(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 973 819**

(51) Int. Cl.:

G06F 21/52 (2013.01)

G06F 21/54 (2013.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **11.12.2020 PCT/FI2020/050833**

(87) Fecha y número de publicación internacional: **15.07.2021 WO21140268**

(96) Fecha de presentación y número de la solicitud europea: **11.12.2020 E 20828029 (7)**

(97) Fecha y número de publicación de la concesión europea: **21.02.2024 EP 4088207**

(54) Título: **Método y sistema para la detección de la manipulación de un código ejecutable**

(30) Prioridad:

07.01.2020 US 202016736167

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
24.06.2024

(73) Titular/es:

SUPERCELL OY (100.0%)
Jätkäsaarenlaituri 1
00180 Helsinki, FI

(72) Inventor/es:

IMMONEN, AKI

(74) Agente/Representante:

DEL VALLE VALIENTE, Sonia

DESCRIPCIÓN

Método y sistema para la detección de la manipulación de un código ejecutable

Campo técnico

La presente descripción se refiere generalmente a evitar la modificación no autorizada de código ejecutable; y más específicamente, a un método para la detección de manipulación en códigos ejecutables. Además, la presente descripción también se refiere a un sistema para la detección de manipulación en códigos ejecutables. Además, la presente descripción también se refiere a productos de programas informáticos para realizar dicho procedimiento.

Antecedentes

Con los desarrollos en la tecnología, se han desarrollado varios programas de software para una variedad de aplicaciones tales como juegos, compras online, aprendizaje online y similares. Sin embargo, estos programas de software a menudo son propensos a un acceso no autorizado que da como resultado alteraciones en el programa de software. En un ejemplo, los videojuegos multijugador son propensos a ataques por usuarios no autorizados de manera que el usuario no autorizado intenta obtener una ventaja sobre otros usuarios manipulando el programa de software mediante acceso no autorizado.

Además, los usuarios no autorizados introducen funciones maliciosas tales como funciones de *hooking* dentro de los programas de software. Como resultado, se redirige el flujo de ejecución del programa de software y, por lo tanto, el programa de software sigue instrucciones de los usuarios no autorizados. Las funciones maliciosas son las funciones del programa de software que se han manipulado por el usuario no autorizado. En un ejemplo, las funciones maliciosas hacen que el programa de software acceda a las bibliotecas proporcionadas por el usuario no autorizado. En otro ejemplo, las funciones maliciosas permiten al usuario no autorizado manipular un número de municiones con el usuario no autorizado en un videojuego.

Las funciones maliciosas proporcionan una ventaja indebida a los usuarios no autorizados. Además, las funciones maliciosas hacen que el programa de software propenso al robo de datos, de modo que las funciones maliciosas permitan que los usuarios no autorizados accedan y extraigan datos confidenciales (como datos relacionados con otros usuarios que acceden al programa de software). Además, los usuarios no autorizados venden códigos tramposos de las funciones maliciosas a otros usuarios. Además, los usuarios no autorizados también buscan obtener beneficios tales como el secuestro mediante el uso de las funciones maliciosas. Además, al introducir funciones maliciosas en los programas de software a los que se accede por otros usuarios, los usuarios no autorizados aumentan la carga computacional en un procesador empleado por los otros usuarios para acceder a los programas de software. Como resultado, la velocidad de procesamiento del procesador se reduce y las funciones de los programas de software se reducen. Además, las funciones maliciosas también permiten que los usuarios no autorizados controlen de forma remota el funcionamiento de los programas de software utilizados por otros usuarios. Convencionalmente, para superar el acceso no autorizado, se utilizan técnicas de coincidencia de patrones. Dichas técnicas de coincidencia de patrones almacenan información conocida relacionada con patrones específicos o secuencias asociadas con funciones maliciosas. Las técnicas de coincidencia de patrones identifican funciones maliciosas en el programa de software identificando los patrones o secuencias en base a la información conocida. Sin embargo, tales técnicas son ineficaces ya que las funciones maliciosas pueden implicar un patrón desconocido o puede implicar una variación marginal en el patrón conocido que hace que la función malintencionada sea indetectable.

El documento US 2015/09562A1 describe un método para detectar un ataque de programación orientado a retorno contra un dispositivo informático verificando una diana de dirección de ramificación durante la ejecución de una rutina.

