(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 553 985**

(51) Int. CI.:

H04L 9/08 (2006.01)
H04L 9/32 (2006.01)
H04N 21/235 (2011.01)
H04N 21/262 (2011.01)
H04N 21/6334 (2011.01)
H04N 21/81 (2011.01)
H04N 7/16 (2011.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Fecha de presentación y número de la solicitud europea: **01.12.2003 E 03777041 (9)**

(97) Fecha y número de publicación de la concesión europea: **02.09.2015 EP 1570648**

(54) Título: **Método de protección de actualizaciones de software**

(30) Prioridad:

03.12.2002 CH 204302

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
15.12.2015

(73) Titular/es:

NAGRAVISION SA (100.0%)
22, ROUTE DE GENÈVE
1033 CHESEAUX-SUR-LAUSANNE, CH

(72) Inventor/es:

SASSELLI, MARCO y
PICAN, NICOLAS

(74) Agente/Representante:

TOMAS GIL, Tesifonte Enrique

DESCRIPCIÓN

Método de protección de actualizaciones de software.

[0001] La presente invención se refiere a un método de protección de las actualizaciones de programas informáticos de utilización que asegura el funcionamiento de los sistemas más diversos.
De una forma más particular, el método de la invención utiliza un mecanismo de firma numérica con una clave privada de un algoritmo de encriptado asimétrico.

[0002] Un sistema se define aquí como un aparato o un conjunto de aparatos cuyo funcionamiento depende de uno o más programas almacenados en una memoria no volátil o un disco duro.
Cuando las funcionalidades del sistema se deben mejorar o completar con el fin de adaptarse a las exigencias crecientes del usuario, es a menudo necesario actualizar únicamente el software que corresponda sin por lo tanto cambiar el hardware que constituye el sistema.

[0003] La actualización de un software dado se efectúa en general mediante el reemplazo de ficheros de un software ya instalado o por adición de nuevos ficheros que completan los que están almacenados.
El conjunto constituye entonces una nueva versión del software previamente instalado en el sistema que se beneficia así de las mejoras deseadas.

[0004] Numerosos aparatos tales como ordenadores y sus periféricos, máquinas expendedoras, teléfonos fijos y móviles, descodificadores de televisión de pago, etc. se pilotan a través de programas adaptados a su configuración y a las funciones de componentes específicos.

[0005] Por ejemplo, en un descodificador de televisión de pago (Set Top Box), un software de explotación gestiona periféricos tales como un disco duro, un lector de tarjetas de chip, interfaces de recepción de datos y memorias.
Con el fin de introducir cambios ya sea a nivel de la configuración, ya sea a nivel de las funcionalidades, a veces es necesario reemplazar el software existente o aportar las mejoras al que ya está instalado en el descodificador.
Este tipo de evolución se efectúa mediante porciones de programas que se llaman actualizaciones o parches proporcionados por el centro de gestión del operador al cual están abonados un cierto número de usuarios.
Estas actualizaciones las proporciona regularmente el centro de gestión y se descargan en los descodificadores de cada abonado que posea los derechos necesarios.

[0006] El documento WO98/43431 describe un método de descarga de aplicaciones en un receptor/descodificador.
El software de la aplicación se divide en módulos y la descarga de los módulos está precedida por una búsqueda de un módulo repertorio en una dirección local determinada.
Los módulos se firman y el módulo repertorio se firma y se encripta de forma que una sola encriptación se aplica a todos los módulos que forman la aplicación.
Varias claves públicas de codificación se almacenan en una memoria de sólo lectura (ROM) del receptor/descodificador.
Las aplicaciones pueden ser así creadas por diferentes fuentes sin necesitar el conocimiento de cada una de sus claves privadas.
Una firma del repertorio se puede disimular en una posición variable dentro de un bloque de datos arbitrarios en el módulo repertorio.
Una aplicación que se va a descargar se puede verificar por comparación con un mapa de bits de validación de aplicación almacenado en el receptor/descodificador.

[0007] El documento WO01/35670 describe un método de autentificación de informaciones transmitidas a un descodificador de televisión de pago.
Un objeto software y una estructura de datos separada que contiene informaciones de autorización se firman digitalmente con una firma global que cubre los dos objetos.
Estos últimos se transmiten separadamente al descodificador.
Cuando estos dos objetos son recibidos por el descodificador, la firma es verificada.

