(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 179 775**

(21) Número de solicitud: 200101022

(51) Int. Cl.[7]: G06F 1/00

(12) PATENTE DE INVENCION B1

(22) Fecha de presentación: **04.05.2001**

(43) Fecha de publicación de la solicitud: **16.01.2003**

Fecha de concesión: **12.09.2003**

(45) Fecha de anuncio de la concesión: **16.10.2003**

(45) Fecha de publicación del folleto de patente: **16.10.2003**

(73) Titular/es: **UNIVERSIDAD DE MÁLAGA**
Plaza de El Ejido, s/n
29071 Málaga, ES

(72) Inventor/es: **López Muñoz, Javier;**
Maña Gómez, Antonio;
Ortega Daza, Juan José y
Troya Linero, José María

(74) Agente: **No consta**

(54) Título: **Sistema para la protección contra el uso ilegítimo y gestión de licencias de software basado en dispositivos procesadores autónomos y criptografía.**

(57) Resumen:
Sistema para la protección contra el uso ilegítimo y gestión de licencias de software basado en dispositivos procesadores autónomos y criptografía.
Sistema para la protección anti-copia y gestión de licencias de software, que usa un dispositivo procesador resistente a ataques físicos (p.ej. una tarjeta inteligente) conteniendo un par de claves asimétricas, basado en que durante la fase de producción del software se sustituyen algunas secciones por código que debe ejecutarse en el dispositivo, siendo tales secciones cifradas mediante un criptosistema simétrico. Para el uso del software se genera una licencia cifrada con la clave pública del dispositivo (y por tanto específica para el mismo) que contiene, entre otras cosas, la clave simétrica con la que se cifraron las secciones de código protegidas e información acerca de las condiciones de uso. Esta licencia queda almacenada en el dispositivo, que posteriormente la gestiona y la usa para la ejecución del software. También está prevista su aplicación para protección de software y datos en entornos de computación móvil y comercialización de información.

ES 2 179 775 B1

DESCRIPCION

Sistema para la protección contra el uso ilegítimo y gestión de licencias de software basado en dispositivos procesadores autónomos y criptografía.

La presente invención se refiere a un sistema para la protección contra el uso ilegítimo y gestión de licencias de software basado en dispositivos procesadores autónomos (como por ejemplo tarjetas inteligentes) y criptografía. El uso ilegal de software ha sido, desde la popularización del uso de los ordenadores, uno de los mayores problemas para su industria. Para atacar este problema han aparecido en la literatura propuestas de esquemas de protección de software basados en dispositivos hardware (dispositivos electrónicos) resistentes a ataques físicos. Todos ellos dependen de dos premisas: (a) la resistencia del dispositivo a los ataques físicos, y (b) la dificultad de analizar y modificar el software para evitar la comprobación de la presencia del dispositivo. La experiencia demuestra que la primera premisa es razonable (e incluso inevitable). Sin embargo, la segunda es poco realista porque el análisis y la modificación del código ejecutable siempre es posible. Además, las técnicas utilizadas para dificultar esta labor no son efectivas para disuadir a un usuario deshonesto con unos recursos medios. En esta invención se describe un esquema robusto de protección de software basado en el uso combinado de criptografía y dispositivos procesadores cuya seguridad depende únicamente de la primera premisa, ya que la modificación del código para evitar la comprobación de la presencia del dispositivo no permite romper este esquema.

Antecedentes de la invención

Un sistema de protección bastante extendido y simple consiste en la incorporación de un chequeo de claves o *passwords* que habilita la instalación. En caso de no disponer de la clave adecuada el software no puede ser instalado, o bien queda con alguna funcionalidad restringida. Este sistema es muy popular en el *shareware*. Pese a ello, el propio software debe contener la función de validación de claves, por lo que es posible el análisis por ingeniería inversa de tal función. Como consecuencia, son usuales tanto la aparición de *generadores de claves* (que el software aceptará como válidas), como la publicación de listas de claves válidas en determinados lugares de Internet.

En otros casos se personaliza el software para que quede ligado a una máquina determinada, por ejemplo, extrayendo información de alguno de los dispositivos hardware (disco duro, tarjeta de red, etc.) o de la configuración del sistema operativo, de forma que el software debe comprobar que se está ejecutando en la máquina adecuada. Esta comprobación, al igual que las anteriores, es susceptible de ser anulada y, además, este esquema resulta inconveniente para el usuario puesto que los cambios en el hardware o el sistema operativo requieren la reinstalación del software y la gestión de una nueva licencia en sustitución de la anterior.

