(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 806 261**

(51) Int. CI.:

G06F 21/00 (2013.01)
H04N 21/258 (2011.01)
H04N 21/442 (2011.01)
H04N 21/426 (2011.01)
H04N 21/6334 (2011.01)
H04N 21/6377 (2011.01)
G06F 21/12 (2013.01)
G06F 21/16 (2013.01)
G06F 16/23 (2009.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **15.11.2011 PCT/IB2011/055083**

(87) Fecha y número de publicación internacional: **24.05.2012 WO12066471**

(96) Fecha de presentación y número de la solicitud europea: **15.11.2011 E 11808938 (2)**

(97) Fecha y número de publicación de la concesión europea: **29.04.2020 EP 2641208**

(54) Título: **Método para detectar software clonado**

(30) Prioridad:

19.11.2010 US 415363 P

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
17.02.2021

(73) Titular/es:

NAGRAVISION S.A. (100.0%)
22-24, route de Genève
1033 Cheseaux-sur-Lausanne, CH

(72) Inventor/es:

FISCHER, JEAN-BERNARD;
MARCACCI, PATRIK;
SCHWARZ, CHRISTIAN y
WYSEUR, BRECHT

(74) Agente/Representante:

TOMAS GIL, Tesifonte Enrique

DESCRIPCIÓN

Método para detectar *software* clonado

CAMPO TÉCNICO

[0001] Esta invención se refiere al campo del receptor/decodificador integrado que recibe datos de un equipo de radiodifusión central; donde el acceso a estos datos está sujeto a un acceso condicional.

ANTECEDENTES

[0002] Un gran problema en la seguridad de *software* es evitar la copia ilegítima y el uso de *software*.

[0003] En una solución de *software* puro, este problema es imposible de resolver en un caso de uso desconectado. Sin embargo, cuando hay una conexión disponible para una entidad confiable (por ejemplo, un servidor de verificación), esta conexión se puede usar para implementar algunos mecanismos de seguridad (tanto en el caso de un caso de uso conectado continuamente como en un caso de uso conectado ocasionalmente). A pesar de esto, en un caso de uso distribuido (donde una gran población de usuarios puede usar el *software*), aun es difícil detectar y bloquear copias.

[0004] El bloqueo del *software* al *hardware* de su plataforma no siempre es una opción. En primer lugar, esto no puede ser factible debido a la falta de *hardware* confiable o un mecanismo de arranque. En segundo lugar, un usuario debería siempre poder migrar su *software* a otra plataforma o cambiar su configuración de *hardware* o *software*.

[0005] En la presente configuración, un reproductor puede clonarse fácilmente con todos su secretos y ejecutarse en miles de ordenadores al mismo tiempo. Por lo tanto, un usuario que paga una tarifa plana para acceder al contenido tendría un buen incentivo para revender copias (clones) de su reproductor a otros usuarios.

[0006] La WO 02/17555 A2 enumera sistemas, métodos y medios legibles por ordenador para permitir que un dispositivo servidor valide uno o más dispositivos cliente. Se genera un secreto imprevisible compartido. El secreto imprevisible compartido se almacena en el dispositivo cliente y en el dispositivo servidor. El dispositivo cliente demuestra la posesión de un secreto imprevisible compartido correcto para el dispositivo servidor. El secreto imprevisible compartido se sustituye por un secreto imprevisible compartido nuevo cada vez que el dispositivo cliente es validado por el dispositivo servidor.

[0007] La US 2006/107323 A1 enumera un sistema y un método para proporcionar comunicaciones seguras entre dispositivos de comunicación cliente y servidores. Un servidor genera un desplazamiento aleatorio. El servidor altera una credencial dinámica del dispositivo de comunicación servidor al aplicar el desplazamiento aleatorio a la credencial dinámica del dispositivo de comunicación servidor. El servidor almacena la credencial dinámica del dispositivo de comunicación servidor.

[0008] La WO 02/052389 A2 enumera un método para evitar el uso de más de un módulo de seguridad idéntico para identificar y utilizar recursos gestionados por un centro de gestión. Por lo tanto, la invención proporciona un método de anticlonación basado en el almacenamiento de números de identificación de unidades de usuario conectadas a dicho módulo de seguridad. Cuando se establece una conexión con un centro de gestión, dichos números se transmiten y comparan con los números de una transmisión precedente.

[0009] La US 7 676 436 menciona permitir a un usuario adquirir un artículo (por ejemplo, una canción digital, un vídeo digital, etc.) usando un dispositivo (por ejemplo, un dispositivo portátil) y mover fácilmente una copia del artículo adquirido desde el primer dispositivo hasta otro dispositivo para que un usuario del otro dispositivo pueda reproducir el artículo.

[0010] La US 2007/174472 A1 menciona un sistema de seguridad para comunicaciones de red con dispositivos cliente. Un servidor compara identificadores escondidos recibidos de los dispositivos cliente para detectar posibles clones.

RESUMEN DE LA INVENCIÓN

[0011] Mientras que la invención se define en la reivindicación independiente, otros aspectos de la invención se exponen en las reivindicaciones dependientes, los dibujos y la siguiente descripción.

BREVE DESCRIPCIÓN DE LOS DIBUJOS

[0012] La presente invención se entenderá mejor gracias a las figuras adjuntas en las que:

La figura 1 muestra un diagrama de detección de clonación, que se basa en una verificación de etiqueta dinámica, según una primera forma de realización de la presente invención,

La figura 2a ilustra una segunda forma de realización de la invención, en particular, una autenticación de mensaje basada en una credencial dinámica para detectar la clonación.

La figura 2b ilustra otra variante de la forma de realización mostrada en la figura 2a.

La figura 3 ilustra una tercera forma de realización de la invención, donde se ejecuta una condición de actualización de etiqueta antes de la entrega de contenido.

DESCRIPCIÓN DETALLADA

[0013] La idea principal de la solución sugerida en la presente invención se basa en la observación de que cada componente de *software* cliente (por ejemplo, un reproductor multimedia) tendrá un uso de contenido diferente. Cada uso de contenido genera un historial de uso único que permite rastrear la legitimidad del *software* y facilitar la detección de copias ilegítimas.

[0014] Más concretamente, un servidor realizará un seguimiento del historial de uso de cada cliente mediante el uso del valor de etiqueta. El valor de etiqueta representa el uso (parcial o completamente) al tiempo que conserva la privacidad del historial. En cada solicitud válida, este valor de etiqueta se actualizará preferiblemente, lo que provocará una desincronización entre las copias realizadas a partir de esta implementación, dado que cuando se realiza una copia, este se bifurcará del historial del cliente original, y su uso se desviará del uso del cliente original. Es precisamente esta desviación la que se puede detectar.

