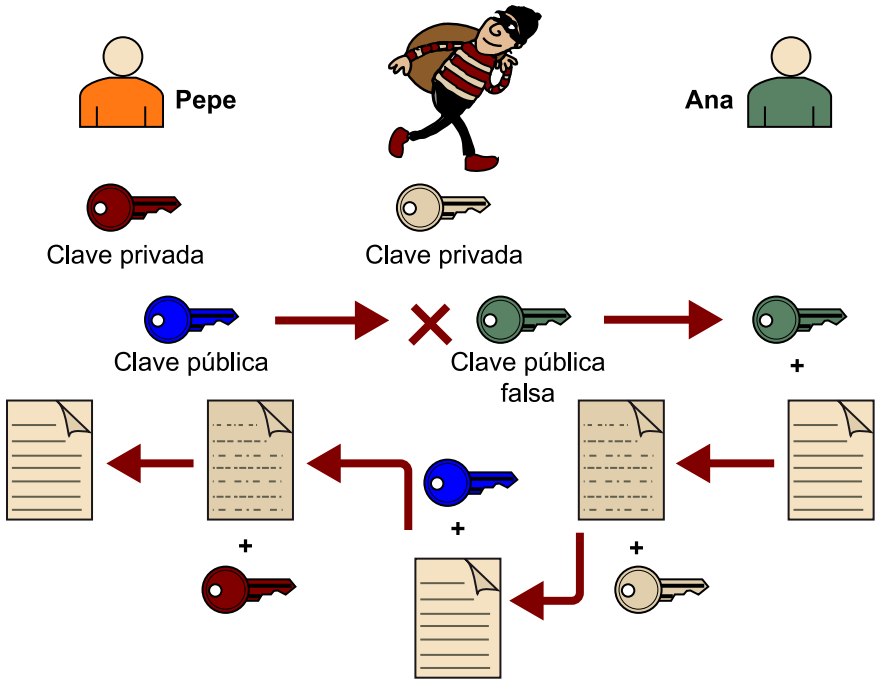
8) Ahora le falta asegurarse de que el vídeo no ha sido alterado y de que quien lo ha enviado ha sido realmente Pepe. Para esto, descifra la firma con la clave pública de Pepe y obtiene un resumen.

9) Aplica la función de resumen al vídeo y obtiene un segundo resumen, que compara con el recibido de Pepe. Si son iguales, el proceso ha terminado con éxito.

4.4.5. Certificados

La solución criptográfica presentada hasta ahora es casi perfecta, pero le falta resolver un problema. Cuando los usuarios no se conocen entre sí, aparece el problema de "cómo puedo estar seguro de que la otra parte es quien afirma ser". Las claves públicas que utilizamos para cifrar contenido pueden ser interceptadas y cambiadas por otras, lo que hace posible la interceptación y la manipulación de las comunicaciones.

Interceptación de la clave pública



Si alguien intercepta la publicación de la clave pública de Pepe, la cambia por una suya y la hace llegar a Ana como si fuera de Pepe, Ana la utilizará para cifrar el mensaje, y entonces el espía puede descifrar el documento (puesto que tiene la clave privada emparejada), verlo, manipularlo y cifrarlo de nuevo con la clave pública de Pepe (la verdadera) y reenviarlo. Pepe no notará la diferencia.

Si puede hacer lo mismo con la clave pública de Ana, podrá cifrar y firmar, y llevará a cabo el engaño perfecto. Tengamos en cuenta que no estamos hablando de obtener las claves privadas, sino las públicas –que se supone que se difunden abiertamente–, de modo que no es una posibilidad inconcebible.

La solución a este problema proviene de otra analogía con el mundo físico. ¿Cómo nos aseguramos de que una persona es la que dice que es en el mundo físico? Porque hay una autoridad en la que todo el mundo confía que valida las identidades y emite algo que lo certifica y a lo que se está vinculado. En abstracto no queda muy claro, pero en realidad hablamos del DNI o algún documento similar. La autoridad en la que se confía (de buen grado o por obligación) es el Estado, que emite un documento –el DNI– que vincula a la persona con su firma (aparece en el documento). La policía, antes de emitir el documento, verifica los datos de la persona (partida de nacimiento, fotos, etc.) de modo que una vez emitido, la posesión del documento y la capacidad de hacer la firma permiten asegurar la identidad de la persona. Esta figura, traspasada al mundo digital, se conoce con el nombre de **autoridad de certificación** (o simplemente, **CA**).

El DNI electrónico

El DNI actual, que incorpora un microchip, contiene dentro un certificado digital y la clave privada para utilizarlo. El mismo Ministerio actúa como autoridad de certificación. Podéis encontrar más información en "DNI electrónico".