Otros muchos sistemas de protección que no usan ningún dispositivo hardware utilizan técnicas como la ofuscación, reorganización y automodificación de código como se describen en el informe técnico # 170 de la Universidad de Auckland escrito por C. Collberg y colaboradores titulado "*Tamper-Proofing, and Obfuscation - Tools for Software Protection*" y en el informe técnico CS-2000-12 de la Universidad de Virginia escrito por C. Wang y colaboradores titulado "*Software Tamper Resistance: Obstructing Static Analysis of Programs*". Estas técnicas proporcionan protección a corto plazo y pueden ser usadas con éxito en entornos donde el software a proteger tiene una vida muy corta, como es el caso de agentes y applets. Algunas de estas técnicas han aparecido por primera vez en un tipo bastante especial de software: los virus informáticos. En definitiva, el objetivo en este caso no es impedir el análisis del código o la ingeniería inversa, sino más bien dificultar y retrasar esas labores.

Una aproximación muy interesante basada en el concepto de función oculta puede hallarse en el trabajo de T. Sander y C.F. Tschudin titulado "*On Software Protection via Function Hiding*", publicado en Proceedings of Information Hiding '98. Springer-Verlag. LNCS 1525. pp 111-123. 1998. Los autores presentan un esquema que permite trabajar con funciones cifradas sobre datos cifrados. La idea es establecer un homomorfismo entre el espacio de datos en claro y el de datos cifrados por algún criptosistema. Se desea que el proceso de cifrar unos datos, aplicarles una función f' y descifrar el resultado sea equivalente a aplicar otra función f a los datos originales. Este proceso puede expresarse de este modo: Sea P el dominio de los datos en claro y Q el de los datos cifrados. Sea f: $P \rightarrow P$ una función que se desea aplicar sobre unos datos $x \in P$, y sean e:$P \rightarrow Q$ y d:$Q \rightarrow P$ respectivamente las funciones de cifrado y descifrado de un cierto criptosistema. En tal caso, bajo ciertas condiciones, es posible encontrar una función f': $Q \rightarrow Q$ tal que $\forall x \in P$ $f(e(x))=e(f(x))$, o expresado de otro modo $\forall x \in P$ $d(f(e(x)))=f(x)$. La utilidad de este esquema consiste en que es posible que una aplicación almacene $e(x)$ e implemente la función f' con objeto de calcular $f'(e(x))$, sin que esto revele a un posible atacante f, x o $f(x)$. Desafortunadamente, el esquema propuesto se encuentra limitado al cálculo de polinomios.

Los estudios teóricos llevados a cabo para la formalización del problema de la protección de software (como el de O. Goldreich titulado "*Towards a theory of software protection*" publicado en Proceedings of the 19th Annual ACM Symposium on Theory of Computing, pp. 182-194. 1987.) han demostrado que es necesario contar con algún tipo de elemento hardware que complemente el diseño software del esquema de protección.

Entre las soluciones que se apoyan en algún tipo de componente hardware, uno de los esquemas más populares consiste en usar dispositivos difíciles de duplicar, los cuales se conectan al ordenador mediante algún puerto de comunicaciones. El software protegido comprueba la presencia del dispositivo y no funciona si tal comprobación falla. Ejemplos de este tipo de sistemas son las "llaves hardware" (*dongle*) y las tarjetas inteligentes (*smart cards*). Un inconveniente de

este tipo de sistemas es la incompatibilidad entre dispositivos pertenecientes a distintas aplicaciones. En el caso de que los dispositivos usados sean tarjetas inteligentes, disponer de un único lector ocasiona que el usuario haya de estar continuamente cambiando la tarjeta, problema que se conoce como *card juggling*, lo cual supone un inconveniente de primer orden.

La comprobación de presencia se realiza de diferentes modos. Por ejemplo, puede consistir en leer un simple valor del puerto de comunicación, o más comúnmente y con objeto de evitar que la intercepción de la comunicación en el puerto permita la duplicación del dispositivo, en enviar un dato (llamado *desafío*) que el dispositivo procesa y cuya respuesta puede ser prevista por el software. En cualquier caso, sea cual sea el tipo de comprobación que se realiza, no resulta complicado evitar este tipo de protección ya que el acceso al puerto de comunicación o al lector se identifican fácilmente en el código ejecutable de la aplicación protegida. Así, puede "trucarse" la comprobación y obtenerse un software totalmente funcional a partir del software protegido. Una vez identificado el chequeo, no es inusual que se produzcan programas (conocidos como "parches") que automaticen el proceso de modificación (o desprotección).

En algunos casos el software a ejecutar se encuentra cifrado y es el dispositivo el que, en tiempo de ejecución, lo descifra para que pueda ser ejecutado. El problema en este caso es que el software, una vez descifrado, pasa a la memoria RAM del ordenador del usuario (y potencial usuario deshonesto) de donde puede ser recuperado con técnicas conocidas (p. ej. provocando un *core dump*).