[0015] En referencia a la figura 1, que muestra una primera forma de realización de la invención, cada cliente (C) recibe un valor de etiqueta aleatorio inicial t_c, que puede insertarse en el reproductor de *software* en el momento de la inicialización o que puede ser transmitido por el servidor durante la primera autenticación con este último. El servidor tendrá los valores de etiqueta de todos los clientes en una base de datos. Este valor de etiqueta inicial t_c también puede ser enviado al cliente de la unidad de usuario medaiante cualquier medio de comunicación convencional como, entre otros, una carta, un SMS, una llamada telefónica, un correo electrónico, una página web, o cualquier combinación de los mismos. A su lado, el servidor ha abierto un nuevo registro en su base de datos para almacenar este valor de etiqueta t_c. De este modo, tanto la unidad de usuario como el servidor poseen el mismo valor de etiqueta t_c al final de esta primera fase, denominada fase de inicialización.

[0016] Para implementar esta fase de inicialización, se llevan a cabo los siguientes pasos:

- definir el valor de etiqueta (t_c) como igual a un valor aleatorio inicial,
- abrir un nuevo registro que almacena dicho valor de etiqueta (t_c) en la base de datos del servidor,
- introducir este valor de etiqueta (t_c) en la unidad de usuario cliente.

[0017] El paso destinado a definir el valor de etiqueta como un valor aleatorio inicial tiene como objetivo tomar un valor imprevisible como primer valor de etiqueta.

[0018] Durante una fase operativa y mientras se sigue haciendo referencia a la figura 1, cuando un cliente solicita un servicio del servidor, a través de una red de comunicación común que conecta la unidad de usuario cliente a este servidor, su valor de etiqueta t_c se enviará, dentro de un mensaje denominado mensaje cliente. El servidor verificará si el valor de etiqueta, incluido en el mensaje cliente, también está presente en su base de datos. Si dicha comparación da un resultado correcto, el servidor otorgará el servicio solicitado (o procederá a verificar otros requisitos). Si el valor de etiqueta enviado dentro del mensaje cliente no figura en la base de datos o no se puede verificar correctamente, el servicio solicitado se rechazará.

[0019] El valor de etiqueta t_c puede ser el resumen de una función de compresión, como una función *hash* criptográfica, aplicada a la solicitud cliente. Aunque las buenas funciones de derivación proporcionan resúmenos que son difíciles de adivinar, un atacante, tal como un intermediario, conocido por el experto en la técnica, aun podrían intentar tener acceso a un servicio cogiendo o adivinando valores de etiqueta.

[0020] Para evitar cualquier ataque, una primera solución es realizar comunicaciones entre el cliente y el servidor a través de un canal protegido (es decir, mediante una comunicación encriptada y autenticada entre el cliente y el servidor). En este caso, el valor de etiqueta se puede adjuntar directamente a la solicitud y, por lo tanto, el servidor puede realizar una verificación directa comprobando si la etiqueta está comprendida en la base de datos del servidor (como se muestra en la figura 1).

[0021] Además, si una función *hash* criptográfica se utiliza para derivar una nueva etiqueta de la etiqueta antigua y el historial de uso, entonces este valor de etiqueta es único por usuario y, por lo tanto, puede usarse como un identificador para identificar a cada cliente. Esto facilita una verificación anticlonación sin la necesidad de asociar solicitudes a individuos y, por lo tanto, conduce a verificaciones transparentes, más rápidas y anónimas (sin embargo, esto no excluye adjuntar un identificador cliente único a la solicitud en el mensaje cliente). Representado en la figura 1, este caso de uso es un protocolo de certificación remota que solo debería usarse en casos en los que una persona maliciosa no pueda falsificar ni alterar activamente la conexión entre el cliente y el servidor (por ejemplo, en un dominio doméstico seguro o cuando la comunicación entre el cliente y el servidor se realiza a través de un canal seguro autenticado).

[0022] Con respecto a cada solicitud válida, tanto el cliente como el servidor actualizarán el valor de etiqueta. El nuevo valor de etiqueta t'_c se derivará de los datos que son conocidos por el cliente y el servidor: por ejemplo, el antiguo valor de etiqueta y la información obtenida de al menos una parte del contenido del mensaje servidor que se proporciona como respuesta a la solicitud cliente. Alternativamente, el nuevo valor de etiqueta t'_c se puede derivar del último valor de etiqueta t_c y de al menos una parte del contenido del mensaje cliente. El mensaje servidor o el mensaje cliente (o una parte de su contenido) puede ser una marca de tiempo que se inserta en el flujo de medios o cualquier otra información de encabezado, claves criptográficas sobreencriptadas que se envían, marcos específicos, etc. El nuevo valor de etiqueta t'_c está destinado a reemplazar el antiguo valor de etiqueta t_c.

[0023] Con este fin, esta fase operativa requiere los pasos de:

- preparar, en el lado de unidad de usuario, un mensaje cliente que comprende una solicitud junto con el valor de etiqueta t_c, luego
- enviar este mensaje cliente, desde la unidad de usuario hasta el servidor,
- realizar una prueba de condición de acceso, en el lado servidor, con el objetivo de probar si este valor de etiqueta t_c está comprendido en la base de datos del servidor. En resultado negativo (es decir, caso negativo): denegar el servicio solicitado, mientras que en resultado positivo (es decir, caso positivo):
- enviar un mensaje servidor a la unidad de usuario, como una respuesta a la solicitud cliente,
- actualizar el valor de etiqueta t_c, tanto en el lado servidor como en el lado unidad de usuario, al reemplazar este valor t_c por un nuevo valor de etiqueta t'_c. Este nuevo valor de etiqueta t'_c se deriva del último valor de etiqueta t_c y de otros datos conocidos por el cliente y el servidor,
- almacenar el nuevo valor de etiqueta t'_c en la unidad de usuario (es decir, en una memoria) y en el registro de la base de datos conectado al servidor (por ejemplo, al reemplazar el antiguo valor de etiqueta t_c).

[0024] Si se hace una copia de la implementación de *software* cliente, se producirá una desincronización cuando uno de ellos (el original o la copia) solicite un servicio. Por lo tanto, un usuario auténtico no tiene ningún incentivo para compartir su implementación de *software* cliente, ya que el uso de la copia denegaría finalmente que su original pueda tener acceso.

