(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 529 426**

(51) Int. CI.:

G06F 21/12 (2013.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Fecha de presentación y número de la solicitud europea: **04.07.2002 E 02762528 (4)**

(97) Fecha y número de publicación de la concesión europea: **19.11.2014 EP 1412838**

(54) Título: **Procedimiento para proteger un software mediante "detección y coerción" contra su uso no autorizado**

(30) Prioridad:

31.07.2001 FR 0110244

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
20.02.2015

(73) Titular/es:

VALIDY (100.0%)
ZONE INDUSTRIELLE, 5, RUE JEAN CHARCOT
26100 ROMANS SUR ISÈRE, FR

(72) Inventor/es:

CUENOD, JEAN-CHRISTOPHE y
SGRO, GILLES

(74) Agente/Representante:

CURELL AGUILÁ, Mireia

Aviso: En el plazo de nueve meses a contar desde la fecha de publicación en el Boletín europeo de patentes, de la mención de concesión de la patente europea, cualquier persona podrá oponerse ante la Oficina Europea de Patentes a la patente concedida. La oposición deberá formularse por escrito y estar motivada; sólo se considerará como formulada una vez que se haya realizado el pago de la tasa de oposición (art. 99.1 del Convenio sobre concesión de Patentes Europeas).

DESCRIPCIÓN

Procedimiento para proteger un software mediante “detección y coerción” contra su uso no autorizado.

La presente invención se refiere al campo técnico de los sistemas de procesado de datos en sentido general y se refiere, de manera más precisa, a los medios para proteger, contra su uso no autorizado, un software que funciona en dichos sistemas de procesado de datos.

El objeto de la invención se refiere, más particularmente, a los medios para proteger un software contra su uso no autorizado, a partir de una unidad de procesado y de memorización, materializándose comúnmente dicha unidad por medio de una tarjeta chip o una llave tangible en un puerto USB.

En el campo técnico mencionado anteriormente, el principal inconveniente se refiere a la utilización no autorizada de softwares por parte de usuarios que no han pagado derechos de licencia. Este uso ilícito de softwares genera un perjuicio manifiesto para los editores de software, los distribuidores de software y/o toda persona que integra dicho software en productos. Para evitar dichas copias ilícitas, se han propuesto, en el estado de la técnica, diversas soluciones para proteger softwares.

De este modo, es conocida una solución de protección que consiste en poner en práctica un sistema material de protección, tal como un elemento físico denominado llave de protección o "dongle" en terminología anglosajona. Dicha llave de protección debería garantizar la ejecución del software únicamente en presencia de la llave. Ahora bien, se debe hacer constar que dicha solución es ineficaz puesto que presenta el inconveniente de ser fácilmente eludible. Con la ayuda de herramientas especializadas, tales como desensambladores, una persona mal intencionada o pirata puede suprimir las instrucciones de control de la llave de protección. Resulta entonces posible realizar copias ilícitas que se corresponden con versiones modificadas de los softwares que ya no tienen ninguna protección. Además, esta solución no se puede generalizar a todos los softwares, en la medida en la que resulta difícil conectar más de dos llaves de protección a un mismo sistema.

Es conocido también por medio del documento WO 99/66387, un procedimiento de protección de un programa de ordenador, que consiste en dividir el programa en por lo menos dos partes, pública y secreta respectivamente, y en ejecutar en la parte secreta, después de la recepción de parámetros, por lo menos una parte del programa, y a continuación en retransmitir los resultados a la parte pública.

La patente US nº 5 754 646 describe un procedimiento de protección de un software, que consiste en dividirlo en dos partes de las cuales por lo menos una parte se ejecuta en una llave física. La parte que se ejecuta en la llave física se descarga a partir de una red. La conexión a la red se realiza durante el arranque del software protegido.

La solicitud de patente FR 2 634 917 describe un procedimiento de protección de un software, que consiste en grabar una parte esencial del software en una memoria conectada a un microprocesador y en confinar el conjunto de memoria-microprocesador dentro de una caja de tipo inviolable que comprende medios de conexión entre dicho microprocesador y un microprocesador exterior de un sistema de utilización del software.

El objetivo de la invención pretende precisamente remediar los inconvenientes mencionados anteriormente al proponer un procedimiento para proteger un software contra su uso no autorizado, a partir de una unidad de procesado y de memorización ad hoc, en la medida en la que la presencia de dicha unidad es necesaria para que el software sea completamente funcional.

Para lograr dicho objetivo, el objeto de la invención se refiere a un procedimiento para proteger, a partir de por lo menos una unidad virgen que consta por lo menos de medios de procesado y medios de memorización, un software vulnerable contra su utilización no autorizada, funcionando dicho software vulnerable en un sistema de procesado de datos. El procedimiento según la invención consiste en:

→ en una fase de protección:

- definir:
  - por lo menos una característica de ejecución de software, susceptible de ser supervisada por lo menos en parte en una unidad,
  - por lo menos un criterio a respetar para por lo menos una característica de ejecución de software,
  - medios de detección a implementar en una unidad y que permiten detectar que por lo menos una característica de ejecución de software no respeta por lo menos un criterio asociado,
  - y medios de coerción a implementar en una unidad y que permiten informar al sistema de procesado de datos y/o modificar la ejecución de un software cuando no se respeta por lo menos un criterio,

- construir medios operativos que permiten transformar la unidad virgen en una unidad con capacidad de poner en práctica los medios de detección y los medios de coerción,

- crear un software protegido:

  - seleccionando por lo menos una característica de ejecución de software a supervisar, entre las características de ejecución susceptibles de ser supervisadas,

  - seleccionando por lo menos un criterio a respetar para por lo menos una característica de ejecución de software seleccionada,

  - seleccionando por lo menos un procesado algorítmico que, durante la ejecución del software vulnerable, utiliza por lo menos un operando y permite obtener por lo menos un resultado, y para el cual se supervisará por lo menos una característica de ejecución de software seleccionada,

  - seleccionando por lo menos una porción de la fuente del software vulnerable que contiene por lo menos un procesado algorítmico seleccionado,

  - generando la fuente del software protegido a partir de la fuente del software vulnerable, modificando por lo menos una porción seleccionada de la fuente del software vulnerable para obtener por lo menos una porción modificada de la fuente del software protegido, siendo dicha modificación tal que:

    - durante la ejecución del software protegido se ejecuta una primera parte de ejecución en el sistema de procesado de datos y se ejecuta una segunda parte de ejecución en una unidad, obtenida a partir de la unidad virgen después de la carga de informaciones,

    - la segunda parte de ejecución ejecuta por lo menos la funcionalidad de por lo menos un procesado algorítmico seleccionado,

    - y durante la ejecución del software protegido, se supervisa por medio de la segunda parte de ejecución por lo menos una característica de ejecución seleccionada y el hecho de no respetar un criterio conduce a una modificación de la ejecución del software protegido,

  - y generando:

    - una primera parte objeto del software protegido, a partir de la fuente del software protegido, siendo esta primera parte objeto tal que durante la ejecución del software protegido, aparece una primera parte de ejecución que se ejecuta en el sistema de procesado de datos y, de la cual, por lo menos una porción tiene en cuenta que se supervisa por lo menos una característica de ejecución de software seleccionada,

    - y una segunda parte objeto del software protegido, que contiene los medios operativos que ponen en práctica los medios de detección y los medios de coerción, siendo esta segunda parte objeto tal que, después de la carga en la unidad virgen y durante la ejecución del software protegido, aparece la segunda parte de ejecución por medio de la cual se supervisa por lo menos una característica de ejecución de software y por medio de la cual el hecho de no respetar un criterio conduce a una modificación de la ejecución del software protegido,

- y cargar la segunda parte objeto en la unidad virgen, con vistas a obtener la unidad,

→ y en una fase de utilización en el curso de la cual se ejecuta el software protegido:

- en presencia de la unidad:

  - en la medida en la que se respeten todos los criterios que se corresponden con todas las características de ejecución supervisadas de todas las porciones modificadas del software protegido, permitir el funcionamiento nominal de estas porciones del software protegido y en consecuencia permitir el funcionamiento nominal del software protegido,

  - y si por lo menos no se respeta uno de los criterios que se corresponde con una característica de ejecución supervisada de una porción del software protegido, informar al sistema de procesado de datos y/o modificar el funcionamiento de la porción del software protegido, de manera que se modifica el funcionamiento del software protegido;

- y en ausencia de la unidad, a pesar de la petición de una porción de la primera parte de ejecución, de iniciar la ejecución en la unidad, de la funcionalidad de un procesado algorítmico seleccionado, no poder responder correctamente a esa petición, de manera que por lo menos esta porción no se ejecuta correctamente y, en consecuencia, el software protegido no es completamente funcional.

Según una variante de realización, el procedimiento de acuerdo con la invención consiste en:

→ en la fase de protección:

- definir:

  - en calidad de característica de ejecución de software susceptible de ser supervisada, una variable de medición del uso de una funcionalidad de un software,

  - en calidad de criterio a respetar, por lo menos un umbral asociado a cada variable de medición,

  - y medios de actualización que permiten actualizar por lo menos una variable de medición,

- construir los medios operativos que permiten que la unidad ponga también en práctica los medios de actualización,

- y modificar el software protegido:

  - seleccionando en calidad de característica de ejecución de software a supervisar, por lo menos una variable de medición del uso de una funcionalidad de un software,

  - seleccionando:

    - por lo menos una funcionalidad del software protegido cuyo uso es susceptible de ser supervisado gracias a una variable de medición,

    - por lo menos una variable de medición que sirve para cuantificar el uso de dicha funcionalidad,

    - por lo menos un umbral asociado a una variable de medición seleccionado en correspondencia con un límite de uso de dicha funcionalidad,

    - y por lo menos un método de actualización de una variable de medición seleccionada en función del uso de dicha funcionalidad,

  - y modificando por lo menos una porción seleccionada de la fuente del software protegido, siendo esta modificación tal que, durante la ejecución del software protegido, la variable de medición se actualiza por medio de la segunda parte de ejecución, en función del uso de dicha funcionalidad y se tiene en cuenta por lo menos una superación de umbral,

→ y en la fase de utilización, en presencia de la unidad, y en el caso en el que se detecte por lo menos una superación de umbral correspondiente a por lo menos un límite de uso, informar al sistema de procesado de datos y/o modificar el funcionamiento de la porción del software protegido, de modo que el funcionamiento del software protegido se modifique.

Según una variante de realización, el procedimiento de acuerdo con la invención consiste en:

→ en la fase de protección:

- definir:

  - para por lo menos una variable de medición, varios umbrales asociados,

  - y medios de coerción diferentes que se corresponden con cada uno de estos umbrales,

- y modificar el software protegido:

  - seleccionando en la fuente del software protegido, por lo menos una variable de medición seleccionada a la cual deben estar asociados varios umbrales correspondientes a límites diferentes de

uso de la funcionalidad,

- seleccionando por lo menos dos umbrales asociados a la variable de medición seleccionada,
- y modificando por lo menos una porción seleccionada de la fuente del software protegido, siendo esta modificación tal que, durante la ejecución del software protegido, se tienen en cuenta las superaciones de los diversos umbrales, por medio de la segunda parte de ejecución, de manera diferente,

→ y en la fase de utilización:

- en presencia de la unidad:
  - en el caso en el que se detecte la superación de un primer umbral, ordenar al software protegido que ya no utilice la funcionalidad correspondiente,
  - y en el caso en el que se detecte la superación de un segundo umbral, convertir en inoperativa la funcionalidad correspondiente y/o por lo menos una porción del software protegido.

Según una variante de realización, el procedimiento de acuerdo con la invención consiste en:

→ en la fase de protección:

- definir medios de recarga que permiten acreditar por lo menos un uso suplementario para por lo menos una funcionalidad de software supervisada por una variable de medición,
- construir los medios operativos que permiten también a la unidad poner en práctica los medios de recarga,
- y modificar el software protegido:
  - seleccionando en la fuente del software protegido, por lo menos una variable de medición seleccionada que permite limitar el uso de una funcionalidad a la cual se debe poder acreditar por lo menos un uso suplementario,
  - y modificando por lo menos una porción seleccionada, siendo esta modificación tal que en una fase denominada de recarga, se puede acreditar por lo menos un uso suplementario de por lo menos una funcionalidad que se corresponde con una variable de medición seleccionada,

→ y en la fase de recarga:

- reactualizar por lo menos una variable de medición seleccionada y/o por lo menos un umbral asociado, de manera que se permita por lo menos un uso suplementario de la funcionalidad.

Según una variante de realización, el procedimiento de acuerdo con la invención consiste en:

→ en la fase de protección:

- definir:
  - en calidad de característica de ejecución de software susceptible de ser supervisada, un perfil de uso de software,
  - y en calidad de criterio a respetar, por lo menos un rasgo de ejecución de software,
- y modificar el software protegido:
  - seleccionando en calidad de característica de ejecución de software a supervisar por lo menos un perfil de uso de software,
  - seleccionando por lo menos un rasgo de ejecución que debe ser respetado por lo menos por un perfil de uso seleccionado,
  - y modificando por lo menos una porción seleccionada de la fuente del software protegido, siendo esta modificación tal que, durante la ejecución del software protegido, la segunda parte de ejecución respeta todos los rasgos de ejecución seleccionados,

→ y en la fase de utilización en presencia de la unidad, y en el caso en el que se detecte que no se respeta por lo menos un rasgo de ejecución, informar al sistema de procesado de datos y/o modificar el funcionamiento de la porción del software protegido, de modo que se modifique el funcionamiento del software protegido.

Según una variante de realización, el procedimiento de acuerdo con la invención consiste en:

→ en la fase de protección:

- definir:
  - un juego de instrucciones cuyas instrucciones son susceptibles de ser ejecutadas en la unidad,
  - un juego de órdenes de instrucciones para este juego de instrucciones, siendo susceptibles estas órdenes de instrucciones de ser ejecutadas en el sistema de procesado de datos y de iniciar en la unidad la ejecución de las instrucciones,
  - en calidad de perfil de uso, el encadenamiento de las instrucciones,
  - en calidad de rasgo de ejecución, un encadenamiento deseado para la ejecución de las instrucciones,
  - en calidad de medios de detección, medios que permiten detectar que el encadenamiento de las instrucciones no se corresponde con el deseado,
  - y en calidad de medios de coerción, medios que permiten informar al sistema de procesado de datos y/o modificar el funcionamiento de la porción de software protegido cuando el encadenamiento de las instrucciones no se corresponde con el deseado,
- construir los medios operativos que permiten también a la unidad ejecutar las instrucciones del juego de instrucciones, siendo iniciada la ejecución de estas instrucciones por la ejecución en el sistema de procesado de datos, de las órdenes de instrucciones,
- y modificar el software protegido:
  - modificando por lo menos una porción seleccionada de la fuente del software protegido, siendo esta modificación tal que:
    - por lo menos un procesado algorítmico seleccionado se descompone de manera que durante la ejecución del software protegido, este procesado algorítmico se ejecuta por medio de la segunda parte de ejecución, utilizando instrucciones,
    - para por lo menos un procesado algorítmico seleccionado, se integran órdenes de instrucciones en la fuente del software protegido, de manera que durante la ejecución del software protegido, cada orden de instrucción es ejecutada por la primera parte de ejecución e inicia en la unidad, la ejecución por medio de la segunda parte de ejecución, de una instrucción,
    - se selecciona una secuenciación de las órdenes de instrucciones entre el conjunto de las secuenciaciones que permiten la ejecución del software protegido,
    - y se especifica el encadenamiento que deben respetar por lo menos ciertas de las instrucciones durante su ejecución en la unidad,

→ y en la fase de utilización, en presencia de la unidad, en el caso en el que se detecte que el encadenamiento de las instrucciones ejecutadas en la unidad no se corresponde con el deseado, informar al sistema de procesado de datos y/o modificar el funcionamiento de la porción del software protegido, de manera que se modifique el funcionamiento del software protegido.

Según una variante de realización, el procedimiento de acuerdo con la invención consiste en:

→ en la fase de protección:

- definir:
  - en calidad de juego de instrucciones, un juego de instrucciones de las cuales por lo menos ciertas instrucciones trabajan sobre registros y utilizan por lo menos un operando con vistas a producir un

resultado,

- para por lo menos una parte de las instrucciones que trabajan sobre registros:
  - una parte que define la funcionalidad de la instrucción,
  - y una parte que define el encadenamiento deseado para la ejecución de las instrucciones y que consta de campos de bits que se corresponden con:
    - un campo de identificación de la instrucción,
    - y para cada operando de la instrucción:
      - un campo de bandera,
      - y un campo de identificación prevista del operando,
- para cada registro pertinente a los medios operativos y utilizado por el juego de instrucciones, un campo de identificación generada en el cual se memoriza automáticamente la identificación de la última instrucción que haya producido su resultado en este registro,
- en calidad de medios de detección, medios que permiten, durante la ejecución de una instrucción, para cada operando, cuando lo impone el campo de bandera, controlar la igualdad entre el campo de identificación generada que se corresponde con el registro utilizado por este operando, y el campo de identificación prevista del origen de este operando,
- y en calidad de medios de coerción, medios que permiten modificar el resultado de las Instrucciones, si por lo menos una de las igualdades controladas es falsa.

Según una forma preferida de realización, el procedimiento de acuerdo con la invención consiste en:

→ en la fase de protección:

- modificar el software protegido:
  - seleccionando por lo menos una variable utilizada en por lo menos un procesado algorítmico seleccionado, que durante la ejecución del software protegido, define parcialmente el estado del software protegido,
  - modificando por lo menos una porción seleccionada de la fuente del software protegido, siendo esta modificación tal que durante la ejecución del software protegido, por lo menos una variable seleccionada o por lo menos una copia de variable seleccionada reside en la unidad,
  - y generando:
    - la primera parte objeto del software protegido, siendo tal esta primera parte objeto que durante la ejecución del software protegido, por lo menos una porción de la primera parte de ejecución tiene también en cuenta que por lo menos una variable o por lo menos una copia de variable reside en la unidad,
    - y la segunda parte objeto del software protegido, siendo esta segunda parte objeto tal que, después de su carga en la unidad y durante la ejecución del software protegido, aparece la segunda parte de ejecución por medio de la cual por lo menos una variable seleccionada, o por lo menos una copia de variable seleccionada reside también en la unidad,

→ y en la fase de utilización:

- en presencia de la unidad cada vez que lo imponga una porción de la primera parte de ejecución, utilizar una variable o una copia de variable que resida en la unidad, de manera que esta porción se ejecuta correctamente y, en consecuencia, el software protegido es completamente funcional,
- y en ausencia de la unidad, a pesar de la petición de una porción de la primera parte de ejecución de utilizar una variable o una copia de variable que reside en la unidad, no poder responder correctamente a esta petición, de manera que por lo menos esta porción no se ejecuta correctamente y, en consecuencia,

el software protegido no es completamente funcional.

Según otra forma preferida de realización, el procedimiento de acuerdo con la invención consiste en:

→ en la fase de protección:

- definir:
  - en calidad de una orden activadora, una orden de instrucción,
  - en calidad de una función dependiente, una instrucción,
  - en calidad de una consigna, por lo menos un argumento para una orden activadora, correspondiente por lo menos en parte a la información transmitida por el sistema de procesado de datos a la unidad, con el fin de iniciar la ejecución de la función dependiente correspondiente,
  - un método de renombramiento de las consignas que permite renombrar las consignas con el fin de obtener órdenes activadoras de consignas renombradas,
  - y medios de restablecimiento destinados a ponerse en práctica en la unidad en el curso de la fase de utilización, y que permiten recuperar la función dependiente a ejecutar, a partir de la consigna renombrada,
- construir medios operativos que permiten que la unidad ponga también en práctica los medios de restablecimiento,
- y modificar el software protegido:
  - seleccionando en la fuente del software protegido, órdenes activadoras,
  - modificando por lo menos una porción seleccionada de la fuente del software protegido renombrando las consignas de las órdenes activadoras seleccionadas, con el fin de ocultar la identidad de las funciones dependientes correspondientes,
  - y generando:
    - la primera parte objeto del software protegido, siendo tal esta primera parte objeto que durante la ejecución del software protegido, se ejecutan las órdenes activadoras de consignas renombradas,
    - y la segunda parte objeto del software protegido que contiene los medios operativos poniendo también en práctica los medios de restablecimiento, siendo tal esta segunda parte objeto que, después de su carga en la unidad y durante la ejecución del software protegido, se restablece, por medio de la segunda parte de ejecución, la identidad de las funciones dependientes cuya ejecución es iniciada por la primera parte de ejecución, y las funciones dependientes son ejecutadas por medio de la segunda parte de ejecución,

→ y en la fase de utilización:

- en presencia de la unidad y cada vez que una orden activadora de consigna renombrada, contenida en una porción de la primera parte de ejecución lo imponga, restablecer en la unidad, la identidad de la función dependiente correspondiente y ejecutar la misma, de modo que esta porción se ejecuta correctamente y, en consecuencia, el software protegido es completamente funcional,
- y en ausencia de la unidad, a pesar de la petición de una porción de la primera parte de ejecución, de iniciar la ejecución de una función dependiente en la unidad, no poder responder correctamente a esta petición, de modo que por lo menos esta porción no se ejecuta correctamente y, en consecuencia, el software protegido no es completamente funcional.

Según una variante de realización, el procedimiento de acuerdo con la invención consiste en:

→ en la fase de protección:

- definir para por lo menos una función dependiente, una familia de funciones dependientes algorítmicamente equivalentes, aunque iniciadas por órdenes activadoras cuyas consignas renombradas son diferentes,

- y modificar el software protegido:

  - seleccionando en la fuente del software protegido por lo menos una orden activadora de consigna renombrada,

  - y modificando por lo menos una porción seleccionada de la fuente del software protegido al sustituir por lo menos la consigna renombrada de una orden activadora de consigna renombrada seleccionada, por otra consigna renombrada, que inicia una función dependiente de la misma familia.

Según una variante de realización, el procedimiento de acuerdo con la invención consiste en:

→ en la fase de protección, definir, para por lo menos una función dependiente, una familia de funciones dependientes algorítmicamente equivalentes:

- concatenando un campo de ruido con la información que define la parte funcional de la función dependiente a ejecutar en la unidad,

- o utilizando el campo de identificación de la instrucción y los campos de identificación prevista de los operandos.

Según una variante de realización, el procedimiento de acuerdo con la invención consiste en:

→ en la fase de protección:

- definir:

  - en calidad de método de renombramiento de las consignas, un método de cifrado para cifrar las consignas,

  - y en calidad de medios de restablecimiento, medios que ponen en práctica un método de descifrado para descifrar las consignas renombradas y restablecer así la identidad de las funciones dependientes a ejecutar en la unidad.

Según otra forma preferida de realización, el procedimiento de acuerdo con la invención consiste en:

→ en la fase de protección:

- modificar el software protegido:

  - seleccionando en la fuente del software protegido, por lo menos un salto condicional efectuado en por lo menos un procesado algorítmico seleccionado,

  - modificando por lo menos una porción seleccionada de la fuente del software protegido, siendo tal esta modificación que durante la ejecución del software protegido, se ejecuta la funcionalidad de por lo menos un salto condicional seleccionado, por medio de la segunda parte de ejecución, en la unidad,

  - y generando:

    ▹ la primera parte objeto del software protegido, siendo tal esta primera parte objeto que durante la ejecución del software protegido, se ejecuta en la unidad la funcionalidad de por lo menos un salto condicional seleccionado,

    ▹ y la segunda parte objeto del software protegido, siendo tal esta segunda parte objeto que, después de su carga en la unidad y durante la ejecución del software protegido, aparece la segunda parte de ejecución por medio de la cual se ejecuta la funcionalidad de por lo menos un salto condicional seleccionado,

→ y en la fase de utilización:

- en presencia de la unidad y cada vez que lo impone una porción de la primera parte de ejecución, ejecutar la funcionalidad de por lo menos un salto condicional a la unidad, de modo que esta porción se ejecuta correctamente y en consecuencia, el software protegido es completamente funcional,

- y en ausencia de la unidad y a pesar de la petición de una porción de la primera parte de ejecución, de

ejecutar la funcionalidad de un salto condicional en la unidad, no poder responder correctamente a esta petición, de modo que por lo menos esta porción no se ejecuta correctamente y, en consecuencia, el software protegido no es completamente funcional.