[0008] Los usuarios de un descodificador poseen en general un contrato de pago con un operador que les garantiza un servicio de mantenimiento regular del software instalado en el descodificador.
Con el fin de limitar el abuso por copias no autorizadas y por introducción de componentes software extranjeros, resulta indispensable proteger las actualizaciones del software del descodificador.
Un método conocido consiste en utilizar una huella codificada con una clave de un algoritmo de encriptado asimétrico de tipo RSA. El software de actualización se proporciona en línea con una huella obtenida por medio de una función de comprobación aleatoria de dirección única (Hash). Esta huella constituye una imagen única que representa el conjunto de la actualización y se admite que no existen dos huellas idénticas en dos mismos conjuntos de datos diferentes.
Esta huella se encripta con ayuda de una clave privada del operador asociada a un conjunto de abonados, lo que constituye una firma propia de este conjunto.
El software acompañado de esta firma se carga en una memoria de acceso aleatorio RAM (Random Access Memory) del descodificador.
Un programa que forma parte del software del descodificador calcula con la función de comprobación aleatoria (Hash)

una huella del software almacenado en la memoria RAM. La firma recibida se descodifica con una clave pública contenida en el descodificador y después se compara con la huella del software calculada anteriormente.
Si la firma descodificada corresponde a la huella resultante de la comprobación aleatoria, la firma que acompaña la actualización almacenada en la memoria RAM se considera como válida.
El software de actualización se instalará en la memoria no volátil del descodificador (memoria Flash).

[0009] La protección se realiza así mediante la verificación de la firma con una clave pública en el descodificador correspondiente a la clave privada del operador.

[0010] La clave pública que reside en el descodificador debe ser fija, así como el programa que permite la verificación de la firma.
La autenticidad de la firma está asegurada por el hecho de que la clave privada depende de la clave pública del descodificador.
La firma no se puede reproducir porque la clave privada sólo la conoce un operador determinado.
Además, una misma firma es inservible para varias actualizaciones diferentes porque es una función de una actualización bien definida.
Un software de actualización firmado y cuyo contenido se modifica en la memoria RAM dará otra huella que no puede por lo tanto ser validada por la firma desencriptada por medio de la clave pública.
De hecho, la huella resultante después de la comprobación aleatoria de la actualización a nivel del descodificador es diferente de la obtenida después de la desencriptación de la firma.

[0011] Sin embargo, este método de protección comprende un punto débil que es la propia clave privada del operador.
De hecho, cuando ésta es descubierta por un tercero, éste puede firmar un software cualquiera e introducir modificaciones abusivas en el sistema.

[0012] Este descubrimiento se puede hacer en forma de iteración sobre la clave pública que este tercero habrá extraído en el descodificador, hasta que descubre el par de claves correcto.
La defensa que consiste en modificar el comportamiento el software del descodificador para que rechace las firmas generadas con la clave descubierta es insuficiente porque el tercero puede esquivar estas modificaciones con programas adecuados.

[0013] El documento WO99/49611A describe un método de transmisión protegida de datos entre un primer dispositivo y un segundo dispositivo.
Cada dispositivo contiene una secuencia idéntica de claves de codificación.
Un marcador indica la misma clave en la secuencia del primer y del segundo dispositivo.
El primer dispositivo transmite al segundo dispositivo datos encriptados con la clave designada por el marcador.
El segundo dispositivo recibe estos datos y los desencripta con la clave designada por el marcador en la secuencia de dicho segundo dispositivo.
Después de la desencriptación de los datos, los marcadores de las secuencias del primer y del segundo dispositivo son incrementados o se desplazan hacia una clave siguiente de manera sincronizada con el fin de preparar cada dispositivo para una nueva transmisión de datos encriptados.

[0014] El objetivo de la presente invención es reducir considerablemente el impacto del descubrimiento de una clave privada mediante un análisis sistemático del funcionamiento del software del descodificador o incrementar considerablemente el tiempo y los medios necesarios en el proceso utilizado para su determinación.

[0015] El objetivo se alcanza mediante un método de protección de actualización de datos según la reivindicación 1.

[0016] Los datos de una actualización los transmite el centro de gestión en forma de un parche y de un bloque de control que comprende una firma constituida por la huella del parche encriptado con una clave privada del centro de gestión.
El descodificador almacena estos datos en la memoria de acceso aleatorio RAM con el fin de llevar a cabo su tratamiento.
Una clave pública, asociada a esta clave privada llamada clave corriente, se selecciona a partir de una lista almacenada en una primera memoria no volátil con el fin de descodificar la firma del parche.
En caso de que la desencriptación y la verificación tengan éxito, se ejecuta un control que da lugar a la instalación del parche en una segunda memoria no volátil (flash) del descodificador.
La clave corriente así utilizada se desactiva en la lista, lo que hace que la clave siguiente esté disponible para la próxima actualización.