Las autoridades de certificación emiten **certificados**, que son documentos electrónicos que relacionan la clave pública de una persona o entidad con su identidad y que permiten a una tercera parte verificar esta identidad. La idea fundamental es que, en un mundo de miles de millones de personas que no nos conocemos, no podemos confiar en que una clave pública que nos envíen corresponda realmente a una persona, pero sí podemos confiar en un pequeño grupo de entidades (autoridades de certificación) que tienen como misión verificar estas identidades y después emitir su certificado. Veamos los diferentes pasos de este proceso:

A) Obtención del certificado

Para obtener el certificado, es preciso en primer lugar que el usuario genere las claves pública y privada. A la clave pública se le añade la información de identidad que se quiere presentar al mundo (nombre, dirección, dirección de correo si es una persona, o URL y nombre de la empresa si es un sitio web) y se envía a la autoridad de certificación. Esta valida los datos (la policía obliga a ir personalmente a identificarse; otras CA permiten hacer el trámite por internet, enviando documentación) y, si son correctos, añade más información (se identifica como la que ha validado los datos, añadiendo fechas de validez), firma la petición con su clave privada y la devuelve.

Para validar que el certificado de Pepe es correcto, solo hay que tomar el certificado de la CA y aplicarlo al certificado de Pepe. Si ha sido manipulado, se detectará. La cantidad y tipología de documentación que hay que presentar

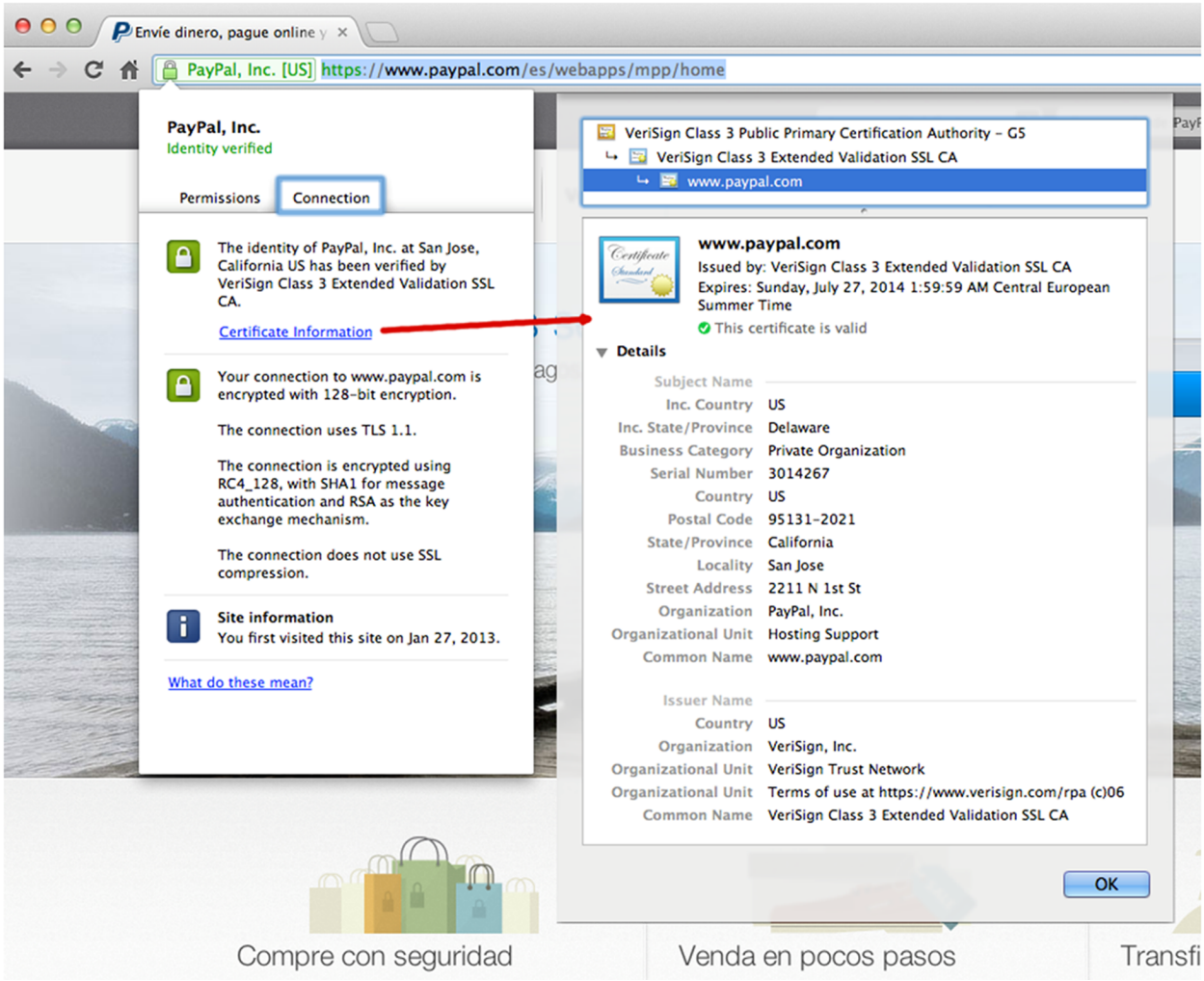
dependerán del uso que se quiera hacer del certificado; no es lo mismo un certificado para identificarse en un servicio de vídeo en línea que un certificado para firmar contratos de millones de euros entre empresas.