[0025] Ventajosamente, un usuario aun puede migrar su implementación de *software* a otra plataforma sin ningún problema.

[0026] En referencia a la figura 2a, la última divulga otra forma de realización de la invención que se puede usar para prevenir cualquier ataque mientras que se usa un canal inseguro. La solución sugerida en esta forma de realización pretende firmar la solicitud cliente mediante el uso de una clave, que deriva directa o indirectamente del valor de etiqueta t_c, obtenido por una función de derivación de clave de firma. Por lo tanto, durante la fase operativa, cuando un cliente solicita un servicio del servidor (por ejemplo, a través de un canal de comunicación inseguro), se enviará una firma de la solicitud junto con la solicitud dentro del mensaje cliente. Como se muestra en la figura 2a, esta firma se puede adjuntar a la solicitud como un código de autenticación del mensaje cliente. Esta firma se obtiene, en primer lugar, al aplicar una función de compresión a la solicitud cliente para obtener un resumen de la solicitud, luego al encriptar este resumen con una clave de firma que se deriva del valor de etiqueta y se obtiene al usar la función de derivación de clave de firma. Por ejemplo, la función de compresión puede ser una función *hash* o una función HMAC que toma como clave el valor de etiqueta t_c o un valor derivado del mismo. De esta manera, el valor de la firma depende del valor de etiqueta. El servidor puede verificar la autenticación de la solicitud cliente comparando la firma comprendida en el mensaje cliente con una firma calculada por el servidor de una manera similar a la determinada por la unidad de usuario cliente. Para este fin, el servidor usa la misma clave de firma (obtenida de la misma función de derivación de firma y del valor de etiqueta almacenado en su base de datos) para desencriptar la firma adjuntada a la solicitud y luego obtener el resumen de esta solicitud. Luego, el servidor calcula un resumen de la solicitud usando la misma función de compresión que la usada para la unidad de usuario. Si este resumen es idéntico al resumen descifrado, la comparación da un resultado correcto y la firma

se define como válida. Si la firma es válida, el servidor otorgará el servicio solicitado (o procederá a verificar otros requisitos) y el valor de etiqueta se puede actualizar como se ha mencionado anteriormente en referencia a la figura 1. Si la firma enviada dentro del mensaje cliente no se puede verificar correctamente, el servicio solicitado se rechazará (el servidor también puede realizar otros pasos como consecuencia de un servicio denegado).

[0027] Para implementar la forma de realización mostrada en la figura 2a, la fase de inicialización descrita con la forma de realización mostrada en la figura 1 tiene que modificarse mediante la realización de los siguientes pasos adicionales:

- definir una función de firma (por ejemplo, una función HMAC) y una función de derivación de clave de firma para obtener una clave de firma (derivada preferiblemente de dicho valor de etiqueta t_c) para encriptar un resumen resultante de esta función de firma,
- compartir la definición de esta función de firma y la definición de la función de derivación de clave de firma entre la unidad de usuario y el servidor.

[0028] Como se ha descrito anteriormente con referencia a la figura 1, compartir estos datos durante esta fase de inicialización se puede lograr de muchas maneras diferentes, siempre y cuando el valor y/o la función se puedan introducir, al final de este paso, en la unidad de usuario cliente. Al final de esta fase de inicialización, la unidad de usuario y el servidor poseen los mismos datos iniciales.

[0029] La fase operativa de la forma de realización ilustrada por la figura 2a también requiere los siguientes pasos, además o en lugar de aquellos relacionados con la primera forma de realización (misma fase):

- calcular un código de autenticación aplicando la función de firma a la solicitud cliente y usando la clave de firma para encriptar el resumen resultante de dicha función de firma, luego cambiar el paso de preparación del mensaje cliente preparando un mensaje cliente que comprenda el código de autenticación y la solicitud cliente,
- cambiar la condición de acceso controlando si el código de autenticación recibido dentro del mensaje cliente es igual a un código de autenticación calculado por el servidor aplicando la misma función de firma a la solicitud cliente y usando la misma clave de firma para desencriptar dicho resumen dicho; donde la clave de firma se deriva preferiblemente del valor de etiqueta esperado que se almacena en la base de datos del servidor.

[0030] Con referencia ahora a la figura 2b, esta última muestra una variante de la forma de realización mostrada en la figura 2a. Durante la fase de inicialización, cada implementación de *software* cliente ha instalado un único identificador IDc y un valor de etiqueta aleatorio inicial t_c. El servidor almacena las tuplas (IDc, t_c) de todos su clientes legítimos. Por lo tanto, con respecto a la fase de inicialización de la forma de realización mostrada en la figura 2a, se llevan a cabo los siguientes pasos adicionales:

- asignar un identificador único IDc al cliente y almacenar este identificador cliente IDc en el nuevo registro asignado a este cliente,
- compartir este identificador IDc entre la unidad de usuario y el servidor (preferiblemente junto con la definición de la función de firma).

[0031] Como se ha mencionado previamente, compartir u obtener estos datos durante esta fase de inicialización se puede lograr de varias maneras, siempre que los datos se puedan introducir, al final de este paso, en la unidad de usuario cliente. El objetivo de este paso es el mismo que para las formas de realización precedentes, es decir, que la unidad de usuario y el servidor poseen los mismos datos iniciales.

[0032] Cuando un cliente solicita un servicio, este envía su identificador y una firma de la solicitud, que autentica su solicitud. Preferiblemente, la firma usa el valor de etiqueta almacenada como una clave (o la clave se deriva de este valor de etiqueta).

[0033] El servidor puede verificar la firma, ya que conoce la función de firma y puede derivar la clave de firma que se usa desde la tupla correspondiente hasta el identificador cliente. Solo cuando la firma es correcta se otorgará un servicio.

[0034] Con respecto a cada solicitud válida, el cliente y el servidor actualizarán el valor de etiqueta t_c de la misma manera que se ha descrito para las formas de realización precedentes. Por lo tanto, un nuevo valor de etiqueta t'_c se calculará a partir del antiguo valor de etiqueta t_c, por un lado, y a partir de la información obtenida del contenido que se proporcona, por otro lado. En la base de datos del servidor, la nueva etiqueta reemplazará a la antigua.

[0035] De lo mencionado anteriormente debe observarse que la fase operativa de la forma de realización ilustrada por la figura 2b también requiere los siguientes pasos, además o en vez de aquellos relacionados con la primera forma de realización mostrada en la figura 2a (misma fase):

- cambiar el paso de preparación del mensaje cliente al incluir el identificador cliente IDc en el mensaje cliente.