[0017] Cuando la verificación de la firma y la desencriptación de una actualización del descodificador se efectúa con una clave pública de la lista, ésta se borra y se convierte en inservible para próximas actualizaciones.
Por lo tanto, en cada actualización, se utiliza una nueva clave que luego se elimina de la lista.
Las claves públicas de la lista deben ser no modificables como el programa que sirve para la verificación de la firma. El documento WO00/56009 describe un sistema de seguridad y un método para proteger un acceso de un terminal distante o de un ordenador a un ordenador host utilizando contraseñas muy largas y/o una gran base de datos de claves

de identificación.
El método de control de acceso a un recurso accesible a partir de una pluralidad de unidades de usuarios comprende las etapas de: generación de una pluralidad de conjuntos de claves únicas largas, creación de una base de datos de claves que comprende una recopilación de cada conjunto de claves, almacenamiento de la base de datos de claves en un medio de almacenamiento conectado a un host, programación de una unidad de usuario para comunicarse con el host y para transmitir o recibir las claves de o a partir del host, programación del host para comunicar con la unidad de usuario y para transmitir o recibir claves de o a partir de la unidad de usuario y distribución de uno o varios conjuntos de claves de usuario a un usuario autorizado del recurso, donde dicho conjunto de llaves de usuario está registrado en un medio de almacenamiento asociado a la unidad de usuario antes o después la distribución del conjunto de claves de usuario al usuario autorizado.
El conjunto de llaves de usuario es del tipo «teclado de un solo uso», es decir, que cada clave del conjunto sólo se utiliza una vez.
Según una variante, el medio de almacenamiento de la unidad de usuario comprende una memoria inscribible en la cual las claves utilizadas del conjunto de claves de usuario son borradas o superpuestas.
La lista sólo es modificable (eliminación de las llaves utilizadas) por el programa de verificación.

[0018] El método descrito anteriormente permite reducir considerablemente las posibilidades de modificaciones del descodificador por un tercero que ha descubierto una clave privada.
Ya que una clave sólo es utilizable una vez, el tercero no efectuará más que una sola modificación.
En consecuencia, una modificación del comportamiento del descodificador para protegerlo de los efectos del pirateo resulta más eficaz porque el tercero, al no no disponer más de una clave privada válida, se encuentra así ante de un aparato inaccesible.

[0019] Debido a que las claves privadas utilizables son más numerosas, un tercero debe por lo tanto atacar sistemáticamente todas las claves para no ser bloqueado en una actualización que corresponda a la clave que ha descubierto.
Hay que saber que la evolución del software de los descodificadores es rápida y que se considera que si una de las claves se descubre, las actualizaciones siguientes volverán a cubrir los fallos de seguridad que el tercero hubiera podido introducir.
Si este tercero es capaz de bloquear toda actualización posterior, las funcionalidades del descodificador se convertirán en obsoletas rápidamente y por lo tanto no tendrán un gran perjuicio para el operador.

[0020] El uso de claves asimétricas es importante en este contexto porque la extracción de las claves públicas de un descodificador no permite fabricar una actualización aceptable puesto que esta actualización debe ser firmada por la clave privada del operador.
Es común el colocar las claves privadas en la parte protegida (el operador) y las claves públicas en la parte de dominio público (el descodificador).
Sin embargo, es posible invertir estas claves sin perjudicar el funcionamiento de la presente invención.

[0021] Una primera forma de realización de la invención propone el uso de claves públicas seleccionadas en un orden predeterminado a partir de la lista.
Así, cada clave de la lista se selecciona en cuanto la clave precedente es utilizada.

[0022] La invención se comprenderá mejor gracias a la descripción detallada que aparece a continuación y que se refiere a las figuras anexadas que sirven de ejemplo de forma no limitativa, a saber:

- la figura 1 representa el desarrollo de una etapa de actualización de un descodificador de una versión N a una versión N+1.

- la figura 2 muestra una actualización de una versión N a una versión R

[0023] En el ejemplo ilustrado por la figura 1, un descodificador de versión inicial se actualiza a la versión 1 con un parche P1. Este parche P1 es transmitido con su firma $(H(P1))_{PK1}$ por el centro de gestión del operador hasta el descodificador.
La etapa de actualización empieza por la carga del parche P1 en la memoria RAM del descodificador.

[0024] La firma $(H(P1))_{PK1}$ se obtiene por encriptación de la huella H(P1) del parche P1 con la clave privada PK1 del operador, operación efectuada en el centro de gestión.
Esta huella se calcula por el operador a partir del parche P1 con una función de comprobación aleatoria H de dirección única de tipo Hash.

[0025] El software del descodificador se carga a continuación de descodificar la firma $(H(P1))_{PK1}$ recibida con una clave pública K1 con el fin obtener la huella del parche $H(P1)_1$. Paralelamente, este mismo software calcula la huella $H(P1)_2$ del parche P1 almacenada en la memoria RAM. La primera huella salida de la desencriptación de la firma $H(P1)_1$ y la segunda $H(P1)_2$ resultante del cálculo por la función de comprobación aleatoria H son comparadas.
Si los dos valores concuerdan, el parche P1 se instala en la memoria no volátil Flash FH del descodificador que realiza

así la actualización del software del descodificador.
La clave pública K1 utilizada para descodificar la firma se borra de la lista.