Una de las primeras propuestas para usar tarjetas inteligentes en la protección de software se presenta en el trabajo de I. Schaumüller-Bichl y E. Piller titulado "*A Method of Software Protection Based on the Use of Smart Caras and Cryptographic Techniques*" publicado en Proceedings of Eurocrypt'84. Springer-Verlag. LNCS 0209, pp. 446-454. 1984. Protective Technologies (Noruega) ofrece una herramienta comercial que se basa en tales ideas para un esquema de protección de software que se basa en tarjetas inteligentes (sin uso de criptografía). Su esquema consiste en sustituir manualmente y como paso previo a la compilación del software ciertas secciones de código de la aplicación a proteger por otras equivalentes para ser procesadas en la tarjeta. De este modo, el software funcionaría únicamente cuando se utilizara la tarjeta, pues estaría distribuido entre ésta y el ordenador. Con un esquema como este, el hecho de necesitar una tarjeta por cada aplicación software a proteger, junto a la dependencia de la capacidad de la tarjeta respecto a la cantidad y complejidad de las funciones protegidas, resultan problemas considerables. Además, las tarjetas deben ser distribuidas junto con el software, lo cual imposibilita la comercialización a través de Internet.

Más recientemente, Tuomas Aura y Dieter Gollman presentan en su trabajo titulado "*Software License Management with Smart Cards*" (publicado en Proceedings of the Usenix Workshop on Smartcard Technology-Smartcard'99-, pp. 75-86. 1999.) un interesante esquema basado en tarjetas inteligentes y certificados digitales que aporta importantes ventajas como el hecho de solucionar los problemas del *card juggling* y de la transferencia de licencias entre usuarios. Además, recopilan y proponen una serie de medidas para dificultar la labor de los usuarios deshonestos. Desafortunadamente, dado que su sistema se basa en el chequeo de la presencia del dispositivo (tarjeta inteligente en este caso), está sujeto a los ataques de modificación de código mencionados anteriormente.

De los estudios presentados se concluye que para lograr un esquema de protección de software cuya seguridad sea demostrable, debe disponerse de un procesador resistente a ataques físicos (tamperproof) que contenga y ejecute el código a proteger como proponen A. Herzberg y S. Pinter en el artículo titulado "*Public Protection of Software*" (publicado en ACM Transactions on Computer Systems, 5(4)-87, pp. 371-393. 1987) y John Phipps en su trabajo titulado "*Physical protection devices*" (publicado en The protection of Computer Software - its technology and applications, British Computer Society -BCS- Monographs in Informatics, chapter 3. Cambridge University Press, 2nd edition, 1992). Una variante de este esquema consiste en distribuir el software cifrado y hacer que el procesador resistente a ataques físicos se encargue de descifrarlo y ejecutarlo como proponen Oded Goldreich y Rafail Ostrovsky en su artículo titulado "*Software Protection and Simulation on Oblivious RAM*" publicado en Journal of the ACM, Vol. 43(3), pp. 431-473. 1996.

Breve descripción de la invención

La presente invención se refiere a un sistema para la protección contra el uso ilegítimo y gestión de licencias de software basado en dispositivos procesadores autónomos (como por ejemplo tarjetas inteligentes) y criptografía. Al contrario que otros sistemas que se basan en la supuesta dificultad de analizar y modificar el software para evitar la comprobación de la presencia del dispositivo, dificultad poco realista porque el análisis de código ejecutable siempre es posible, la modificación del código para evitar la comprobación de la presencia del dispositivo no permite romper este esquema, ya que el dispositivo es necesario para la ejecución del software porque colabora en tal ejecución.

La invención descrita permite proteger múltiples aplicaciones software con un único dispositivo, aumenta la complejidad de las funciones que pueden protegerse respecto a desarrollos previos, elimina la limitación en el número de funciones a proteger, hace posible la distribución de software a través de redes de comunicación abiertas como Internet, es más resistente a ataques lógicos y permite la definición de diferentes tipos de licencias de uso del software protegido y la transferencia selectiva de las mismas.

Explicación de las figuras

La figura 1 presenta el esquema de la producción del software protegido y de la licencia de uso. Se puede observar que algunas secciones del código original se sustituyen por sus equiva-

lentes en código del dispositivo durante la fase de producción (1). La fase de producción incluye también el cifrado mediante un criptosistema simétrico de las secciones de código a proteger (2).

Tras la transformación, las secciones de código a proteger son sustituidas por llamadas a una función que transfiere la sección correspondiente de código cifrado (p. ej. B") al dispositivo para su ejecución (3).

Al producirse la compra se genera una licencia (4) que contiene la clave simétrica con la que se cifraron las secciones de código protegidas, información acerca de las condiciones de uso (p. ej. límites temporales, número de ejecuciones, etc.), la identificación del software (identificador, versión, etc.) y por último, la identificación de la licencia. Esta licencia es cifrada con la clave pública del dispositivo. Cuando el cliente recibe la licencia, ésta queda almacenada en el propio dispositivo (5).

En tiempo de ejecución, una vez que el dispositivo recibe cada sección de código y datos desde la aplicación protegida, procede a descifrar el código, a ejecutarlo sobre los datos y a devolver los resultados.