El problema del huevo y la gallina

¿De dónde obtienen su certificado las autoridades de certificación (CA)? Es el problema del huevo y la gallina. El usuario obtiene el certificado de la CA (que lo firma), ¿pero de dónde lo obtiene esta? Bien, algunas CA lo obtienen de otras CA que les generan un certificado que permite generar más, lo que crea una cadena de CA que confían en la de nivel superior, hasta llegar a las primeras, que han firmado sus propios certificados. Aquí terminan las relaciones de confianza verificables y se tiene que hacer el único acto de fe en todo el proceso y confiar en su certificado.

Habitualmente, estos (pocos) certificados ya vienen incluidos para todos los productos que utilizan criptografía de clave pública (los navegadores, las bibliotecas de seguridad, los lenguajes de programación), lo que hace posible verificar cualquier certificado hasta el origen. En la imagen siguiente, se puede ver el certificado de un sitio web (que se utiliza para garantizar que realmente nos hemos conectado al sitio web esperado, y no a otro que simula serlo) y la cadena de certificados hasta el certificado raíz. El hecho de que el navegador muestre el candado, el color verde y el texto "Identity verified" indica que se ha verificado el certificado y que la conexión es correcta.

Certificado de un sitio web



B) Firma

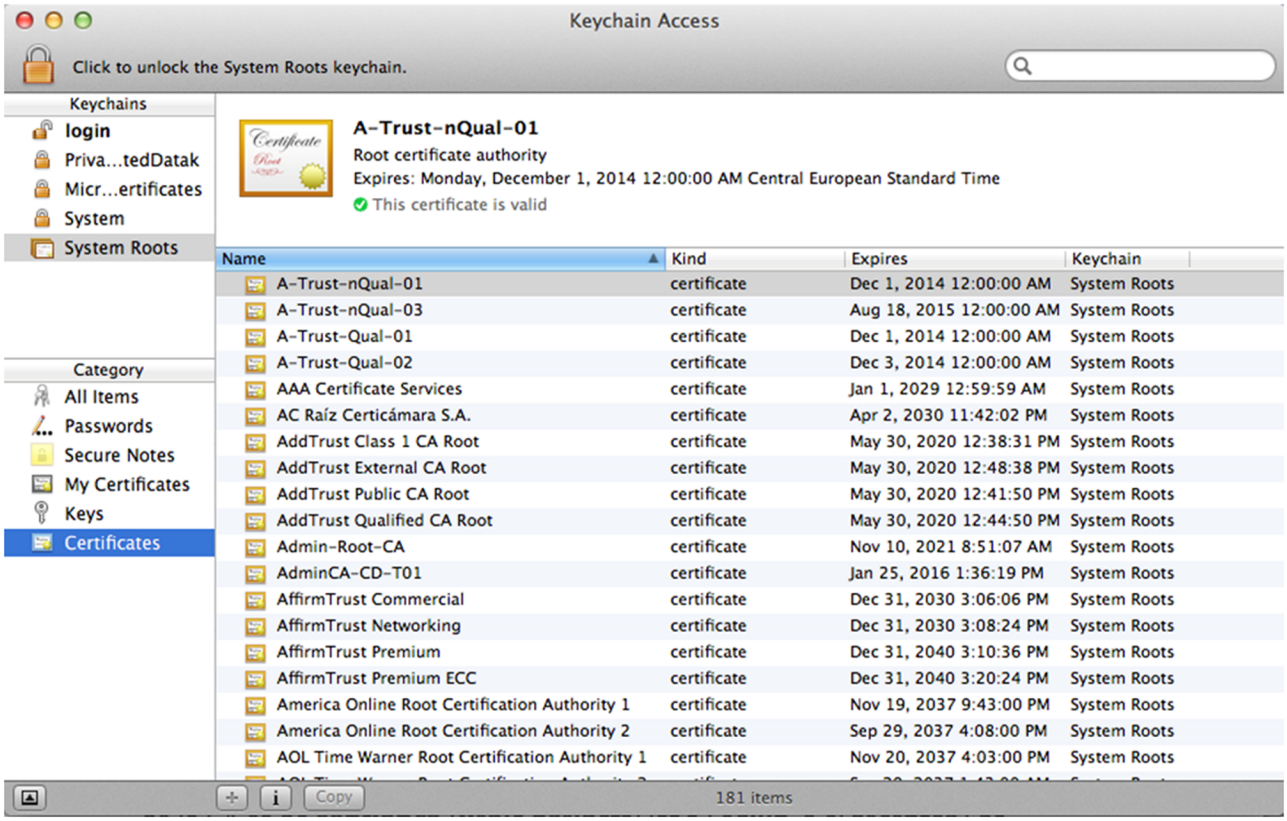
La firma es el proceso que menos ha cambiado. El que quiere firmar un contenido, utiliza su clave privada para cifrar el resumen y envía el documento, la firma y su certificado al receptor.

C) Verificación de firma

El receptor, antes de verificar la firma, verificará el certificado. Para hacerlo, obtendrá el certificado de la CA y lo utilizará para verificar. Si el certificado de la CA es **de confianza** (venía preinstalado en el equipo, o el receptor lo ha instalado porque confía en este), ya hemos terminado. Si el certificado de la CA está firmado por otra CA, tendremos que repetir el proceso de validación hasta llegar a una CA de confianza.