[0026] Una segunda actualización de la versión 1 a la versión 2 transmitida por el centro de gestión en forma de un nuevo parche P2 acompañado de su firma $(H(P2))_{PK2}$ pasa por el mismo procedimiento de descarga y de verificación.
Una nueva clave pública K2 seleccionada de la lista será entonces utilizada del lado del descodificador.
Todas las actualizaciones siguientes transmitidas se verifican de la misma manera utilizando cada vez una nueva clave pública seleccionada de la lista.
Las claves de las actualizaciones precedentes son neutralizadas ya sea mediante borrado, ya sea mediante una marcación adecuada.

[0027] Al aplicar este procedimiento, se efectúa una actualización de un software de la versión 1 hacia una versión N en N-1 etapas.
El centro de gestión transmitirá N-1parches con N-1 firmas correspondientes encriptadas cada una por una clave privada propia de cada versión.
La instalación de diferentes parches comprende por lo tanto la neutralización de N-1 claves públicas de la lista.

[0028] La lista de las claves públicas se puede almacenar por ejemplo en una memoria no volátil del tipo EEPROM (memoria de sólo lectura programable y borrable eléctricamente).
Después de cada uso de una clave en el momento de una actualización, ésta se borra definitivamente de la EEPROM , lo que autoriza el acceso a la clave siguiente para la próxima actualización.

[0029] Según otra forma de realización, la lista de las claves públicas no se altera por un borrado o una marcación de una clave.
Después de cada instalación de una versión de software en la memoria no volátil Flash, se incrementa un contador o se desplaza un marcador para indicar la fila de la clave que se va a seleccionar de la lista en el momento de la próxima actualización.
Así en el momento de cada actualización, sólo la clave que servirá para descodificar la firma del parche es designada, y las llaves precedentes ya no pueden ser seleccionadas porque el contador o el marcador no pueden progresar más que en un sólo sentido, el de las filas crecientes.

[0030] Según una variante, el parche puede ser encriptado por la clave privada del operador.
Una etapa suplementaria de desencriptación se agrega por lo tanto al procedimiento descrito anteriormente.
El parche P recibido y cargado en la memoria RAM se puede desencriptar con la clave pública delantera que tiene el cálculo por la función de comprobación aleatoria H de la huella que sirve para la verificación de la firma.
El cálculo de la huella puede igualmente ser efectuado sobre los parches en su forma encriptada.

[0031] La instalación de actualizaciones por terceros resulta más difícil porque cada cambio de versión exige el conocimiento de la clave actual.
Esta última cambia en cada actualización, lo que obliga el tercero a conocer todas las claves para seguir las diferentes actualizaciones.

[0032] El procedimiento descrito previamente puede presentar un problema cuando el descodificador se queda fuera de servicio cuando se deben llevar a cabo varias actualizaciones.
El paso de una versión antigua de un software a una nueva cuyo número no es consecutivo al de la versión precedente se efectúa secuencialmente en varias etapas sucesivas.
Estas últimas utilizan claves públicas diferentes seleccionadas de la lista una tras otra y en orden.
Se recuerda que el parche en sí mismo no contiene un orden que permita seleccionar una clave diferente de la clave actual.
Si ese fuera el caso, un tercero podría utilizar este control para forzar el uso de una clave que él conoce.

[0033] La figura 2 ilustra el caso de un paso de un software de una versión N a una versión R donde la diferencia R-n entre la nueva versión y la precedente excede de la unidad.
El ejemplo que aparece a continuación se refiere a un caso donde N=2 y R=5.

[0034] Un descodificador cuyo software es de versión 2 no puede descodificar directamente la firma $(H(P))_{PK5}$ de la nueva versión 5 porque la clave disponible en la lista de las claves públicas es la de la versión inmediatamente superior, es decir, la clave K3. Para la instalación de la nueva versión 5, debe poder acceder a la clave correspondiente a esta versión, es decir, la clave K5.

[0035] La solución consiste en transmitir un flujo de datos que contiene el parche P para la actualización del software del descodificador a la versión 5 firmado con la clave PK5 al cual se agrega una pluralidad de mensajes M1, M2, M3, M4 encriptados cada uno con una clave privada PK1, PK2, PK3, PK4 sacada de la lista de claves.
La memoria de acceso aleatorio RAM almacena estos mensajes así como el parche P con su firma $(H(P))_{PK5}$. Cuando la versión del descodificador es la 2, la clave de la actualización de la versión 1 a 2 ya se ha desactivado mediante la primera actualización.

El mensaje M1 es entonces ignorado porque la clave K1 que sirve para descodificarlo ya no está disponible.