Según una variante de realización, el procedimiento de acuerdo con la invención consiste, en la fase de protección, en modificar el software protegido:

- seleccionando, en la fuente del software protegido por lo menos una serie de saltos condicionales seleccionados,
- modificando por lo menos una porción seleccionada de la fuente del software protegido, siendo tal esta modificación que durante la ejecución del software protegido, se ejecuta la funcionalidad global de por lo menos una serie seleccionada de saltos condicionales por medio de la segunda parte de ejecución, en la unidad,
- y generando:
  - la primera parte objeto del software protegido, siendo tal esta primera parte objeto que durante la ejecución del software protegido, se ejecuta la funcionalidad de por lo menos una serie seleccionada de saltos condicionales en la unidad,
  - y la segunda parte objeto del software protegido, siendo tal esta segunda parte objeto que después de su carga en la unidad y durante la ejecución del software protegido, aparece la segunda parte de ejecución por medio de la cual se ejecuta la funcionalidad global de por lo menos una serie seleccionada de saltos condicionales.

El procedimiento según la invención permite así proteger la utilización de un software al implementar una unidad de procesado y de memorización que presenta la particularidad de contener una parte del software en ejecución. De aquí se deduce que toda versión derivada del software que intenta funcionar sin la unidad de procesado y de memorización impone la reconstrucción de la parte del software contenida en la unidad de procesado y de memorización durante la ejecución, so pena que esta versión derivada del software no sea completamente funcional.

Otras diversas características se desprenden de la descripción realizada anteriormente en referencia a los dibujos adjuntos que muestran, a título de ejemplo no limitativo, formas de realización y de implementación del objeto de la invención.

Las figuras 10 y 11 son diagramas de bloques funcionales que ilustran las diversas representaciones de un software respectivamente no protegido y protegido por el procedimiento de acuerdo con la invención.

Las figuras 20 a 22 ilustran a título de ejemplo, diversas formas de realización de un dispositivo de implementación del procedimiento según la invención.

Las figuras 30 y 31 son diagramas de bloques funcionales que explicitan el principio general del procedimiento de acuerdo con la invención.

Las figuras 40 a 43 son esquemas que ilustran el procedimiento de protección de acuerdo con la invención poniendo en práctica el principio de protección por variable.

Las figuras 70 a 74 son esquemas que ilustran el procedimiento de protección de acuerdo con la invención poniendo en práctica el principio de protección por detección y coerción.

Las figuras 80 a 85 son esquemas que ilustran el procedimiento de protección de acuerdo con la invención poniendo en práctica el principio de protección por renombramiento.

Las figuras 90 a 92 son esquemas que ilustran el procedimiento de protección de acuerdo con la invención poniendo en práctica el principio de protección por salto condicional.

La figura 100 es un esquema que ilustra las diferentes fases de implementación del objeto de la invención.

La figura 110 ilustra un ejemplo de realización de un sistema que permite la implementación del estadio de construcción de la fase de protección de acuerdo con la invención.

La figura 120 ilustra un ejemplo de realización de una unidad de prepersonalización utilizada en el procedimiento de protección de acuerdo con la invención.

La figura 130 ilustra un ejemplo de realización de un sistema que permite la implementación del estadio de elaboración de herramientas de la fase de protección de acuerdo con la invención.

La figura 140 ilustra un ejemplo de realización de un sistema que permite la implementación del procedimiento de protección de acuerdo con la invención.

La figura 150 ilustra un ejemplo de realización de una unidad de personalización utilizada en el procedimiento de protección de acuerdo con la invención.

En adelante en la descripción, se utilizarán las siguientes definiciones:

- Un sistema de procesado de datos 3 es un sistema con capacidad de ejecutar un programa.
- Una unidad de procesado y de memorización es una unidad con capacidad:
  - de aceptar datos suministrados por un sistema de procesado de datos 3,
  - de restituir datos al sistema de procesado de datos 3,
  - de almacenar datos por lo menos en parte de manera secreta y de conservar por lo menos una parte de los mismos incluso cuando la unidad está sin alimentación,
  - y de efectuar un procesado algorítmico sobre datos, siendo secreta una parte o la totalidad de este procesado.
- Una unidad 6 es una unidad de procesado y de memorización que implementa el procedimiento según la invención.
- Una unidad virgen 60 es una unidad que no implementa el procedimiento según la invención, pero que puede recibir informaciones que la transforman en una unidad 6.
- Una unidad prepersonalizada 66 es una unidad virgen 60 que ha recibido una parte de las informaciones que le permiten, después de la recepción de informaciones complementarias, transformarse en una unidad 6.
- La carga de informaciones en una unidad virgen 60 o una unidad prepersonalizada 66 se corresponde con una transferencia de informaciones a la unidad virgen 60 o la unidad prepersonalizada 66, y con un almacenamiento de dichas informaciones transferidas. Eventualmente, la transferencia puede constar de un cambio de formato de las informaciones.
- Una variable, un dato o una función contenida en el sistema de procesado de datos 3 se indicará con una mayúscula, mientras que una variable, un dato o una función contenida en la unidad 6 se indicará con una minúscula.
- Un "software protegido", es un software que se ha protegido mediante por lo menos un principio de protección implementado a través del procedimiento de acuerdo con la invención.
- Un "software vulnerable", es un software que no se ha protegido mediante ningún principio de protección implementado a través del procedimiento de acuerdo con la invención.
- En el caso en el que la diferenciación entre un software vulnerable y un software protegido no tenga importancia, se utiliza el término "software".
- Un software se presenta bajo diversas representaciones según el instante considerado en su ciclo de vida:
  - una representación fuente,
  - una representación objeto,
  - una distribución,
  - o una representación dinámica.
- Una representación fuente de un software se interpreta como una representación que después de una transformación, proporciona una representación objeto. Una representación fuente se puede presentar según diferentes niveles, desde un nivel conceptual abstracto hasta un nivel ejecutable directamente por un sistema de procesado de datos o una unidad de procesado y de memorización.

- Una representación objeto de un software se corresponde con un nivel de representación que, después de una transferencia en una distribución y a continuación una carga en un sistema de procesado de datos o una unidad de procesado y de memorización, puede ser ejecutado. Se puede tratar, por ejemplo, de un código binario, de un código interpretado, etcétera.

- Una distribución es un soporte físico o virtual que contiene la representación objeto, debiéndose poner esta distribución a disposición del usuario para permitirle usar el software.

- Una representación dinámica se corresponde con la ejecución del software a partir de su distribución.

- Una porción de software se corresponde con una parte cualquiera de software y se puede corresponder, por ejemplo, con una o varias instrucciones consecutivas o no, y/o con uno o varios bloques funcionales consecutivos o no, y/o con una o varias funciones, y/o uno o varios subprogramas, y/o uno o varios módulos. Una porción de un software se puede corresponder también con la totalidad de este software.

Las figuras 10 y 11 ilustran las diversas representaciones respectivamente de un software vulnerable 2v en sentido general, y de un software protegido 2p según el procedimiento de la invención.

La figura 10 ilustra diversas representaciones de un software vulnerable 2v que aparece en el transcurso de su ciclo de vida. El software vulnerable 2v puede aparecer, por lo tanto, bajo una de las siguientes representaciones:

- una representación fuente 2vs,

- una representación objeto 2vo,

- una distribución 2vd. Esta distribución se puede presentar comúnmente bajo la forma de un medio de distribución físico tal como un CDROM o bajo la forma de archivos distribuidos a través de una red (GSM, Internet, ...),

- o una representación dinámica 2ve que se corresponde con la ejecución del software vulnerable 2v en un sistema de procesado de datos 3 de cualquier tipo conocido, el cual consta usualmente de por lo menos un procesador 4.

La figura 11 ilustra diversas representaciones de un software protegido 2p que aparece en el transcurso de su ciclo de vida. Así, el software protegido 2p puede aparecer bajo una de las siguientes representaciones:

- una representación fuente 2ps que consta de una primera parte fuente destinada al sistema de procesado de datos 3 y una segunda parte fuente destinada a la unidad 6, y pudiendo estar contenida comúnmente una parte de estas partes fuente en archivos comunes,

- una representación objeto 2po que consta de una primera parte objeto 2pos destinada al sistema de procesado de datos 3 y una segunda parte objeto 2pou destinada a la unidad 6,

- una distribución 2pd que consta de:

  - una primera parte de distribución 2pds que contiene la primera parte objeto 2pos, estando destinada esta primera parte de distribución 2pds al sistema de procesado de datos 3 y pudiéndose presentar comúnmente bajo la forma de un medio de distribución físico tal como un CDROM, o bajo la forma de archivos distribuidos a través de una red (GSM, Internet, ...),

  - y una segunda parte de distribución 2pdu que se presenta bajo la forma:

    - de por lo menos una unidad prepersonalizada 66 en la cual se ha cargado una parte de la segunda parte objeto 2pou y para la cual el usuario debe terminar la personalización al cambiar informaciones complementarias, con el fin de obtener una unidad 6, obteniéndose estas informaciones complementarias, por ejemplo, mediante carga o descarga a través de una red,

    - o de por lo menos una unidad 6 en la cual se ha cargado la segunda parte objeto 2pou,

- o una representación dinámica 2pe que se corresponde con la ejecución del software protegido 2p. Esta representación dinámica 2pe consta de una primera parte de ejecución 2pes que se ejecuta en el sistema de procesado de datos 3 y una segunda parte de ejecución 2peu que se ejecuta en la unidad 6.

En el caso en el que la diferenciación entre las diferentes representaciones del software protegido 2p no tenga

importancia, se utilizan las expresiones primera parte del software protegido y segunda parte del software protegido.

La implementación del procedimiento según la invención de acuerdo con la representación dinámica de la figura 11, utiliza un dispositivo 1p que consta de un sistema de procesado de datos 3 conectado a través de un enlace 5 a una unidad 6. El sistema de procesado de datos 3 es de cualquier tipo y, normalmente, consta de por lo menos un procesador 4. El sistema de procesado de datos 3 puede ser un ordenador o puede formar parte, por ejemplo, de diversas máquinas, dispositivos, productos fijos o móviles, o vehículos en sentido general. El enlace 5 se puede realizar de cualquier manera posible, tal como por ejemplo a través de una línea serie, un bus USB, un enlace de radiocomunicaciones, un enlace óptico, un enlace de red o una conexión eléctrica directa a un circuito del sistema de procesado de datos 3, etcétera. Debe indicarse que la unidad 6 se puede encontrar de manera eventual físicamente en el interior del mismo circuito integrado que el procesador 4 del sistema de procesado de datos 3. En este caso, la unidad 6 se puede considerar como un coprocesador con respecto al procesador 4 del sistema de procesado de datos 3 y el enlace 5 es interno al circuito integrado.

Las figuras 20 a 22 muestran de manera ilustrativa y a título no limitativo, diversas formas de realización del dispositivo 1p que permiten la implementación del procedimiento de protección de acuerdo con la invención.

En el ejemplo de realización ilustrado en la figura 20, el dispositivo de protección 1p consta de, en calidad de sistema de procesado de datos 3, un ordenador y, en calidad de unidad 6, una tarjeta chip 7 y su interfaz 8 comúnmente denominada lector de tarjeta. El ordenador 3 está conectado a la unidad 6 mediante un enlace 5. Durante la ejecución de un software protegido 2p, la primera parte de ejecución 2pes que se ejecuta en el ordenador 3 y la segunda parte de ejecución 2peu que se ejecuta en la tarjeta chip 7 y su interfaz 8, deben ser funcionales, las dos, con el fin de que el software protegido 2p sea completamente funcional.

En el ejemplo de realización ilustrado en la figura 21, el dispositivo de protección 1p equipa a un producto 9 en sentido general, que consta de diversos órganos 10 adaptados a la o las funciones asumidas por dicho producto 9. El dispositivo de protección 1p consta de, por una parte, un sistema de procesado de datos 3 incorporado en el producto 9 y, por otra parte, una unidad 6 asociada al producto 9. Para que el producto 9 sea totalmente funcional, el software protegido 2p debe ser completamente funcional. Así, durante la ejecución del software protegido 2p, la primera parte de ejecución 2pes que se ejecuta en el sistema de procesado de datos 3 y la segunda parte de ejecución 2peu que se ejecuta en la unidad 6, deben ser funcionales, las dos. Este software protegido 2p permite así, de manera indirecta, proteger contra un uso no autorizado, el producto 9 o una de sus funcionalidades. Por ejemplo, el producto 9 puede ser una instalación, un sistema, una máquina, un juguete, un aparato electrodoméstico, un teléfono, etcétera.

En el ejemplo de realización ilustrado en la figura 22, el dispositivo de protección 1p incluye varios ordenadores, así como una parte de una red de comunicación. El sistema de procesado de datos 3 es un primer ordenador conectado mediante un enlace 5 de tipo red, a una unidad 6 constituida por un segundo ordenador. Para implementar la invención, el segundo ordenador 6 se utiliza como servidor de licencias para un software protegido 2p. Durante la ejecución del software protegido 2p, la primera parte de ejecución 2pes que se ejecuta en el primer ordenador 3 y la segunda parte de ejecución 2peu que se ejecuta en el segundo ordenador 6, deben ser funcionales, las dos, de manera que el software protegido 2p sea completamente funcional.

La figura 30 permite explicitar de manera más precisa el procedimiento de protección de acuerdo con la invención. Debe indicarse que un software vulnerable 2v se considera de manera que está siendo ejecutado totalmente en un sistema de procesado de datos 3. Por el contrario, en el caso de la implementación de un software protegido 2p, el sistema de procesado de datos 3 consta de medios de transferencia 12 conectados mediante el enlace 5, a medios de transferencia 13 que forman parte de la unidad 6 permitiendo conseguir que se comuniquen entre ellas la primera parte de ejecución 2pes y la segunda parte de ejecución 2peu del software protegido 2p.

Debe considerarse que los medios de transferencia 12, 13 tienen naturaleza de software y/o material y son aptos para garantizar y, eventualmente, optimizar la comunicación de los datos entre el sistema de procesado de datos 3 y la unidad 6. Estos medios de transferencia 12, 13 están adaptados para permitir disponer de un software protegido 2p que, preferentemente, es independiente del tipo del enlace 5 utilizado. Estos medios de transferencia 12, 13 no forman parte del objeto de la invención y no se describen de manera más precisa ya que son bien conocidos por los expertos en la materia. La primera parte del software protegido 2p consta de órdenes. Durante la ejecución del software protegido 2p, la ejecución de estas órdenes por la primera parte de ejecución 2pes permite la comunicación entre la primera parte de ejecución 2pes y la segunda parte de ejecución 2peu. En adelante en la descripción, estas órdenes se representan con IN, OUT o TRIG.

Tal como se ilustra en la figura 31, para permitir la implementación de la segunda parte de ejecución 2peu del software protegido 2p, la unidad 6 consta de medios de protección 14. Los medios de protección 14 constan de medios de memorización 15 y de medios de procesado 16.

En adelante para simplificar descripción, se prefiere considerar, durante la ejecución del software protegido 2p, la presencia de la unidad 6 ó la ausencia de la unidad 6. En realidad, una unidad 6 que presenta medios de protección

14 inadecuados para la ejecución de la segunda parte de ejecución 2peu del software protegido 2p también se considera como ausente, cada vez que la ejecución del software protegido 2p no es correcta. En otras palabras:

- una unidad 6 físicamente presente y que consta de medios de protección 14 adaptados para la ejecución de la segunda parte de ejecución 2peu del software protegido 2p, siempre se considera como presente,

- una unidad 6 físicamente presente pero que consta de medios de protección 14 inadecuados, es decir, que no permiten la implementación correcta de la segunda parte de ejecución 2peu del software protegido 2p se considera como presente, cuando funciona correctamente, y como ausente cuando no funciona correctamente,

- y una unidad 6 físicamente ausente se considera siempre como ausente.

En el caso en el que la unidad 6 esté constituida por una tarjeta chip 7 y su interfaz 8, los medios de transferencia 13 se descomponen en dos partes de las cuales una se encuentra en la interfaz 8 y otra se encuentra en la tarjeta chip 7. En este ejemplo de realización, la ausencia de la tarjeta chip 7 se considera como equivalente a la ausencia de la unidad 6. En otras palabras, en ausencia de la tarjeta chip 7 y/o de su interfaz 8, los medios de protección 14 no son accesibles y no permiten por tanto la ejecución de la segunda parte de ejecución 2peu del software protegido, de modo que el software protegido 2p no es completamente funcional.

Según la invención, el procedimiento de protección pretende implementar un principio de protección, que se denomina por "detección y coerción", del cual se efectúa una descripción en relación con las figuras 70 a 74.

Para la implementación del principio de protección por detección y coerción, se define:

- por lo menos una característica de ejecución de software susceptible de ser supervisada por lo menos en parte en la unidad 6,

- por lo menos un criterio que se debe respetar para por lo menos una característica de ejecución de software,

- medios de detección 17 a poner en práctica en la unidad 6 y que permiten detectar que por lo menos una característica de ejecución de software no respeta por lo menos un criterio asociado,

- y medios de coerción 18 a poner en práctica en la unidad 6 y que permiten informar al sistema de procesado de datos 3 y/o modificar la ejecución de un software, cuando no se respeta por lo menos un criterio.

Para la implementación del principio de protección por detección y coerción, también se construyen medios operativos que permiten transformar una unidad virgen 60 en una unidad 6 que ponga en práctica por lo menos los medios de detección 17 y los medios de coerción 18.

La figura 70 ilustra los medios de necesarios para la implementación de este principio de protección por detección y coerción. La unidad 6 consta de los medios de detección 17 y de los medios de coerción 18 pertinentes a los medios de procesado 16. A los medios de coerción 18 se les informa del hecho de no respetar un criterio por parte de los medios de detección 17.

De una forma más precisa, los medios de detección 17 utilizan informaciones procedentes de los medios de transferencia 13 y/o de los medios de memorización 15 y/o de los medios de procesado 16, con el fin de supervisar una o varias características de ejecución de software. Para cada característica de ejecución de software se fija por lo menos un criterio que se debe respetar.

En el caso en el que se detecte que por lo menos una característica de ejecución de software no respeta por lo menos un criterio, los medios de detección 17 informan de ello a los medios de coerción 18. Estos medios de coerción 18 están adaptados para modificar, de la manera apropiada, el estado de la unidad 6.

Para la implementación del principio de protección por detección y coerción, también se selecciona:

- por lo menos una característica de ejecución de software que se debe supervisar, entre las características de ejecución susceptibles de ser supervisadas,

- por lo menos un criterio a respetar para por lo menos una característica de ejecución de software seleccionada,

- en la fuente del software vulnerable 2vs, por lo menos un procesado algorítmico para el cual por lo menos se debe supervisar una característica de ejecución de software,

- y en la fuente del software vulnerable 2vs, por lo menos una porción que contiene por lo menos un procesado

algorítmico seleccionado.

A continuación se modifica por lo menos una porción seleccionada de la fuente del software vulnerable 2vs, de manera que se obtenga la fuente del software protegido 2ps. Esta modificación es tal que especialmente durante la ejecución del software protegido 2p:

- por lo menos una porción de la primera parte de ejecución 2pes, que se ejecuta en el sistema de procesado de datos 3, tiene en cuenta que por lo menos se supervisará una característica de ejecución de software seleccionada, por lo menos en parte en la unidad 6,
- y la segunda parte de ejecución 2peu, que se ejecuta en la unidad 6, supervisa por lo menos en parte, una característica de ejecución de software seleccionada.

Durante la ejecución del software protegido 2p, protegido por este principio de protección por detección y coerción, en presencia de la unidad 6:

- mientras se respeten todos los criterios correspondientes a todas las características de ejecución supervisadas de todas las porciones modificadas del software protegido 2p, estas porciones modificadas del software protegido 2p funcionan de manera nominal, y en consecuencia, el software protegido 2p funciona de manera nominal,
- y si no se respeta por lo menos uno de los criterios que se corresponden con una característica de ejecución supervisada de una porción del software protegido 2p, al sistema de procesado de datos 3 se le informa de ello y/o el funcionamiento de la porción del software protegido 2p se modifica, de modo que se modifica el funcionamiento del software protegido 2p.

Evidentemente, en ausencia de la unidad 6, no se puede satisfacer correctamente por lo menos una petición de una porción de la primera parte de ejecución 2pes del software protegido 2p, de utilizar la unidad 6, de modo que por lo menos esta porción no se ejecuta correctamente y en consecuencia, el software protegido 2p no es completamente funcional.

Para la implementación del principio de protección por detección y coerción, se utilizan preferentemente dos tipos de características de ejecución de software.

El primer tipo de característica de ejecución de software se corresponde con una variable de medición de la ejecución de un software y el segundo tipo se corresponde con un perfil de uso de un software. Estos dos tipos de características se pueden utilizar de manera independiente o combinados.

Para la implementación del principio de protección por detección y coerción que utiliza, en calidad de característica de ejecución, una variable de medición de la ejecución de software, se define:

- en los medios de memorización 15, la posibilidad de memorizar por lo menos una variable de medición que sirve para cuantificar el uso de por lo menos una funcionalidad de software,
- en los medios de detección 17, la posibilidad de supervisar por lo menos un umbral asociado a cada variable de medición,
- y medios de actualización que permiten actualizar cada variable de medición en función del uso de la funcionalidad a la cual está asociada la misma.

También se construyen medios operativos que ponen en práctica, además de los medios de detección 17 y de los medios de coerción 18, los medios de actualización.

También se selecciona, en la fuente del software vulnerable 2vs:

- por lo menos una funcionalidad del software vulnerable 2v cuyo uso es susceptible de ser supervisado gracias a una variable de medición,
- por lo menos una variable de medición que sirve para cuantificar el uso de dicha funcionalidad,
- por lo menos un umbral asociado a la variable de medición que se corresponde con un límite de uso de dicha funcionalidad,
- y por lo menos un método de actualización de la variable de medición en función del uso de dicha funcionalidad.

La fuente del software vulnerable 2vs se modifica a continuación, de manera que se obtenga la fuente del software protegido 2ps, siendo esta modificación tal que, durante la ejecución del software protegido 2p, la segunda parte de ejecución 2peu:

- actualiza la variable de medición en función del uso de dicha funcionalidad,
- y tiene en cuenta por lo menos una superación de umbral.

En otras palabras, durante la ejecución del software protegido 2p, la variable de medición se actualiza en función del uso de dicha funcionalidad, y cuando se supera el umbral, los medios de detección 17 informan de ello a los medios de coerción 18 los cuales toman una decisión adaptada para informar al sistema de procesado de datos 3 y/o modificar los procesados efectuados por los medios de procesado 16 permitiendo modificar el funcionamiento de la porción del software protegido 2p, de modo que se modifica el funcionamiento del software protegido 2p.

Para la implementación de una primera variante preferida de realización del principio de protección por detección y coerción que utiliza, como característica, una variable de medición, se definen:

- para por lo menos una variable de medición, varios umbrales asociados,
- y medios de coerción diferentes que se corresponden con cada uno de estos umbrales.

También se selecciona, en la fuente del software vulnerable 2vs:

- por lo menos una variable de medición que sirve para cuantificar el uso de por lo menos una funcionalidad del software y a la cual deben estar asociados varios umbrales que se corresponden con límites diferentes de uso de dicha funcionalidad,
- y por lo menos dos umbrales asociados a la variable de medición.

La fuente del software vulnerable 2vs se modifica a continuación, de manera que se obtenga la fuente del software protegido 2ps, siendo esta modificación tal que, durante la ejecución del software protegido 2p, la segunda parte de ejecución 2peu:

- actualiza la variable de medición en función del uso de dicha funcionalidad,
- y tiene en cuenta, de manera diferente, las superaciones de diversos umbrales.