El documento US 2015/213260 A1 describe un método para evitar la ejecución de un código malicioso mediante la detección de acceso erróneo o ejecución de código en un área completa de memoria.

Por lo tanto, en vista de la explicación anterior, existe la necesidad de superar los inconvenientes mencionados anteriormente asociados con la detección de funciones maliciosas en los programas informáticos.

Resumen

La presente descripción busca proporcionar un método para la detección de manipulación en un código ejecutable que comprenda uno o más bloques de código. La presente descripción también busca proporcionar un sistema para la detección de manipulación de un código ejecutable. La presente descripción busca además proporcionar un producto de programa informático para realizar dicho método. La presente descripción busca proporcionar una solución al problema existente de usuarios no autorizados que buscan una ventaja sobre otros usuarios manipulando el código ejecutable. Un objetivo de la presente descripción es proporcionar una solución que supere, al menos parcialmente, los problemas encontrados en la técnica anterior, y proporcione la detección de manipulación en el código ejecutable que sea aún anteriormente desconocido y evite cualquier usuario del código ejecutable a partir de buscar ventajas indebidas.

En un aspecto, una realización de la presente descripción proporciona un método para la detección de manipulación en un código ejecutable que comprende uno o más bloques de código, que comprende:

- monitorizar la ejecución del código ejecutable con una estructura de datos de pila de llamadas asociada con el mismo, implicando la ejecución acceder a uno o más espacios de dirección;

- recibir información sobre uno o más espacios de dirección accedidos;

- comparar la información recibida sobre uno o más espacios de dirección accedidos con información sobre uno o más espacios de dirección permitidos definidos en la estructura de datos de pila de llamadas del código ejecutable;

- activar una alerta tras la detección de que uno o más espacios de dirección accedidos son diferentes del uno o más espacios de dirección permitidos, en base a la comparación; y

- ejecutar una acción basada en la alerta; y

- añadir la alerta como una variable global a cada uno de los uno o más bloques de código ejecutable, en donde la alerta de cada uno de los uno o más bloques de código tiene un valor falso, y un valor de la alerta de un bloque de código dado se cambia a un valor verdadero cuando un espacio de dirección accedido del bloque de código dado es diferente del uno o más espacios de dirección permitidos; además en donde si el valor de la alerta del bloque de código dado es un valor falso, un espacio de dirección accedido de uno o más bloques de código posteriores no se compara con el uno o más espacios de dirección permitidos.

En otro aspecto, una realización de la presente descripción proporciona un sistema para la detección de manipulación en un código ejecutable, que comprende:

- una memoria configurada para almacenar el código ejecutable que comprende uno o más bloques de código y una estructura de datos de pila de llamadas asociada con el mismo, y un código de detección de manipulación; y

- un procesador configurado para ejecutar el código de detección de manipulación:

- monitorizar la ejecución del código ejecutable, implicando la ejecución acceder a uno o más espacios de dirección;

- recibir información acerca del uno o más espacios de dirección accedidos;

- comparar la información recibida sobre uno o más espacios de dirección accedidos con información sobre uno o más espacios de dirección permitidos definidos en la estructura de datos de pila de llamadas del código ejecutable;

- activar una alerta tras la detección de que uno o más espacios de dirección accedidos son diferentes del uno o más espacios de dirección permitidos, 15 en base a la comparación; y

- ejecutar una acción basada en la alerta; y

- añadir la alerta como una variable global a cada uno de los uno o más bloques de código ejecutable, en donde la alerta de cada uno de los uno o más bloques de código tiene un valor falso, y un valor de la alerta de un bloque de código dado se cambia a un valor verdadero cuando un espacio de dirección accedido del bloque de código dado es diferente del uno o más espacios de dirección permitidos, además en donde si el valor de la alerta del bloque de código dado es un valor falso, un espacio de dirección accedido de uno o más bloques de código posteriores no se compara con el uno o más espacios de dirección permitidos.

En otro aspecto más, una realización de la presente descripción proporciona un producto de programa informático, que comprende un código de detección de manipulación legible por ordenador que, cuando se ejecuta en un ordenador, es dispuesto para hacer que el ordenador realice el procedimiento para la detección de manipulación en un código ejecutable que comprende uno o más bloques de código.