[0036] Los mensajes siguentes M2, M3 y M4 sirven para desactivar una tras otra las claves públicas K2, K3, y K4 que corresponden a cada versión intermedia desde la versión 2 hasta la versión 4 antes de la versión 5. Así, para instalar la versión 5 en la memoria no volátil Flash, cada clave pública K2, K3, y K4 de la lista es utilizada y después neutralizada o borrada.
En el momento de la desencriptación del mensaje por la clave correcta, el contenido de este mensaje se reconoce y provoca la operación de neutralización de la clave actual.
Si el mensaje no se reconoce, esto significa que la clave de codificación de este mensaje no es la clave actual.

[0037] Después las desencriptaciones sucesivas y con éxito de los mensajes M2, M3, M4, la clave K5 necesaria para la desencriptación de la firma del parche $(H(P))_{PK5}$ (y del parche P) se convierte en la clave actual.
Esta última será también borrada de la lista después de la instalación del parche y la clave K6 estará presente en la cabeza de la lista para la próxima actualización de la versión 5 a la versión 6.

[0038] Un tal flujo puede por lo tanto actualizar todo un parque de descodificadores sea cual sea su versión de software gracias a los mensajes de cambio de clave que acompañan al parche.
Cada descodificador dispone de una clave pública en la lista capaz de descodificar una actualización de la versión actual después de la neutralización de los claves antiguas.

[0039] En el momento de una actualización del software de un descodificador de una versión N a una versión R donde la diferencia R-n se convierte en grande, por ejemplo superior a 10, resulta molesto para un descodificador que tiene una versión R-1 descodificar sistemáticamente todos los mensajes con el fin de verificar las órdenes de desactivación.
Este descodificador va a aplicar su clave corriente (R-1) a cada uno de estos mensajes para constatar que no puede interpretar su contenido.

[0040] Una primera solución consiste en introducir en abierto en el membrete de los mensajes del índice correspondiente a los números de diferentes versiones.
Este índice sirve únicamente para evitar la desencriptación de los mensajes que han sido encriptados por una clave diferente de la clave actual.
Este índice no selecciona la fila de la clave actual, sólo la desencriptación con éxito de un mensaje con dicha clave actual provoca el avance de una fila en la lista de claves.

[0041] Según una segunda solución, la huella del parche de actualización se encripta sucesivamente mediante todas las claves privadas de las actualizaciones precedentes.
Este procedimiento obliga una utilización de cada clave pública de la lista, una tras otra, para descodificar la firma.
En tal caso de codificación en cadena, al contrario que lo anterior, todas las claves públicas deben quedar disponibles en la memoria EEPROM del descodificador.
Por ejemplo, para una actualización de la versión 1 hacia la versión N, la huella del parche P se encripta con una clave privada de la versión N. El conjunto es a continuación encriptado con la clave privada de la versión N-1, luego con la clave de la versión N-2 y así sucesivamente hasta la versión 1. La desencriptación necesita por lo tanto el uso sucesivo de las claves públicas K1 a KN-1 correspondientes a las actualizaciones de la versión 1 a la versión N. La interrupción de este mecanismo reiterativo se hace por el reconocimiento de una marca apropiada en el resultado de la desencriptación.

[0042] Si se desea proteger los datos de actualización, una manera de proceder consiste en utilizar una clave de sesión SK generada aleatoriamente por el centro de gestión por ejemplo.
Por razones operativas de rapidez, esta clave es de tipo simétrico.
El centro de gestión codifica el parche con la clave de sesión SK y compone un conjunto de datos que comprenden la clave de sesión SK y la huella del parche de actualización.
Este conjunto es encriptado por la clave privada corriente del operador para constituir el bloque de control.

[0043] El parche codificado y el bloque de control se cargan en la memoria de acceso aleatorio RAM del descodificador.
El bloque es desencriptado por la clave pública actual en la lista, lo que da la huella del parche y la clave de sesión SK.
Esta última se aplica al parche cargado en la memoria RAM que permite su desencriptación.
Después, la huella del parche se verifica y, en caso de concordancia, el parche se instala en la memoria no volátil Flash.

[0044] La clave de sesión se puede introducir como medio de protección complementaria en una u otra de las variantes descritas anteriormente, por ejemplo:

- la actualización sencilla de una versión 1 a una versión N en varias etapas,

- la actualización de una versión N a R con ayuda de un parche y mensajes de desactivación de las claves.

[0045] En un descodificador que ha sido actualizado a numerosas reactivaciones, el número de claves públicas a disposición disminuye mientras que el número de llaves desactivadas aumenta al mismo tiempo que las actualizaciones

logradas.
Con el fin de reconstituir una lista de claves para permitir las futuras actualizaciones, una nueva lista de claves públicas se puede mandar al descodificador por el centro de gestión.
Esta lista se puede incorporar en el flujo de datos y se puede acompañar de una firma como para un parche de actualización.
Ésta se almacena en la memoria EEPROM y reemplaza la antigua lista con las llaves desactivadas.