Una vez validado el certificado, ya sabemos que la persona que envía es realmente quien afirma que es, y podemos proceder a validar la firma con la clave pública incluida en el certificado.

Certificados raíz incluidos en un ordenador



El estándar que rige los formatos y procesos de firma digital es el X.509.

Reflexión

El mundo de la criptografía es más amplio, y hay otras figuras alrededor de la certificación digital (servidores de tiempo, autoridades de registro), que en conjunto forman lo que se denomina infraestructura de clave pública (*PKI*, del inglés *public-key infrastructure*). No obstante, para el propósito de este módulo, que es explicar la base en la que se fundamenta el DRM, con lo que hemos visto hay más que suficiente y no nos extenderemos sobre lo mismo.

Con los certificados, completamos el cajón de herramientas que nos permiten proteger el contenido en sistemas de DRM.

4.5. Proceso del DRM

Vamos a ver con más detalle cómo funciona un sistema de DRM utilizando las herramientas criptográficas acabadas de introducir. Cada implementación presenta variaciones sobre este modelo general y añade algunas características extra para cubrir necesidades específicas de los diferentes modelos de negocio, pero los fundamentos son comunes.

Sin embargo, antes de empezar, es preciso notar que hay un punto especialmente importante en todo el proceso. Si la aplicación de la criptografía se hace correctamente, una vez cifrado el fichero hay garantías de seguridad hasta el final del camino, en donde está el **reproductor**. Este es un elemento clave, por los motivos siguientes:

- Está instalado dentro del equipo que tiene el usuario final y, por lo tanto, más expuesto a manipulaciones y ataques.

- Es el punto en el que se tiene que descodificar el contenido y, por lo tanto, el lugar ideal para acceder al mismo.

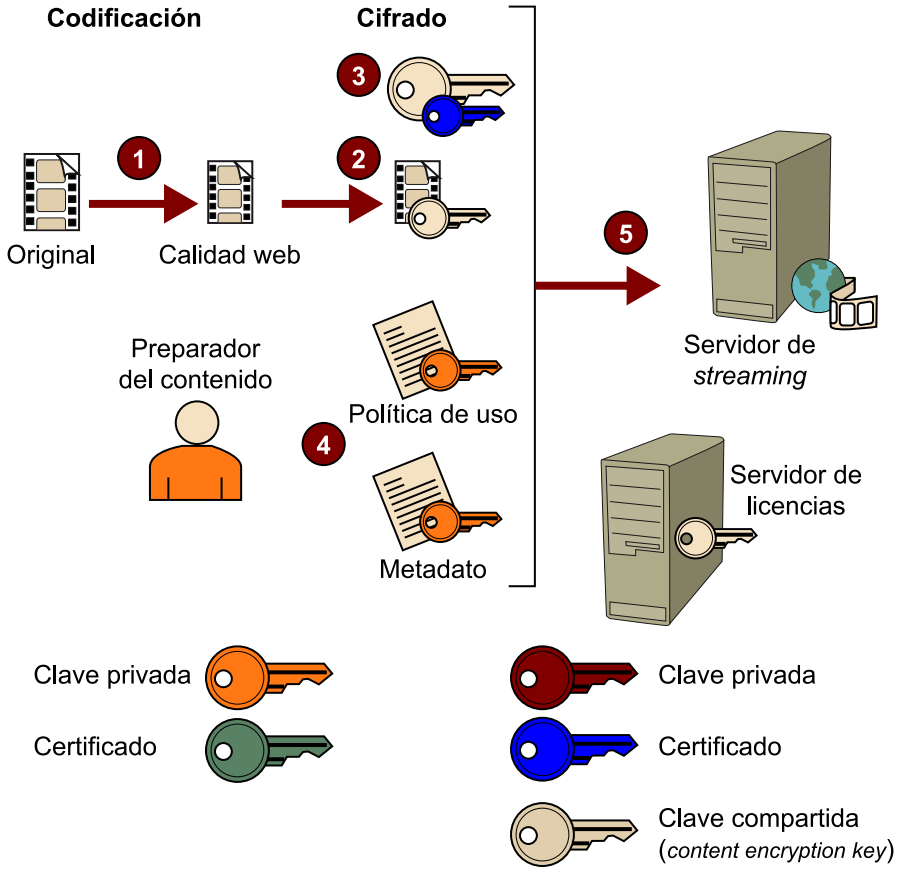
Por este motivo, resulta una pieza clave de cualquier sistema de DRM. Todos los fabricantes ponen mucho cuidado en su desarrollo y, si es viable, implementan en el hardware de los equipos la lógica de seguridad (especialmente en televisores, *set-top boxes*, teléfonos, etc.). El reproductor tiene que seguir de manera escrupulosa las normas que encontrará descritas en la licencia, para garantizar que todo funciona como está previsto.

Veamos el proceso desde la generación de contenido hasta el usuario final; para hacerlo más digerible, dividiremos el ciclo de trabajo en dos partes: preparación del contenido y acceso al contenido.

4.5.1. Ciclo de trabajo de preparación del contenido

El proceso de preparación del contenido solo se hace una vez para cada contenido que se incorpora al sistema protegido por DRM.

Ciclo de trabajo de preparación de contenido



1) Lo primero que se hace es la codificación del contenido original en el formato o los formatos necesarios para la distribución.