Descripción del nuevo sistema de protección de software

El nuevo esquema que presentamos en esta sección se basa en la idea ya expuesta del uso de un dispositivo procesador resistente a ataques físicos para la ejecución de parte del código de la aplicación software que se protege. La popularización de las tarjetas inteligentes y su evolución en las capacidades de proceso y almacenamiento nos han hecho considerar ésta como la forma de realización preferida de este sistema. No obstante el diseño no se limita al uso de tarjetas inteligentes, y en consecuencia, la solución puede ser aplicada usando cualquier dispositivo hardware de características similares a las tarjetas inteligentes (algunos tipos de llaves hardware, tarjetas coprocesadoras como por ejemplo el modelo "IBM 4758 PCI Cryptographic Coprocessor", o algunos dispositivos que integran la funcionalidad de una tarjeta inteligente y un lector). Por tanto, en el resto de la descripción, cuando se utilicen los términos "tarjeta" y "tarjeta inteligente" nos referiremos tanto a estas como a cualquier otro dispositivo con funcionalidades similares.

Esquemas de protección de software que se basen únicamente en la ejecución de parte del código en dispositivos de protección consiguen frustrar el ataque de modificación de código descrito anteriormente. De hecho, el único ataque posible habría de consistir en el análisis tanto de los datos enviados a la tarjeta como de las respuestas de la misma con el objeto de deducir las funciones realizadas en su interior. Aún así, y como se verá con mayor detalle más adelante, mediante un número adecuado de funciones con una relevancia y complejidad suficientemente altas tal ataque podría frustrarse. Los problemas principales en este caso son la limitada capacidad de almacenamiento de código y datos de la tarjeta que hace que el número y complejidad de las funciones protegidas sea muy reducido.

Con objeto de evitar estos problemas, y tal y como se muestra en el siguiente apartado, introducimos en nuestro esquema la utilización de la criptografía como segundo elemento básico a utilizar.

Consecución del sistema

Para el funcionamiento del sistema la tarjeta debe contener un par de claves (pública y privada) correspondientes a un criptosistema asimétrico. El par de claves debe generarse en la tarjeta y la clave privada nunca debe abandonar la misma. Para garantizar la autenticidad de las claves, o mejor dicho, para garantizar que el par de claves ha sido generado en la tarjeta y por consiguiente la clave privada no se conoce fuera de la misma, la tarjeta también contiene un certificado de la clave pública de la tarjeta firmado por el fabricante o emisor de la tarjeta.

El funcionamiento del sistema es el siguiente: Algunas secciones del código original se sustituyen por sus equivalentes en código ejecutable en la tarjeta (1) durante la fase de producción. En ese mismo momento, dichas secciones son cifradas mediante un criptosistema simétrico (2). Tras la transformación, las secciones de código y datos protegidas son sustituidas en el código de la aplicación por llamadas a una función que las transfiere a la tarjeta para su descifrado y ejecución (3).

En el momento de la adquisición del software se genera una licencia que contiene la clave simétrica con la que se cifraron las secciones de código protegidas, información acerca de las condiciones de uso (p. ej. límites temporales, número de ejecuciones, etc.), la identificación del software (identificador, versión, etc.) y, por último, la identificación de la licencia. Esta licencia es cifrada con un criptosistema asimétrico usando la clave pública de la tarjeta (4), de forma que sólo la tarjeta con la correspondiente clave privada puede descifrar y ejecutar el código. Cuando el cliente recibe la licencia, ésta queda almacenada en la propia tarjeta (5). Dado que la licencia se personaliza para esa tarjeta, se elimina la necesidad de que el software tenga que residir en la misma.

En tiempo de ejecución, una vez que la tarjeta recibe el código y los datos desde la aplicación protegida, procede a descifrarlos, ejecuta el código sobre los datos y devuelve los resultados.

Un ataque al sistema necesitaría, al igual que en el caso anterior, del conocimiento de las funciones ejecutadas en la tarjeta, o lo que es lo mismo, del análisis de los datos de entrada y salida y el tiempo empleado en la ejecución interna de cada función. Hay que hacer notar que, en este caso y dado que la tarjeta sólo contiene una función en cada momento, podemos aumentar la complejidad de cada función, utilizando toda la capacidad de almacenamiento de código y datos de la tarjeta. Más aún, se posibilita que la tarjeta ejecute de forma protegida tantas secciones como sea necesario. De este modo, el usuario deshonesto habría de invertir un esfuerzo mucho mayor para poder descubrir la función de cada una de ellas.

El agrupamiento de información en una "licencia" permite que la autorización pueda ajustarse a las necesidades concretas de cada caso, haciendo

al sistema más flexible. Además, y debido a que con este sistema cada aplicación tiene su propia clave, se habilitan nuevas características, como la posibilidad de transferir o eliminar licencias de forma selectiva.