[0046] Según una variante del método de la invención, el centro de gestión y los descodificadores disponen respectivamente de una lista de claves privadas y públicas fija.
Con cada actualización, el centro de gestión elige aleatoriamente un conjunto de claves privadas entre las de la lista y codifica la huella del parche sucesivamente con cada clave del conjunto.
El centro compone unos datos masivos que comprenden la huella encriptada (firma) y una continuación de números que corresponden a las filas de las claves seleccionadoas anteriormente.
Esta continuación se puede transmitir en abierto o encriptada con una clave de sesión.
El descodificador que recibe la continuación de números selecciona en la lista de las claves públicas, según su fila, las claves necesarias para descodificar la huella del parche.
La lista no puede entender más de una vez el mismo número de clave y la longitud de esta lista (número de llaves utilizadas) se conoce y no es modificable.
En esta variante, las listas de claves quedan fijas y no se alteran tras una instalación con éxito de un parche.
Con cada actualización, una nueva combinación de claves seleccionadas de la lista se utiliza para la firma del parche.
Un tercero deberá por lo tanto siempre disponer de un conjunto de claves para introducir una actualización en un aparato, lo que necesita medios más consecuentes que para la determinación de una sola clave.

[0047] El procedimiento de protección de actualización según la invención es independiente del servicio de emisión utilizado entre un proveedor y un usuario.
De hecho, el procedimiento puede también aplicarse sobre parches distribuidos en CD-ROM, en disquete o en cualquier otro soporte de datos digitales.

REIVINDICACIONES

1. Método de protección de actualización de datos de una pluralidad de aparatos, donde cada aparato recibe las actualizaciones de un centro de gestión, donde estas actualizaciones (P_R) permiten pasar el aparato de una versión inicial N a una versión final R, donde la diferencia (R-N) entre la versión inicial N y la versión final R supera la unidad, donde estas actualizaciones (P_R) van acompañadas de un bloque de control que comprende al menos una firma ($H(P_R)$)PK_R asociada a dicha actualización, donde esta firma ($H(P_R)$)PK_R está encriptada por una clave (PK_R) asimétrica seleccionada de una lista de claves asimétricas contenida en el centro de gestión, donde la lista correspondiente de las claves asimétricas está almacenada en el aparato, **caracterizado por** las etapas siguientes:

a) preparación por el centro de gestión de una serie de mensajes (M_1...M_{R-1}) donde cada mensaje está encriptado con una clave asimétrica corriente (PK_1 ...PK_{R-1}) y envío de la serie de mensajes (M_1 ...M_{R-1}) encriptados al aparato,
b) recepción de dichos mensajes (M_1 ...M_{R-1}) por el aparato y utilización de cada clave actual correspondiente (K_1 ...K_{R-1}) para la desencriptación de cada mensaje (M_1 ...M_{R-1}),
c) si la desencriptación del mensaje con la clave corriente tiene éxito, desactivación de cada clave corriente correspondiente (K_1 ...K_{R-1}) y activación de la clave corriente (K_R) correspondiente a la versión (R) de la actualización (P_R),
d) preparación por el centro de gestión de la actualización (P_R) y de su firma ($H(P_R)$) PK_R encriptada con la clave asimétrica actual (PK_R) y transmisión al aparato, donde dicha transmisión no comprende ninguna indicación que permita al aparato seleccionar una clave asimétrica de su lista de claves asimétricas,
e) recepción de la actualización (P_R) y de su firma ($H(P_R)$)PK_R por el aparato,
f) utilización de la clave actual correspondiente (K_R) para la desencriptación de la firma ($H(P_R)$)PK_R para obtener una primera huella $H(P_R)1$ esperada de la actualización (P_R),
g) verificación de la conformidad de la actualización (P_R) mediante el cálculo de un segunda huella $H(P_R)2$ sobre los datos de la actualización (P_R) y comparación de la primera y la segunda huella,
h) instalación de la actualización (P_R) recibida si la verificación ha sido positiva y,
i) desactivación de la clave correspondiente actual (K_R) del aparato y activación de la clave siguiente (K_{R+1}), donde dicha clave corriente correspondiente (K_R) se vuelve inservible para todo uso posterior, donde dicha desactivación se lleva a cabo después cada uso de una clave.

2. Método según la reivindicación 1, **caracterizado por el hecho de que** la segunda huella $H(P_R)_2$ es el resultado de una función de comprobación aleatoria, y por el hecho de que la verificación de la firma ($H(P_R)$) PK_R comprende la etapa de establecimiento de la segunda huella $H(P_R)_2$ de la actualización (P_R) recibida y la comparación con la primera huella $H(P_R)_1$ obtenida después de la desencriptación de la firma ($H(P_R)$)PK_R con la clave actual correspondiente (K_R).

3. Método según la reivindicación 1 o 2, **caracterizado por el hecho de que** el bloque de control comprende además una clave de sesión simétrica (SK) determinada por el centro de gestión, donde esta clave (SK) se utiliza para encriptar los datos de la actualización (P_R), donde dicha clave de sesión (SK) es encriptada por la clave asimétrica actual (PK_R).