[0036] Opcionalmente, el servidor puede enviar una actualización de *software* a un cliente, cambiar la función de firma usada y/o los parámetros usados, o puede decidir reemplazar el valor de etiqueta t_c por un nuevo valor de etiqueta t'_c. Esta técnica se puede aplicar a cualquier forma de realización y podría usarse para inhabilitar a los piratas informáticos que han podido obtener un valor de etiqueta y/o invalidar la ingeniería a la función de firma usada o a la función para calcular un nuevo valor de etiqueta (por ejemplo, la funcion *hash* criptográfica). Como en este caso, podrían intentar combatir la desincronización que ocurre cuando se usan clones, mediante la implementación de un servicio de "resincronización" central o un *proxy* entre los clones y el servidor.

[0037] Se recomenda el uso de una función *hash* para realizar la invención conforme a la primera forma de realización mostrada en la figura 1. Esto se deduce del hecho de que los valores de etiqueta (es decir, los valores *hash*) almacenados en la base de datos del servidor deben ser difíciles de adivinar y deben ser diferentes cuando se actualizan. No debería producirse ninguna colisión entre dos implementaciones de clientes auténticos. Es decir, si todos los clientes auténticos comienzan con un valor de etiqueta inicial diferente, una colisión en los valores hash implicaría que la función *hash* criptográfica se muestre insegura (con respecto a la propiedad de resistencia a la colisión que se requiere para las funciones *hash* criptográficas).

[0038] Sin embargo, según la forma de realización mostrada en la figura 2b, esta recomendación se puede relajar, ya que cada valor de etiqueta está vinculado a un identificador único, es decir, al identificador cliente IDc. La única recomendación es que la clave de firma, que es idéntica al valor de etiqueta t_c o preferiblemente derivada de este valor de etiqueta, sigue siendo imprevisible para usuarios maliciosos que no tienen ningún conocimiento del valor de etiqueta t_c. Por lo tanto, debería quedar suficiente entropía de la entrada. La función de derivación de etiqueta calcula el nuevo valor de etiqueta t'_c basándose en el (antiguo) valor de etiqueta t_c y al menos una parte del contenido del mensaje. De esta manera, se lleva a cabo una cadena que contiene información obtenida a lo largo de todo el historial anterior.

[0039] La presente invención también sugiere un mecanismo de recuperación en caso de desincronización accidental, por ejemplo, cuando un cliente necesita recurrir a una copia precedente después de que su sistema se haya bloqueado, o cuando su *software* se haya clonado involuntariamente. Un procedimiento de recuperación se puede implementar a través de un proceso de autenticación convencional, por ejemplo, presentando las credenciales correctas. Al final de este procedimiento, el valor de etiqueta correspondiente al identificador cliente único se puede sustituir en el lado cliente (en la unidad de usuario) y en el lado servidor (en la memoria del servidor). Este hará que cada clon que use el mismo identificador sea inútil.

[0040] Para realizar dicho mecanismo de recuperación, la fase operativa de cualquier forma de realización comprenderá un paso de resincronización que cambia el paso de actualización por los siguientes pasos:

- sustituir el valor de etiqueta t_c por un nuevo valor de etiqueta t'_c igual a un nuevo valor aleatorio,

- enviar luego dicho nuevo valor de etiqueta t'_c a la unidad de usuario.

[0041] Con referencia a la figura 3, este último sugiere una tercera forma de realización principal para la presente invención. Según esta variante, la unidad de usuario cliente adjuntará a cada solicitud una firma, como también se hace en la segunda forma de realización de esta invención (véase la figura 2a y la figura 2b). Una vez que el servidor haya verificado que la firma es válida, el servidor puede decidir aplicar una actualización de la etiqueta (lo que obliga al cliente a solicitar el servicio de nuevo, y permite que el servidor compruebe que el valor de etiqueta de la unidad de usuario cliente se ha actualizado), o no hacerlo y enviar directamente el contenido en respuesta a la solicitud. A diferencia de las formas de realización precedentes de esta invención, el nuevo valor de etiqueta t'_c ya no se actualiza en función del contenido que se envía. La decisión de ejecutar o no una actualización de etiqueta depende de la lógica empresarial. La comprobación de si el valor de etiqueta t_c tiene que ser actualizado se puede llevar a cabo mediante una prueba basada en un parámetro temporal o en una característica que puede depender, por ejemplo, de la importancia de la solicitud cliente. Se puede ejecutar una actualización de etiqueta para cada solicitud de contenido, o solo para contenido valioso, o una vez al día, o dependiendo de cualquier otro parámetro imaginable. La decisión de ejecutar o no una actualización del valor de etiqueta también se podría aplicar a las otras formas de realización mostradas en las figuras 1, 2a y 2b al añadir, a la fase operativa, un paso con el objetivo de tomar tal decisión. En este caso, la unidad de usuario cliente debe ser informada, cada vez que se envía un mensaje servidor como respuesta a una solicitud cliente, si el valor de etiqueta se ha actualizado o no por el servidor. Por ejemplo, se podría añadir o adjuntar un valor específico al mensaje servidor para proporcionar dicha información a la unidad de usuario cliente.

[0042] Volviendo a la forma de realización mostrada en la figura 3, para ejecutar una actualización de etiqueta, el servidor enviará a la unidad de usuario cliente un mensaje específico (es decir, un mensaje de actualización) que incluye un valor de actualización (indicado con la letra X en esta figura) que se usará al derivar el nuevo valor de etiqueta t'_c. Por ejemplo, este valor de actualización (X) puede ser un valor aleatorio, una información de encabezado o información de metadatos. Después de este paso, el servidor esperará que la unidad de usuario cliente inicie el procedimiento de solicitud de nuevo desde el principio, es decir, justo después de la fase de inicialización y después de haber introducido el nuevo valor de etiqueta t'_c que, por un lado, se deriva del anterior valor de etiqueta t_c y, por otro lado, del valor de actualización (X) comprendido en el mensaje de actualización. De esta manera, el servidor puede verificar fácilmente si la etiqueta se ha actualizado o no. Mientras que no se haya recibido ninguna firma que corresponda al nuevo valor de etiqueta, el servidor usará la antigua etiqueta y seguirá repitiendo el paso de ejecución de la actualización de la etiqueta. Alternativamente, el servicio solicitado se puede denegar, por ejemplo después de un cierto número de intentos fallidos. Por lo tanto, se puede realizar una prueba con el objetivo de contar (por medio de un contador específico) el número de intentos fallidos, hasta un umbral predefinido, para decidir enviar un mensaje servidor a la unidad de usuario sin actualizar el valor de etiqueta t_c o denegar el servicio solicitado. Una vez que se ha recibido una firma que corresponde a la nueva etiqueta, el servicio solicitado se puede entregar al cliente.