Si la transmisión de la licencia se realiza a través de un canal de comunicación no confiable sería posible, en teoría, realizar un ataque desde el intermedio (También conocido como "Man-in-the-middle" en la literatura relacionada con los sistemas de cifrado "de clave pública" o "asimétricos" -p. ej. RSA o ECC-.). Sin embargo, y como se mostrará en el apartado siguiente, el productor del software exigirá para la venta el certificado de la clave pública de la tarjeta firmado por el fabricante de la misma, que el usuario deshonesto que pretenda realizar este tipo de ataques no podrá falsificar.

En resumen, este sistema aporta la ventaja de que con una única tarjeta pueden protegerse múltiples aplicaciones, aumenta la complejidad de las funciones que pueden protegerse, elimina la limitación en el número de funciones a proteger, hace posible la distribución de software a través de redes de comunicación como Internet, es más resistente a ataques lógicos y permite la definición de diferentes tipos de licencias y su transferencia selectiva.

Gestión de licencias

La gestión de licencias es parte inseparable del sistema ya que la dependencia entre el sistema de protección en sí y la gestión de las licencias es mutua. A continuación se detallan cada uno de los procesos relacionados con la gestión de las licencias.

Venta

Dado que la licencia para el cliente (que contiene la clave para descifrar las secciones protegidas de software) se encuentra cifrada con la clave pública de la tarjeta, se hace necesario evitar que la correspondiente clave privada pueda ser conocida fuera de la tarjeta, ya que, en tal caso, el código protegido podría ser descifrado. A este efecto la solución más práctica consiste en utilizar tarjetas específicamente preparadas para este cometido (bien por el fabricante o bien por el emisor de las tarjetas). Tales tarjetas contendrán el software de base necesario para cumplir con la función asignada en nuestro esquema, que es: manejar licencias, recibir secciones de código y datos cifrados, descifrarlos, ejecutar el código sobre los datos y devolver los resultados.

Un escenario típico para la compra de una aplicación protegida puede ser este: el cliente envía al productor de la aplicación el certificado de la clave pública de su tarjeta. El productor verifica la validez del certificado recibido y, en caso de que la verificación tenga éxito, genera la licencia, la cifra con la clave pública recibida y la envía al cliente. El productor, además, registra en una base de datos de licencias las claves a las que se autoriza el uso del producto para, en caso necesario (pérdida, destrucción, etc.), poder generar una nueva la licencia para el cliente. Por tanto, en caso de recibir una petición de licencia de la clave que ya tiene registrada podría emitir una nueva licencia que sustituirá a la anterior sin cargo para el cliente. Cada vez que se emite una licencia a partir de una existente, el productor cambia el número de emisión que es parte de la identificación de la licencia.

El software en sí puede ser distribuido libremente sin protección adicional, ya que sólo podrá ser utilizado con una licencia válida.

Transferencia

La transferencia de licencias puede ser necesaria para ceder el derecho de uso del producto software a otro usuario o simplemente para trasladar las licencias a una nueva tarjeta del mismo usuario. En este sistema se introduce la posibilidad de realizar transferencias selectivas de licencias.

Para transferir una licencia se sigue un protocolo que se divide en dos fases: *delegación* (pasos 1 a 3) y *recuperación* (pasos 4 al 6). Llamamos a este protocolo *transferencia directa* para distinguirlo del protocolo de *transferencia prevista* que se usa principalmente para prevención y recuperación de fallos en la tarjeta. El proceso es el siguiente:

1. El usuario selecciona qué licencia o licencias de entre las disponibles en la tarjeta de origen va a transferir. Nótese que, contrariamente a lo que sucede en otros sistemas, no es necesario transferir todas las licencias de la tarjeta, lo cual supondría una seria limitación. En lo sucesivo y para simplificar la explicación supondremos que vamos a transferir una licencia concreta.

2. La tarjeta destino envía el certificado de su clave pública a la tarjeta origen.

3. La tarjeta origen crea un certificado de delegación de la licencia a la clave de la tarjeta destino, destruye su propia licencia y por último, envía el certificado de delegación a la tarjeta destino.

4. La tarjeta destino solicita al productor del software una nueva licencia presentando el certificado de su clave pública y el certificado de delegación de la tarjeta origen.

5. El vendedor comprueba ambos certificados y, si son correctos, crea una nueva licencia para la tarjeta destino. La información en la base de licencias se actualiza consecuentemente.

6. La tarjeta destino descifra y almacena la nueva licencia.

Para evitar ataques de duplicación de licencias, se introduce en la licencia un número de emisión que es diferente para cada nueva copia de la licencia emitida por el productor del software.

Recuperación

En casos como el del presente sistema, que incluyen el uso de algún tipo de componente hardware, es necesario prever las consecuencias fallos en tales componentes. Debemos recordar que las licencias se ligan a una tarjeta en base a que su clave privada no se conoce fuera de la misma, por lo que, en caso de que la tarjeta falle, será imposible utilizar el software. Para tal eventualidad es necesario tomar medidas de precaución.