4. Método según una de las reivindicaciones 1 a 3, **caracterizado por el hecho de que** la clave asimétrica actual correspondiente (K_R) es eliminada de la lista después su uso por el aparato.

5. Método según una de las reivindicaciones 1 a 4, **caracterizado por el hecho de que** las claves asimétricas de la lista se utilizan secuencialmente en un orden predeterminado en el momento de cada desencriptación lograda.

6. Método según una de las reivindicaciones 1 a 5, **caracterizado por el hecho de que** la lista de las claves asimétricas se almacena en una memoria no volátil del aparato, donde una clave que corresponde a una desencriptación lograda se borra definitivamente de la memoria que autoriza el acceso a la clave siguiente para el próximo uso.

7. Método según la reivindicación 1, **caracterizado por el hecho de que** el número (R-1) de mensajes (M_1 ...M_{R-1}) corresponde al número de versiones de actualización que separa la versión inicial N del aparato y la versión final R de la actualización (P_R) menos uno.

8. Método según una de las reivindicaciones 1 a 7, **caracterizado por el hecho de que** el aparato comprende un contador que, en el momento de una instalación de una actualización, se incrementa en el momento de cada verificación positiva.

9. Método según una de las reivindicaciones 1 a 7, **caracterizado por el hecho de que** el aparato comprende un marcador que indica la fila de la clave asimétrica actual que se va a seleccionar de la lista, donde este marcador se desplaza en el momento de cada verificación positiva.

10. Método según la reivindicación 6, **caracterizado por el hecho de que** una nueva lista de claves públicas se transmite al aparato, dicha lista reemplaza la lista contenida en la memoria no volátil que contiene las claves desactivadas a través de actualizaciones logradas previamente.

11. Método según una de las reivindicaciones 1 a 10, **caracterizado por el hecho de que** los aparatos consisten en descodificadores de televisión de pago.

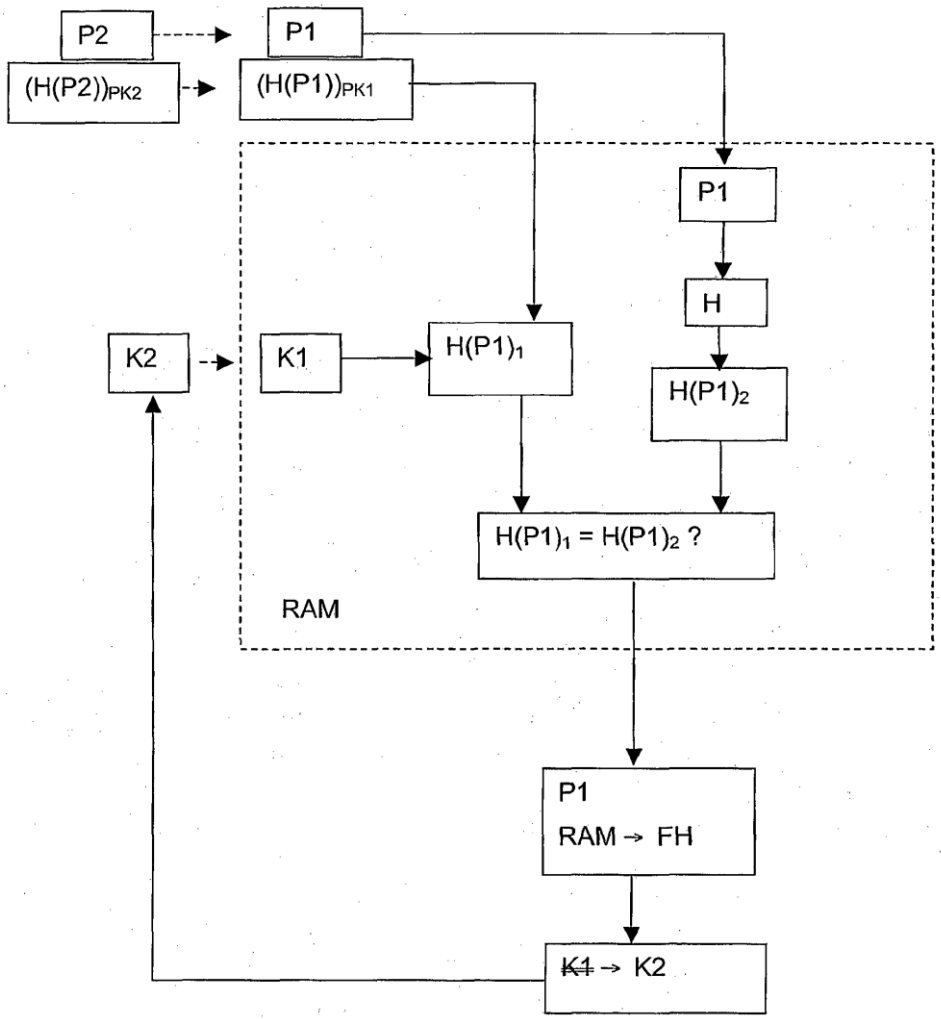
P2
P1
(H(P2))PK2
(H(P1))PK1
P1
H
K2
K1
H(P1)1
H(P1)2
H(P1)1 = H(P1)2 ?
RAM
P1
RAM → FH
K1 → K2