[0043] Esta tercera forma de realización principal, como se muestra en la figura 3, requiere además:

a) reemplazar los pasos en el resultado positivo de la prueba de condición de acceso realizada dentro de la fase operativa por un paso condicional con el objetivo de verificar si el valor de etiqueta (t_c) debe actualizarse:

- en caso positivo: en primer lugar, actualizar el valor de etiqueta t_c, tanto en el lado servidor como en el lado de unidad de usuario, enviando el servidor a la unidad de usuario un mensaje de actualización con un valor de actualización X y reemplazando el valor de etiqueta t_c por un nuevo valor de etiqueta t'_c derivado del último valor de etiqueta t_c y del valor de actualización X, almacenar luego el nuevo valor de etiqueta t'_c en el registro de la base de datos del servidor y en la memoria de la unidad de usuario,
- en caso negativo: enviar directamente el mensaje servidor a la unidad de usuario, como una respuesta a la solicitud cliente (en este ultimo caso, el valor de etiqueta t_c no se actualiza),

b) reemplazar los pasos en el resultado negativo de dicha prueba de condición de acceso por una prueba que tiene como objetivo, por medio de un contador de intentos fallidos usado para contar una serie de resultados negativos sucesivos relacionados con dicha condición de acceso, verificar si este número ha alcanzado un umbral predeterminado:

- en caso negativo: enviar un mensaje servidor a la unidad de usuario sin actualizar el valor de etiqueta (t_c),
- en caso positivo: denegar el servicio solicitado.

[0044] Con respecto al caso positivo del paso condicional que pretende verificar si el valor de etiqueta debe actualizarse (como se mencionó anteriormente en la parte denominada a) dentro de la fase operativa), debe observarse que la unidad de usuario cliente se ve obligada a reenviar la solicitud cliente usando el valor de etiqueta actualizado. Según otra variante apropiada, se puede enviar un mensaje a la unidad de usuario cliente para requerir que esta última solicite el servicio de nuevo usando el valor de etiqueta actualizado. Esto permite que el servidor se asegure de que el valor de etiqueta se haya actualizado en el lado cliente.

[0045] Aunque la forma de realización ilustrada por la figura 3 se muestra como una variante de las formas de realización mostradas en las figuras 2a y 2b, debe tenerse en cuenta que los pasos mencionados anteriormente realizados en estos casos positivos y negativos también se pueden aplicar al método según la primera forma de realización, reemplazando los pasos realizados en los resultados positivos y negativos de la prueba de condición de acceso de la primera forma de realización.

[0046] Independientemente de la forma de realización, debe observarse que el valor de etiqueta asociado a cada cliente captura el historial de uso del cliente (y se inicializa con un valor aleatorio). Preferiblemente, este debería ser único para cada cliente, en particular si no se asigna ningún identificador IDc al cliente y cambiará después de cada solicitud válida. Por lo tanto, este se puede usar como una fuente de aleatoriedad para derivar otras claves (por ejemplo, una clave de sesión para comunicaciones seguras entre el cliente y el servidor) o para habilitar la diversidad de *software* en el tiempo de ejecución.

[0047] Los valores aleatorios pueden ser proporcionados por el servidor, por ejemplo durante la primera comunicación con el cliente, o por el *software*, por ejemplo durante su primer uso y/o la primera comunicación con el servidor.

[0048] Preferiblemente, los mensajes y/o valores enviados entre el servidor y la unidad de usuario cliente en el método de la presente invención se intercambian dentro de un canal de comunicación seguro, independientemente

de la forma de realización usada. Dicho canal de comunicación seguro se puede obtener mediante la encriptación de al menos una parte de contenidos/comunicaciones intercambiados. Por ejemplo, al menos una parte del contenido de un mensaje se puede encriptar. Además, las comunicaciones intercambiadas se pueden firmar. Asimismo, el valor de etiqueta t_c, t'_c se puede usar para derivar al menos una encriptación de clave para encriptar dichos contenidos intercambiados.

[0049] Ventajosamente, la presente invención permite, en primer lugar, producir copias de *software* idénticas destinadas a ser distribuidas a clientes individuales y, en segundo lugar, después de la fase de inicialización, proporcionar una individualización de cada implementación de *software* que llevará su propia vida al tener datos únicos a su disposición, que se puede usar para permitir la diversidad espacial.

[0050] Dependiendo de la lógica comercial, a veces puede desearse permitir una cantidad limitada de clones de *software*. Se puede permitir a los usuarios copiar una implementación de *software* cliente en otro dispositivo y tener una cantidad limitada de clones que se ejecutan de forma independiente. Para lograr esto, el servidor bifurcará una nueva tupla (IDc1, tc1) desde la tupla original (IDc, tc) tan pronto como se haya detectado una etiqueta incorrecta y las políticas permitan un nuevo clon. El identificador presentado debe ser un identificador IDc correcto. La nueva tupla se puede asociar con la identidad original, por ejemplo, almacenando el nuevo identificador cliente IDc1 clonado en un registro específico del identificador cliente IDc original (dentro de la base de datos del servidor), para mantener la asociación entre la nueva tupla clonada y el identificador cliente IDc original (o el registro de usuario cliente original). Al proporcionar medios de conteo (por ejemplo, medios para incrementar en una unidad) para contar el número de asociaciones, es decir, el número de aquellos registros específicos que están asociados a un identificador cliente original, esto permite controlar cuántos clones tiene este identificador original y conocer el identificador (IDc1, IDc2, ...) de estos clones autorizados. Se pueden proporcionar medios de comparación para comparar el número de estos registros específicos con un umbral predefinido que determina el número máximo de clones autorizados para un cierto identificador cliente original. Si el resultado de esta comparación es igual o superior a este umbral, se denegará la solicitud y no se permitirá ningún nuevo clon autorizado para este identificador cliente original. Al contrario, si no se alcanza el umbral (o no se excede), se aplicará un nuevo identificador IDc1 y una nueva etiqueta tc1 en la unidad de usuario cliente clonado usando un comando dedicado o cualquier operación adecuada para proporcionar estos nuevos datos a esta nueva unidad de usuario cliente . A partir de entonces, el clon puede verse como una nueva unidad de usuario auténtica.