Las realizaciones de la presente descripción sustancialmente eliminan, o al menos parcialmente abordan los problemas mencionados anteriormente del estado de la técnica, y permiten la detección de manipulación del código ejecutable, y toma acción contra el usuario no autorizado que realiza la manipulación y evita que cualquier usuario del código ejecutable busque ventajas indebidas.

Se deducirán otros aspectos, ventajas, características y objetos de la presente descripción a partir de los dibujos y la descripción detallada de las realizaciones ilustrativas interpretadas junto con las reivindicaciones anexas que siguen.

Se apreciará que las características de la presente descripción son susceptibles de unirse en varias combinaciones sin abandonar el ámbito de la presente descripción definido por las reivindicaciones anexas.

Breve descripción de los dibujos

El resumen anterior, así como la siguiente descripción detallada de realizaciones ilustrativas, se comprende mejor al leerlo junto con los dibujos anexos. Para ilustrar la presente descripción, se muestran estructuras ilustrativas de la descripción en los dibujos. Sin embargo, la presente descripción no se limita a métodos e instrumentales específicos descritos en la presente memoria. Además, los expertos en la técnica entenderán que los dibujos no están a escala. En la medida de lo posible, los elementos similares se han indicado con números idénticos.

A continuación se describirán realizaciones de la presente descripción, a modo de ejemplo únicamente, con referencia a los siguientes diagramas, en donde:

la Figura 1A es un diagrama de bloques para detectar la manipulación de un código ejecutable, según una realización de la presente descripción;

la Figura 1B es un entorno de red del sistema de la Figura 1 según una realización de la presente descripción;

la Figura 1C es un entorno de red del sistema de la Figura 1A según una realización de la presente descripción;

Figura 2 es un diagrama de bloques que representa detalles de una memoria del sistema de la Figura 1A, según una realización de la presente descripción; y

Figura 3 es un diagrama de flujo de un método para la detección de manipulación en un código ejecutable, según una realización de la presente descripción.

En los dibujos adjuntos, se emplea un número subrayado para representar un elemento sobre el que se coloca el número subrayado o un elemento al que está adyacente el número subrayado. Un número no subrayado se refiere a un elemento identificado por una línea que vincula el número no subrayado al elemento. Cuando un número no está subrayado y está acompañado por una flecha asociada, el número no subrayado se utiliza para identificar un elemento general al que la flecha apunta.

Descripción detallada de las realizaciones

La siguiente descripción detallada ilustra realizaciones de la presente descripción y formas en las que pueden implementarse. Aunque se han descrito algunos modos de realización de la presente descripción, los expertos en la técnica reconocerán que también son posibles otras realizaciones para llevar a cabo o poner en práctica la presente descripción.

En un aspecto, una realización de la presente descripción proporciona un método para la detección de manipulación en un código ejecutable que comprende uno o más bloques de código, que comprende:

- monitorizar la ejecución del código ejecutable con una estructura de datos de pila de llamadas asociada con el mismo, implicando la ejecución acceder a uno o más espacios de dirección;

- recibir información sobre uno o más espacios de dirección, según se acceden;

- comparar la información recibida sobre uno o más espacios de dirección accedidos con información sobre uno o más espacios de dirección permitidos definidos en la estructura de datos de pila de llamadas del código ejecutable;

- activar una alerta tras la detección de que uno o más espacios de dirección accedidos son diferentes del uno o más espacios de dirección permitidos, en base a la comparación; y

- ejecutar una acción basada en la alerta activada.

En otro aspecto, una realización de la presente descripción proporciona un sistema para la detección de manipulación en un código ejecutable, que comprende:

- una memoria configurada para almacenar el código ejecutable que comprende uno o más bloques de código y una estructura de datos de pila de llamadas asociada con el mismo, y un código de detección de manipulación; y

- un procesador configurado para ejecutar el código de detección de manipulación:

- monitorizar la ejecución del código ejecutable, la ejecución que implica acceder a uno o más espacios de dirección;

- recibir información acerca del uno o más espacios de dirección accedidos;

- comparar la información recibida sobre uno o más espacios de dirección accedidos con información sobre uno o más espacios de dirección permitidos definidos en la estructura de datos de pila de llamadas del código ejecutable;

- activar una alerta tras la detección de que uno o más espacios de dirección accedidos son diferentes del uno o más espacios de dirección permitidos, en base a la comparación; y

- ejecutar una acción basada en la alerta activada.