2) A continuación, se cifra el fichero con una clave compartida que se denomina *content encryption key* (*CEK*). Esta puede ser la misma para todos los contenidos, o distinta para cada fichero, con el objetivo de mejorar la seguridad.

3) La CEK se cifra con la clave pública del servidor de licencias (de modo que solo este pueda acceder a la misma).

4) La política de uso y el metadato asociado al fichero se cifran con la clave privada del preparador del contenido:

- La política indica qué uso se puede hacer de este contenido (el mínimo derecho posible es que se pueda reproducir, pero puede incorporar fechas límite, tipos de dispositivos aceptados, si se puede copiar, etc.).

- La figura del preparador de contenidos tiene entidad (y por lo tanto, dispone de claves y certificado para llevar a cabo acciones criptográficas) porque puede ser una entidad/persona diferente de la que hará la distribución, y limita los derechos que se podrán dar a los consumidores sobre el con-

tenido. De este modo, hace cumplir los contratos de cesión de derechos entre creador y distribuidor.

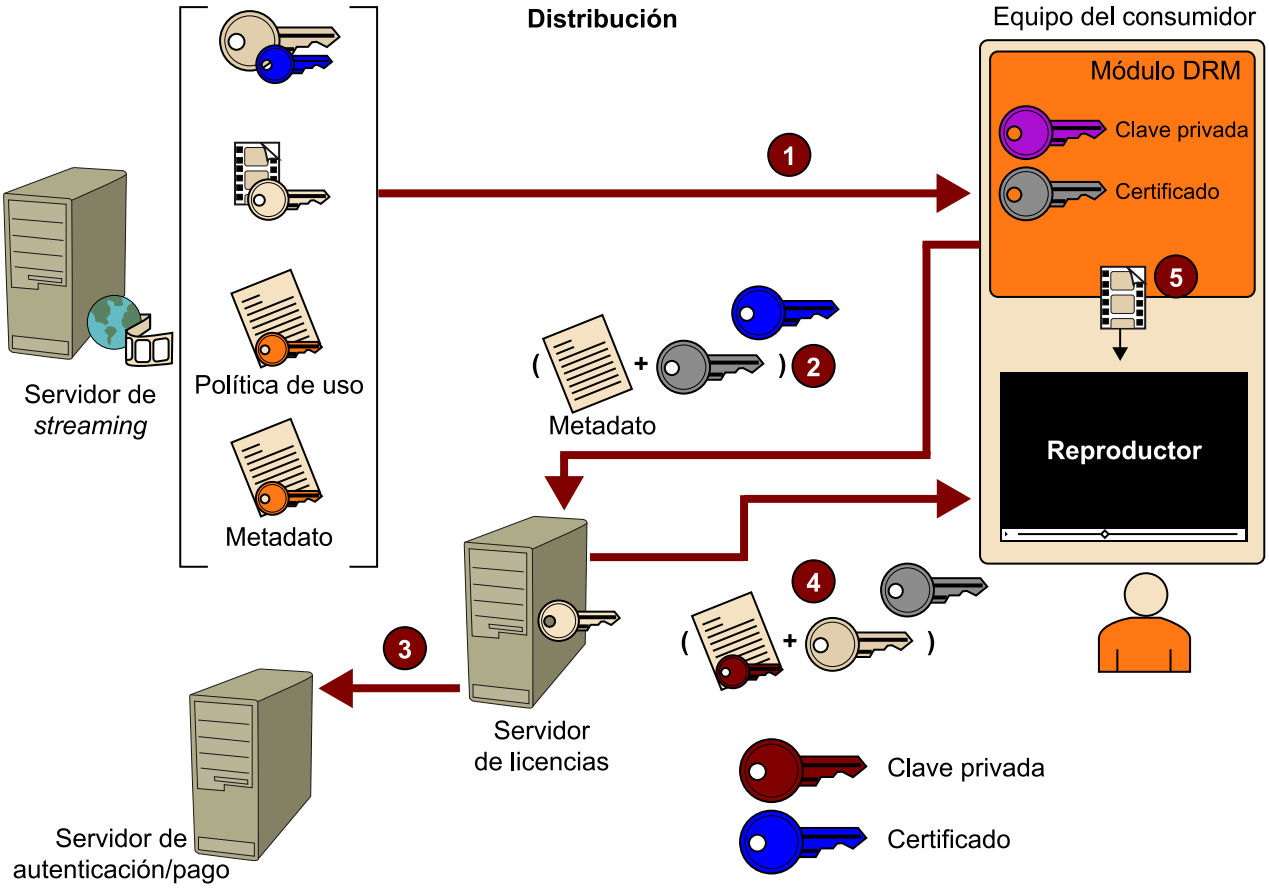
- El metadato es información sobre el contenido. Como mínimo incluye la URL en la que el reproductor podrá encontrar el servidor de licencias, un identificador único del contenido y la CEK cifrada en el paso 2. Aparte de esto, puede incorporar cualquier otro dato descriptivo (título, autor, etc.).

- El hecho de firmar la política se hace para que en todo momento se pueda comprobar que esta se cumple y que no ha sido alterada.