[0051] Según una forma ligeramente diferente, para controlar cuántos clones tiene una determinada identidad, cada registro cliente (identificado por su identificador cliente IDc) puede incluir un valor de conteo. Este valor es incrementado (o disminuido) en uno cada vez que se deriva un clon de *software* de este cliente. De esta manera, el servido sabe, en cualquier momento, cuántos clones de *software* han sido generados por un determinado cliente.

[0052] Para evitar que un identificador cliente clonado genere más clones de *software* a su vez, cada registro resultante de una generación de clones de *software* recibirá una etiqueta de clonación usada para identificar qué identificador cliente es un denominado "identificador clonado" y qué identificador cliente es un identificador inicial. La misma función que la proporcionada por tal etiqueta de clonación se puede obtener al almacenar, en el nuevo registro correspondiente al denominado "identificador cliente clonado", un valor de conteo que alcanza inmediatamente el umbral anteriormente mencionado.

[0053] La cantidad de limitación de clones de *software* es una variante que es aplicable a las formas de realización mostradas en la figura 2b y la figura 3. Para llevar a cabo dicha limitación de clones de *software*, los siguientes pasos, que se refieren a la forma de realización mostrada en la figura 2b, deben tenerse en cuenta con respecto al resultado negativo de la prueba de condición de acceso del método. Este resultado negativo será sustituido por (o también incluirá) el siguiente paso condicional con los otros pasos siguientes:

- (o) verificar si el identificador cliente IDc incluido en el mensaje cliente ya está almacenado en uno de los registros de la base de datos: en caso negativo, denegar el servicio solicitado, mientras que en caso positivo:

- incrementar en una unidad un valor de conteo de un contador cliente asociado a dicho identificador cliente IDc en su registro, luego verificar si este contador cliente alcanza un umbral predeterminado, en caso positivo: denegar el servicio solicitado, mientras que en caso negativo (la generación de un nuevo clon está autorizada):

- almacenar el valor de conteo del contador cliente incrementado en dicho registro,

- asignar un nuevo identificador cliente único IDc al cliente al almacenar, en un nuevo registro de la base de datos, este nuevo identificador cliente único IDc junto con un nuevo valor de etiqueta t'_c (valor aleatorio) y con un valor de conteo, almacenado como contador cliente, que alcanza dicho umbral predeterminado,

- enviar un mensaje servidor a la unidad de usuario, como una respuesta a la solicitud cliente,

- proporcionar el nuevo identificador cliente único IDc y el nuevo valor de etiqueta t'_c al cliente (unidad de usuario cliente),

- almacenar el nuevo valor de etiqueta t'_c en el registro de la base de datos del servidor y en la unidad de usuario cliente (es decir, en la memoria de la unidad de usuario cliente).

[0054] Para proporcionar el nuevo cliente IDc y el nuevo valor de etiqueta a la nueva unidad de usuario cliente, esta operación (este paso) se puede conseguir mediante cualquier medio de comunicación convencional, como, entre otros, una carta, un SMS, una llamada telefónica, un correo electrónico, una página web o cualquier combinación de los mismos. Alternativamente, esta información se puede transmitir al cliente, incluido el nuevo identificador cliente único y el nuevo valor de etiqueta en el mensaje servidor enviado a la unidad de usuario cliente. Esta información también puede se puede adjuntar al mensaje servidor o se puede enviar por separado.

[0055] Cabe señalar que el método de la presente invención no evita que los clones reproduzcan contenido ya comprado: solo evita que los clones puedan solicitar nuevos servicios. Esto es un problema inherente con reproductores multimedia cliente que se pueden usar desconectados, ya que solo se puede realizar una verificación del lado servidor cuando el componente de *software* cliente se conecta al servicio en línea.

[0056] La implementación de *software* cliente debe ser con estado.

[0057] En cuanto a los problemas de privacidad, el método sugerido en esta invención divulga que los valores de etiqueta se derivan del historial de uso y del valor de etiqueta inicial. Sin embargo, cuando se usan funciones *hash* criptográficas para derivar nuevos valores de etiqueta, estos valores no exponen ninguna información de historial de uso (debido a la propiedad de resistencia preimagen de las funciones *hash* criptográficas).

[0058] Además, también es posible incluir información adicional que se envía desde el servidor hasta el cliente; por ejemplo, un desplazamiento aleatorio que se puede incluir en el proceso de actualización o un nuevo valor *hash* encriptado.

REIVINDICACIONES

1. Método para detectar un *software* clonado para ser usado en una unidad de usuario cliente que se comunica con un servidor para solicitar un servicio enviando una solicitud desde la unidad de usuario hasta el servidor, donde el último está conectado a una base de datos que comprende registros cliente, donde cada uno de estos registros comprenden al menos un valor de etiqueta (t_c), donde dicho método incluye una fase de inicialización y una fase operativa,

a) la fase de inicialización comprende los pasos de:

- definir dicho valor de etiqueta (t_c) como igual a un valor inicial,
- abrir un nuevo almacenamiento de registro que almacena dicho valor de etiqueta (t_c) en el servidor,
- introducir dicho valor de etiqueta (t_c) en la unidad de usuario cliente,

b) la fase operativa comprende los pasos de:

- preparar, en el lado de unidad de usuario, un mensaje cliente para el servidor que comprende dicha solicitud y dicho valor de etiqueta (t_c), luego
- enviar este mensaje cliente, desde la unidad de usuario hasta el servidor,
- realizar una prueba de condición de acceso, en el lado servidor, controlando si el valor de etiqueta (t_c) de dicho mensaje cliente está comprendido en la base de datos, en resultado negativo: denegar el servicio solicitado,

en resultado positivo:

- enviar un mensaje servidor a la unidad de usuario, como una respuesta a dicha solicitud,
- calcular en el lado servidor y en el lado unidad de usuario un nuevo valor de etiqueta (t'_c) usando una función *hash* configurada para usar el último valor de etiqueta (t_c) y al menos una parte del mensaje cliente o una parte del mensaje servidor como datos de entrada de dicha función *hash* y para proporcionar dicho nuevo valor de etiqueta (t'_c) como datos de salida de dicha función *hash*,
- actualizar dicho valor de etiqueta (t_c), tanto en el lado servidor como en el lado unidad de usuario, reemplazándolo por el nuevo valor de etiqueta (t'_c),
- almacenar el nuevo valor de etiqueta (t'_c) en el registro de la base de datos del servidor y en la unidad de usuario.