Fig. 1

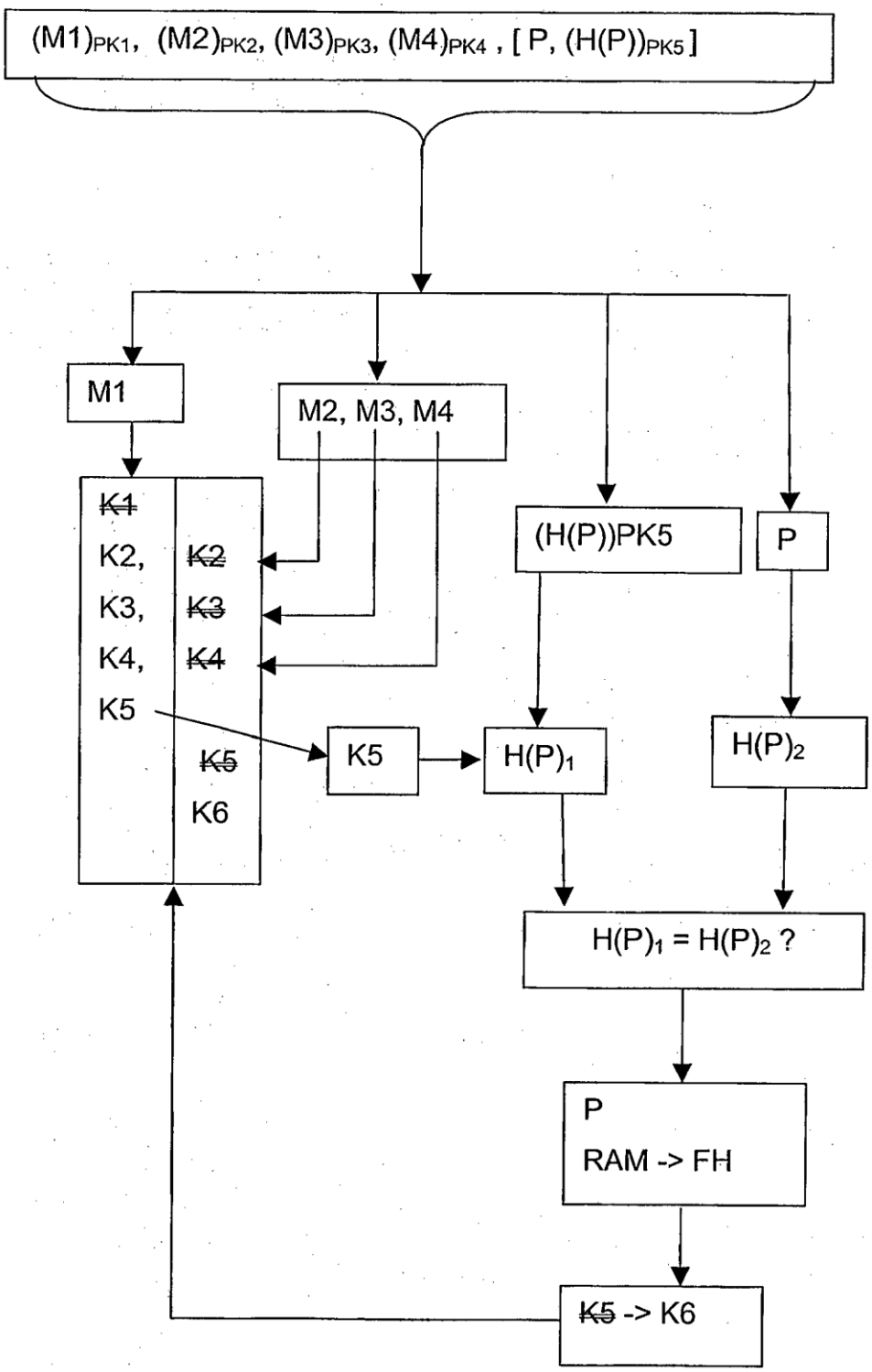
(M1)PK1, (M2)PK2, (M3)PK3, (M4)PK4 , [P, (H(P))PK5]
M1
M2, M3, M4
K1
K2,
K3,
K4,
K5
K2
K3
K4
K5
K6
(H(P))PK5
P
K5
H(P)1
H(P)2
H(P)1 = H(P)2 ?
P
RAM -> FH
K5 -> K6

Fig. 2