5) Estos elementos se empaquetan de manera conjunta y se ponen a disposición del consumidor en un servidor.

4.5.2. Ciclo de trabajo de acceso al contenido

Ciclo de trabajo de obtención de contenido



Antes de explicar el ciclo de trabajo, hay que precisar que en los sistemas de DRM es necesario que cada reproductor esté identificado de manera individual, con el objetivo de entregar contenido de manera exclusiva a este reproductor. Y hemos visto que la forma que tenemos de gestionar las identidades digitales son los certificados, y por esto a cada reproductor se le asigna una pareja de clave privada y un certificado. Este certificado puede venir pregenerado de fábrica o ser generado la primera vez que se accede a un servicio.

Acostumbra a estar firmado por la CA del fabricante del DRM, lo que permite verificarlo en cualquier momento y compartir una identidad única aunque se utilice el mismo equipo para diferentes servicios.

1) El consumidor accede al sitio web del proveedor y elige visualizar un contenido. En este momento, el módulo de DRM del reproductor solicita la descarga del paquete formado por el vídeo, la política, los metadatos y la CEK cifrada.

2) Una vez el reproductor tiene los metadatos y ha validado que no han sido manipulados (descifrándolo con la clave pública del preparador de contenidos), extrae la URL del servidor de licencias, hace una petición de licencia al servidor y adjunta el certificado (prueba de identidad) y los metadatos del vídeo, todo cifrado con la clave pública del servidor de licencias. Cuando llegan al servidor de licencias, este descifra la petición y extrae sus datos.

3) Primero utiliza los datos para invocar el servidor de autentificación y la autorización del proveedor para comprobar si aquel reproductor tiene derechos para acceder al contenido (por ejemplo, si tiene la suscripción al corriente de pago).

4) Si este responde de manera positiva con una licencia, descifra la CEK (recordemos que en la fase de preparación del contenido, había sido cifrada con la clave pública del servidor de licencias justamente para este momento) y la vuelve a cifrar de modo conjunto con la licencia, ahora con la clave pública del reproductor. Finalmente, cifra todo el contenido con su clave pública y lo devuelve.

5) El reproductor descifra el mensaje con la clave pública del servidor de licencias (de modo que verifica que el contenido no se ha alterado) y extrae la CEK que le permitirá acceder al contenido. Sin embargo, antes de dar acceso al contenido, verificará que la acción solicitada (reproducir, grabar, etc.) está admitida por la licencia entregada.

Observemos que la clave que permite acceder al contenido ha sido almacenada y ha viajado cifrada todo el tiempo. Solo el servidor de licencias tiene acceso a la misma y, al final del proceso, también lo tiene el reproductor autorizado. Por este motivo es tan importante proteger este último elemento, puesto que si se puede extraer la CEK, se pone en compromiso todo el sistema.

4.5.3. Dominios

Un caso de negocio frecuente que algunos sistemas de DRM contemplan es la posibilidad de asociar distintos dispositivos a una misma persona (por ejemplo, alquilar una película que se puede reproducir en el televisor, el móvil y el ordenador).

Esta agrupación de dispositivos en una misma persona se denomina **dominio**. Para implantar esta opción, hay que añadir un nuevo elemento, el **servidor de dominio**, que gestionará las asociaciones entre personas (dominios) y dispositivos. Cada dominio, como cualquier identidad digital, significará un certificado y una clave. Los dispositivos se registrarán en este servidor y se relacionarán con uno o varios dominios.

Cuando el servidor de licencias emita una licencia, esta puede estar dirigida al dispositivo (caso que ya hemos visto) o a un dominio; en este caso, el dispositivo no podrá descifrar la CEK (puesto que será necesaria la clave privada del dominio X, y no del dispositivo) y será necesario enviarla al servidor de dominio para que emita una CEK cifrada con su clave.

Si el servidor de dominio tiene que emitir certificados, entonces deberá disponer de un certificado de CA que le permita hacer justamente esto.

Las asociaciones dispositivo-dominio varían en función de cada proveedor.

Los sistemas de DRM no suelen fijar reglas sobre los derechos y las limitaciones de los dominios, algo que dejan a la lógica de negocio de cada proveedor de servicios. Solo proporcionan las primitivas para gestionar adición y eliminación de dispositivos del dominio y generar autorizaciones.

4.6. Principales DRM en el mercado

A principios del apartado, donde describíamos a los principales actores, ya hemos mencionado algunos fabricantes de soluciones de DRM. Acabaremos el apartado recapitulando sobre las distintas opciones que podemos encontrar en el mercado.

Al hacer una búsqueda, podemos observar claramente tres orígenes que marcan los terrenos en los que son más fuertes, así como unas determinadas maneras de trabajar.

4.6.1. DRM proveniente del mercado de los ordenadores personales

Encontramos dos grandes fabricantes que provienen de este entorno:

1) Microsoft tiene una larga experiencia en soluciones de DRM. En primer lugar, sacó un producto denominado Windows Media DRM; la primera versión apareció en 1999, y la última, en el 2004. Ambas utilizaban Windows Media Player como reproductor y estaban, por lo tanto, centradas en los PC con Windows. En el 2005, Windows Media DRM empezó a tener problemas

de seguridad que Microsoft fue remendando, hasta que sustituyó el producto por otro: PlayReady, que apareció en el 2007. Las principales novedades que aportaba eran las siguientes:

- PlayReady es independiente de la plataforma, y soporta equipos que no son de Microsoft. Microsoft ha creado un kit de desarrollo para permitir a fabricantes de dispositivos incorporar la tecnología.