Dado que el precio de las tarjetas es pequeño en comparación al coste del software, es razonable preparar una tarjeta de repuesto para el caso de que la tarjeta principal falle. El proceso preventivo consiste simplemente en ejecutar la fase de *delegación* del protocolo de *transferencia prevista* parcialmente para cada licencia. En caso de que la tarjeta principal falle el protocolo continuará con la fase de *recuperación* con lo que, al finalizar el mismo, las licencias habrán sido transferidas a la nueva tarjeta.

Las diferencias entre el protocolo de transferencia directa y el de transferencia prevista consiste en la inclusión de un parámetro (posiblemente fecha, aunque también puede ser número de ejecuciones, etc.) que indica cuando tendrá lugar la transferencia. Este parámetro se incluye en el certificado de delegación. Los pasos 3 y 4 del protocolo de transferencia directa se sustituyen por los siguientes en el protocolo de transferencia prevista.

3' La tarjeta origen crea un certificado de delegación de la licencia a la clave de la tarjeta destino en la fecha F y envía el certificado de delegación a la tarjeta destino. La tarjeta origen no creará más certificados de delegación de tal licencia hasta pasada la fecha F.

4' Posteriormente pueden darse dos situaciones:

4.1 Si el usuario decide continuar usando la tarjeta principal puede hacer que la tarjeta de repuesto envíe antes de la fecha F una nueva petición de *delegación prevista* para una fecha F' posterior a F. En este caso la tarjeta principal si aceptará tal petición y enviará un nuevo certificado de delegación para la fecha F' que sustituirá al anterior en la tarjeta de repuesto.

4.2 En otro caso, cuando se llegue a la fecha F:

4.2.1 La tarjeta principal destruirá la licencia correspondiente al certificado de delegación prevista que se cumple en la fecha F.

4.2.2 Como en el caso de la transferencia directa ambas tarjetas podrán solicitar al productor del software una nueva licencia, pero sólo la primera petición tendrá éxito. El usuario puede decidir cuál de las tarjetas desea usar.

Caducidad

Puesto que las licencias se mantienen en todo momento protegidas, bien mediante cifrado o bien porque residen en la tarjeta, es posible que la tarjeta (cuyo software es confiable) elimine una licencia cuando corresponda (según diferentes criterios como número de usos, tiempo de uso, etc.) e incluso que avise poco antes de que esto suceda. Uno de los parámetros que más se usan en la expedición de licencias de software es la fecha de caducidad. Para este fin, se usarán tarjetas que dispongan de un reloj autónomo interno (cuya comercialización está prevista en breve).

Detalles de realización

Se describen a continuación algunos detalles que se usan en la realización para garantizar la seguridad del esquema.

Funciones a ejecutar por la tarjeta

Para complicar la labor de análisis del pirata se utilizan técnicas como la inclusión de datos falsos (*dummy*) que no se utilizan en la computación real pero que son transformados para dificultar la identificación de los datos auténticos. En la parte externa de la aplicación (la parte no protegida que se ejecuta normalmente sobre el ordenador del cliente) esos datos se usan con otros datos reales en cálculos no útiles a la función del software.

Una técnica usada que complica fuertemente la identificación de las funciones, consiste en intercalar los cálculos de varias funciones simples, de forma que el resultado de una llamada a la tarjeta no depende sólo de los datos recibidos en esa llamada sino también de otros de llamadas anteriores, o incluso de resultados calculados en llamadas previas que no se han emitido como resultado desde la tarjeta por lo que el atacante no puede conocer.

Lectores

Probablemente uno de los ataques más comunes se produzca dentro de la propia organización del cliente legítimo del software. Este ataque suele consistir en que un cliente utiliza múltiples copias de un programa que ha adquirido legalmente. En un esquema como el nuestro, este ataque se materializaría en que varios ordenadores compartiesen el acceso al dispositivo de protección que contiene la licencia (p. ej. a un lector donde se introduce la tarjeta).

En esquemas anteriores, la solución propuesta a este problema consiste en controlar el acceso al lector o al dispositivo directamente (a bajo nivel) desde el software protegido. Esta solución introduce innumerables problemas y costes computacionales en el software protegido ya que debe adaptarse a todas las posibles configuraciones y lectores posibles, función que recaería normalmente en el sistema operativo.

En nuestra propuesta, contra este ataque, se ha diseñado un sistema basado en la técnica descrita al final del apartado anterior. Este sistema consiste en que el resultado de cada llamada a una función de la tarjeta depende de las llamadas anteriores, por lo que una secuenciación incorrecta de las llamadas (que se produciría en el caso de que varios ordenadores intentasen compartir un único lector o dispositivo) dará como resultado la ejecución incorrecta del software protegido.

Otras aplicaciones

La utilidad del sistema descrito no se limita a la protección contra el uso no autorizado del software sino que va más allá, proporcionando importantes ventajas en otros ámbitos de aplicación.