2. Método según la reivindicación 1, donde dicha parte de mensaje es una marca de tiempo.

3. Método según cualquiera de las reivindicaciones precedentes, donde el mensaje cliente comprende además un identificador único (IDc) asignado al cliente.

4. Método según cualquiera de las reivindicaciones precedentes, donde

a) la fase de inicialización comprende además los pasos de:

- definir una función de firma y una función de derivación de clave de firma para encriptar un resumen resultante de dicha función de firma,
- compartir la definición de dicha función de firma y dicha función de derivación de clave de firma entre la unidad de usuario y el servidor,

b) la fase operativa comprende además los pasos de:

- calcular un código de autenticación aplicando dicha función de firma a la solicitud y usando la clave de firma para encriptar el resumen resultante de dicha función de firma, luego cambiar el paso de preparación del mensaje cliente al preparar un mensaje cliente que comprende dicho código de autenticación y dicha solicitud,
- cambiar la condición de acceso verificando si el código de autenticación recibido en el mensaje cliente es igual a un código de autenticación calculado por el servidor aplicando la misma función de firma a la solicitud comprendida en el mensaje cliente y usando la misma clave de firma para procesar dicho resumen; donde esta clave de firma se deriva preferiblemente del valor de etiqueta esperado que se almacena en la base de datos del servidor.

5. Método según la reivindicación 4, donde

a) la fase de inicialización comprende además los pasos de:

- asignar un identificador único (IDc) al cliente y almacenar este identificador (IDc) en dicho nuevo registro,
- compartir dicho identificador (IDc) entre la unidad de usuario y el servidor,

b) la fase operativa comprende además el paso de:

- cambiar el paso de preparación del mensaje cliente al incluir el identificador cliente (IDc) en el mensaje cliente.

6. Método según cualquiera de las reivindicaciones precedentes, donde la fase operativa comprende un paso de resincronización que reemplaza el valor de etiqueta (t_c) por un nuevo valor de etiqueta (t'_c) igual a un nuevo valor aleatorio, luego enviar dicho nuevo valor de etiqueta (t'_c) a la unidad de usuario.

7. Método según la reivindicación 4 o 5, donde dicha clave de firma es idéntica al valor de etiqueta (t_c) del cliente.

8. Método según la reivindicación 4 o 5, donde dicha clave de firma se deriva del valor de etiqueta (t_c) del cliente.

9. Método según cualquiera de reivindicaciones precedentes, donde dichos mensajes y/o valores enviados entre el servidor y la unidad de usuario se intercambian dentro de un canal de comunicación seguro.

10. Método según la reivindicación 9, donde dicha comunicación segura se obtiene al encriptar al menos una parte de los contenidos intercambiados.

11. Método según la reivindicación 10, donde el valor de etiqueta (t_c, t'_c) se usa para derivar al menos una encriptación de clave para encriptar dichos contenidos intercambiados.

12. Método según cualquiera de reivindicaciones precedentes, donde el resultado positivo en la prueba de condición de acceso realizada en la fase operativa se sustituye por la realización de un paso condicional destinado a verificar si el valor de etiqueta (t_c) debe actualizarse:

- en caso positivo: en primer lugar, actualizar el valor de etiqueta (t_c), tanto en el lado servidor como en el lado unidad de usuario, enviando desde el servidor hasta la unidad de usuario un mensaje de actualización con un valor de actualización (X) y sustituyendo dicho valor de etiqueta (t_c) por un nuevo valor de etiqueta (t'_c) derivado del último valor de etiqueta (t_c) y de dicho valor de actualización (X), y almacenar el nuevo valor de etiqueta (t'_c) en el registro de la base de datos del servidor y en la unidad de usuario;
- en caso negativo: enviar directamente dicho mensaje servidor a la unidad de usuario, como una respuesta a dicha solicitud;

y donde el resultado negativo en dicha prueba de condición de acceso se sustituye de la siguiente manera:

- realizar un paso condicional, por medio de un contador de intentos fallidos contador usado para contar una serie de consecuencias negativas sucesivas sobre dicha condición de acceso, para verificar si este número ha alcanzado un umbral predeterminado:
- en caso negativo: enviar un mensaje servidor a la unidad de usuario sin actualizar el valor de etiqueta (t_c),
- en caso positivo: denegar el servicio solicitado.

13. Método según cualquiera de las reivindicaciones 5 a 12, donde el resultado negativo en dicha prueba de condición de acceso comprende además los siguientes pasos:

- verificar si el identificador cliente (IDc) incluido en el mensaje cliente ya está almacenado en uno de los registros de la base de datos: en caso negativo, denegar el servicio solicitado, mientras que en caso positivo:
- incrementar en una unidad el valor de un contador cliente asociado a dicho identificador cliente (IDc) en su registro, luego verificar si este contador cliente alcanza un umbral predeterminado, en caso positivo: denegar el servicio solicitado, mientras que en caso negativo:
- almacenar el valor del contador cliente incrementado en dicho registro,
- asignar un nuevo identificador cliente único (IDc) al cliente al almacenar, en un nuevo registro de la base de datos, este nuevo identificador cliente único (IDc) junto con un nuevo valor de etiqueta (t'_c) y con un valor, almacenado como contador cliente, que alcanza dicho umbral predeterminado,
- enviar un mensaje servidor a la unidad de usuario, como una respuesta a dicha solicitud,
- proporcionar dicho nuevo identificador cliente único (IDc) y dicho nuevo valor de etiqueta (t'_c) a su unidad de usuario cliente,
- almacenar el nuevo valor de etiqueta (t'_c) en el registro de la base de datos del servidor y en la unidad de usuario cliente.