En otras palabras, usualmente, durante la ejecución del software protegido 2p, cuando se supera el primer umbral, la unidad 6 informa al sistema de procesado de datos 3 ordenando al software protegido 2p que ya no utilice esta funcionalidad. Si el software protegido 2p continúa utilizando esta funcionalidad, se podrá superar el segundo umbral. En el caso en el que se supere el segundo umbral, los medios de coerción 18 pueden convertir en inoperativa la funcionalidad seleccionada y/o convertir en inoperativo el software protegido 2p.

Para la implementación de una segunda variante preferida de realización del principio de protección por detección y coerción que utiliza, como característica, una variable de medición, se definen medios de recarga que permiten acreditar por lo menos un uso suplementario para por lo menos una funcionalidad de software supervisada por una variable de medición.

También se construyen medios operativos que ponen en práctica los medios de recarga, además de los medios de detección 17, de los medios de coerción 18 y de los medios de actualización.

También se selecciona, en la fuente del software vulnerable 2vs, por lo menos una variable de medición que sirve para limitar el uso de por lo menos una funcionalidad del software y a la cual se debe poder acreditar por lo menos un uso suplementario.

La fuente del software vulnerable 2vs se modifica a continuación, de manera que se obtenga la fuente del software protegido 2ps, siendo esta modificación tal que, en una fase denominada de recarga, se puede acreditar por lo menos un uso suplementario de por lo menos una funcionalidad que se corresponde con una variable de medición seleccionada.

En la fase de recarga, se procede a la reactualización de por lo menos una variable de medición seleccionada y/o de por lo menos un umbral asociado, de manera que se permita por lo menos un uso suplementario de la funcionalidad correspondiente. En otras palabras, es posible, en la fase de recarga, acreditar usos suplementarios de por lo menos una funcionalidad del software protegido 2p.

Para la implementación del principio de protección por detección y coerción que utiliza, como característica, un perfil

de uso de software, se define en calidad de criterio a respetar para este perfil de uso, por lo menos un rasgo de ejecución de software.

También se selecciona, en la fuente del software vulnerable 2vs:

- por lo menos un perfil de uso que se debe supervisar,
- y por lo menos un rasgo de ejecución que debe ser respetado por lo menos por un perfil de uso seleccionado.

La fuente del software vulnerable 2vs se modifica a continuación, de manera que se obtenga la fuente del software protegido 2ps, siendo esta modificación tal que durante la ejecución del software protegido 2p, la segunda parte de ejecución 2peu respeta todos los rasgos de ejecución seleccionados. En otras palabras, la propia unidad 6 supervisa la manera en la cual se ejecuta la segunda parte de ejecución 2peu y puede informar al sistema de procesado de datos 3 y/o modificar el funcionamiento del software protegido 2p, en caso de que no se respete por lo menos un rasgo de ejecución.

Durante la ejecución del software protegido 2p, protegido por este principio, en presencia de la unidad 6:

- mientras se respeten todos los rasgos de ejecución de todas las porciones modificadas del software protegido 2p, estas porciones modificadas del software protegido 2p funcionan de manera nominal y en consecuencia, el software protegido 2p funciona de manera nominal,
- y si no se respeta por lo menos un rasgo de ejecución de una porción de software protegido 2p, se informa de ello al sistema de procesado de datos 3 y/o se modifica el funcionamiento de la porción del software protegido 2p, de modo que se modifica el funcionamiento del software protegido 2p.

Se puede considerar la supervisión de diferentes rasgos de ejecución, como por ejemplo la supervisión de la presencia de instrucciones que constan de un marcador o la supervisión del encadenamiento de ejecución para por lo menos una parte de las instrucciones.

Para la implementación del principio de protección por detección y coerción que utiliza, en calidad de rasgo de ejecución a respetar, la supervisión del encadenamiento de ejecución para por lo menos una parte de las instrucciones, se define:

- un juego de instrucciones cuyas instrucciones son susceptibles de ser ejecutadas en la unidad 6,
- un juego de órdenes de instrucción para este juego de instrucciones, siendo susceptibles estas órdenes de instrucciones de ser ejecutadas en el sistema de procesado de datos 3. La ejecución de cada una de estas órdenes de instrucciones en el sistema de procesado de datos 3 inicia en la unidad 6, la ejecución de la instrucción correspondiente,
- medios de detección 17 que permiten detectar que el encadenamiento de las instrucciones no se corresponde con el deseado,
- y medios de coerción 18 que permiten informar al sistema de procesado de datos 3 y/o modificar la ejecución de un software cuando el encadenamiento de las instrucciones no se corresponde con el deseado.

Asimismo se construyen medios operativos que permiten, en la unidad 6, ejecutar también las instrucciones del juego de instrucciones, siendo iniciada la ejecución de estas instrucciones por la ejecución en el sistema de procesado de datos 3, de las órdenes de instrucciones.

También se selecciona, en la fuente del software vulnerable 2vs, por lo menos un procesado algorítmico que se debe desviar a la unidad 6 y para el cual se va a supervisar el encadenamiento de por lo menos una parte de las instrucciones.

La fuente del software vulnerable 2vs se modifica a continuación de manera que se obtenga la fuente del software protegido 2ps, siendo tal esta modificación que, durante la ejecución del software protegido 2p:

- la segunda parte de ejecución 2peu ejecuta por lo menos la funcionalidad del procesado algorítmico seleccionado,
- el procesado algorítmico seleccionado se descompone en instrucciones,
- se especifica el encadenamiento que deben respetar por lo menos algunas de las instrucciones durante su ejecución en la unidad 6,

- y la primera parte de ejecución 2pes del software protegido 2p ejecuta órdenes de instrucciones que inician la ejecución de las instrucciones en la unidad 6.

Durante la ejecución del software protegido 2p, protegido por este principio, en presencia de la unidad 6:

- mientras que el encadenamiento de las instrucciones de todas las porciones modificadas del software protegido 2p se corresponda con el deseado, estas porciones modificadas del software protegido 2p funcionan de manera nominal y, en consecuencia, el software protegido 2p funciona de manera nominal,

- y si el encadenamiento de las instrucciones de una porción de software protegido 2p ejecutadas en la unidad 6 no se corresponde con el deseado, se informa de ello al sistema de procesado de datos 3 y/o se modifica el funcionamiento de la porción del software protegido 2p, de modo que se modifica el funcionamiento del software protegido 2p.

La figura 71 ilustra un ejemplo de implementación del principio de protección por detección y coerción que utiliza, en calidad de rasgo de ejecución a respetar, la supervisión del encadenamiento de ejecución de por lo menos una parte de las instrucciones, en el caso en el que se respeta el encadenamiento deseado.

La primera parte de ejecución 2pes del software protegido 2p, ejecutada en el sistema de procesado de datos 3, ejecuta órdenes de instrucciones CI_i que inician, en la unidad 6, la ejecución de instrucciones i_i pertinentes al juego de instrucciones. En el juego de instrucciones, por lo menos algunas de las instrucciones constan, cada una de ellas, de una parte que define la funcionalidad de la instrucción y una parte que permite verificar el encadenamiento deseado para la ejecución de las instrucciones. En este ejemplo, las órdenes de instrucciones CI_i se representan con $TRIG(i_i)$ y el encadenamiento deseado para la ejecución de las instrucciones es i_n, i_{n+1} e i_{n+2}. La ejecución en la unidad 6, de la instrucción i_n produce el resultado a, y la ejecución de la instrucción i_{n+i} produce el resultado b. La instrucción i_{n+2} utiliza como operando, los resultados a y b de las instrucciones i_n e i_{n+1} y su ejecución produce el resultado c.

Teniendo en cuenta que este encadenamiento de las instrucciones ejecutadas en la unidad 6 se corresponde con el deseado, de aquí se deduce un funcionamiento normal o nominal del software protegido 2p.

La figura 72 ilustra un ejemplo de implementación del principio de protección por detección y coerción que utiliza, en calidad de rasgo de ejecución a respetar, la supervisión del encadenamiento de ejecución de por lo menos una parte de las instrucciones, en el caso en el que no se respeta el encadenamiento deseado.

De acuerdo con este ejemplo, el encadenamiento deseado para la ejecución de las instrucciones es siempre i_n, i_{n+1} e i_{n+2}. Sin embargo, el encadenamiento de ejecución de las instrucciones es modificado por la sustitución de la instrucción i_n por la instrucción i'_n, de modo que el encadenamiento efectivamente ejecutado es i'_n, i_{n+1} e i_{n+2}. La ejecución de la instrucción i'_n produce el resultado a, es decir, el mismo resultado que la ejecución de la instrucción i_n. Sin embargo, como muy tarde durante la ejecución de la instrucción i_{n+2}, los medios de detección 17 detectan que la instrucción i'_n no se corresponde con la instrucción deseada para generar el resultado a utilizado como operando de la instrucción i_{n+2}. Los medios de detección 17 informan de ellos a los medios de coerción 18 que modifican en consecuencia, el funcionamiento de la instrucción i_{n+2}, de modo que la ejecución de la instrucción i_{n+2} produce el resultado c' que puede ser diferente de c. Evidentemente, si la ejecución de la instrucción i'_n produce un resultado a' diferente al resultado a de la instrucción i_n, es evidente que el resultado de la instrucción i_{n+2} también puede ser diferente a c.

En la medida en la que el encadenamiento de ejecución de las instrucciones ejecutadas en la unidad 6 no se corresponda con el deseado, se puede obtener por tanto una modificación del funcionamiento del software protegido 2p.

Las figuras 73 y 74 ilustran una variante preferida de realización del principio de protección por detección y coerción que utiliza, en calidad de rasgo de ejecución a respetar, la supervisión del encadenamiento de ejecución de por lo menos una parte de las instrucciones. De acuerdo con esta variante preferida, se define un juego de instrucciones de las cuales por lo menos ciertas instrucciones trabajan sobre registros y utilizan por lo menos un operando con el fin de producir un resultado.

Tal como se ilustra en la figura 73, se define para por lo menos algunas de las instrucciones que trabajan sobre registros, una parte PF que define la funcionalidad de la instrucción y una parte PE que define el encadenamiento deseado para la ejecución de las instrucciones. La parte PF se corresponden con el código de operación conocido por los expertos en la materia. La parte PE que define el encadenamiento deseado, consta de campos de bits que se corresponden con:

- un campo de identificación de la instrucción CII,

- y para cada operando k de la instrucción, variando k entre 1 y K, y siendo K el número de operandos de la

instrucción:

- un campo de bandera CD_k, que indica si es conveniente verificar la procedencia del operando k,
- y un campo de identificación prevista CIP_k del operando, que indica la identidad esperada de la instrucción que ha generado el contenido del operando k.

Tal como se ilustra en la figura 74, el juego de instrucciones consta de V registros pertinentes a los medios de procesado 16, denominándose R_v cada registro, y variando v entre 1 y V. Para cada registro R_v, se definen dos campos, a saber:

- un campo funcional CF_v, conocido por los expertos en la materia y que permite almacenar el resultado de la ejecución de las instrucciones,
- y un campo de identificación generada CIG_v que permite memorizar la identidad de la instrucción que ha generado el contenido del campo funcional CF_v. Este campo de identificación generada CIG_v se actualiza automáticamente con el contenido del campo de identificación de la instrucción CII que ha generado el campo funcional CF_v. Este campo de identificación generada CIG_v no es accesible, ni modificable por ninguna instrucción y es útil únicamente para los medios de detección 17.

Durante la ejecución de una instrucción, los medios de detección 17 efectúan para cada operando k las siguientes operaciones:

- se lee el campo de bandera CD_k,
- si el campo de bandera CD_k lo impone, se leen tanto el campo de identificación prevista CIP_k como el campo de identificación generada CIG_v que se corresponden con el registro utilizado para el operando k,
- se controla la igualdad de los dos campos CIP_k y CIG_v,
- y si la igualdad es falsa, los medios de detección 17 consideran que no se ha respetado el encadenamiento de ejecución de las instrucciones.

Los medios de coerción 18 permiten modificar el resultado de las instrucciones cuando los medios de detección 17 les han informado de un encadenamiento de instrucciones no respetado. Una forma de realización preferida consiste en modificar la parte funcional PF de la instrucción en el curso de la ejecución o la parte funcional PF de instrucciones ulteriores.

Según otra característica ventajosa de la invención, el procedimiento de protección pretende implementar un principio de protección que se denomina por "variable", cuya descripción se efectúa en relación con las figuras 40 a 43.

Para la implementación del principio de protección por variable, se selecciona en la fuente del software vulnerable 2vs por lo menos una variable que, durante la ejecución del software vulnerable 2v, define parcialmente el estado de este último. Por estado de un software, se debe interpretar el conjunto de las informaciones, en un momento dado, necesarias para la ejecución completa de este software, de modo que la ausencia de una de dichas variables seleccionadas perjudica la ejecución completa de este software. También se selecciona por lo menos una porción de la fuente del software vulnerable 2vs que contiene por lo menos una variable seleccionada.

Por lo menos una porción seleccionada de la fuente del software vulnerable 2vs se modifica entonces de manera que se obtenga la fuente del software protegido 2ps. Esta modificación es tal que durante la ejecución del software protegido 2p, por lo menos una porción de la primera parte de ejecución 2pes que se ejecuta en el sistema de procesado de datos 3, tiene en cuenta que por lo menos una variable seleccionada o por lo menos una copia de variable seleccionada reside en la unidad 6.

La figura 40 ilustra un ejemplo de ejecución de un software vulnerable 2v. En este ejemplo, aparece en el transcurso de la ejecución del software vulnerable 2v en el sistema de procesado de datos 3:

- en el instante t_1, la asignación del dato X a la variable V_1, representada por $V_1 \leftarrow X$,
- en el instante t_2, la asignación del valor de la variable V_1 a la variable Y, representada por $Y \leftarrow V_1$,
- y en el instante t_3, la asignación del valor de la variable V_1 a la variable Z, representada por $Z \leftarrow V_1$.

La figura 41 ilustra un ejemplo de una primera forma de implementación de la invención para la cual la variable reside en la unidad 6. En este ejemplo, durante la ejecución en el sistema de procesado de datos 3 de la primera

parte de ejecución 2pes del software protegido 2p, y en presencia de la unidad 6, aparece:

- en el instante t_1, la ejecución de una orden de transferencia que inicia la transferencia del dato X desde el sistema de procesado de datos 3 hacia la variable v_1 situada en los medios de memorización 15 de la unidad 6, representándose esta orden de transferencia con OUT(v_1, X) y correspondiéndose finalmente con la asignación del dato X a la variable v_1,

- en el instante t_2, la ejecución de una orden de transferencia que inicia la transferencia del valor de la variable v_1 que reside en la unidad 6 hacia el sistema de procesado de datos 3 con el fin de asignarla a la variable Y, representándose esta orden de transferencia con IN(v_1) y correspondiéndose finalmente con la asignación del valor de la variable v_1 a la variable Y,

- y en el instante t_3, la ejecución de una orden de transferencia que inicia la transferencia del valor de la variable v_1 que reside en la unidad 6 hacia el sistema de procesado de datos 3 con el fin de asignarla a la variable Z, representándose esta orden de transferencia con IN(v_1) y correspondiéndose finalmente con la asignación del valor de la variable v_1 a la variable Z.

Debe indicarse que durante la ejecución del software protegido 2p, por lo menos una variable reside en la unidad 6. Así, cuando una porción de la primera parte de ejecución 2pes del software protegido 2p lo impone, y en presencia de la unidad 6, el valor de esta variable que reside en la unidad 6 se transfiere al sistema de procesado de datos 3 para ser utilizado por la primera parte de ejecución 2pes del software protegido 2p, de modo que esta porción se ejecuta correctamente y, en consecuencia, el software protegido 2p se completamente funcional.

La figura 42 ilustra un ejemplo de una segunda forma de implementación de la invención para la cual una copia de la variable reside en la unidad 6. En este ejemplo, durante la ejecución en el sistema de procesado de datos 3 de la primera parte de ejecución 2pes del software protegido 2p, y en presencia de la unidad 6, aparece:

- en el instante t_1, la asignación del dato X a la variable V_1 situada en el sistema de procesado de datos 3, así como la ejecución de una orden de transferencia que inicia la transferencia del dato X desde el sistema de procesado de datos 3 hacia la variable v_1 situada en los medios de memorización 15 de la unidad 6, representándose esta orden de transferencia con OUT(v_1, X),

- en el instante t_2, la asignación del valor de la variable V_1 a la variable Y,

- y en el instante t_3, la ejecución de una orden de transferencia que inicia la transferencia del valor de la variable v_1 que reside en la unidad 6 al sistema de datos 3 con el fin de asignarlo a la variable Z, representándose esta orden de transferencia con IN(v_1).

Debe indicarse que, durante la ejecución del software protegido 2p, por lo menos una copia de una variable reside en la unidad 6. Así, cuando una porción de la primera parte de ejecución 2pes del software protegido 2p lo impone, y en presencia de la unidad 6, el valor de esta copia de variable que reside en la unidad 6 se transfiere al sistema de procesado de datos 3 para ser utilizado por la primera parte de ejecución 2pes del software protegido 2p, de modo que esta porción se ejecuta correctamente y, en consecuencia, el software protegido 2p es completamente funcional.

La figura 43 ilustra un ejemplo de intento de ejecución del software protegido 2p, mientras la unidad 6 está ausente. En este ejemplo, durante la ejecución en el sistema de procesado de datos 3 de la primera parte de ejecución 2pes del software protegido 2p:

- en el instante t_1, la ejecución de la orden de transferencia OUT(v_1, X) no puede iniciar la transferencia del dato X a la variable v_1, teniendo en cuenta la ausencia de la unidad 6,

- en el instante t_2, la ejecución de la orden de transferencia IN(v_1) no puede iniciar la transferencia del valor de la variable v_1 al sistema de procesado de datos 3, teniendo en cuenta la ausencia de la unidad 6,

- y en el instante t_3, la ejecución de la orden de transferencia IN(v_1) no puede iniciar la transferencia del valor de la variable v_1 al sistema de procesado de datos 3, teniendo en cuenta la ausencia de la unidad 6.

Parece entonces que en ausencia de la unidad 6, no se puede satisfacer correctamente por lo menos una petición de una porción de la primera parte de ejecución 2pes, de utilizar una variable o una copia de variable que reside en la unidad 6, de modo que por lo menos esta porción no se ejecuta correctamente y en consecuencia, el software protegido 2p no es completamente funcional.

Debe indicarse que las transferencias de datos entre el sistema de procesado de datos 3 y la unidad 6 ilustrados en los ejemplos que preceden no utilizan más que asignaciones simples, aunque los expertos en la materia sabrán combinarlas con otras operaciones para derivar en operaciones complejas tales como por ejemplo OUT(v1, 2 * X +

3) o bien Z ←(5 * v1 + v2).

De acuerdo con otra característica ventajosa de la invención, el procedimiento de protección pretende implementar un principio de protección, que se denomina por "renombramiento", del cual se efectúa una descripción con relación a las figuras 80 a 85.

Para la implementación del principio de protección por renombramiento, se define:

- un conjunto de funciones dependientes, cuyas funciones dependientes son susceptibles de ser ejecutadas, por medio de la segunda parte de ejecución 2peu, en la unidad 6, y eventualmente de transferir datos entre el sistema de procesado de datos 3 y la unidad 6, de manera que este conjunto de funciones dependientes puede ser finito o infinito,

- un conjunto de órdenes activadoras para estas funciones dependientes, siendo estas órdenes activadoras susceptibles de ser ejecutadas en el sistema de procesado de datos 3 y de iniciar en la unidad 6, la ejecución de funciones dependientes correspondientes,

- para cada orden activadora, una consigna correspondiente por lo menos en parte a la información transmitida por la primera parte de ejecución 2pes, a la segunda parte de ejecución 2peu, con el fin de iniciar la ejecución de la función dependiente correspondiente, presentándose esta consigna bajo la forma de por lo menos un argumento de la orden activadora,

- un método de renombramiento de las consignas destinado a ser puesto en práctica durante la modificación del software vulnerable, permitiendo dicho método renombrar las consignas con el fin de obtener órdenes activadoras de consignas renombradas que permiten ocultar la identidad de las funciones dependientes correspondientes,

- y medios de restablecimiento 20 destinados a ponerse en práctica en la unidad 6 durante la fase de utilización y que permiten recuperar la consigna inicial, a partir de la consigna renombrada, con el fin de recuperar la función dependiente a ejecutar.

Para la implementación del principio de protección por renombramiento, también se construyen medios operativos que permiten transformar una unidad virgen 60 en una unidad 6 que pone por lo menos en práctica los medios de restablecimiento 20.

Para la implementación del principio de protección por renombramiento, también se selecciona, en la fuente del software vulnerable 2vs:

- por lo menos un procesado algorítmico que utiliza por lo menos un operando y que produce por lo menos un resultado,

- y por lo menos una porción de la fuente del software vulnerable 2vs, que contiene por lo menos un procesado algorítmico seleccionado.

La fuente del software vulnerable 2vs se modifica a continuación, de manera que se obtiene la fuente del software protegido 2ps. Siendo tal esta modificación que especialmente:

- durante la ejecución del software protegido 2p, por lo menos una porción de la primera parte de ejecución 2pes, que se ejecuta en el sistema de procesado de datos 3, tiene en cuenta que la funcionalidad de por lo menos un procesado algorítmico seleccionado se ejecuta en la unidad 6,

- durante la ejecución del software protegido 2p, la segunda parte de ejecución 2peu, que se ejecuta en la unidad 6, ejecuta por lo menos la funcionalidad de por lo menos un procesado algorítmico seleccionado,

- cada procesado algorítmico seleccionado se descompone de manera que durante la ejecución del software protegido 2p, cada procesado algorítmico seleccionado se ejecuta, por medio de la segunda parte de ejecución 2peu, utilizando funciones dependientes. Preferentemente, cada procesado algorítmico seleccionado se descompone en funciones dependientes fd_n (variando n entre 1 y N), a saber:

  - eventualmente una o varias funciones dependientes que permiten la puesta a disposición de uno o varios operandos para la unidad 6,

  - funciones dependientes de entre las cuales algunas utilizan el o los operandos y que, combinadas, ejecutan la funcionalidad del procesado algorítmico seleccionado, utilizando este o estos operandos,

  - y eventualmente una o varias funciones dependientes que permiten la puesta a disposición por la unidad

6, para el sistema de procesado de datos 3 del resultado del procesado algorítmico seleccionado,

- durante la ejecución del software protegido 2p, la segunda parte de ejecución 2peu ejecuta las funciones dependientes fd_n,

- durante la ejecución de software protegido 2p, las funciones dependientes son iniciadas por órdenes activadoras de consignas renombradas,

- y se selecciona una secuenciación de las órdenes activadoras entre el conjunto de las secuenciaciones que permiten la ejecución del software protegido 2p.

La primera parte de ejecución 2pes del software protegido 2p, ejecutada en el sistema de procesado de datos 3, ejecuta órdenes activadoras de consignas renombradas transfiriendo a la unidad 6 consignas renombradas, e iniciando en la unidad 6, el restablecimiento, a través de los medios de restablecimiento 20, de las consignas, y a continuación la ejecución por medio de la segunda parte de ejecución 2peu, de cada una de las funciones dependientes fd_n previamente definidas.

En otras palabras, el principio de protección por renombramiento consiste en renombrar las consignas de las órdenes activadoras, de manera que se obtengan órdenes activadoras de consignas renombradas cuya ejecución en el sistema de procesado de datos 3, inicia en la unidad 6, la ejecución de las funciones dependientes que habrían sido iniciadas por las órdenes activadoras de consignas no renombradas, aunque sin que el examen del software protegido 2p permita determinar la identidad de las funciones dependientes ejecutadas.

La figura 80 ilustra un ejemplo de ejecución de un software vulnerable 2v. En este ejemplo, aparece en el transcurso de la ejecución del software vulnerable 2v en el sistema de procesado de datos 3, en un momento dado, el cálculo de $Z \leftarrow F(X, Y)$ que se corresponde con la asignación a una variable Z del resultado de un procesado algorítmico representado por una función F y que utiliza los operandos X e Y.

Las figuras 81 y 82 ilustran un ejemplo de implementación de la invención.