Aplicación a la protección de software en entornos de computación móvil

Los sistemas basados en software móvil (como agentes o applets) son cada vez más populares. El sistema descrito en la presente invención puede

aplicarse a este tipo de software, de modo que se genera una licencia (probablemente para un solo uso) en el momento de descargar el software, lo cual permite controlar su ejecución, evitando que múltiples clientes ejecuten el mismo software móvil o que un cliente ejecute el software móvil múltiples veces.

Aplicación a la protección de acceso a datos distribuidos

Ejemplos de tales aplicaciones son las bases de datos distribuidas y las extranets.

En este caso cuando un cliente desea acceder a cierta información, envía al proveedor de la información el certificado de su tarjeta, un certificado que incluye la expresión de las autorizaciones de acceso y la petición concreta. El proveedor produce en ese momento un elemento de software móvil (p. ej. applet o agente) específico para satisfacer la petición. Este software, que será ejecutado en el ordenador del cliente, contiene código que debe ser ejecutado parcialmente por la tarjeta, junto con una licencia para que la tarjeta del cliente pueda ejecutarlo. Una vez el software es recibido por el ordenador del cliente, se ejecuta usando la tarjeta y dando acceso a los datos que contiene.

Aplicación del sistema a la comercialización de información sobre redes abiertas

Ejemplos de este tipo de aplicaciones pueden ser los periódicos en línea con pago por acceso o las librerías digitales. En este esquema cada cliente debe disponer de una tarjeta con su correspondiente par de claves y su certificado Cuando un cliente desea acceder a cierta información, envía al vendedor de la información el certificado de su tarjeta y, la petición concreta. Este paso puede necesitar un proceso de negociación de condiciones.

El vendedor, produce en ese momento un elemento de software móvil específico para satisfacer la petición según las condiciones pactadas. Este software, que será ejecutado en el ordenador del cliente, contiene código que debe ser ejecutado parcialmente por la tarjeta, junto con una licencia para que la tarjeta del cliente pueda ejecutarlo. Una vez el software es recibido por el ordenador del cliente, se ejecuta usando la tarjeta para dar acceso a la información que contiene según los términos establecidos previamente, llevar a cabo su cometido.

Puesto que el software de la tarjeta es confiable y está certificado por el fabricante de la misma, es posible controlar aspectos como que la licencia autorice a una única ejecución, o incorporar a la tarjeta un monedero electrónico (electronic purse) al cual se carga el coste de la información accedida.

REIVINDICACIONES

1. Un sistema de protección de software basado en el uso combinado de la criptografía y de tarjetas inteligentes u otros dispositivos funcionalmente equivalentes **caracterizado**, esencialmente, porque las tarjetas o dispositivos funcionalmente equivalentes reciben licencias que les permiten descifrar y ejecutar secciones de código y datos de la aplicación que se protege, las cuales se distribuyen cifradas y no tienen que residir en el dispositivo, de forma que con una única tarjeta pueden protegerse múltiples aplicaciones; porque el código y datos protegidos no residen en la tarjeta más que en el momento de su ejecución, y porque permite la definición de diferentes tipos de licencias y su transferencia selectiva.

2. Un sistema como el reivindicado en la reivindicación 1, **caracterizado** porque algunas secciones del código original se sustituyen por sus equivalentes en código ejecutable en la tarjeta u otro dispositivo funcionalmente equivalente durante la fase de producción siendo cifradas mediante un criptosistema simétrico.

3. Un sistema como el reivindicado en la reivindicación 1, **caracterizado** porque en el momento de la adquisición del software se genera una licencia cifrada con un criptosistema asimétrico usando la clave pública de la tarjeta u otro dispositivo funcionalmente equivalente, que contiene la clave simétrica con la que se cifraron las secciones de código protegidas, la información acerca de las condiciones de uso, la identificación del software y, por último, la identificación de la licencia.

4. Un sistema como el reivindicado en la reivindicación 1, **caracterizado** porque toda la capacidad de las tarjetas u otros dispositivos funcionalmente equivalentes están disponibles para cada sección protegida de la aplicación.

5. Un sistema como el reivindicado en la reivindicación 1, **caracterizado** porque el número de secciones protegidas de una aplicación no está limitado.

6. Un sistema como el reivindicado en la reivindicación 1, **caracterizado** porque es posible la distribución del software protegido a través de redes de comunicación abiertas como Internet.

7. Un sistema de protección de software en entornos de computación móvil, **caracterizado** porque el software móvil (agentes o applets) está protegido a través del sistema reivindicado en las reivindicaciones 1 a 6.

8. Un sistema de protección de acceso a datos distribuidos, **caracterizado** porque los datos se transmiten mediante software móvil (agentes o applets) protegido a través del sistema reivindicado en la reivindicación 7.

9. Un sistema de comercialización de información sobre redes abiertas, **caracterizado** porque los datos se transmiten mediante software móvil (agentes o applets) protegido a través del sistema reivindicado en la reivindicación 7 y el pago se realiza en base a un monedero electrónico incluido en la tarjeta u otro dispositivo funcionalmente equivalente.