En otro aspecto más, una realización de la presente descripción proporciona un producto de programa informático, que comprende un código de detección de manipulación de por ordenador que, cuando se ejecuta en un ordenador, está dispuesto para hacer que el ordenador realice el método para la detección de manipulación en un código ejecutable que comprende uno o más bloques de código.

En otro aspecto, una realización de la presente descripción proporciona un sistema para la detección de manipulación en un código ejecutable, que comprende:

- una memoria configurada para almacenar el código ejecutable que comprende uno o más bloques de código y una estructura de datos de pila de llamadas asociada con el mismo, y un producto de programa informático; y

- Un procesador configurado para ejecutar el código de detección de manipulación.

La presente descripción proporciona un método y un sistema que permiten la detección de la manipulación en el código ejecutable, tomando la acción contra el usuario no autorizado que realiza la manipulación y evitan que cualquier usuario de busque ventajas indebidas. El método discutido en la presente descripción no depende del patrón previamente conocido para la detección de manipulación en el código ejecutable y, por lo tanto, es más preciso para detectar la manipulación en comparación con las técnicas convencionales usadas para la detección de manipulación. El método tras la detección de manipulación en el código ejecutable está configurado para terminar la ejecución del código ejecutable en el dispositivo cliente que tiene el código ejecutable. Como resultado, el usuario no autorizado no obtiene ninguna ventaja indebida sobre otros usuarios. De forma ventajosa, el método está configurado para comprobar la manipulación del código ejecutable solo para uno o más bloques de código predeterminados. Como resultado, se reduce el espacio informático y la velocidad asociada con la detección de manipulación de código ejecutable y se mejora la experiencia del usuario. El método comprende emplear segmentos de solo lectura en el código ejecutable que mejora la detección de manipulación en el código ejecutable. El código de detección de manipulación empleado en la presente descripción también es a prueba de manipulaciones, y los usuarios no autorizados no pueden manipular el código de detección de manipulación. En un ejemplo, la detección de manipulación en el código ejecutable de un juego de estrategia de combate permite a cualquier usuario obtener ventajas indebidas.

La presente descripción comprende el método y el sistema para la detección de manipulación en el código ejecutable. A lo largo de la presente descripción, el término *"código ejecutable"* se refiere a cualquier colección o conjunto de instrucciones/comandos ejecutables por un ordenador u otro sistema digital para configurar el ordenador o el sistema digital para realizar una o más tareas que es la intención del código ejecutable. En otras palabras, el código ejecutable es un programa o una aplicación para un dispositivo cliente (es decir, un ordenador) tal como un teléfono inteligente, una tableta. Además, el código ejecutable pretende abarcar las instrucciones mencionadas anteriormente almacenadas en el medio de almacenamiento (analizado más adelante en detalle) tales como RAM, un disco duro, disco óptico, etc., y también está destinado a abarcar el denominado "firmware" que es un software almacenado en ROM u otros. Opcionalmente, el código ejecutable puede invocar código de nivel de sistema o llamadas a otro software que reside en un servidor u otra ubicación para realizar ciertas funciones. Además, el código ejecutable puede estar preconfigurado y preintegrado con un sistema operativo, construyendo un servidor de software. En un ejemplo, el código ejecutable es un juego multijugador que se ejecuta en teléfono inteligente o tableta y se conecta a un servidor de juego. En otro ejemplo, el código ejecutable está configurado para habilitar a los usuarios realizar tareas de compras y financieras, establecer comunicación con otros usuarios, recibir educación a partir de fuentes electrónicas tales como libros electrónicos y similares.

El código ejecutable comprende uno o más bloques de código. El término *"bloque de código"* como se utiliza en la presente memoria, se refiere a un módulo del código ejecutable que está configurado para realizar una tarea específica de una o más tareas previstas del código ejecutable. Generalmente, el bloque de código está configurado para recibir una entrada, procesar la entrada y devolver un resultado como una salida basada en el procesamiento de la entrada. Cada uno de los uno o más bloques de código está asociado con un espacio de direcciones. En un ejemplo, en un juego de estrategia de combate, un primer bloque de código se configura para realizar una tarea de mover un avatar del juego en una dirección de avance, un segundo bloque de código se configura para realizar una tarea de mover el avatar en una dirección hacia atrás, un tercer bloque de código se configura para realizar una tarea de mover el avatar en una dirección hacia la izquierda y un cuarto bloque de código se configura para realizar una tarea de mover el avatar en una dirección hacia la derecha.