La figura 81 ilustra la implementación parcial de la invención. En este ejemplo, durante la ejecución en el sistema de procesado de datos 3, de la primera parte de ejecución 2pes del software protegido 2p y en presencia de la unidad 6, aparece:

- en los instantes t_1, t_2, la ejecución de las órdenes activadoras CD_1, CD_2 que inician en la unidad 6, la ejecución por medio de la segunda parte de ejecución 2peu, de las funciones dependientes fd_1, fd_2 correspondientes que garantizan la transferencia de los datos X, Y desde el sistema de procesado de datos 3 hacia zonas de memorización respectivamente x, y situadas en los medios de memorización 15 de la unidad 6, representándose estas órdenes activadoras CD_1, CD_2 respectivamente con OUT(x, X), OUT(y, Y),

- en los instantes t_3 a t_{N-1}, la ejecución de las órdenes activadoras CD_3 a CD_{N-1}, que inician en la unidad 6, la ejecución por medio de la segunda parte de ejecución 2peu, de las funciones dependientes fd_3 a fd_{N-1} correspondientes, representándose estas órdenes activadoras CD_3 a CD_{N-1}, respectivamente, con TRIG(fd_3) a TRIG(fd_{N-1}). El desarrollo de las funciones dependientes fd_3 a fd_{N-1} ejecutadas en combinación es algorítmicamente equivalente a la función F. De manera más precisa, la ejecución de estas órdenes activadoras conduce a la ejecución en la unidad 6, de las funciones dependientes fd_3 a fd_{N-1} que se sirven del contenido de las zonas de memorización x, y, y producen el resultado en una zona de memorización z de la unidad 6,

- y en el instante t_N, la ejecución de una orden activadora CD_N que inicia en la unidad 6, la ejecución por medio de la segunda parte de ejecución 2peu, de la función dependiente fd_N que garantiza la transferencia del resultado del procesado algorítmico contenido en la zona de memorización z de la unidad 6 al sistema de procesado de datos 3, con el fin de asignarlo a la variable Z, representándose esta orden con IN(z).

En este ejemplo, para implementar completamente la invención, se selecciona como consigna, el primer argumento de las órdenes activadoras OUT y el argumento de las órdenes activadoras TRIG e IN. Las consignas así seleccionadas son renombradas por el método de renombramiento de las consignas. De esta manera, las consignas de las órdenes activadoras CD_1 a CD_N, a saber x, y, fd_3, fd_{N-1}, z se renombran de manera que se obtiene respectivamente R(x), R(y), R(fd_3)..., R(fd_{N-1}), R(z).

La figura 82 ilustra la implementación completa de la invención. En este ejemplo, durante la ejecución en el sistema de procesado de datos 3, de la primera parte de ejecución 2pes del software protegido 2p, y en presencia de la unidad 6, aparece:

- en los instantes t_1, t_2, la ejecución de las órdenes activadoras de consignas renombradas $CDCR_1$, $CDCR_2$, que transfieren hacia la unidad 6, las consignas renombradas R(x), R(y) así como los datos X, Y iniciando en

la unidad 6 el restablecimiento, por medio de los medios de restablecimiento 20, de las consignas renombradas para restablecer las consignas, a saber, la identidad de las zonas de memorización x, y, desde la ejecución por medio de la segunda parte de ejecución 2peu, de las funciones dependientes fd_1, fd_2 correspondientes que garantizan la transferencia de los datos X, Y desde el sistema de procesado de datos 3 hacia las zonas de memorización respectivamente x, y situadas en los medios de memorización 15 de la unidad 6, representándose estas órdenes activadoras de consignas renombradas $CDCR_1$, $CDCR_2$ respectivamente con OUT (R(x), X), OUT (R(y), Y),

- en los instantes t_3 a $t_{N-1,}$ la ejecución de las órdenes activadoras de consignas renombradas $CDCR_3$ a $CDCR_{N-1}$, que transfieren hacia la unidad 6, las consignas renombradas $R(fd_3)$ a $R(fd_{N-1})$, iniciando en la unidad 6 el restablecimiento a través de los medios de restablecimiento 20, de las consignas, a saber, fd3 a fd_{N-1}, y a continuación la ejecución por medio de la segunda parte de ejecución 2peu, de las funciones dependientes fd_3 a fd_{N-1}, representándose estas órdenes activadoras de consignas renombradas $CDCR_3$ a $CDCR_{N-1}$ respectivamente con TRIG ($R(fd_3)$) a TRIG ($R(fd_{N-1})$),

- y en el instante t_N, la ejecución de la orden activadora de consigna renombrada $CDCR_N$ que transfiere hacia la unidad 6, la consigna renombrada R(z) iniciando en la unidad 6 el restablecimiento, a través de los medios de restablecimiento 20, de la consigna, a saber, la identidad de la zona de memorización z, y a continuación la ejecución por medio de la segunda parte de ejecución 2peu, de la función dependiente fd_N que garantiza la transferencia del resultado del procesado algorítmico contenido en la zona de memorización z de la unidad 6 hacia el sistema de procesado de datos 3 con el fin de asignarlo a la variable Z, representándose esta orden activadora de consigna renombrada $CDCR_N$ con IN (R(z)).

En el ejemplo ilustrado, las órdenes activadoras de consignas renombradas 1 a N se ejecutan sucesivamente. Debe indicarse que se pueden aportar dos mejoras:

- La primera mejora se refiere al caso en el que varios procesados algorítmicos son desviados a la unidad 6 y por lo menos el resultado de un procesado algorítmico es utilizado por otro procesado algorítmico. Eventualmente, en este caso, ciertas órdenes activadoras de consignas renombradas que sirven para la transferencia, se pueden suprimir.

- La segunda mejora tiene como objetivo optar por una secuenciación pertinente de las órdenes activadoras de consignas renombradas entre el conjunto de las secuenciaciones que permiten la ejecución del software protegido 2p. A este respecto, es preferible seleccionar una secuenciación de las órdenes activadoras de consignas renombradas que disocia temporalmente la ejecución de las funciones dependientes, intercalando entre ellas porciones de código ejecutado en el sistema de procesado de datos 3 y que consta o no de las órdenes activadoras de consignas renombradas que sirven para la determinación de otros datos. Las figuras 83 y 84 ilustran el principio de una realización de este tipo.

La figura 83 muestra un ejemplo de ejecución de un software vulnerable 2v. En este ejemplo, aparece, en el transcurso de la ejecución del software vulnerable 2v, en el sistema de procesado de datos 3, la ejecución de dos procesados algorítmicos que conducen a la determinación de Z y Z', tales como Z ← F (X, Y) y Z' ← F'(X', Y').

La figura 84 ilustra un ejemplo de implementación del procedimiento de acuerdo con la invención para el cual los dos procesados algorítmicos seleccionados en la figura 83 son desviados a la unidad 6. De acuerdo con dicho ejemplo, durante la ejecución en el sistema de procesado de datos 3, de la primera parte de ejecución 2pes del software protegido 2p y en presencia de la unidad 6, aparece, tal como se ha explicado anteriormente en la presente, la ejecución de las órdenes activadoras de consignas renombradas $CDCR_1$ a $CDCR_N$ que se corresponden con la determinación de Z y la ejecución de las órdenes activadoras de consignas renombradas $CDCR'_1$ a $CDCR'_M$ que se corresponden con la determinación de Z'. Tal como se ilustra, las órdenes activadoras de consignas renombradas $CDCR_1$ a $CDCR_N$ no se ejecutan consecutivamente, en la medida en la que las órdenes activadoras de consignas renombradas $CDCR'_1$ a $CDCR'_M$ así como otras porciones de códigos están intercaladas. En el ejemplo, se materializa así la siguiente secuenciación: $CDCR_1$, porción de código intercalado, $CDCR'_1$, $CDCR_2$, porción de código intercalado, $CDCR'_2$, $CDCR'_3$, porción de código intercalado, $CDCR'_4$, $CDCR_3$, $CDCR_4$, ..., $CDCR_N$, $CDCR'_M$.

Debe indicarse que durante la ejecución de una porción de la primera parte de ejecución 2pes del software protegido 2p, las órdenes activadoras de consignas renombradas ejecutadas en el sistema de procesado de datos 3, inician en la unidad 6 el restablecimiento de la identidad de las funciones dependientes correspondientes y a continuación la ejecución de las mismas. Así, resulta que en presencia de la unidad 6, esta porción se ejecuta correctamente y, en consecuencia, el software protegido 2p es completamente funcional.

La figura 85 ilustra un ejemplo de intento de ejecución del software protegido 2p, mientras la unidad 6 está ausente. En este ejemplo, durante la ejecución en el sistema de procesado de datos 3 de la primera parte de ejecución 2pes del software protegido 2p, en todo momento, la ejecución de una orden activadora de consigna renombrada no puede iniciar el restablecimiento de la consigna ni la ejecución de la función dependiente correspondiente, debido a la ausencia de la unidad 6. El valor a asignar a la variable Z no se puede determinar entonces correctamente.

Resulta así, que en ausencia de la unidad 6, no se puede satisfacer correctamente por lo menos una petición de una porción de la primera parte de ejecución 2pes del software protegido 2p, de iniciar el restablecimiento de una consigna y la ejecución de una función dependiente en la unidad 6, de modo que por lo menos esta porción no se ejecuta correctamente y, en consecuencia, el software protegido 2p no es completamente funcional.

Gracias a este principio de protección por renombramiento, el examen en el software protegido 2p de las órdenes activadoras de consignas renombradas no permite determinar la identidad de las funciones dependientes que se deben ejecutar en la unidad 6. Debe indicarse que el renombramiento de las consignas se efectúa durante la modificación del software vulnerable 2v en un software protegido 2p.

De acuerdo con una variante del principio de protección por renombramiento, se define para por lo menos una función dependiente, una familia de funciones dependientes algorítmicamente equivalentes aunque iniciadas por órdenes activadoras de consignas renombradas diferentes. De acuerdo con esta variante, para por lo menos un procesado algorítmico que utiliza funciones dependientes, este procesado algorítmico se descompone en funciones dependientes que para por lo menos una de entre ellas se sustituye por una función dependiente de la misma familia en lugar de conservar varias apariciones de la misma función dependiente. Con este fin, las órdenes activadoras de consignas renombradas se modifican para tener en cuenta la sustitución de funciones dependientes por funciones dependientes de la misma familia. En otras palabras, dos funciones dependientes de la misma familia tienen consignas diferentes y, en consecuencia, órdenes activadoras de consignas renombradas diferentes y, no es posible, en el examen del software protegido 2p, descubrir que las funciones dependientes llamadas son algorítmicamente equivalentes.

De acuerdo con una primera forma de realización preferida de la variante del principio de protección por renombramiento, se define para por lo menos una función dependiente, una familia de funciones dependientes algorítmicamente equivalente, concatenando un campo de ruido a la información que define la parte funcional de la función dependiente a ejecutar en la unidad 6.

De acuerdo con una segunda forma de realización preferida de la variante del principio de protección por renombramiento, se define para por lo menos una función dependiente, una familia de funciones dependientes algorítmicamente equivalente utilizado campos de identificación.

De acuerdo con una variante preferida de realización del principio de protección por renombramiento, se define en calidad de método de renombramiento de las consignas, un método de cifrado que permite cifrar las consignas para transformarlas en consignas renombradas. Se recuerda que el renombramiento de las consignas se efectúa en la fase de protección P. Para esta variante preferida, los medios de restablecimiento 20 son medios que ponen en práctica un método de descifrado que permite descifrar las consignas renombradas y restablecer así la identidad de las funciones dependientes a ejecutar en la unidad 6. Estos medios de restablecimiento se ponen en práctica en la unidad 6 y pueden ser de naturaleza de software o material. Estos medios de restablecimiento 20 se solicitan en la fase de utilización U cada vez que se ejecuta una orden activadora de consigna renombrada en el sistema de procesado de datos 3 con el objetivo de iniciar en la unidad 6 la ejecución de una función dependiente.

De acuerdo con otra característica ventajosa de la invención, el procedimiento de protección pretende implementar un principio de protección denominado por "salto condicional" cuya descripción se efectúa con relación a las figuras 90 a 92.

Para la implementación del principio de protección por salto condicional, se selecciona en la fuente del software vulnerable 2vs, por lo menos un salto condicional BC. También se selecciona por lo menos una porción de la fuente del software vulnerable 2vs que contiene por lo menos un salto condicional BC seleccionado.

Se modifica entonces por lo menos una porción seleccionada de la fuente del software vulnerable 2vs, de manera que se obtiene la fuente del software protegido 2ps. Esta modificación es tal que particularmente durante la ejecución del software protegido 2p:

- por lo menos una porción de la primera parte de ejecución 2pes, que se ejecuta en el sistema de procesado de datos 3, tiene en cuenta que la funcionalidad de por lo menos un salto condicional BC seleccionado se ejecuta en la unidad 6,

- y la segunda parte de ejecución 2peu, que se ejecuta en la unidad 6, ejecuta por lo menos la funcionalidad de por lo menos un salto condicional BC seleccionado y pone a disposición del sistema de procesado de datos 3, una información que permite que la primera parte de ejecución 2pes prosiga con su ejecución en el lugar seleccionado.

La primera parte de ejecución 2pes del software protegido 2p, ejecutada en el sistema de procesado de datos 3, ejecuta órdenes de saltos condicionales, que inician en la unidad 6, la ejecución por medio de la segunda parte de ejecución 2peu, de saltos condicionales desviados bc cuya funcionalidad es equivalente a la funcionalidad de los

saltos condicionales BC seleccionados.

La figura 90 ilustra un ejemplo de ejecución de un software vulnerable 2v. En este ejemplo, aparece, en el transcurso de la ejecución del software vulnerable 2v en el sistema de procesado de datos 3 en un momento determinado, un salto condicional BC que indica al software vulnerable 2v el lugar en el que continuar su desarrollo, a saber, uno de los tres lugares posibles B_1, B_2 o B_3. Debe entenderse que el salto condicional BC toma la decisión de proseguir con la ejecución del software en el lugar B_1, B_2 o B_3.

La figura 91 ilustra un ejemplo de implementación de la invención para el cual el salto condicional seleccionado para ser desviado en la unidad 6, se corresponde con el salto condicional BC. En este ejemplo, durante la ejecución en el sistema de procesado de datos 3 de la primera parte de ejecución 2pes del software protegido 2p y en presencia de la unidad 6, aparece:

- en el instante t1, la ejecución de la orden de salto condicional CBC_1 que inicia en la unidad 6, la ejecución por medio de la segunda parte de ejecución 2peu, del salto condicional desviado bc algorítmicamente equivalente al salto condicional BC, representándose esta orden de salto condicional CBC_1 con TRIG(bc),

- y en el instante t_2, la transferencia de la unidad 6 al sistema de procesado de datos 3, de la información que permite que la primera parte de ejecución 2pes prosiga con su ejecución en el lugar seleccionado, a saber el lugar B_1, B_2 o B_3.

Debe indicarse que durante la ejecución de una porción de la primera parte de ejecución 2pes del software protegido 2p, las órdenes de saltos condicionales ejecutadas en el sistema de procesado de datos 3 inician la ejecución de los saltos condicionales desviados correspondientes en la unidad 6. Así, resulta que en presencia de la unidad 6, esta porción se ejecuta correctamente y, en consecuencia, el software protegido 2p es completamente funcional.

La figura 92 ilustra un intento de ejecución del software protegido 2p, mientras la unidad 6 está ausente. En este ejemplo, durante la ejecución en el sistema de procesado de datos 3 de la primera parte de ejecución 2pes del software protegido 2p:

- en el instante t_1, la ejecución de la orden de salto condicional CBC_1, no puede iniciar la ejecución del salto condicional desviado bc, teniendo en cuenta la ausencia de la unidad 6,

- y en el instante t_2, la transferencia de información que permite que la primera parte de ejecución 2pes prosiga en el lugar seleccionado fracasa, teniendo en cuenta la ausencia de la unidad 6.

Resulta así que, en ausencia de la unidad 6, no se puede satisfacer correctamente por lo menos una petición de una porción de la primera parte de ejecución 2pes de iniciar la ejecución de un salto condicional desviado en la unidad 6, de modo que por lo menos esta porción no se ejecuta correctamente y en consecuencia, el software protegido 2p no es completamente funcional.

En la descripción que precede con relación a las figuras 90 a 92, el objeto de la invención pretende desviar en la unidad 6, un salto condicional. Evidentemente, una forma de realización preferida de la invención puede consistir en desviar a la unidad 6, una serie de saltos condicionales cuya funcionalidad global es equivalente al conjunto de las funcionalidades de los saltos condicionales que han sido desviados. La ejecución de la funcionalidad global de esta serie de saltos condicionales desviados conduce a la puesta a disposición, para el sistema de procesado de datos 3, de una información que permite que la primera parte de ejecución 2pes del software protegido 2p prosiga con su ejecución en el lugar seleccionado.

En la descripción que precede en relación con las figuras 40 a 92, se han explicado cuatro principios diferentes de protección de un software, en general de manera independiente unos con respecto a los otros. El procedimiento de protección según la invención se pone en práctica al utilizar el principio de protección por detección y coerción, asociado eventualmente a otro u otros principios de protección. En el caso en el que el principio de protección por detección y coerción se complete mediante la puesta en práctica de por lo menos otro principio de protección, el principio de protección por detección y coerción se completa ventajosamente con el principio de protección por variable y/o el principio de protección por renombramiento y/o el principio de protección por salto condicional.

Y cuando se pone en práctica el principio de protección por renombramiento, este último se puede completar a su vez con el principio de protección por salto condicional.

Según la variante preferida de realización, el principio de protección por detección y coerción se completa con el principio de protección por variable y con el principio de protección por renombramiento, completado con el principio de protección por salto condicional.

En caso de que se aplique un principio de protección, como complemento del principio de protección por detección y coerción, su descripción efectuada anteriormente debe constar, para tener en cuenta su puesta en práctica

combinada, las siguientes modificaciones:

- la noción de software vulnerable se debe entender como software vulnerable con respecto al principio de protección en el transcurso de la descripción. Así, en el caso en el que un principio de protección ya se haya aplicado al software vulnerable, la expresión "software vulnerable" debe ser interpretada por el lector como la expresión "software protegido por el o los principios de protección ya aplicado(s)";

- la noción de software protegido se debe entender como software protegido con respecto al principio de protección en el transcurso de la descripción. Así, en el caso en el que ya se haya aplicado un principio de protección, la expresión "software protegido" debe ser interpretada por el lector como la expresión "nueva versión del software protegido";

- y la o las elecciones efectuada(s) para la implementación del principio de protección en el transcurso de la descripción debe(n) tener en cuenta la o las elecciones efectuada(s) para la implementación del o de los principio(s) de protección ya aplicado(s).

La continuación de la descripción permite comprender mejor la implementación del procedimiento de protección de acuerdo con la invención. Este procedimiento de protección de acuerdo con la invención hace que intervengan, tal como aparece de manera más precisa en la figura 100:

- en primer lugar, una fase de protección P en el transcurso de la cual un software vulnerable 2v se modifica en un software protegido 2p,

- a continuación, una fase de utilización U en el transcurso de la cual se implementa el software protegido 2p. En esta fase de utilización U:

  - en presencia de la unidad 6 y cada vez que lo imponga una porción de la primera parte de ejecución 2pes ejecutada en el sistema de procesado de datos 3, en la unidad 6 se ejecuta una funcionalidad impuesta, de modo que esta porción se ejecuta correctamente y, en consecuencia, el software protegido 2p es completamente funcional,

  - en ausencia de la unidad 6 y a pesar de la petición de una porción de la primera parte de ejecución 2pes de ejecutar una funcionalidad en la unidad 6, esta petición no se puede liquidar correctamente, de modo que por lo menos esta porción no se ejecuta correctamente y en consecuencia, el software protegido 2p no es completamente funcional,

- y eventualmente una fase de recarga R en el transcurso de la cual se acredita por lo menos un uso suplementario de una funcionalidad protegida por la implementación de la segunda variante preferida de realización del principio de protección por detección y coerción utilizando como característica, una variable de medición.

La fase de protección P se puede descomponer en dos subfases de protección P_1 y P_2. La primera, denominada subfase de protección aguas arriba P_1, se implementa independientemente del software vulnerable 2v que se debe proteger. La segunda, denominada subfase de protección aguas abajo P_2 depende del software vulnerable 2v a proteger. Debe indicarse que las subfases de protección aguas arriba P_1 y aguas abajo P_2 pueden ser materializadas ventajosamente por dos personas diferentes o dos equipos diferentes. Por ejemplo, la subfase de protección aguas arriba P_1 puede ser materializada por una persona o una sociedad que garantice el desarrollo de sistemas de protección de softwares, mientras que la subfase de protección aguas abajo P_2 puede ser materializada por una persona o una sociedad que garantice el desarrollo de softwares que deben ser protegidos. Evidentemente, es obvio que las subfases de protección aguas arriba P_1 y aguas abajo P_2 pueden también ser materializadas por la misma persona o el mismo equipo.

La subfase de protección aguas arriba P_1 implica varios estadios S_{11}, …, S_{1i} para cada uno de los cuales se van a efectuar diferentes tareas o trabajos.