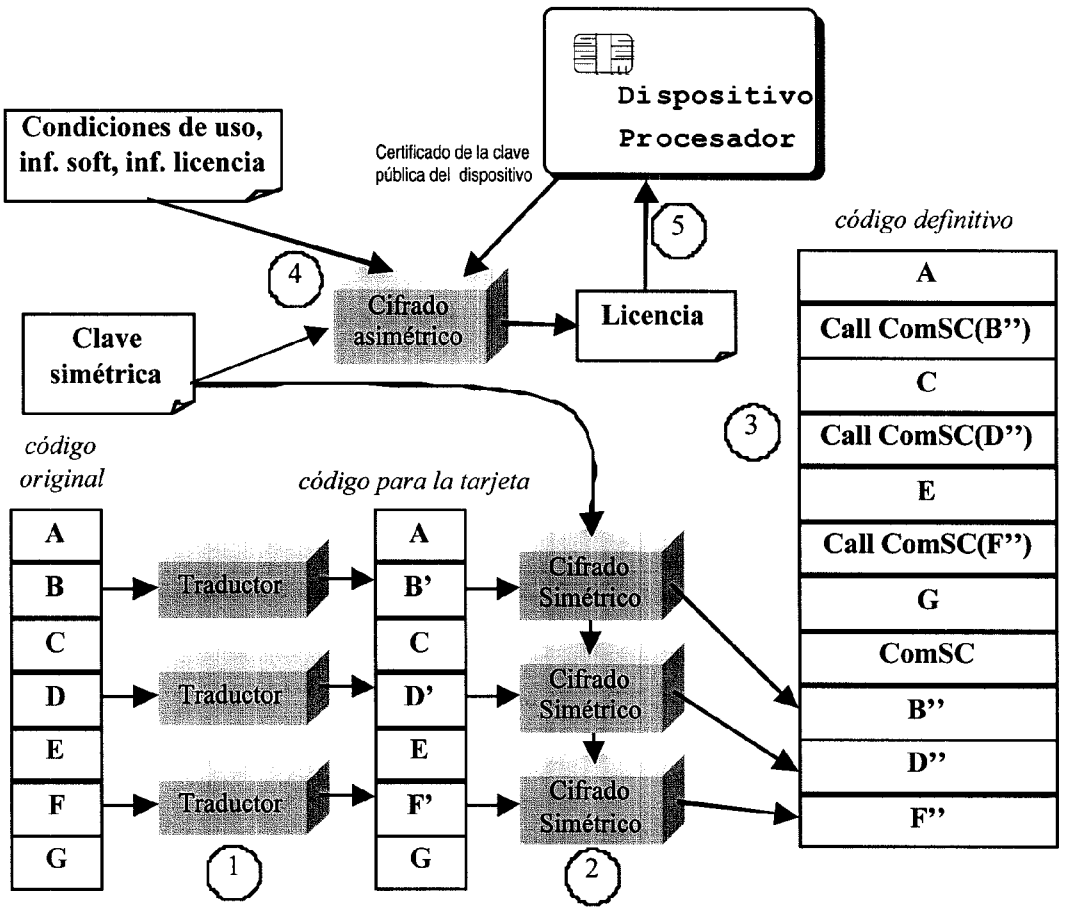

Fig. 1. Transformación del código y producción de la licencia

OFICINA ESPAÑOLA
DE PATENTES Y MARCAS

ESPAÑA

(11) ES 2 179 775

(21) N.° solicitud: 200101022

(22) Fecha de presentación de la solicitud: **04.05.2001**

(32) Fecha de prioridad:

INFORME SOBRE EL ESTADO DE LA TECNICA

(51) **Int. Cl.**[7]**:** G06F 1/00

DOCUMENTOS RELEVANTES

Categoría	Documentos citados	Reivindicaciones afectadas
X	Un esquema eficiente de protección de software basado en tarjetas inteligentes. Informe Técnico de Investigación LCC2000-9 (2000). ANTONIO MAÑA y otros. Recuperado de Internet: <url:www.lcc.uma.es>	1-9
A	WO 8502310 A (SOFTNET) 23.05.1985, páginas 5-12.	1-6
A	US 5666411 A (JOHNNIE C. MCCARTY) 09.09.1997	

Categoría de los documentos citados

X: de particular relevancia

Y: de particular relevancia combinado con otro/s de la misma categoría

A: refleja el estado de la técnica

O: referido a divulgación no escrita

P: publicado entre la fecha de prioridad y la de presentación de la solicitud

E: documento anterior, pero publicado después de la fecha de presentación de la solicitud

El presente informe ha sido realizado

[×] para todas las reivindicaciones

[ ] para las reivindicaciones n°:

Fecha de realización del informe	Examinador	Página
11.12.2002	A. Astudillo Lizaga	1/1