El término *"manipulación"* utilizado en la presente descripción se refiere al acceso no autorizado por usuarios no autorizados que dan como resultado modificaciones en el código ejecutable. El acceso no autorizado permite la adición de códigos nuevos y fraudulentos al código ejecutable de manera que el código ejecutable no realice la tarea prevista del código ejecutable. Las modificaciones no autorizadas se realizan en el código ejecutable de manera que los usuarios no autorizados obtienen un beneficio sobre otros usuarios utilizando el código ejecutable. Los usuarios no autorizados realizan una función de hooking para manipular la ejecución del código ejecutable. El hooking de función se implementa modificando una pequeña pieza de código del código ejecutable, generalmente al comienzo de un bloque de código, para redirigir el flujo de ejecución del código ejecutable a una biblioteca proporcionada por el usuario no autorizado. La presente descripción detecta la manipulación del código ejecutable para identificar los usuarios no autorizados que realizan la manipulación, evitar que los usuarios no autorizados realicen tal manipulación en el futuro, protegen a otros usuarios de los efectos de la manipulación realizada por usuarios no autorizados y, por lo tanto, proporcionan un entorno seguro y de igualdad para todos los usuarios que acceden al código ejecutable. En un ejemplo, en un juego de estrategia de combate, un usuario no autorizado puede realizar la manipulación del código ejecutable para aumentar una cantidad de munición para un avatar del usuario no autorizado. En tal caso, el usuario no autorizado obtiene un beneficio sobre otros usuarios.

El sistema comprende la memoria y el procesador. El término *"memoria"* como se usa en la presente memoria se refiere a un medio volátil o persistente, tal como un circuito eléctrico, disco magnético, memoria virtual o disco óptico, en el que los datos o software se almacenan durante cualquier duración. Opcionalmente, la memoria es una unidad de almacenamiento masivo no volátil, tal como medios de almacenamiento físico. Opcionalmente, la memoria está conectada a un servidor para recibir los datos o software mencionados anteriormente. El término *"procesador"* como se usa en la presente descripción, se refiere a un elemento informático que es operable para responder y procesar instrucciones que ejecutan el método. Opcionalmente, el procesador incluye, pero no se limita a, un microprocesador, un microcontrolador, un microprocesador de cálculo de conjunto de instrucciones complejas (CISC), un microprocesador de conjunto de instrucciones reducidas (RISC), un microprocesador de palabra de instrucción muy larga (VLIW) o cualquier otro tipo de circuito de procesamiento. Además, el término *"procesador"* puede referirse a uno o más procesadores, dispositivos de procesamiento y diversos elementos individuales asociados con un dispositivo de procesamiento que puede ser compartido por otros dispositivos de procesamiento. Además, el uno o más procesadores individuales, dispositivos de procesamiento y elementos están dispuestos en diversas arquitecturas para responder a y procesar las instrucciones que accionan el método. La memoria y el procesador son parte del dispositivo cliente.

La memoria se configura para almacenar el código ejecutable que comprende uno o más bloques de código y la estructura de datos de pila de llamadas asociada con el mismo, y el código de detección de manipulación. Opcionalmente, la memoria se configura para recibir el código ejecutable, la estructura de datos de la pila de llamadas y el código de detección de manipulación del servidor. El término *"estructura de datos de pila de llamadas"* como se usa en la presente memoria, se refiere a una estructura de datos que está configurada para almacenar información relacionada con una secuencia de ejecución del uno o más de los bloques de código. Opcionalmente, la estructura de datos de pila de llamadas permite determinar un bloque de código cuya salida debe usarse como una entrada para el bloque del código posterior. Además, la estructura de datos de la pila de llamadas comprende además el espacio de direcciones asociado con cada uno de los bloques de código. En un ejemplo, la estructura de datos de la pila de llamadas comprende una secuencia de un primer bloque de código, un segundo bloque de código y un tercer bloque de código de una manera que una salida producida por el primer bloque de código se proporciona como una entrada al tercer bloque de código y además una salida producida por el tercer bloque de código se proporciona como entrada al segundo bloque de código. El término *"código de detección de manipulación"* como se usa en la presente memoria, se refiere a una colección o un conjunto de instrucciones que están configuradas para detectar la manipulación realizada por usuarios no autorizados en el código ejecutable identificando variaciones en los espacios de dirección asociados con uno o más bloques de código. El código de detección de manipulación está configurado para realizar la comprobación de tiempo de ejecución de la estructura de datos de pila de llamadas. El código de detección de manipulación valida todos los espacios de direcciones asociados con uno o más bloques de código. Si todos los espacios de dirección están dentro del código ejecutable, entonces no hay manipulación en el código ejecutable como un código ejecutable manipulado tiene uno o más espacios de dirección dirigidos al exterior del código ejecutable.