Al primer estadio de esta subfase de protección aguas arriba P_1 se le denomina "estadio de definiciones S_{11}". Durante este estadio de definiciones S_{11}:

- se selecciona:

  - el tipo de la unidad 6. A título ilustrativo, se puede seleccionar, en calidad de unidad 6, un lector 8 de tarjetas chip y la tarjeta chip 7 asociada al lector,

  - y los medios de transferencia 12, 13 destinados a ser puestos en práctica respectivamente en el sistema de procesado de datos 3 y en la unidad 6, en el transcurso de la fase de utilización U y aptos para garantizar la transferencia de datos entre el sistema de procesado de datos 3 y la unidad 6,

- se define:
  - por lo menos una característica de ejecución de software, susceptible de ser supervisada por lo menos en parte en una unidad 6,
  - por lo menos un criterio a respetar para por lo menos una característica de ejecución de software,
  - medios de detección 17 a implementar en una unidad 6 y que permiten detectar que por lo menos una característica de ejecución del software no respeta por lo menos un criterio asociado
  - y medios de coerción 18 a implementar en una unidad 6 y que permiten informar al sistema de procesado de datos 3 y/o modificar la ejecución de un software cuando no se respeta por lo menos un criterio,
- y en el caso en el que el procedimiento de protección de acuerdo con la invención implementa el principio de protección por detección y coerción utilizando como característica una variable de medición de la ejecución del software, también se define:
  - en calidad de característica de ejecución de software susceptible de ser supervisada, una variable de medición del uso de una funcionalidad de un software,
  - en calidad de criterio a respetar, por lo menos un umbral asociado a cada variable de medición,
  - y medios de actualización que permiten actualizar por lo menos una variable de medición,
- y en el caso en el que el procedimiento de protección de acuerdo con la invención implementa una primera variante preferida de realización del principio de protección por detección y coerción utilizando como característica una variable de medición de la ejecución del software, también se define:
  - para por lo menos una variable de medición, varios umbrales asociados,
  - y medios de coerción diferentes que se corresponden con cada uno de estos umbrales,
- y en el caso en el que el procedimiento de protección de acuerdo con la invención implementa una segunda variante preferida de realización del principio de protección por detección y coerción utilizando como característica una variable de medición de la ejecución del software, también se definen medios de recarga que permiten acreditar por lo menos un uso suplementario para por lo menos una funcionalidad de software supervisada por una variable de medición,
- y en el caso en el que el procedimiento de protección de acuerdo con la invención implementa el principio de protección por detección y coerción utilizando como característica un perfil de uso de software, también se define:
  - en calidad de característica de ejecución de software susceptible de ser supervisada, un perfil de uso de software,
  - y en calidad de criterio a respetar, por lo menos un rasgo de ejecución de software,
- y en el caso en el que el procedimiento de protección de acuerdo con la invención implementa el principio de protección por detección y coerción utilizando en calidad de rasgo de ejecución a respetar, la supervisión del encadenamiento de ejecución, también se define:
  - un juego de instrucciones cuyas instrucciones son susceptibles de ser ejecutadas en la unidad 6,
  - un juego de órdenes de instrucciones para este juego de instrucciones, siendo estas órdenes de instrucción susceptibles de ser ejecutadas en el sistema de procesado de datos 3 y de iniciar en la unidad 6 la ejecución de las instrucciones,
  - en calidad de perfil de uso, el encadenamiento de las instrucciones,
  - en calidad de rasgo de ejecución, un encadenamiento deseado para la ejecución de las instrucciones,
  - en calidad de medios de detección 17, medios que permiten detectar que el encadenamiento de las instrucciones no se corresponde con el deseado,
  - y en calidad de medios de coerción 18, medios que permiten informar al sistema de procesado de datos 3

y/o modificar el funcionamiento de la porción de software protegido 2p cuando el encadenamiento de las instrucciones no se corresponde con el deseado,

- y en el caso en el que el procedimiento de protección de acuerdo con la invención implementa una variante preferida de realización del principio de protección por detección y coerción utilizando en calidad de rasgo de ejecución a respetar, la supervisión del encadenamiento de ejecución, también se define:

  - en calidad de juego de instrucciones, un juego de instrucciones de las cuales por lo menos ciertas instrucciones trabajan sobre registros y utilizan por lo menos un operando con vistas a producir un resultado,

  - para por lo menos una parte de las instrucciones que trabajan sobre registros:

    - una parte PF que define la funcionalidad de la instrucción,

    - y una parte que define el encadenamiento deseado para la ejecución de las instrucciones y que consta de campos de bits que se corresponden con:

      - un campo de identificación de la instrucción CII,

      - y para cada operando de la instrucción:

        - un campo de bandera CD_K,

        - y un campo de identificación prevista CIP_K del operando,

  - para cada registro pertinente a los medios operativos y utilizado por el juego de instrucciones, un campo de identificación generada CIG_v en el cual se memoriza automáticamente la identificación de la última instrucción que ha producido su resultado en este registro,

  - en calidad de medios de detección 17, medios que permiten, durante la ejecución de una instrucción, para cada operando, cuando lo impone el campo de bandera CD_k, controlar la igualdad entre el campo de identificación generada CIG_v que se corresponde con el registro utilizado por este operando, y el campo de identificación prevista CIP_k del origen de este operando,

  - y en calidad de medios de coerción 18, medios que permiten modificar el resultado de las instrucciones, si por lo menos una de las igualdades controladas es falsa,

- y en el caso en el que el procedimiento de protección de acuerdo con la invención implementa el principio de protección por renombramiento, también se define:

  - en calidad de una orden activadora, una orden de instrucción,

  - en calidad de una función dependiente, una instrucción,

  - en calidad de una consigna, por lo menos un argumento para una orden activadora, correspondiente por lo menos en parte a la información transmitida por el sistema de procesado de datos 3 a la unidad 6, con el fin de iniciar la ejecución de la función dependiente correspondiente,

  - un método de renombramiento de las consignas que permite renombrar las consignas con el fin de obtener órdenes activadoras de consignas renombradas,

  - y medios de restablecimiento 20 destinados a ponerse en práctica en la unidad 6 en el transcurso de la fase de utilización U, y que permiten recuperar la función dependiente a ejecutar, a partir de la consigna renombrada,

- y en el caso en el que el procedimiento de protección de acuerdo con la invención implementa una variante del principio de protección por renombramiento, también se define para por lo menos una función dependiente, una familia de funciones dependientes algorítmicamente equivalentes, aunque iniciadas por órdenes activadoras cuyas consignas renombradas son diferentes,

- y en el caso en el que el procedimiento de protección de acuerdo con la invención implementa una u otra de las formas de realización preferidas de la variante del principio de protección por renombramiento, también se define para por lo menos una función dependiente, una familia de funciones dependientes algorítmicamente equivalentes:

- concatenando un campo de ruido con la información que define la parte funcional de la función dependiente a ejecutar en la unidad 6,

- o utilizando el campo de identificación de la instrucción CII y los campos de identificación prevista CIP_k de los operandos,

- y en el caso en el que el procedimiento de protección de acuerdo con la invención implementa una variante preferida del principio de protección por renombramiento, también se define:

  - en calidad de método de renombramiento de las consignas, un método de cifrado para cifrar las consignas,

  - y en calidad de medios de restablecimiento 20, medios que ponen en práctica un método de descifrado para descifrar las consignas renombradas y restablecer así la identidad de las funciones dependientes a ejecutar en la unidad 6.

Durante la subfase de protección aguas arriba, al estadio de definición S_{11} le sigue un estadio denominado "estadio de construcción S_{12}". Durante un estadio S_{12} del tipo mencionado, se construyen los medios de transferencia 12,13 y los medios operativos que se corresponden con las definiciones del estadio de definición S_{11}.

Durante este estadio de construcción S_{12}, se procede así:

- a la construcción de los medios de transferencia 12,13 que permiten, en el transcurso de la fase de utilización U, la transferencia de datos entre el sistema de procesado de datos 3 y la unidad 6,

- a la construcción:

  - de los medios operativos que permiten que la unidad 6, en el transcurso de la fase de utilización U, ponga en práctica los medios de detección 17 y los medios de coerción 18,

  - y eventualmente de los medios operativos que permiten que la unidad 6, en el transcurso de la fase de utilización U, ponga también en práctica los medios de actualización,

  - y eventualmente de los medios operativos que permiten que la unidad 6, en el transcurso de la fase de recarga, ponga también en práctica los medios de recarga,

  - y eventualmente de los medios operativos que permiten también que la unidad 6, en el transcurso de la fase de utilización U, ejecute las instrucciones del juego de instrucciones,

- y cuando también se pone en práctica el principio de protección por renombramiento, a la construcción de los medios operativos que permiten que la unidad 6, en el transcurso de la fase de utilización U, ponga también en práctica los medios de restablecimiento.

La construcción de los medios operativos se realiza de manera usual, a través de una unidad de desarrollo de programa que tiene en cuenta las definiciones que se implican en el estadio de definiciones S_{11}. Dicha unidad se describe en la continuación de la descripción en la figura 110.

Durante la subfase de protección aguas arriba P_1, al estadio de construcción S_{12} le puede seguir un estadio denominado "estadio de prepersonalización S_{13}". Durante este estadio de prepersonalización S_{13}, por lo menos una parte de los medios de transferencia 13 y/o los medios operativos se cargan en por lo menos una unidad virgen 60 con vistas a obtener por lo menos una unidad prepersonalizada 66. Debe indicarse que una parte de los medios operativos, una vez transferida a una unidad prepersonalizada 66, ya no es directamente accesible desde el exterior de esta unidad prepersonalizada 66. La transferencia de los medios operativos a una unidad virgen 60 se puede realizar a través de una unidad de prepersonalización adaptada, la cual se describe en la continuación de la descripción en la figura 120. En el caso de una unidad prepersonalizada 66, constituida por una tarjeta chip 7 y por su lector 8, la prepersonalización no se refiere más que a la tarjeta chip 7.

Durante la subfase de protección aguas arriba P_1, se puede implementar, después del estadio de definición S_{11} y, eventualmente después del estadio de construcción S_{12}, un estadio denominado "estadio de elaboración de herramientas S_{14}". Durante este estadio de elaboración de herramientas S_{14}, se elaboran herramientas que permiten ayudar a la generación de softwares protegidos o automatizar la protección de softwares. Dichas herramientas permiten:

- ayudar a seleccionar o seleccionar automáticamente en el software vulnerable 2v a proteger:

- la o las características de ejecución a supervisar y, eventualmente, el o los procesados algorítmicos susceptibles de descomponerse en instrucciones desviables a la unidad 6,

- la o las porciones susceptibles de ser modificadas,

- y cuando también se pone en práctica el principio de protección por variable, la o las variables susceptibles de ser desviadas a la unidad 6,

- y cuando también se pone en práctica el principio de protección por renombramiento, el o los procesados algorítmicos susceptibles de descomponerse en funciones dependientes desviables a la unidad 6 y para las cuales se pueden renombrar las consignas de las órdenes activadoras,

- y cuando también se pone en práctica el principio de protección por salto condicional, el o los saltos condicionales cuya funcionalidad es susceptible de ser desviada en la unidad 6,

- y, eventualmente, ayudar a generar softwares protegidos o automatizar la protección de softwares.

Estas diferentes herramientas se pueden materializar independientemente o combinadas y cada herramienta puede adoptar diversas formas, como por ejemplo un preprocesador, un ensamblador, un compilador, etcétera.

A la subfase de protección aguas arriba P_1 le sigue una subfase de protección aguas abajo P_2 que depende del software vulnerable 2v a proteger. Esta subfase de protección aguas abajo P_2 implica asimismo varios estadios. El primer estadio correspondiente a la puesta en práctica del principio de protección por detección y coerción se denomina "estadio de creación S_{21}". Durante este estadio de creación S_{21}, se utilizan las elecciones implicadas en el estadio de definición S_{11}. Con la ayuda de estas elecciones y eventualmente de herramientas construidas en el estadio de elaboración de herramientas S_{14}, se crea el software protegido 2p:

- seleccionando por lo menos una característica de ejecución de software a supervisar, entre las características de ejecución susceptibles de ser supervisadas,

- seleccionando por lo menos un criterio a respetar para por lo menos una característica de ejecución de software seleccionado,

- seleccionando por lo menos un procesado algorítmico el cual, durante la ejecución del software vulnerable 2v, utiliza por lo menos un operando y permite obtener por lo menos un resultado, y para el cual se supervisará por lo menos una característica de ejecución de software seleccionado,

- seleccionando por lo menos una porción de la fuente del software vulnerable 2vs que contiene por lo menos un procesado algorítmico seleccionado,

- generando la fuente del software protegido 2ps a partir de la fuente del software vulnerable 2vs, al modificar por lo menos una porción seleccionada de la fuente del software vulnerable 2vs para obtener por lo menos una porción modificada de la fuente del software protegido 2ps, siendo tal esta modificación que:

  - durante la ejecución del software protegido 2p se ejecuta una primera parte de ejecución 2pes en el sistema de procesado de datos 3 y se ejecuta una segunda parte de ejecución 2peu en una unidad 6, que se obtiene a partir de la unidad virgen 60 después de la carga de informaciones,

  - la segunda parte de ejecución 2peu ejecuta por lo menos la funcionalidad de por lo menos un procesado algorítmico seleccionado,

  - y durante la ejecución del software protegido 2p, por lo menos una característica de ejecución seleccionada se supervisa por medio de la segunda parte de ejecución 2peu, y el hecho de no respetar un criterio conduce a una modificación de la ejecución del software protegido 2p,

- y generando:

  - una primera parte objeto 2pos del software protegido 2p, a partir de la fuente del software protegido 2ps, siendo tal esta primera parte objeto 2pos que durante la ejecución del software protegido 2p, aparece una primera parte de ejecución 2pes que se ejecuta en el sistema de procesado de datos 3 y de la cual por lo menos una porción tiene en cuenta que se supervisa por lo menos una característica de ejecución de software seleccionada,

  - y una segunda parte objeto 2pou del software protegido 2p, que contiene los medios operativos que ponen en práctica los medios de detección 17 y los medios de coerción 18, siendo tal esta segunda parte objeto 2pou que, después de su carga en la unidad virgen 60 y durante la ejecución del software

protegido 2p, aparece la segunda parte de ejecución 2peu por medio de la cual se supervisa por lo menos una característica de ejecución de software, y por medio de la cual el hecho de no respetar un criterio conduce a una modificación de la ejecución del software protegido 2p.

Evidentemente, el principio de protección por detección y coerción según la invención puede aplicarse directamente durante el desarrollo de un nuevo software sin que sea necesaria la materialización previa de un software vulnerable 2v. De esta manera, se obtiene directamente un software protegido 2p.

Para la implementación del principio de protección por detección y coerción que utiliza como característica una variable de medición de la ejecución del software, el software protegido 2p se modifica:

- seleccionando en calidad de característica de ejecución de software a supervisar, por lo menos una variable de medición del uso de una funcionalidad de un software,
- seleccionando:
  - por lo menos una funcionalidad del software protegido 2p cuyo uso es susceptible de ser supervisado gracias a una variable de medición,
  - por lo menos una variable de medición que sirve para cuantificar el uso de dicha funcionalidad,
  - por lo menos un umbral asociado a una variable de medición seleccionada que se corresponde con un límite de uso de dicha funcionalidad,
  - y por lo menos un método de actualización de una variable de medición seleccionada en función del uso de dicha funcionalidad,
- y modificando por lo menos una porción seleccionada de la fuente del software protegido 2ps, siendo tal esta modificación que, durante la ejecución del software protegido 2p, la variable de medición se actualiza por medio de la segunda parte de ejecución 2peu, en función del uso de dicha funcionalidad y se tiene en cuenta por lo menos una superación de umbral.

Para la implementación de una primera variante preferida de realización del principio de protección por detección y coerción que utiliza, como característica, una variable de medición, el software protegido 2p se modifica:

- seleccionando en la fuente del software protegido 2ps, por lo menos una variable de medición seleccionada a la cual deben estar asociados varios umbrales correspondientes a límites diferentes de uso de la funcionalidad,
- seleccionando por lo menos dos umbrales asociados a la variable de medición seleccionada,
- y modificando por lo menos una porción seleccionada de la fuente del software protegido 2ps, siendo tal esta modificación que, durante la ejecución del software protegido 2p, se tienen en cuenta las superaciones de los diversos umbrales, por medio de la segunda parte de ejecución 2peu, de manera diferente.

Para la implementación de una segunda variante preferida de realización del principio de protección por detección y coerción que utiliza como característica, una variable de medición, el software protegido 2p se modifica:

- seleccionando en la fuente del software protegido 2ps, por lo menos una variable de medición seleccionada que permite limitar el uso de una funcionalidad a la cual se debe poder acreditar por lo menos un uso suplementario,
- y modificando por lo menos una porción seleccionada, siendo tal esta modificación que en una fase denominada de recarga, se puede acreditar por lo menos un uso suplementario de por lo menos una funcionalidad que se corresponde con una variable de medición seleccionada.

Para la implementación del principio de protección por detección y coerción que utiliza como característica, un perfil de uso de software, el software protegido 2p se modifica:

- seleccionando en calidad de característica de ejecución de software a supervisar por lo menos un perfil de uso de software,
- seleccionando por lo menos un rasgo de ejecución que debe ser respetado por lo menos por un perfil de uso seleccionado,
- y modificando por lo menos una porción seleccionada de la fuente del software protegido 2ps, siendo tal esta

modificación que, durante la ejecución del software protegido 2p, la segunda parte de ejecución 2peu respeta todos los rasgos de ejecución seleccionados.

Para la implementación del principio de protección por detección y coerción que utiliza como rasgo de ejecución a respetar, la supervisión del encadenamiento de ejecución, el software protegido 2p se modifica:

- modificando por lo menos una porción seleccionada de la fuente del software protegido 2ps, siendo tal esta modificación que:
  - se descompone por lo menos un procesado algorítmico seleccionado, de manera que durante la ejecución del software protegido 2p, este procesado algorítmico se ejecuta por medio de la segunda parte de ejecución 2peu, utilizando instrucciones,
  - para por lo menos un procesado algorítmico seleccionado, se integran órdenes de instrucciones en la fuente del software protegido 2ps, de manera que durante la ejecución del software protegido 2p, cada orden de instrucción es ejecutada por la primera parte de ejecución 2pes y la misma inicia en la unidad 6, la ejecución por medio de la segunda parte de ejecución 2peu, de una instrucción,
  - se selecciona una secuenciación de las órdenes de instrucciones entre el conjunto de las secuenciaciones que permiten la ejecución del software protegido 2p,
  - y se especifica el encadenamiento que deben respetar por lo menos ciertas de las instrucciones durante su ejecución en la unidad 6.

Durante la subfase de protección aguas abajo P_2, y cuando se aplica por lo menos otro principio de protección además del principio de protección por detección y coerción, se pone en práctica un "estadio de modificación S_{22}". Durante este estadio de modificación S_{22}, se utilizan las definiciones implicadas en el estadio de definiciones S_{11}. Con la ayuda de estas definiciones y eventualmente de herramientas construidas en el estadio de elaboración de herramientas S_{14}, se modifica el software protegido 2p para permitir la puesta en práctica de los principios de protección según una de las disposiciones definidas anteriormente.

Cuando se pone en práctica el principio de protección por variable, se modifica el software protegido 2p:

- seleccionando por lo menos una variable utilizada en por lo menos un procesado algorítmico seleccionado, que durante la ejecución del software protegido 2p, define parcialmente el estado del software protegido 2p,
- modificando por lo menos una porción seleccionada de la fuente del software protegido 2ps, siendo esta modificación tal que durante la ejecución del software protegido 2p, por lo menos una variable seleccionada o por lo menos una copia de variable seleccionada reside en la unidad 6,
- y generando:
  - la primera parte objeto 2pos del software protegido 2p, siendo tal esta primera parte objeto 2pos que durante la ejecución del software protegido 2p, por lo menos una porción de la primera parte de ejecución 2pes tiene también en cuenta que por lo menos una variable o por lo menos una copia de variable reside en la unidad 6,
  - y la segunda parte objeto 2pou del software protegido 2p, siendo esta segunda parte objeto 2pou tal que, después de su carga en la unidad 6 y durante la ejecución del software protegido 2p, aparece la segunda parte de ejecución 2peu por medio de la cual por lo menos una variable seleccionada o por lo menos una copia de variable seleccionada reside también en la unidad 6.

Cuando se implementa el principio de protección por renombramiento, el software 2p se modifica:

- seleccionando en la fuente del software protegido 2ps, órdenes activadoras,
- modificando por lo menos una porción seleccionada de la fuente del software protegido 2ps renombrando las consignas de las órdenes activadoras seleccionadas, con el fin de ocultar la identidad de las funciones dependientes correspondientes,
- y generando:
  - la primera parte objeto 2pos del software protegido 2p, siendo tal esta primera parte objeto 2pos que durante la ejecución del software protegido 2p, se ejecutan las órdenes activadoras de consignas renombradas,

- y la segunda parte objeto 2pou del software protegido 2p que contiene los medios operativos que ponen también en práctica los medios de restablecimiento 20, siendo tal esta segunda parte objeto 2pou que, después de su carga en la unidad 6 y durante la ejecución del software protegido 2p, la identidad de las funciones dependientes cuya ejecución es iniciada por la primera parte de ejecución 2pes se restablece por medio de la segunda parte de ejecución 2peu, y las funciones dependientes son ejecutadas por medio de la segunda parte de ejecución 2peu.

Para la implementación de una variante del principio de protección por renombramiento, el software protegido 2p se modifica:

- seleccionando en la fuente del software protegido 2ps por lo menos una orden activadora de consigna renombrada,
- y modificando por lo menos una porción seleccionada de la fuente del software protegido 2ps al sustituir por lo menos la consigna renombrada de una orden activadora de consigna renombrada seleccionada, por otra consigna renombrada, que inicia una función dependiente de la misma familia.

Cuando se implementa el principio de protección por salto condicional, el software protegido 2p se modifica:

- seleccionando en la fuente del software protegido 2ps, por lo menos un salto condicional efectuado en por lo menos un procesado algorítmico seleccionado,
- modificando por lo menos una porción seleccionada de la fuente del software protegido 2ps, siendo tal esta modificación que durante la ejecución del software protegido 2p, se ejecuta la funcionalidad de por lo menos un salto condicional seleccionado, por medio de la segunda parte de ejecución 2peu, en la unidad 6,
- y generando:
  - la primera parte objeto 2pos del software protegido 2p, siendo tal esta primera parte objeto 2pos que durante la ejecución del software protegido 2p, se ejecuta la funcionalidad de por lo menos un salto condicional seleccionado en la unidad 6,
  - y la segunda parte objeto 2pou del software protegido 2p, siendo tal esta segunda parte objeto 2pou que, después de su carga en la unidad 6 y durante la ejecución del software protegido 2p, aparece la segunda parte de ejecución 2peu por medio de la cual se ejecuta la funcionalidad de por lo menos un salto condicional seleccionado.

Para la implementación de una forma de realización preferida del principio de protección por salto condicional, el software protegido 2p se modifica:

- seleccionando, en la fuente del software protegido 2ps por lo menos una serie de saltos condicionales seleccionados,
- modificando por lo menos una porción seleccionada de la fuente del software protegido 2ps, siendo tal esta modificación que durante la ejecución del software protegido 2p, la funcionalidad global de por lo menos una serie seleccionada de saltos condicionales se ejecuta por medio de la segunda parte de ejecución 2peu, en la unidad 6,
- y generando:
  - la primera parte objeto 2pos del software protegido 2p, siendo tal esta primera parte objeto 2pos que durante la ejecución del software protegido 2p, la funcionalidad de por lo menos una serie seleccionada de saltos condicionales se ejecuta en la unidad 6,
  - y la segunda parte objeto 2pou del software protegido 2p, siendo tal esta segunda parte objeto 2pou que, después de su carga en la unidad 6 y durante la ejecución del software protegido 2p, aparece la segunda parte de ejecución 2peu por medio de la cual se ejecuta la funcionalidad global de por lo menos una serie seleccionada de saltos condicionales.

Evidentemente, los principios de protección de acuerdo con la invención se pueden aplicar directamente durante el desarrollo de un nuevo software sin requerir la materialización previa de softwares protegidos intermediarios. De esta forma, los estadios de creación S_{21} y de modificación S_{22} se pueden efectuar de manera concomitante con el fin de obtener directamente el software protegido 2p.

Durante la subfase de protección aguas abajo P_2, se pone en práctica después del estadio de creación S_{21} del software protegido 2p, y eventualmente después del estadio de modificación S_{22}, un estadio denominado "estadio de

personalización S_{23}". Durante este estadio de personalización S_{23}, la segunda parte objeto 2pou que contiene los medios operativos se carga en por lo menos una unidad virgen 60, para obtener por lo menos una unidad 6, o una parte de la segunda parte objeto 2pou que contiene eventualmente los medios operativos se carga en por lo menos una unidad prepersonalizada 66, para obtener por lo menos una unidad 6. La carga de estas informaciones de personalización permite convertir en operativa por lo menos una unidad 6. Debe indicarse que una parte de estas informaciones, una vez que es transferida a una unidad 6, ya no es accesible directamente desde el exterior de esta unidad 6. La transferencia de las informaciones de personalización a una unidad virgen 60 o una unidad prepersonalizada 66 se puede realizar a través de una unidad de personalización adaptada que se describe en la continuación de la descripción en la figura 150. En el caso de una unidad 6, constituida por una tarjeta chip 7 y por su lector 8, la personalización no se refiere más que a la tarjeta chip 7.

Para la implementación de la fase de protección P, se describen de forma más precisa diferentes medios técnicos en relación con las figuras 110, 120, 130, 140 y 150.

La figura 110 ilustra un ejemplo de realización de un sistema 25 que permite implementar el estadio de construcción S_{12} que tiene en cuenta las definiciones implicadas en el estadio de definiciones S_{11} y en el transcurso del cual se construyen los medios de transferencia 12, 13 y eventualmente, los medios operativos destinados a la unidad 6. Un sistema 25 de este tipo consta de una unidad de desarrollo de programa o estación de trabajo que se presenta usualmente bajo la forma de un ordenador que comprende una unidad central, una pantalla, periféricos del tipo teclado-ratón, y que consta, particularmente, de los siguientes programas: editores de archivos, ensambladores, pre-procesadores, compiladores, intérpretes, depuradores y enlazadores.

La figura 120 ilustra un ejemplo de realización de una unidad de prepersonalización 30 que permite cargar por lo menos en parte los medios de transferencia 13 y/o los medios operativos en por lo menos una unidad virgen 60 con vistas a obtener por lo menos una unidad prepersonalizada 66. Esta unidad de prepersonalización 30 consta de un medio de lectura y de escritura 31 que permite prepersonalizar de manera eléctrica, una unidad virgen 60, para obtener una unidad prepersonalizada 66 en la cual se han cargado los medios de transferencia 13 y/u operativos. La unidad de prepersonalización 30 puede constar también de medios de personalización física 32 de la unidad virgen 60 que se pueden presentar, por ejemplo, en forma de una impresora. En el caso en el que la unidad 6 esté constituida por una tarjeta chip 7 y su lector 8, la prepersonalización se refiere en general únicamente a la tarjeta chip 7.