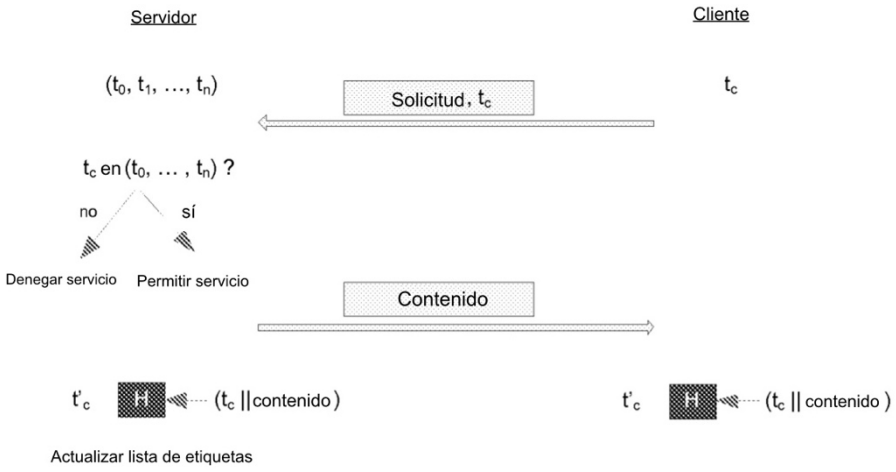

Figura 1

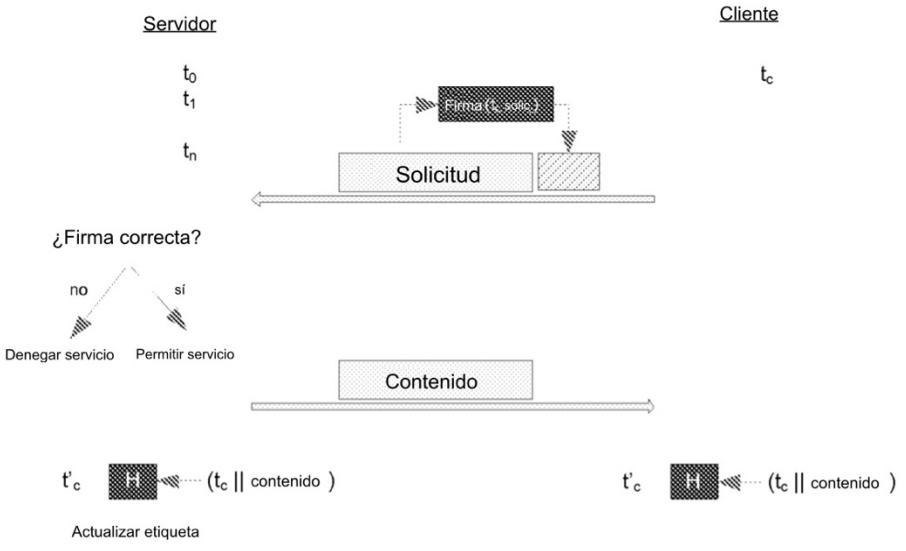

Figura 2a

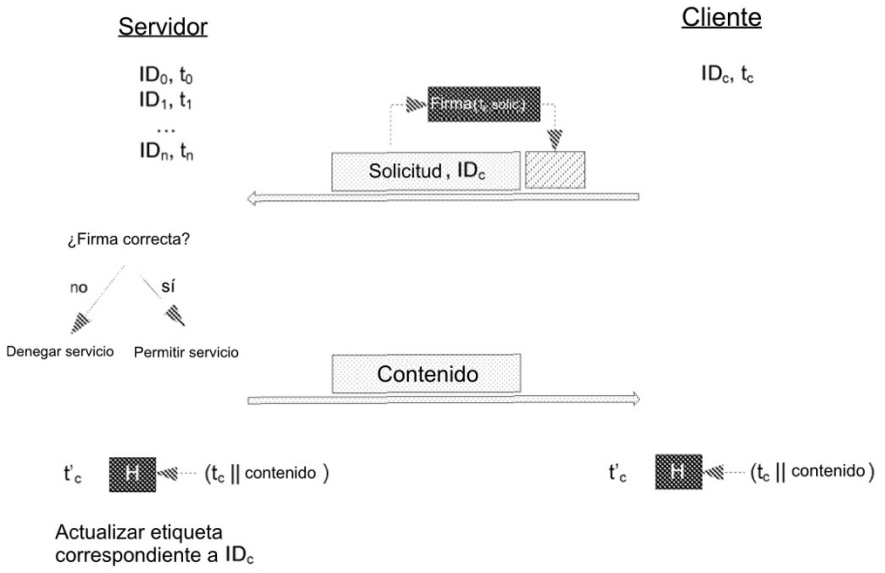

Figura 2b

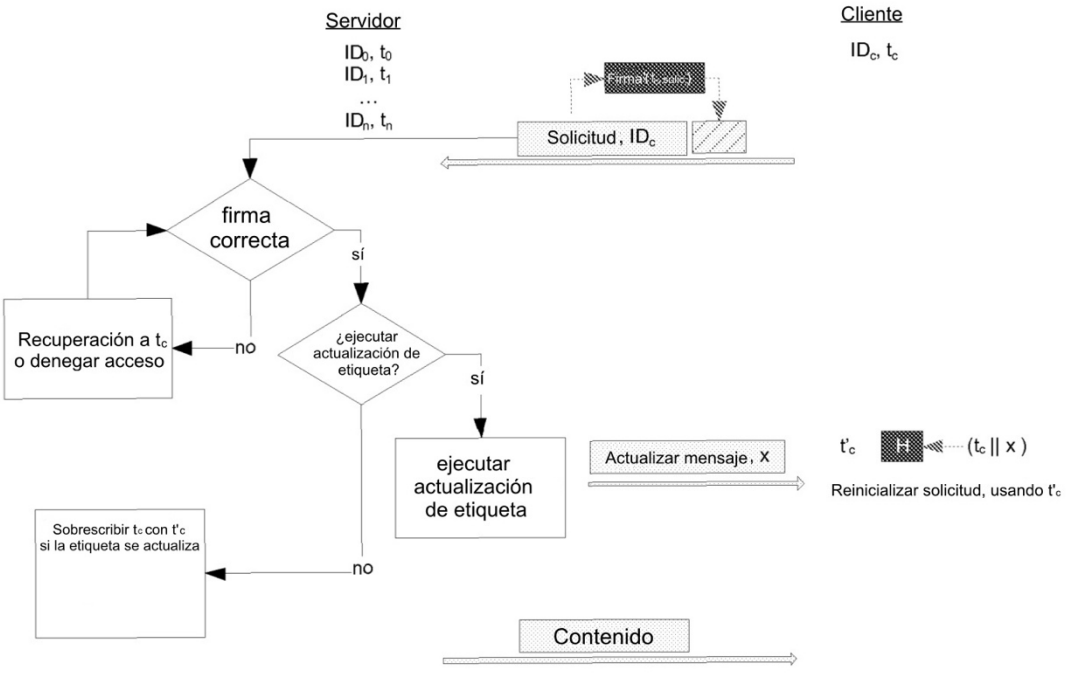

Figura 3