El código de detección de manipulación se configura para monitorizar la ejecución del código ejecutable, recibir información sobre los espacios de direcciones del uno o más bloques de código, comparar la información recibida sobre los espacios de dirección accedidos con información acerca de los espacios de dirección permitidos, activar una alerta tras la detección de que los espacios de dirección accedidos son diferentes de los espacios de dirección permitidos, en base a la comparación; y ejecutar una acción basada en la alerta activada.

En una realización, el código de detección de manipulación se añade en línea al código ejecutable. Opcionalmente, el código de detección de manipulación se añade en línea al código ejecutable antes de uno o más bloques de código del código ejecutable. En una realización, el código de detección de manipulación es una función denominada para la ejecución del código ejecutable. En otras palabras, el código de detección de manipulación está separado del código ejecutable. La función asociada con el código de detección de manipulación se denomina para verificar el espacio de direcciones asociado con uno o más bloques de código. Opcionalmente, el código de detección de manipulación puede implementarse en una o más posiciones en el código ejecutable. En una

realización, el código de detección de manipulación es una función de puntero de pila. Opcionalmente, la función de puntero de pila representa una parte superior de la estructura de datos de pila de llamadas. Opcionalmente, cuando se envían datos en la estructura de datos de la pila de llamadas, se copiará en la memoria donde la función del puntero de la pila apunta y se incrementa la función puntera del apilamiento. Sin embargo, cuando se presentan datos de la estructura de datos de pila de llamadas, la función de puntero de pila se reduce pero los datos permanecen en la estructura de datos de pila de llamadas y los datos pueden leerse hasta que algunos otros datos sean empujados a la estructura de datos de pila de llamadas.

El método comprende monitorizar la ejecución del código ejecutable con la estructura de datos de pila de llamadas asociada con el mismo. En la presente memoria, la ejecución implica acceder a uno o más espacios de dirección. El procesador se configura para ejecutar el código de detección de manipulación para monitorizar la ejecución del código ejecutable. La ejecución del código ejecutable comprende ejecutar el uno o más bloques de código en una secuencia o un orden de modo que cada una de las una o más tareas previstas del código ejecutable se logra. La estructura de datos de la pila de llamadas proporciona la secuencia de ejecución del uno o más bloques de código. Opcionalmente, el método para monitorizar la ejecución del código ejecutable se realiza repetidamente por cada período de tiempo predefinido. En un ejemplo, un usuario no autorizado puede manipular el uno o más bloques de código después del código de detección de manipulación ha monitorizado la ejecución del uno o más bloques de código. En tal caso, monitorizar repetidamente la ejecución del código ejecutable permite detectar la manipulación del código ejecutable. Opcionalmente, el método para monitorizar la ejecución del código ejecutable se realiza cada vez que el usuario reinicia la aplicación de software con la que el código ejecutable está asociado. En un ejemplo, un usuario no autorizado puede manipular el uno o más bloques de código después de reiniciar el software cuando el código de detección de manipulación ha monitorizado la ejecución del uno o más bloques de código mientras ejecuta el software por primera vez. En tal caso, el método es capaz de detectar la manipulación en el código ejecutable incluso después del reinicio del software. En un ejemplo, la ejecución del código ejecutable se controla en un juego multijugador para evitar que el jugador tenga una ventaja sobre otros jugadores. En otro ejemplo, la ejecución del código ejecutable se controla en un juego de estrategia para evitar que el haqueo por parte de un jugador le otorgue una ventaja indebida sobre otros jugadores que compiten contra el jugador.