La figura 130 ilustra un ejemplo de realización de un sistema 35 que permite efectuar la elaboración de las herramientas que permiten ayudar a generar softwares protegidos o automatizar la protección de softwares. Un sistema 35 de este tipo consta de una unidad de desarrollo de programa o estación de trabajo que se presenta usualmente bajo la forma de un ordenador que comprende una unidad central, una pantalla, periféricos del tipo teclado-ratón, y que consta, particularmente, de los siguientes programas: editores de archivos, ensambladores, pre-procesadores, compiladores, intérpretes, depuradores y enlazadores.

La figura 140 ilustra un ejemplo de realización de un sistema 40 que permite crear directamente un software protegido 2p o modificar un software vulnerable 2v con vistas a obtener un software protegido 2p. Un sistema 40 de este tipo consta de una unidad de desarrollo de programa o estación de trabajo que se presenta usualmente bajo la forma de un ordenador que comprende una unidad central, una pantalla, periféricos del tipo teclado-ratón, y que consta, particularmente, de los siguientes programas: editores de archivos, ensambladores, pre-procesadores, compiladores, intérpretes, depuradores y enlazadores, así como herramientas que permiten ayudar a generar softwares protegidos o automatizar la protección de softwares.

La figura 150 ilustra un ejemplo de realización de una unidad de personalización 45 que permite cargar la segunda parte objeto 2pou en por lo menos una unidad virgen 60 con vistas a obtener por lo menos una unidad 6 ó una parte de la segunda parte objeto 2pou en por lo menos una unidad prepersonalizada 66 con vistas a obtener por lo menos una unidad 6. Esta unidad de personalización 45 consta de un medio de lectura y de escritura 46 que permite personalizar de manera eléctrica, por lo menos una unidad virgen 60 o por lo menos una unidad prepersonalizada 66, para obtener por lo menos una unidad 6. A la finalización de esta personalización, una unidad 6 consta de las informaciones necesarias para la ejecución del software protegido 2p. La unidad de personalización 45 puede también constar de medios de personalización física 47 para por lo menos una unidad 6 que se pueden presentar por ejemplo, en forma de una impresora. En el caso en el que una unidad 6 esté constituida por una tarjeta chip 7 y su lector 8, la personalización se refiere en general únicamente a la tarjeta chip 7.

El procedimiento de protección de la invención se puede implementar con las siguientes mejoras:

- Se puede prever la utilización conjunta de varias unidades de procesado y de memorización en las cuales se distribuye la segunda parte objeto 2pou del software protegido 2p de manera que su utilización conjunta permite ejecutar el software protegido 2p, en donde la ausencia de por lo menos una de estas unidades de procesado y de memorización impide el uso del software protegido 2p.

- Asimismo, después del estadio de prepersonalización S_{13} y durante el estadio de personalización S_{23}, la parte

de la segunda parte objeto 2pou necesaria para transformar la unidad prepersonalizada 66 en una unidad 6 puede estar contenida en una unidad de procesado y de memorización utilizada por la unidad de personalización 45 con el fin de limitar el acceso a esta parte de la segunda parte objeto 2pou. Evidentemente, esta parte de la segunda parte objeto 2pou se puede distribuir en varias unidades de procesado y de memorización de manera que esta parte de la segunda parte objeto 2pou sea accesible únicamente durante la utilización conjunta de estas unidades de procesado y de memorización.

REIVINDICACIONES

1. Procedimiento para proteger, a partir de por lo menos una unidad virgen (60) que consta de por lo menos unos medios de memorización (15) y unos medios de procesado (16), un software vulnerable (2v) contra su uso no autorizado, funcionando dicho software vulnerable (2v) en un sistema de procesado de datos (3), caracterizado por que consiste en:

→ en una fase de protección (P):

- definir:
  - por lo menos una característica de ejecución de software, susceptible de ser supervisada por lo menos en parte en una unidad (6),
  - por lo menos un criterio que se debe respetar para por lo menos una característica de ejecución de software,
  - unos medios de detección (17) que se deben implementar en una unidad (6) y que permiten detectar que por lo menos una característica de ejecución de software no respeta por lo menos un criterio asociado,
  - y unos medios de coerción (18) que se deben implementar en una unidad (6) y que permiten informar al sistema de procesado de datos (3) y/o modificar la ejecución de un software cuando no se respeta por lo menos un criterio,
- construir unos medios operativos que permiten transformar la unidad virgen (60) en una unidad (6) que puede implementar los medios de detección (17) y los medios de coerción (18),
- crear un software protegido (2p):
  - seleccionando por lo menos una característica de ejecución de software que se debe supervisar, entre las características de ejecución susceptibles de ser supervisadas,
  - seleccionando por lo menos un criterio que se debe respetar para por lo menos una característica de ejecución de software seleccionada,
  - seleccionando por lo menos un procesado algorítmico que, durante la ejecución del software vulnerable (2v), utiliza por lo menos un operando y permite obtener por lo menos un resultado, y para el cual se supervisará por lo menos una característica de ejecución de software seleccionada,
  - seleccionando por lo menos una porción de la fuente del software vulnerable (2vs) que contiene por lo menos un procesado algorítmico seleccionado,
  - produciendo la fuente del software protegido (2ps) a partir de la fuente del software vulnerable (2vs), modificando por lo menos una porción seleccionada de la fuente del software vulnerable (2vs) para obtener por lo menos una porción modificada de la fuente del software protegido (2ps), siendo tal esta modificación que:
    - durante la ejecución del software protegido (2p) se ejecuta una primera parte de ejecución (2pes) en el sistema de procesado de datos (3) y se ejecuta una segunda parte de ejecución (2peu) en una unidad (6), obtenida a partir de la unidad virgen (60) después de la carga de informaciones,
    - la segunda parte de ejecución (2peu) ejecuta por lo menos la funcionalidad de por lo menos un procesado algorítmico seleccionado,
    - y durante la ejecución del software protegido (2p), por lo menos una característica de ejecución seleccionada se supervisa por medio de la segunda parte de ejecución (2peu), y no respetar un criterio conduce a una modificación de la ejecución del software protegido (2p),
  - y produciendo:
    - una primera parte objeto (2pos) del software protegido (2p), a partir de la fuente del software protegido (2ps), siendo tal esta primera parte objeto (2pos) que durante la ejecución del software protegido (2p), aparece una primera parte de ejecución (2pes) que se ejecuta en el sistema de

procesado de datos (3) y de la cual por lo menos una porción tiene en cuenta que se supervisa por lo menos una característica de ejecución de software seleccionada,

- ▹ y una segunda parte objeto (2pou) del software protegido (2p), que contiene los medios operativos que implementan los medios de detección (17) y los medios de coerción (18), siendo tal esta segunda parte objeto (2pou) que, después de su carga en la unidad virgen (60) y durante la ejecución del software protegido (2p), aparece la segunda parte de ejecución (2peu) por medio de la cual se supervisa por lo menos una característica de ejecución de software, y por medio de la cual no respetar un criterio conduce a una modificación de la ejecución del software protegido (2p),

- y cargar la segunda parte objeto (2pou) en la unidad virgen (60), para obtener la unidad (6),

→ y en una fase de utilización (U) en el curso de la cual se ejecuta el software protegido (2p):

- en presencia de la unidad (6):
  - en la medida en la que se respeten todos los criterios que corresponden a todas las características de ejecución supervisadas de todas las porciones modificadas del software protegido (2p), permitir el funcionamiento nominal de estas porciones del software protegido (2p) y en consecuencia permitir el funcionamiento nominal del software protegido (2p),
  - y si no se respeta por lo menos uno de los criterios que corresponde a una característica de ejecución supervisada de una porción del software protegido (2p), informar al sistema de procesado de datos (3) y/o modificar el funcionamiento de la porción del software protegido (2p), de manera que se modifica el funcionamiento del software protegido (2p),
- y en ausencia de la unidad (6), a pesar de la petición de una porción de la primera parte de ejecución (2pes) de activar la ejecución en la unidad (6), de la funcionalidad de un procesado algorítmico seleccionado, no poder responder correctamente a esta petición, de manera que por lo menos esta porción no se ejecuta correctamente y, en consecuencia, el software protegido (2p) no es completamente funcional.

2. Procedimiento según la reivindicación 1, para limitar la utilización de un software protegido (2p), caracterizado por que consiste en:

→ en la fase de protección (P):

- definir:
  - como característica de ejecución de software susceptible de ser supervisada, una variable de medición del uso de una funcionalidad de un software,
  - como criterio que se debe respetar, por lo menos un umbral asociado a cada variable de medición,
  - y unos medios de actualización que permiten actualizar por lo menos una variable de medición,
- construir los medios operativos que permiten que la unidad (6) implemente asimismo los medios de actualización,
- y modificar el software protegido (2p):
  - seleccionando como característica de ejecución de software que se debe supervisar, por lo menos una variable de medición del uso de una funcionalidad de un software,
  - seleccionando:
    - ▹ por lo menos una funcionalidad del software protegido (2p) cuyo uso es susceptible de ser supervisado gracias a una variable de medición,
    - ▹ por lo menos una variable de medición que sirve para cuantificar el uso de dicha funcionalidad,
    - ▹ por lo menos un umbral asociado a una variable de medición seleccionada correspondiente a un límite de uso de dicha funcionalidad,

- ▹ y por lo menos un método de actualización de una variable de medición seleccionada en función del uso de dicha funcionalidad,

- y modificando por lo menos una porción seleccionada de la fuente del software protegido (2ps), siendo esta modificación tal que, durante la ejecución del software protegido (2p), la variable de medición se actualiza por medio de la segunda parte de ejecución (2peu), en función del uso de dicha funcionalidad y se tiene en cuenta por lo menos una superación de umbral,

→ y en la fase de utilización (U), en presencia de la unidad (6), y en el caso en el que se detecte por lo menos una superación de umbral correspondiente a por lo menos un límite de uso, informar al sistema de procesado de datos (3) y/o modificar el funcionamiento de la porción del software protegido (2p), de modo que se modifica el funcionamiento del software protegido (2p).

3. Procedimiento según la reivindicación 2, caracterizado por que consiste en:

→ en la fase de protección (P):

- definir:

  - para por lo menos una variable de medición, varios umbrales asociados,

  - y unos medios de coerción diferentes correspondientes a cada uno de estos umbrales,

- y modificar el software protegido (2p):

  - seleccionando en la fuente del software protegido (2ps), por lo menos una variable de medición seleccionada a la cual se deben asociar varios umbrales correspondientes a los límites diferentes de uso de la funcionalidad,

  - seleccionando por lo menos dos umbrales asociados a la variable de medición seleccionada,

  - y modificando por lo menos una porción seleccionada de la fuente del software protegido (2ps), siendo esta modificación tal que, durante la ejecución del software protegido (2p), se tienen en cuenta las superaciones de los diversos umbrales, por medio de la segunda parte de ejecución (2peu), de manera diferente,

→ y en la fase de utilización (U):

- en presencia de la unidad (6):

  - en el caso en el que se detecte la superación de un primer umbral, ordenar al software protegido (2p) que no utilice más la funcionalidad correspondiente,

  - y en el caso en el que se detecte la superación de un segundo umbral, convertir en inoperativa la funcionalidad correspondiente y/o por lo menos una porción del software protegido (2p).

4. Procedimiento según la reivindicación 2 o 3, caracterizado por que consiste en:

→ en la fase de protección (P):

- definir unos medios de recarga que permiten acreditar por lo menos un uso suplementario para por lo menos una funcionalidad de software supervisada por una variable de medición,

- construir los medios operativos que permiten asimismo que la unidad (6) implemente los medios de recarga,

- y modificar el software protegido (2p):

  - seleccionando en la fuente del software protegido (2ps), por lo menos una variable de medición seleccionada que permite limitar el uso de una funcionalidad a la cual se debe poder acreditar por lo menos un uso suplementario,

  - y modificando por lo menos una porción seleccionada, siendo esta modificación tal que en una fase denominada de recarga, se puede acreditar por lo menos un uso suplementario de por lo menos una funcionalidad correspondiente a una variable de medición seleccionada,

→ y en la fase de recarga:

- reactualizar por lo menos una variable de medición seleccionada y/o por lo menos un umbral asociado, de manera que se permita por lo menos un uso suplementario de la funcionalidad.

5. Procedimiento según la reivindicación 1, caracterizado por que consiste en:

→ en la fase de protección (P):

- definir:
  - como característica de ejecución de software susceptible de ser supervisada, un perfil de uso de software,
  - y como criterio que se debe respetar, por lo menos un rasgo de ejecución de software,
- y modificar el software protegido (2p):
  - seleccionando como característica de ejecución de software que se debe supervisar por lo menos un perfil de uso de software,
  - seleccionando por lo menos un rasgo de ejecución que debe ser respetado por lo menos por un perfil de uso seleccionado,
  - y modificando por lo menos una porción seleccionada de la fuente del software protegido (2ps), siendo esta modificación tal que, durante la ejecución del software protegido (2p), la segunda parte de ejecución (2peu) respeta todos los rasgos de ejecución seleccionados,

→ y en la fase de utilización (U) en presencia de la unidad (6), y en el caso en el que se detecte que no se respeta por lo menos un rasgo de ejecución, informar al sistema de procesado de datos (3) y/o modificar el funcionamiento de la porción del software protegido (2p), de modo que se modifique el funcionamiento del software protegido (2p).

6. Procedimiento según la reivindicación 5, caracterizado por que consiste en:

→ en la fase de protección (P):

- definir:
  - un juego de instrucciones cuyas instrucciones son susceptibles de ser ejecutadas en la unidad (6),
  - un juego de órdenes de instrucciones para este juego de instrucciones, siendo susceptibles estas órdenes de instrucciones de ser ejecutadas en el sistema de procesado de datos (3) y de activar en la unidad (6) la ejecución de las instrucciones,
  - como perfil de uso, el encadenamiento de las instrucciones,
  - como rasgo de ejecución, un encadenamiento deseado para la ejecución de las instrucciones,
  - como medios de detección (17), unos medios que permiten detectar que el encadenamiento de las instrucciones no corresponde al deseado,
  - y como medios de coerción (18), unos medios que permiten informar al sistema de procesado de datos (3) y/o modificar el funcionamiento de la porción de software protegido (2p) cuando el encadenamiento de las instrucciones no corresponde al deseado,
- construir los medios operativos que permiten asimismo que la unidad (6) ejecute las instrucciones del juego de instrucciones, siendo activada la ejecución de estas instrucciones por la ejecución en el sistema de procesado de datos (3), de las órdenes de instrucciones,
- y modificar el software protegido (2p):
  - modificando por lo menos una porción seleccionada de la fuente del software protegido (2ps), siendo esta modificación tal que:

- ▹ se descompone por lo menos un procesado algorítmico seleccionado, de manera que durante la ejecución del software protegido (2p), este procesado algorítmico se ejecuta por medio de la segunda parte de ejecución (2peu), utilizando instrucciones;

- ▹ para por lo menos un procesado algorítmico seleccionado, se integran órdenes de instrucciones en la fuente del software protegido (2ps), de manera que durante la ejecución del software protegido (2p), cada orden de instrucción es ejecutada por la primera parte de ejecución (2pes) y la misma activa en la unidad (6), la ejecución por medio de la segunda parte de ejecución (2peu), de una instrucción,

- ▹ se selecciona una secuenciación de las órdenes de instrucciones entre el conjunto de las secuenciaciones que permiten la ejecución del software protegido (2p),

- ▹ y se especifica el encadenamiento que deben respetar por lo menos algunas de las instrucciones durante su ejecución en la unidad (6),

→ y en la fase de utilización (U), en presencia de la unidad (6), en el caso en el que se detecte que el encadenamiento de las instrucciones ejecutadas en la unidad (6) no corresponde al deseado, informar al sistema de procesado de datos (3) y/o modificar el funcionamiento de la porción del software protegido (2p), de manera que se modifique el funcionamiento del software protegido (2p).

7. Procedimiento según la reivindicación 6, caracterizado por que consiste en:

→ en la fase de protección (P):

- definir:

  - como juego de instrucciones, un juego de instrucciones de las cuales por lo menos ciertas instrucciones trabajan sobre los registros y utilizan por lo menos un operando con el fin de producir un resultado,

  - para por lo menos una parte de las instrucciones que trabajan sobre los registros:

    - ▹ una parte (PF) que define la funcionalidad de la instrucción,

    - ▹ y una parte que define el encadenamiento deseado para la ejecución de las instrucciones y que consta de unos campos de bits que corresponden a:

      - ◊ un campo de identificación de la instrucción (CII),

      - ◊ y para cada operando de la instrucción:

        - ∗ un campo de bandera (CD_k),

        - ∗ y un campo de identificación prevista (CIP_k) del operando,

  - para cada registro perteneciente a los medios operativos y utilizado por el juego de instrucciones, un campo de identificación generada (CIG_V) en el cual se memoriza automáticamente la identificación de la última instrucción que ha producido su resultado en este registro,

  - como medios de detección (17), unos medios que permiten, durante la ejecución de una instrucción, para cada operando, cuando lo impone el campo de bandera (CD_k), controlar la igualdad entre el campo de identificación generada (CIG_v) correspondiente al registro utilizado por este operando, y el campo de identificación prevista (CIPk) del origen de este operando,

  - y como medios de coerción (18), unos medios que permiten modificar el resultado de las instrucciones, si por lo menos una de las igualdades controladas es falsa.

8. Procedimiento según una de las reivindicaciones 1 a 7, caracterizado por que consiste en:

→ en la fase de protección (P):

- modificar el software protegido (2p):

- seleccionando por lo menos una variable utilizada en por lo menos un procesado algorítmico seleccionado, que durante la ejecución del software protegido (2p), define parcialmente el estado del software protegido (2p),

- modificando por lo menos una porción seleccionada de la fuente del software protegido (2ps), siendo esta modificación tal que durante la ejecución del software protegido (2p), por lo menos una variable seleccionada o por lo menos una copia de variable seleccionada reside en la unidad (6),

- y produciendo:

  ▹ la primera parte objeto (2pos) del software protegido (2p), siendo tal esta primera parte objeto (2pos) que durante la ejecución del software protegido (2p), por lo menos una porción de la primera parte de ejecución (2pes) tiene asimismo en cuenta que por lo menos una variable o por lo menos una copia de variable reside en la unidad (6),

  ▹ y la segunda parte objeto (2pou) del software protegido (2p), siendo esta segunda parte objeto (2pou) tal que, después de su carga en la unidad (6) y durante la ejecución del software protegido (2p), aparece la segunda parte de ejecución (2peu) por medio de la cual por lo menos una variable seleccionada o por lo menos una copia de variable seleccionada reside asimismo en la unidad (6),

→ y en la fase de utilización (U):

- en presencia de la unidad (6) cada vez que lo impone una porción de la primera parte de ejecución (2pes), utilizar una variable o una copia de variable que reside en la unidad (6), de manera que esta porción se ejecuta correctamente y, en consecuencia, el software protegido (2p) es completamente funcional,

- y en ausencia de la unidad (6), a pesar de la petición de una porción de la primera parte de ejecución (2pes) utilizar una variable o una copia de variable que reside en la unidad (6), no poder responder correctamente a esta petición, de manera que por lo menos esta porción no se ejecuta correctamente y, en consecuencia, el software protegido (2p) no es completamente funcional.

9. Procedimiento según la reivindicación 6, caracterizado por que consiste en:

→ en la fase de protección (P):

- definir:

  - como una orden activadora, una orden de instrucción,

  - como una función dependiente, una instrucción,

  - como una consigna, por lo menos un argumento para una orden activadora, correspondiente por lo menos en parte a la información transmitida por el sistema de procesado de datos (3) a la unidad (6), con el fin de activar la ejecución de la función dependiente correspondiente,

  - un método de renombramiento de las consignas que permite renombrar las consignas con el fin de obtener unas órdenes activadoras de consignas renombradas,

  - y unos medios de restablecimiento (20) destinados a ser implementados en la unidad (6) en el curso de la fase de utilización (U), y que permiten recuperar la función dependiente que se debe ejecutar, a partir de la consigna renombrada,

- construir unos medios operativos que permiten que la unidad (6) implemente asimismo los medios de restablecimiento,

- y modificar el software protegido (2p):

  - seleccionando en la fuente del software protegido (2ps), unas órdenes activadoras,

  - modificando por lo menos una porción seleccionada de la fuente del software protegido (2ps) al renombrar las consignas de las órdenes activadoras seleccionadas, con el fin de ocultar la identidad de las funciones dependientes correspondientes,

  - y produciendo:

- la primera parte objeto (2pos) del software protegido (2p), siendo tal esta primera parte objeto (2pos) que durante la ejecución del software protegido (2p), se ejecutan las órdenes activadoras de consignas renombradas,

- y la segunda parte objeto (2pou) del software protegido (2p) que contiene los medios operativos que implementan asimismo los medios de restablecimiento (20), siendo tal esta segunda parte objeto (2pou) que, después de su carga en la unidad (6) y durante la ejecución del software protegido (2p), se restablece por medio de la segunda parte de ejecución (2peu) la identidad de las funciones dependientes cuya ejecución es activada por la primera parte de ejecución (2pes), y las funciones dependientes son ejecutadas por medio de la segunda parte de ejecución (2peu),

→ y en la fase de utilización (U):

- en presencia de la unidad (6) y cada vez que una orden activadora de consigna renombrada, contenida en una porción de la primera parte de ejecución (2pes) lo impone, restablecer en la unidad (6), la identidad de la función dependiente correspondiente y ejecutar la misma, de modo que esta porción se ejecuta correctamente y que, en consecuencia, el software protegido (2p) es completamente funcional,

- y en ausencia de la unidad (6), a pesar de la petición de una porción de la primera parte de ejecución (2pes), de activar la ejecución de una función dependiente en la unidad (6), no poder responder correctamente a esta petición, de modo que por lo menos esta porción no se ejecuta correctamente y, en consecuencia, el software protegido (2p) no es completamente funcional.

10. Procedimiento según la reivindicación 9, caracterizado por que consiste en:

→ en la fase de protección (P):

- definir para por lo menos una función dependiente, una familia de funciones dependientes algorítmicamente equivalentes, aunque activadas por órdenes activadoras cuyas consignas renombradas son diferentes,

- y modificar el software protegido (2p):

  - seleccionando en la fuente del software protegido (2ps) por lo menos una orden activadora de consigna renombrada,

  - y modificando por lo menos una porción seleccionada de la fuente del software protegido (2ps) sustituyendo por lo menos la consigna renombrada de una orden activadora de consigna renombrada seleccionada, por otra consigna renombrada, que activa una función dependiente de la misma familia.

11. Procedimiento según la reivindicación 10, caracterizado por que consiste en:

→ en la fase de protección (P), definir, para por lo menos una función dependiente, una familia de funciones dependientes algorítmicamente equivalentes:

- concatenando un campo de ruido con la información que define la parte funcional de la función dependiente que se debe ejecutar en la unidad (6),

- o utilizando el campo de identificación de la instrucción (CII) y los campos de identificación prevista (CIP_k) de los operandos.

12. Procedimiento según la reivindicación 9, 10 u 11, caracterizado por que consiste en:

→ en la fase de protección (P):

- definir:

  - como método de renombramiento de las consignas, un método de cifrado para cifrar las consignas,

  - y como medios de restablecimiento (20), unos medios que implementan un método de descifrado para descifrar las consignas renombradas y restablecer así la identidad de las funciones dependientes que se deben ejecutar en la unidad (6).