- Soporta modelos de negocio más avanzados, como por ejemplo dominios y la distribución de licencias con el contenido por acceso sin conexión al servidor de licencias.

Más información

Podéis encontrar más información sobre PlayReady en las direcciones siguientes.

- Sitio web de Microsoft
- Documentos introductorios

2) Adobe, como fabricante de soluciones de reproducción en tiempo real, lanzó en el 2009 Adobe Flash Media Rights Management Server, la primera versión de DRM, que se renovó en el 2009 y cambió el nombre por **Adobe Access 2.0**. En el 2012 ha aparecido la versión 4.0, que soporta las plataformas Windows, Mac, Linux, iOS y Android.

Más información

Podéis encontrar más información sobre Adobe Access en las direcciones siguientes:

- Sitio web de Adobe
- Introducción a Adobe Access
- Documentación del producto
- Página en la que se puede encontrar un reproductor que utiliza DRM

Los dos fabricantes poseen unos puntos fuertes importantes:

a) Tienen una gran presencia en las plataformas en las que actualmente se consume más vídeo: ordenadores y terminales con Android o iOS. En televisores conectados, Microsoft ha conseguido convencer a bastantes fabricantes para incorporar su tecnología y posicionarse mejor que Adobe.

b) Hay una gran base de desarrolladores con experiencia en estas plataformas, lo que facilita la creación de productos.

c) Ambos han independizado sus soluciones de DRM de sus tecnologías de reproducción en tiempo real, con el objetivo de ampliar el mercado.

Más información

Para los que estéis interesados, podéis encontrar un artículo –ya un poco antiguo– de Tim Siglin, que compara estas dos tecnologías en la web streamingmedia.com:

"DRM: The big two".

4.6.2. DRM provenientes del mercado de móviles

La industria del móvil definió un estándar de DRM en el 2002 –OMA DRM– más centrado en contenido de pequeñas dimensiones, como por ejemplo politonos y fondos de pantalla. Desde aquella primera versión, el estándar ha ido evolucionando para dar apoyo a las nuevas necesidades. Actualmente, la última versión publicada es la 2.1.2.

El proceso está dirigido por la OMA[23], que engloba a los principales fabricantes y que define estándares para una gran cantidad de aspectos relacionados con la tecnología de móviles.

(23)OMA son las siglas de la entidad Open Mobile Alliance.

Lo que la OMA define es un estándar, no un producto desarrollado y operativo. Los diferentes fabricantes pueden implementar el software como deseen, siguiendo las especificaciones definidas, y según Wikipedia ("OMA DRM") algunos ya lo han hecho.

Más información

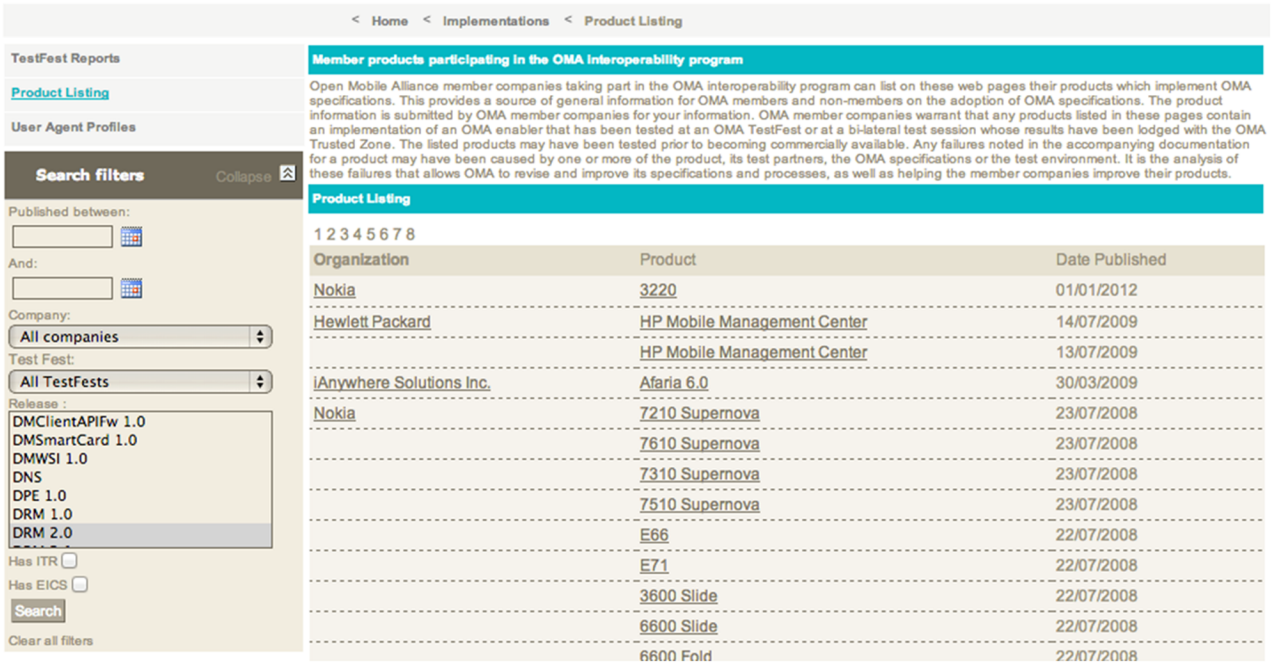
Las especificaciones del estándar OMA DRM se pueden encontrar en la web de la OMA, en:

"OMA Digital Rights Management V2.1.2".

El problema de OMA DRM es que el mercado de móviles ha dado un salto enorme y los fabricantes "tradicionales" han sido barridos. No tanto en cuanto a la fabricación como a los sistemas operativos que incorporan, y al final es esta pieza la que determina qué se ejecuta en el terminal. Actualmente, los líderes son iOS, Android y Windows, y no se adhieren a este estándar.

Si hacemos una consulta a la web de la Open Mobile Alliance para los dispositivos que soportan DRM, únicamente aparecen dispositivos del 2009 y tan solo uno del 2012.

Últimos dispositivos que soportan OMA DRM



4.6.3. DRM provenientes del mercado TV

El DRM lleva tiempo presente en el mercado de la televisión. Las plataformas de IPTV lo han utilizado para proteger el acceso a los contenidos, y hay numerosos fabricantes que tienen soluciones. En estos sistemas, la implementación se suele hacer por hardware. La empresa que quiere lanzar un canal adquiere *set-top boxes* que distribuirá a sus clientes, los cuales controlarán todo el proceso. Un dispositivo es para un solo servicio.

El punto de inflexión se produce al aparecer el vídeo por internet –*over the top*, como se acostumbra a denominar en el sector–, en el que varios servicios pueden proporcionar contenido al mismo dispositivo. Esto obliga a la interoperabilidad, y los fabricantes de dispositivos intentan ofrecer soporte a varias tecnologías. Es aquí por donde los fabricantes del mundo internet, especialmente Microsoft, han empezado a entrar en este mercado.

Por otro lado, los fabricantes provenientes de este sector han empezado a desarrollar soluciones para dispositivos móviles y ordenadores personales, de modo que han ampliado la competencia. Dentro de estos movimientos estratégicos, Google compró en el 2010 una de las empresas más importantes del sector –Widevine– y todavía se está a la espera de cuáles son los objetivos de esta compra.

Los principales fabricantes de este sector son Widevine, Irdeto y NDS.

Aparte de estas iniciativas, hay otra interesante: un conjunto de empresas conpuesto por Intertrust, Panasonic, Philips, Samsung y Sony, que se unieron para formar la Marlin Developer Community con el objetivo de desarrollar un sistema de DRM, liberado como código abierto, que ha estado implantándose en numerosos televisores conectados. Uno de los aspectos que hacen interesante este producto es que ha sido elegido, junto con PlayReady, como solución para proteger la televisión híbrida en el Estado español por la asociación que agrupa a fabricantes y cadenas de televisión (AEDETI).

Se denomina **televisión híbrida** a la televisión que combina emisión en *broadcast* (por la TDT) con los contenidos bajo demanda distribuidos por internet. Está estandarizada con la especificación HbbTV, vigente tanto en el Estado español como en otros muchos países europeos.

Más información

Podéis obtener más detalles sobre la televisión híbrida en la dirección siguiente:

"HbbTV® = More entertainment at your command"

Marlin está ganando relevancia rápidamente y parece tener el favor del sector *broadcast*.

Resumen

Como podéis ver, la seguridad en el vídeo por internet presenta diferentes problemas:

- Todas las opciones que no incluyen DRM presentan deficiencias.

- Hay una gran fragmentación de tecnologías, fabricantes y dispositivos. Incluso de manera independiente de la seguridad, encontrar una solución para llevar un servicio a diferentes plataformas puede requerir el uso de más de una tecnología. Si a esto le añadimos la necesidad de protección, todavía tenemos más problemas.

- Los sistemas de DRM tienen mala fama. No por su funcionamiento, sino porque son complejos, caros de implantar y limitan mucho el acceso al contenido. Incluso servicios que tienen contenido de valor intentan evitar el uso de DRM.

Tomar decisiones en este contexto no es simple. Habitualmente, se utiliza una técnica iterativa que pasa por los puntos siguientes:

- Definir plataformas objetivo: dónde queremos que el contenido esté presente.

- Identificar las tecnologías que permiten cumplir el objetivo.

- Definir las necesidades de seguridad.

- Analizar las posibilidades que ofrecen las tecnologías del segundo punto y ver si se encuentra una que cumpla dichas condiciones de seguridad.

- Si no se encuentra una solución que encaje, ya sea porque no cumple los requerimientos de seguridad o porque para cubrirlos hace falta más de una tecnología, se vuelve al primer punto para reducir el número de plataformas, o al tercero para reducir las necesidades de seguridad.

- La iteración se hace hasta llegar a un compromiso satisfactorio.