13. Procedimiento según una de las reivindicaciones 1 a 12, caracterizado por que consiste en:

→ en la fase de protección (P):

- modificar el software protegido (2p):

  - seleccionando en la fuente del software protegido (2ps), por lo menos un salto condicional efectuado en por lo menos un procesado algorítmico seleccionado,

  - modificando por lo menos una porción seleccionada de la fuente del software protegido (2ps), siendo tal esta modificación que durante la ejecución del software protegido (2p), se ejecuta la funcionalidad de por lo menos un salto condicional seleccionado, por medio de la segunda parte de ejecución (2peu), en la unidad (6),

  - y produciendo:

    ▹ la primera parte objeto (2pos) del software protegido (2p), siendo tal esta primera parte objeto (2pos) que durante la ejecución del software protegido (2p), se ejecuta la funcionalidad de por lo menos un salto condicional seleccionado en la unidad (6),

    ▹ y la segunda parte objeto (2pou) del software protegido (2p), siendo tal esta segunda parte objeto (2pou) que, después de su carga en la unidad (6) y durante la ejecución del software protegido (2p), aparece la segunda parte de ejecución (2peu) por medio de la cual se ejecuta la funcionalidad de por lo menos un salto condicional seleccionado,

→ y en la fase de utilización (U):

- en presencia de la unidad (6) y cada vez que lo impone una porción de la primera parte de ejecución (2pes), ejecutar la funcionalidad de por lo menos un salto condicional en la unidad (6), de modo que esta porción se ejecuta correctamente y, en consecuencia, el software protegido (2p) es completamente funcional,

- y en ausencia de la unidad (6) y a pesar de la petición de una porción de la primera parte de ejecución (2pes) de ejecutar la funcionalidad de un salto condicional en la unidad (6), no poder responder correctamente a esta petición, de modo que por lo menos esta porción no se ejecuta correctamente y que, en consecuencia, el software protegido (2p) no es completamente funcional.

14. Procedimiento según la reivindicación 13, caracterizado por que consiste, en la fase de protección (P), en modificar el software protegido (2p):

- seleccionando, en la fuente del software protegido (2ps) por lo menos una serie de saltos condicionales seleccionados,

- modificando por lo menos una porción seleccionada de la fuente del software protegido (2ps), siendo tal esta modificación que durante la ejecución del software protegido (2p), se ejecuta la funcionalidad global de por lo menos una serie seleccionada de saltos condicionales por medio de la segunda parte de ejecución (2peu), en la unidad (6),

- y produciendo:

  ▹ la primera parte objeto (2pos) del software protegido (2p), siendo tal esta primera parte objeto (2pos) que durante la ejecución del software protegido (2p), se ejecuta la funcionalidad de por lo menos una serie seleccionada de saltos condicionales en la unidad (6),

  ▹ y la segunda parte objeto (2pou) del software protegido (2p), siendo tal esta segunda parte objeto (2pou) que, después de su carga en la unidad (6) y durante la ejecución del software protegido (2p), aparece la segunda parte de ejecución (2peu) por medio de la cual se ejecuta la funcionalidad global de por lo menos una serie seleccionada de saltos condicionales.

15. Procedimiento según una de las reivindicaciones 1 a 14, caracterizado por que consiste en descomponer la fase de protección (P) en una subfase de protección aguas arriba (P1), independiente del software que se debe proteger y una subfase de protección aguas abajo (P2), dependiente del software que se debe proteger.

16. Procedimiento según la reivindicación 15, caracterizado por que consiste, durante la subfase de protección aguas arriba (P1), en hacer intervenir un estadio de definiciones (S11) en el cual se efectúan todas las definiciones.

17. Procedimiento según la reivindicación 16, caracterizado por que consiste, después del estadio de definiciones (S11), en hacer intervenir un estadio de construcción (S12) en el cual se construyen los medios operativos.

18. Procedimiento según la reivindicación 17, caracterizado por que consiste, después del estadio de construcción (S12), en hacer intervenir un estadio de prepersonalización (S13), que consiste en cargar en una unidad virgen (60), por lo menos una parte de los medios operativos con el fin de obtener una unidad prepersonalizada (66).

19. Procedimiento según la reivindicación 16 o 17, caracterizado por que consiste, durante la subfase de protección aguas arriba (P1), en hacer intervenir un estadio de elaboración de herramientas (S14) en el cual se elaboran las herramientas que permiten ayudar a generar softwares protegidos o automatizar la protección de softwares.

20. Procedimiento según las reivindicaciones 15 y 18, caracterizado por que consiste en descomponer la subfase de protección aguas abajo (P2), en:

- un estadio de creación (S21) en el cual se crea el software protegido (2p), a partir del software vulnerable (2v),
- eventualmente, un estadio de modificación (S22) en el cual se modifica el software protegido (2p),
- y un estadio de personalización (S23) en el cual:
  - la segunda parte objeto (2pou) del software protegido (2p) que contiene los medios operativos se carga en por lo menos una unidad virgen (60) con el fin de obtener por lo menos una unidad (6),
  - o una parte de la segunda parte objeto (2pou) del software protegido (2p) que contiene eventualmente los medios operativos se carga en por lo menos una unidad prepersonalizada (66) con el fin de obtener por lo menos una unidad (6).

21. Procedimiento según las reivindicaciones 19 y 20, caracterizado por que consiste, durante el estadio de creación (S21) y eventualmente el estadio de modificación (S22), en utilizar por lo menos una de las herramientas de ayuda para la generación de softwares protegidos o de automatización de la protección de softwares.

FIG. 10

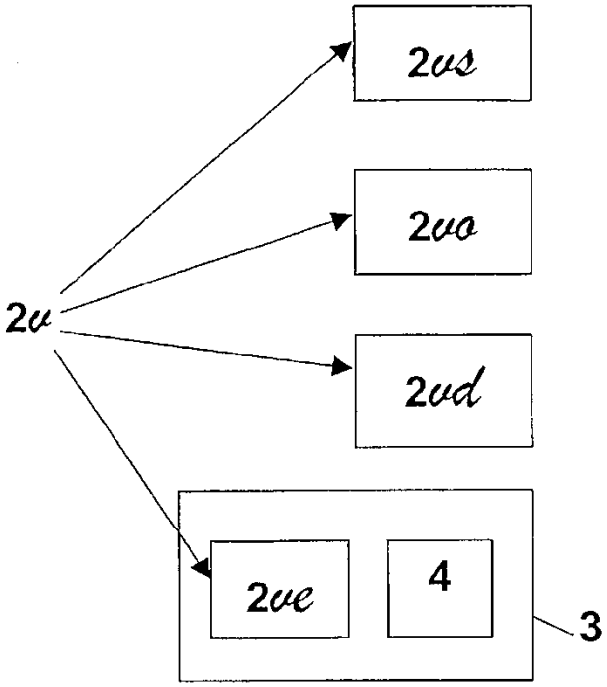

2vs
2vo
2u
2vd
2ve
4
3

FIG. 11

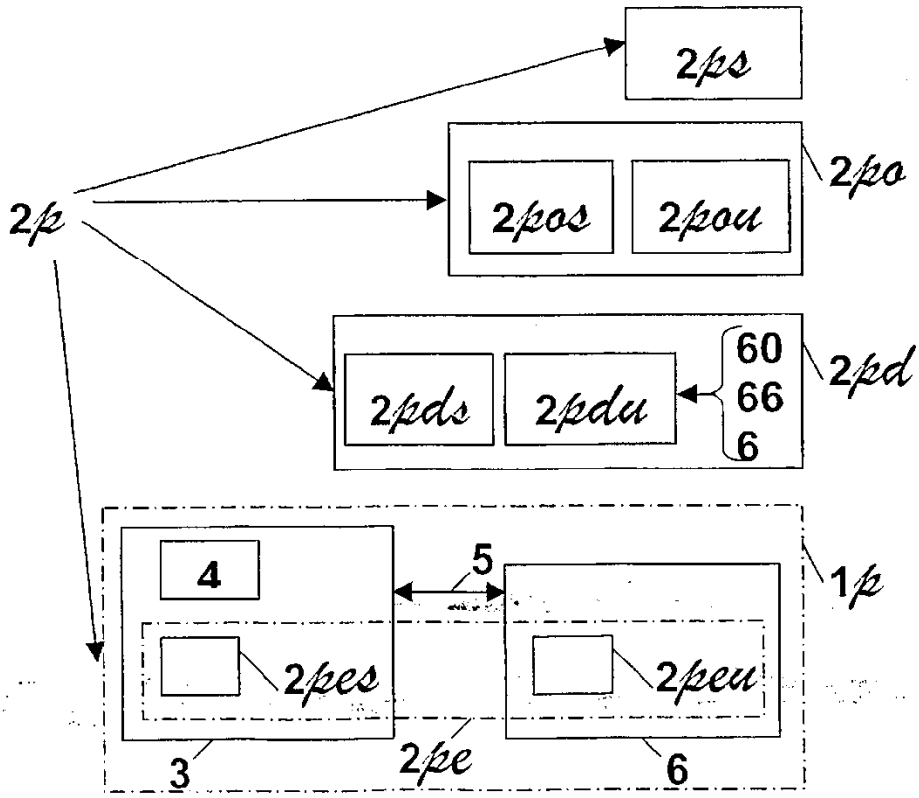

2ps
2p
2pos
2pou
2po
2pds
2pdu
60
66
6
2pd
4
5
1p
2pes
2peu
3
2pe
6

FIG. 20

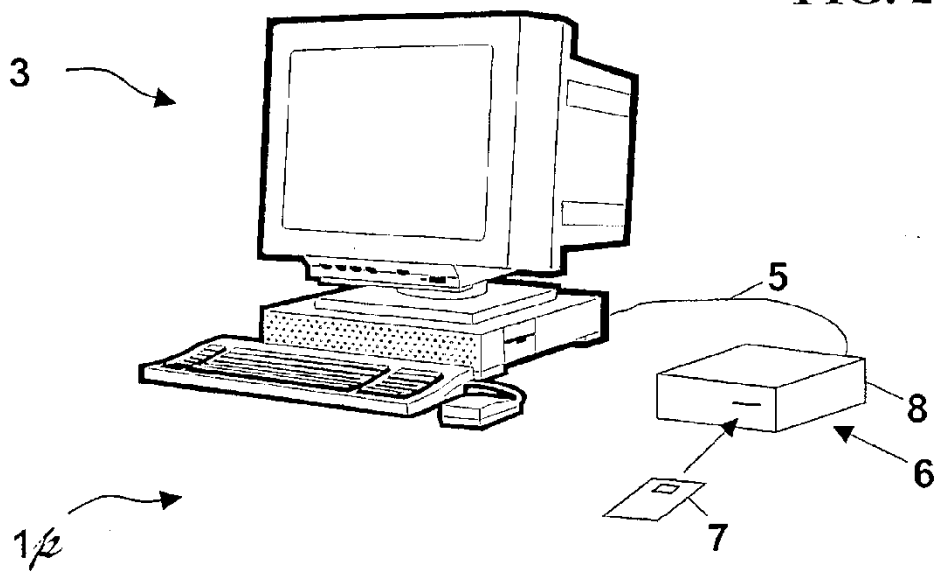


FIG. 21

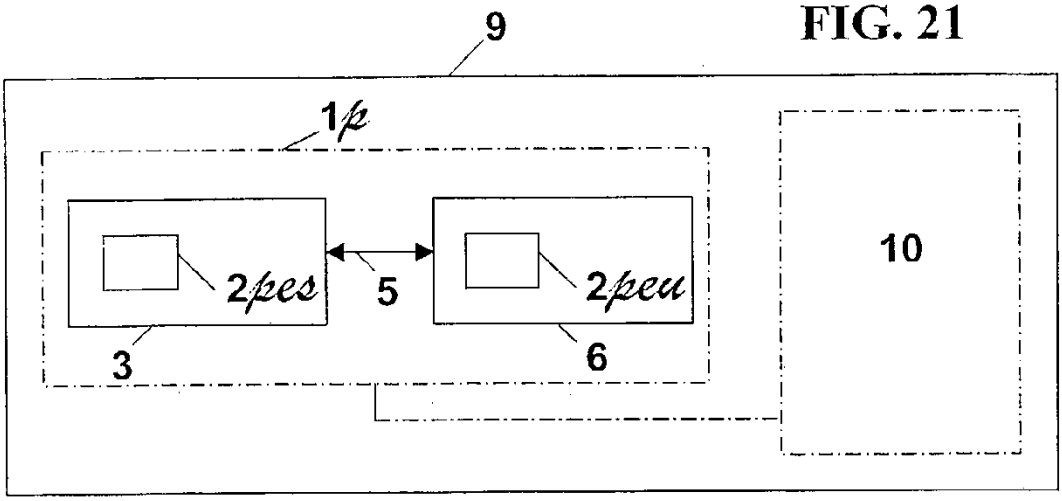


FIG. 22

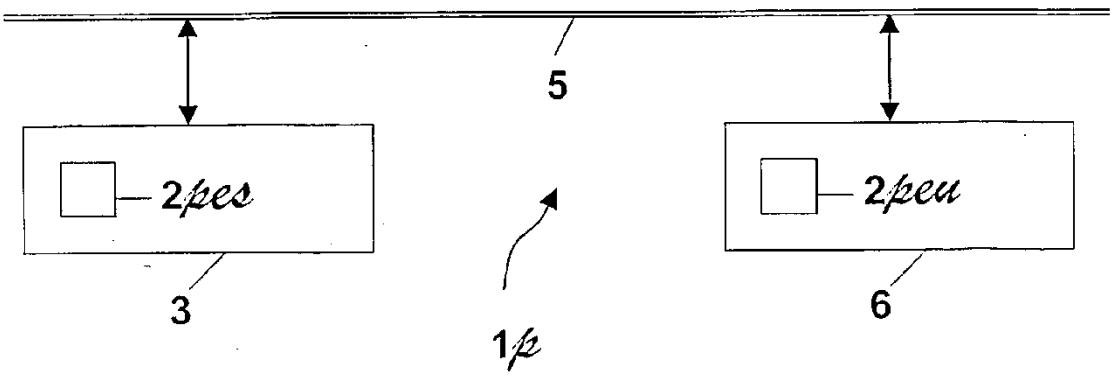


FIG. 30

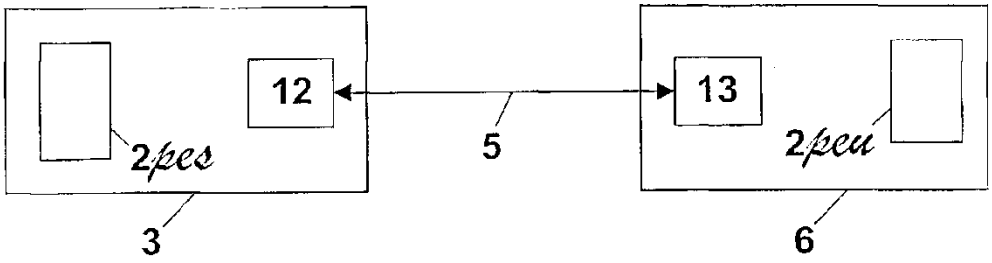


FIG. 31

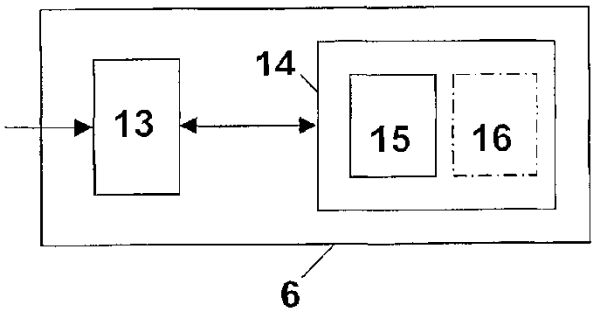


FIG. 40

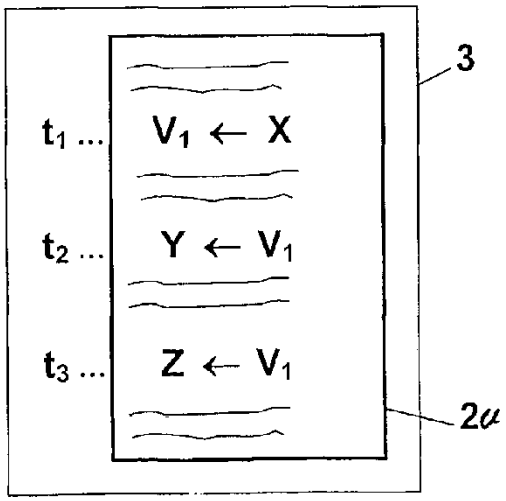


FIG. 41

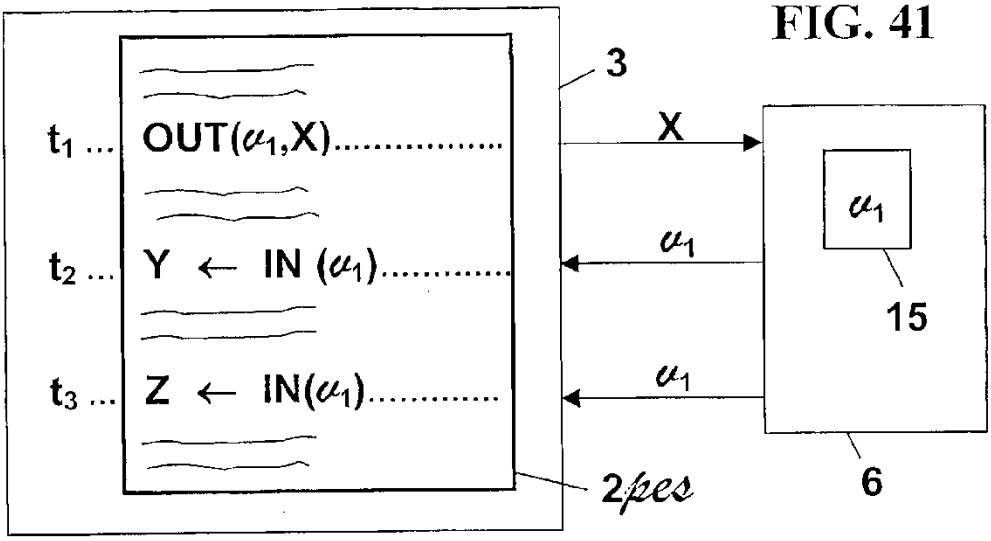


FIG. 42

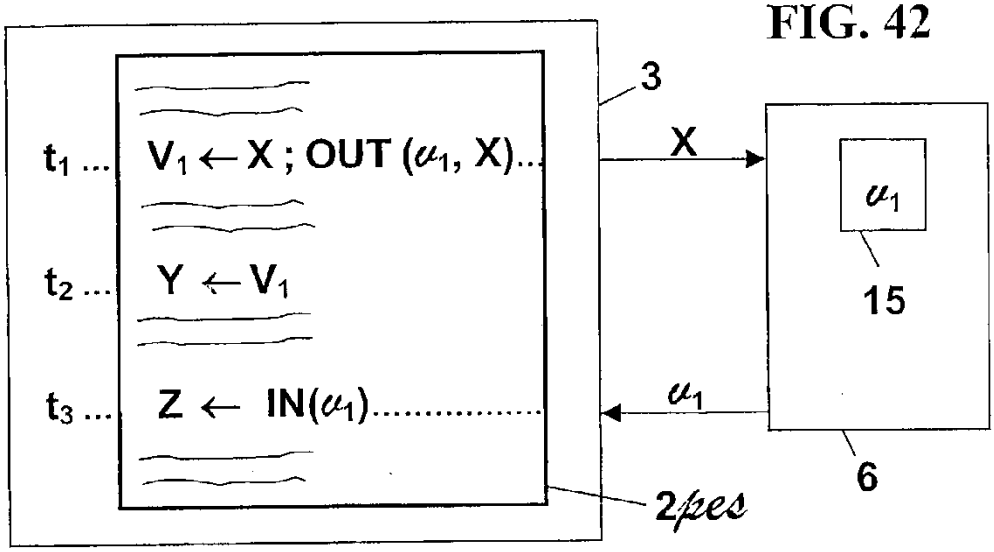


FIG. 43

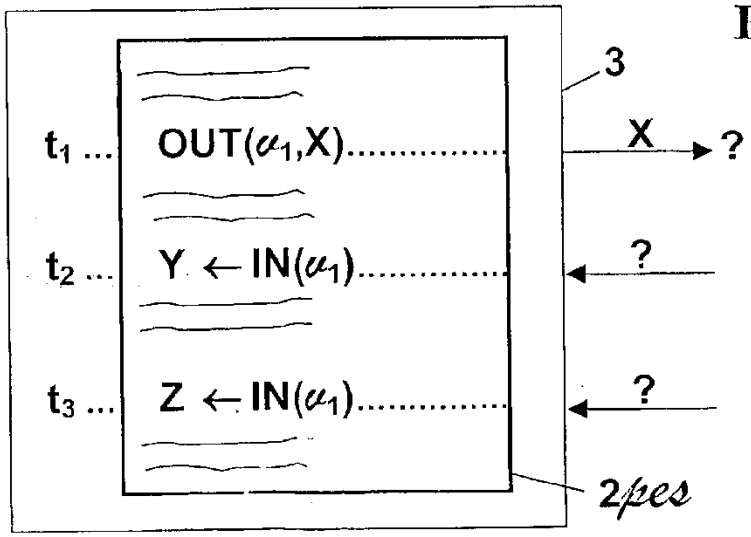


FIG. 70

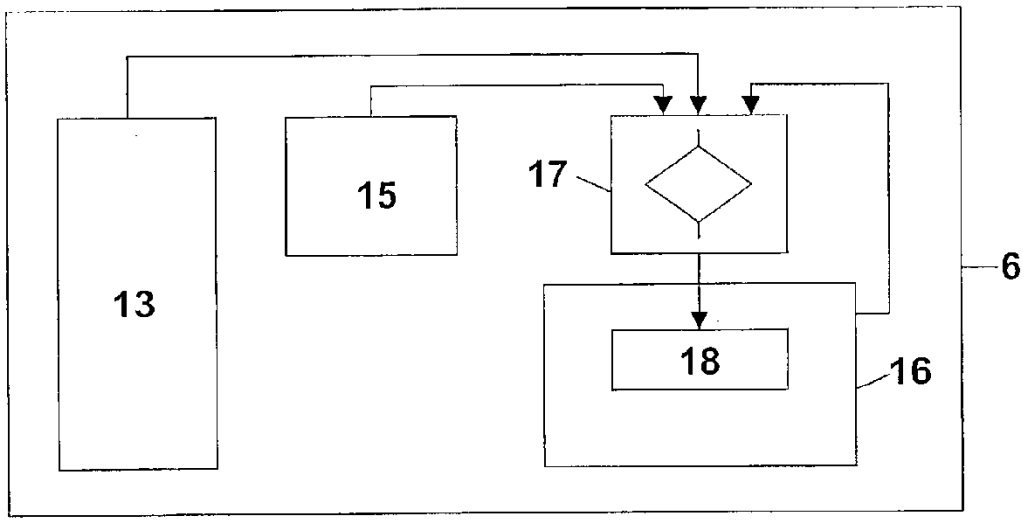

13
15
17
18
16
6

FIG. 71

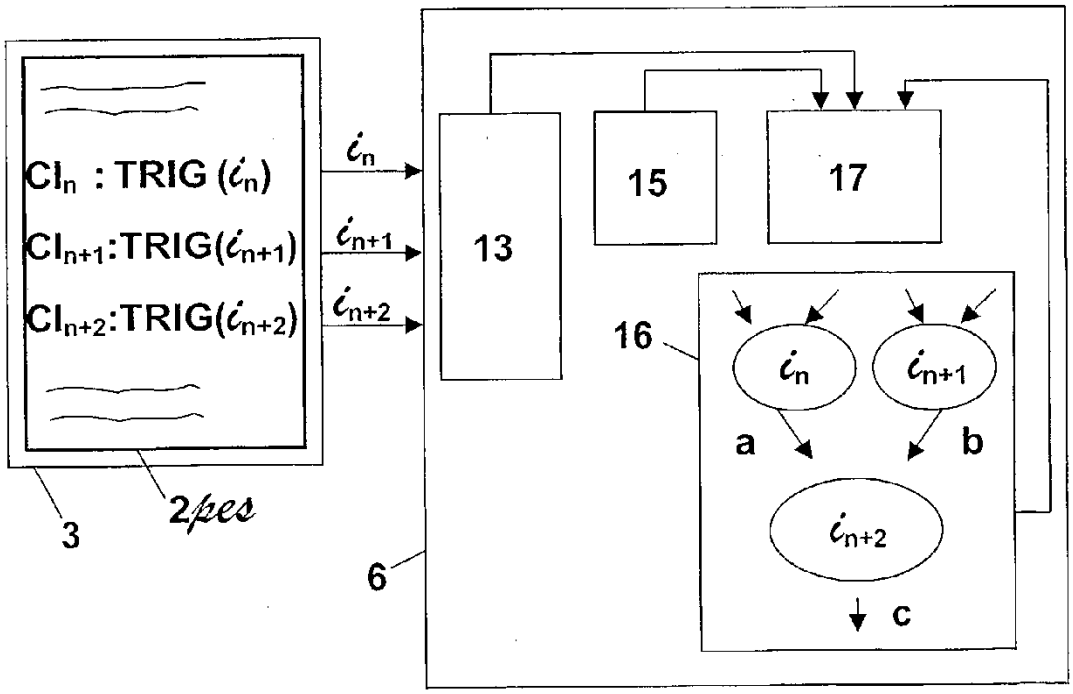

CI_n : TRIG (i_n)
CI_{n+1}:TRIG(i_{n+1})
CI_{n+2}:TRIG(i_{n+2})
i_n
i_{n+1}
i_{n+2}
13
15
17
16
a
b
c
3
2pes
6

FIG. 72

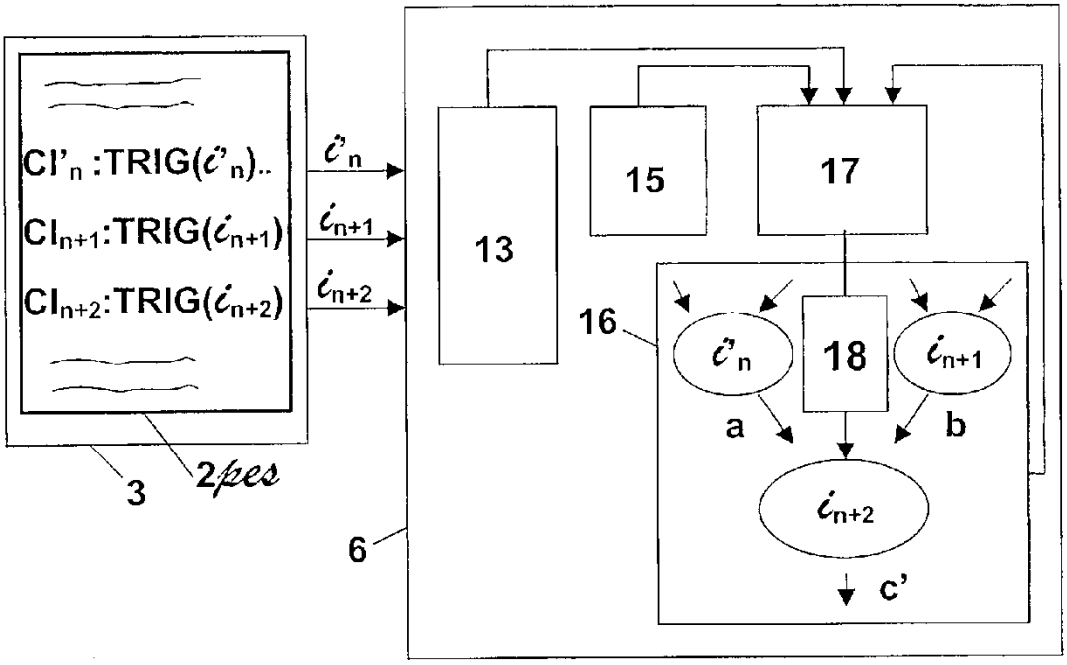


FIG. 73

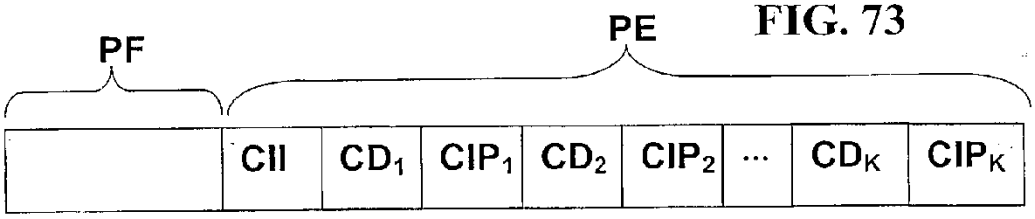


FIG. 74

FIG. 80

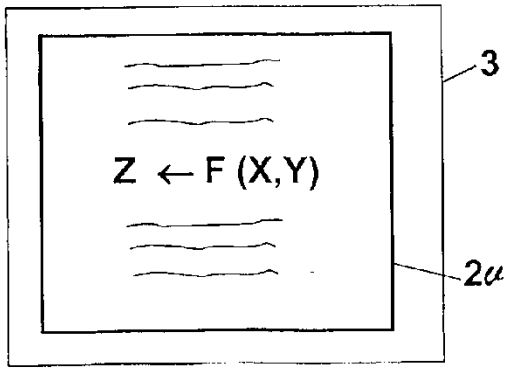


FIG. 81

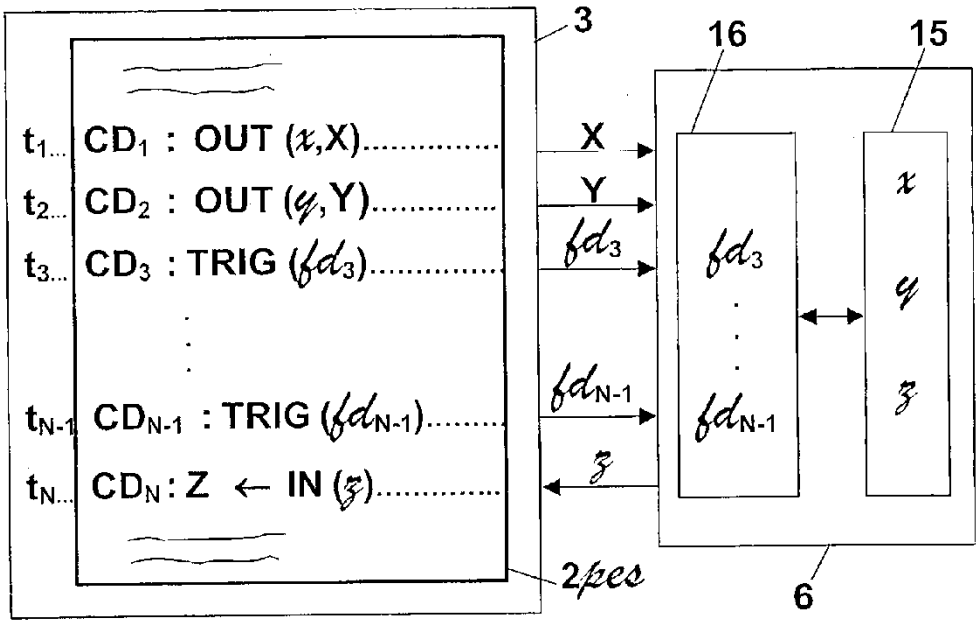
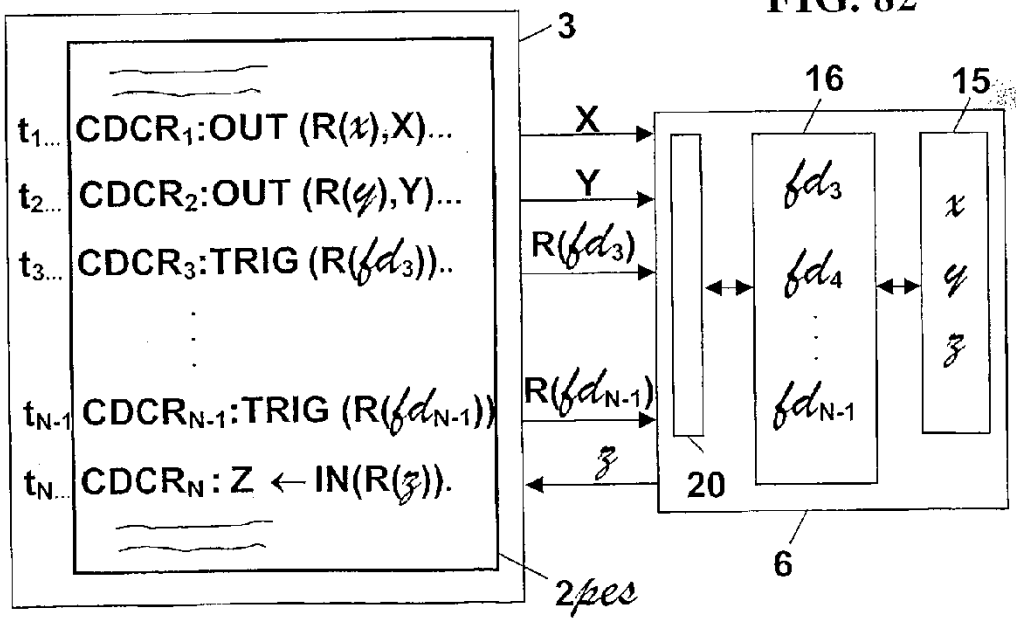

3
16
15
t1... CD1 : OUT (x,X).....
t2... CD2 : OUT (y,Y).....
t3... CD3 : TRIG (fd3).....
tN-1 CDN-1 : TRIG (fdN-1).....
tN... CDN : Z ← IN (z).....
X
Y
fd3
fdN-1
z
fd3
fdN-1
x
y
z
2pes
6

FIG. 82


3
16
15
t1... CDCR1:OUT (R(x),X)...
t2... CDCR2:OUT (R(y),Y)...
t3... CDCR3:TRIG (R(fd3))..
tN-1 CDCRN-1:TRIG (R(fdN-1))
tN... CDCRN : Z ← IN(R(z)).
X
Y
R(fd3)
R(fdN-1)
z
fd3
fd4
fdN-1
x
y
z
20
2pes
6

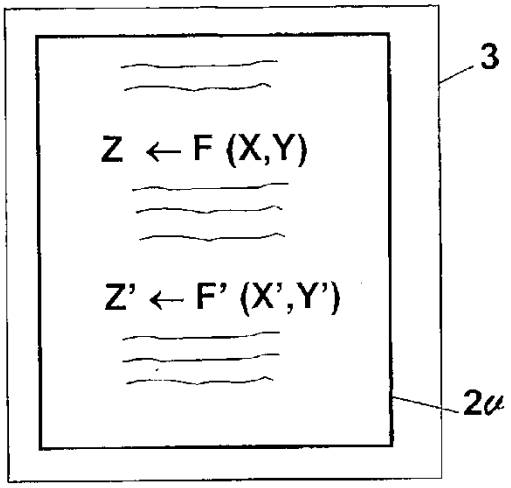

Z ← F (X,Y)
Z' ← F' (X',Y')
3
2u

FIG. 83

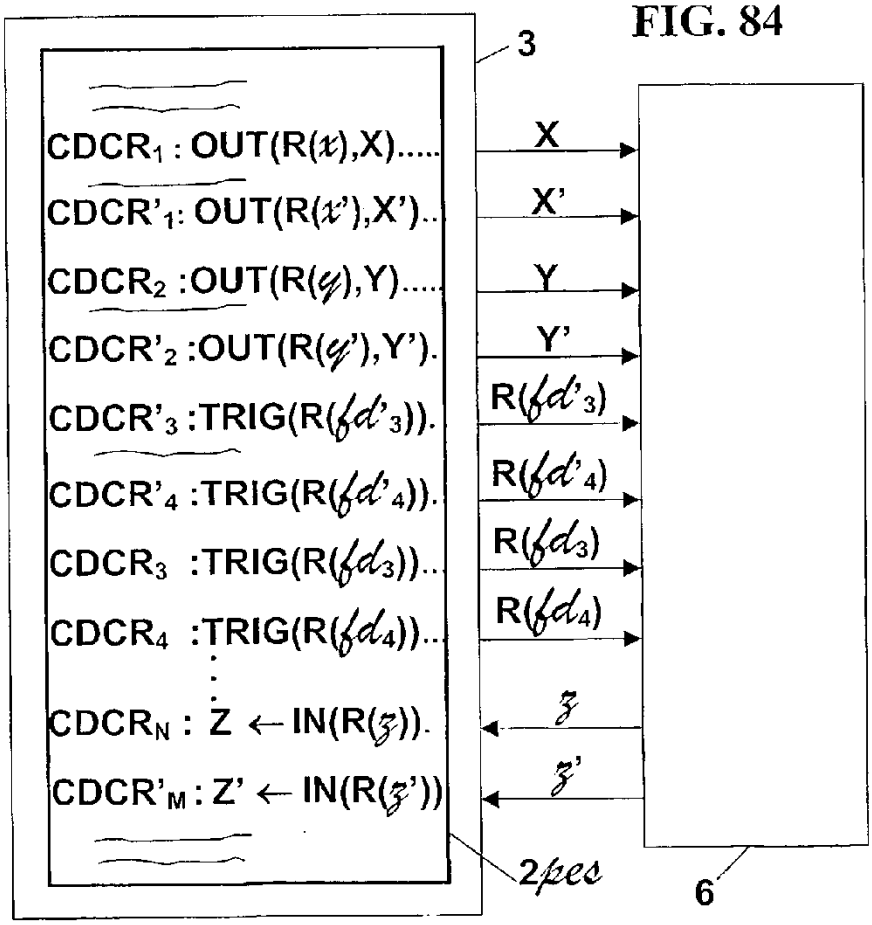

CDCR1 : OUT(R(x),X).....
CDCR'1: OUT(R(x'),X')..
CDCR2 :OUT(R(y),Y).....
CDCR'2 :OUT(R(y'),Y').
CDCR'3 :TRIG(R(fd'3)).
CDCR'4 :TRIG(R(fd'4)).
CDCR3 :TRIG(R(fd3))..
CDCR4 :TRIG(R(fd4))..
CDCRN : Z ← IN(R(z)).
CDCR'M : Z' ← IN(R(z'))
X
X'
Y
Y'
R(fd'3)
R(fd'4)
R(fd3)
R(fd4)
z
z'
3
2pes
6

FIG. 84

FIG. 85

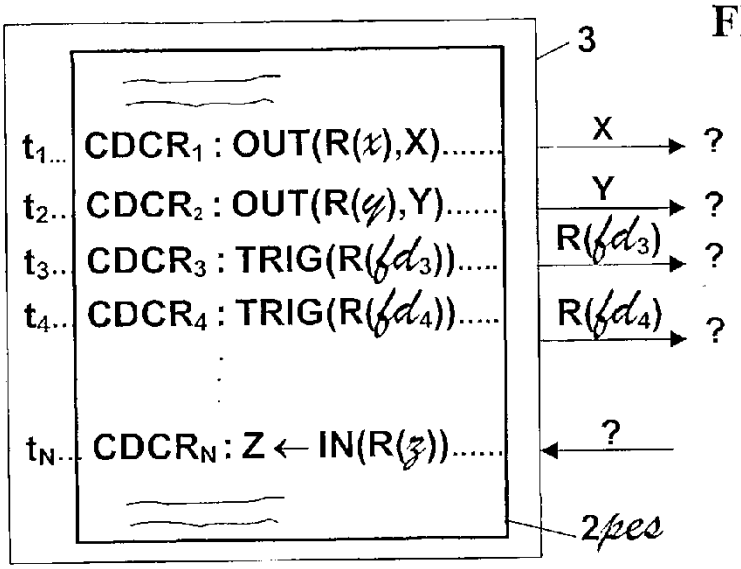


FIG. 90

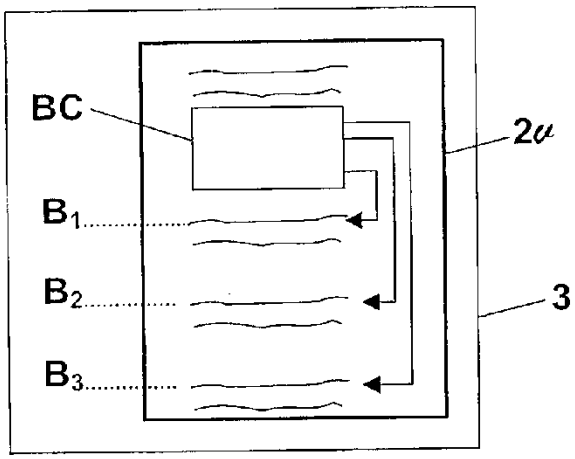


FIG. 91

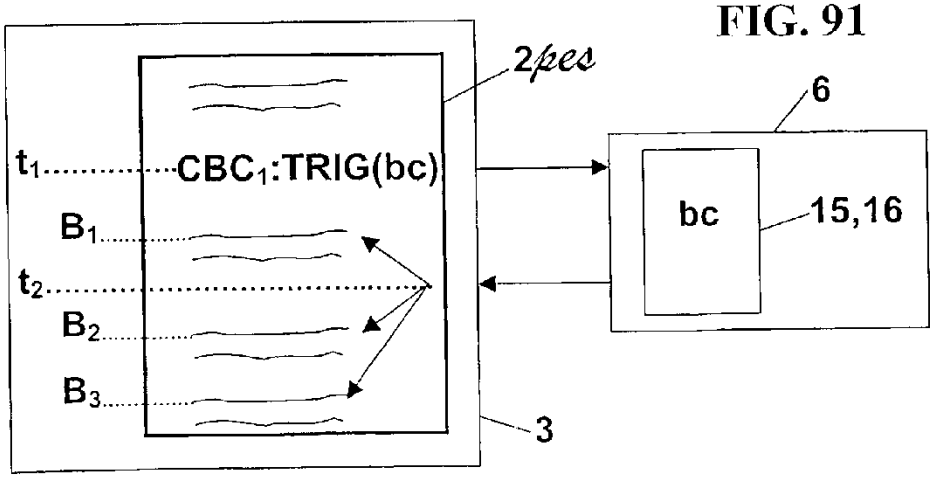


FIG. 92

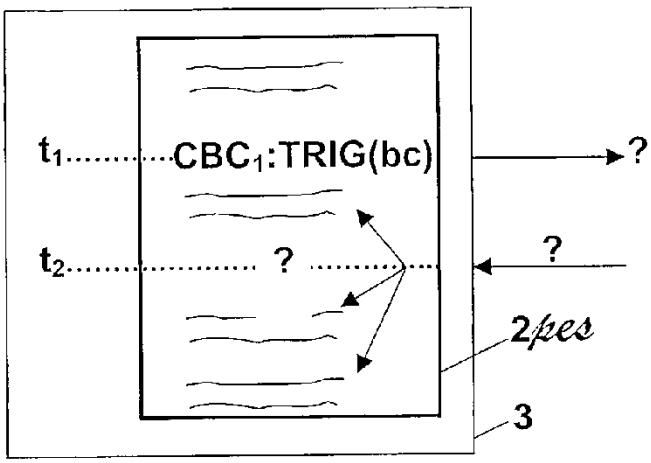


FIG. 100

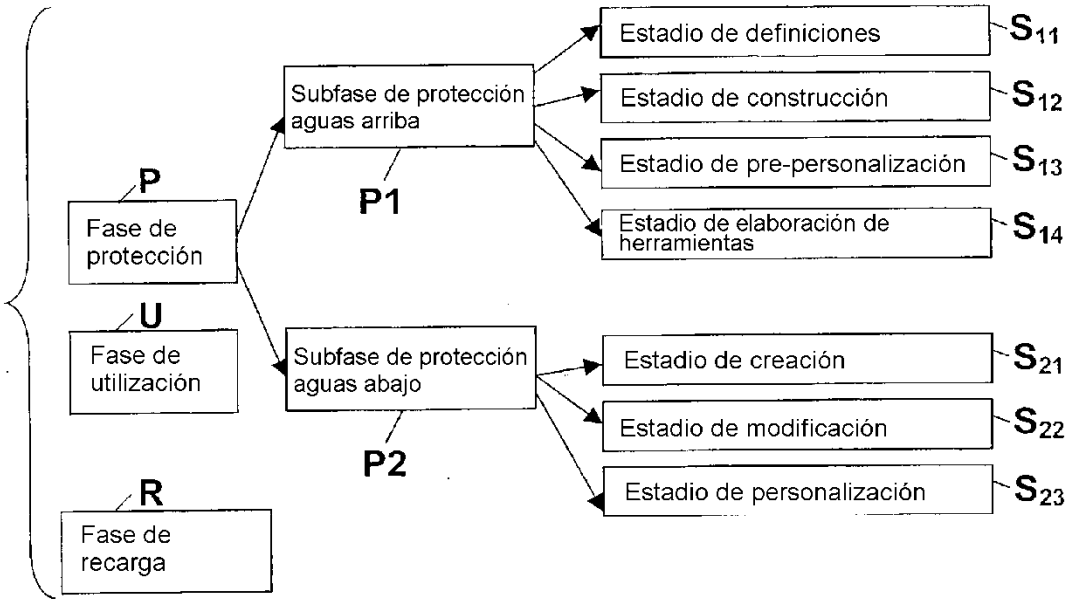


FIG. 110

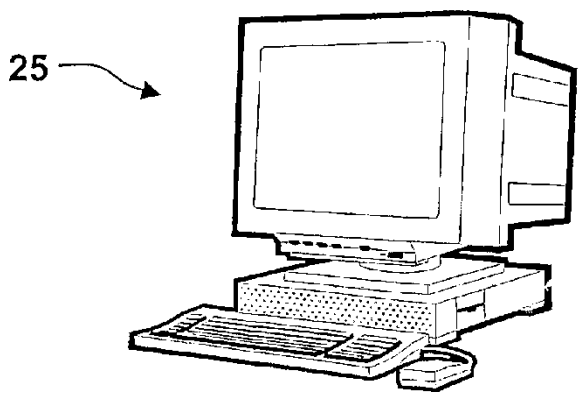


FIG. 120

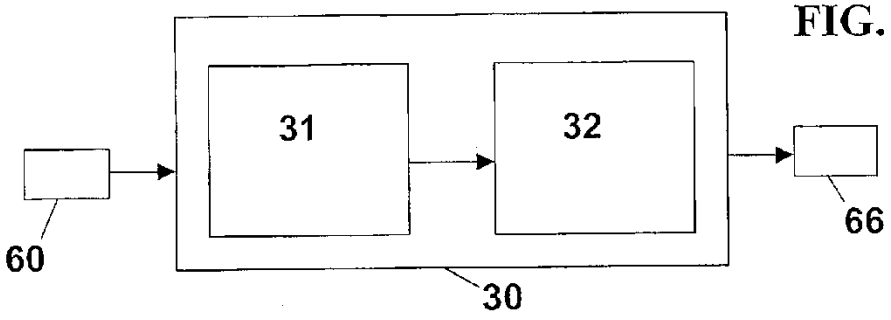


FIG. 130

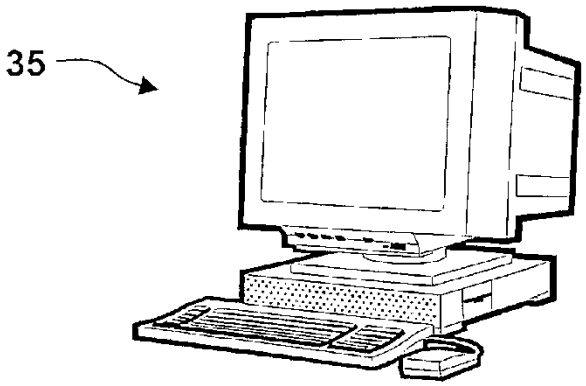


FIG. 140

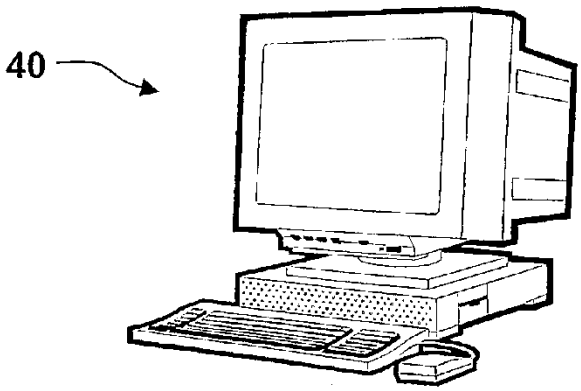


FIG. 150