En una realización, el método comprende además monitorizar la ejecución de cada uno de los uno o más bloques de código del código ejecutable. Opcionalmente, cada uno de los uno o más bloques de código se monitoriza mientras el uno o más bloques de código se ejecutan en base a su secuencia de ejecución como en la estructura de datos de pila de llamadas. En un ejemplo, en el código ejecutable de un juego de estrategia de combate, una secuencia de ejecución de uno o más bloques de código incluye un primer bloque de código para proporcionar instrucciones a un avatar en el juego para moverse hacia adelante, un segundo bloque de código para proporcionar instrucciones al avatar para detenerse en un almacén de municiones, un tercer bloque de código para proporcionar instrucciones al avatar para recoger una munición del almacenamiento de munición y un cuarto bloque de código para proporcionar instrucciones al avatar para disparar la munición. En tal caso, el método se configura para monitorizar cada uno del primer, segundo, tercer y cuarto bloque de código. En una realización, el procesador está configurado para ejecutar el código de detección de manipulación para monitorizar la ejecución de cada uno de los uno o más bloques de código del código ejecutable.

En una realización, el método comprende además monitorizar la ejecución de uno o más bloques de código predeterminados del código ejecutable. Opcionalmente, la memoria se configura para recibir información desde el servidor alrededor de uno o más bloques de código predeterminados que son bloques de código reasignados que son tales bloques de código si se manipulan, pueden proporcionar beneficios al usuario no autorizado. Se añade el código de detección de manipulación únicamente a dichos bloques de código de reasignación. Opcionalmente, la memoria está configurada para almacenar información relacionada con uno o más bloques de código predeterminados que anteriormente han sido manipulado por otros usuarios no autorizados. Opcionalmente, la memoria recibe la información mencionada anteriormente del servidor en un momento de recibir el código ejecutable. El código de detección de manipulación únicamente se añade a uno o más bloques de códigos identificados.

En un ejemplo, en un juego de estrategia de combate, uno o más bloques de código predeterminados pueden incluir un bloque de código para un número de munición disponible con un avatar, un bloque de código para una cantidad de dinero virtual asociado con el avatar, un bloque de código para un número de monedas asociadas con el avatar y similares.

En una realización, el procesador está configurado para ejecutar el código de detección de manipulación para monitorizar la ejecución de uno o más bloques de código predeterminados del código ejecutable. Como resultado de la ejecución de monitorización de solo el uno o más bloques de código predeterminados, la velocidad de procesamiento del procesador se mejora y el tiempo asociado con la detección de la manipulación se reduce, lo que mejora la experiencia general del usuario.

En una realización, el código ejecutable se asocia con una aplicación de juego. En un ejemplo, el uno o más bloques de código predeterminados del código ejecutable que son bloques de código de recompensa pueden ser cualquier de los enumerados en la tabla 1.

Tabla 1

SetDeviceTokenMessage::encode
ModeBase::updateLoading
GameMain::showNativeDialog
HUD::setLeftSideVisible
LogicBattleLog::calculateAvailableResources
Serverconnection::update
LogicClientAvatar::decode
GameMain::IsIAPWarningNeeded
Building::getBuildingMovie
EventTracker::matchMakingSuccess
GameMode::~GameMode
Trap::update
HUDButton::goHomePressed

En una realización, el método comprende además monitorizar la ejecución de uno a tres bloques de código inicial del código ejecutable. Opcionalmente, monitorizar la ejecución de uno a tres bloques de código inicial del código ejecutable por el código de detección de manipulación permite determinar si el código ejecutable está manipulado. En una realización, el procesador se configura para ejecutar el código de detección de manipulación para monitorizar la ejecución de uno a tres bloques de código inicial del código ejecutable. De manera beneficiosa, la monitorización de la ejecución de solo el uno de los bloques de código iniciales a tres en lugar de todos los bloques de código permite aumentar la velocidad de procesamiento del procesador y el tiempo asociado con la determinación de la manipulación en el código ejecutable. En un ejemplo, en un juego de estrategia de combate, del uno a tres bloques iniciales de código puede incluir un bloque de código para un número de munición disponible con un avatar, un bloque de código para una cantidad de dinero virtual asociado con el avatar, un bloque de código para una serie de vidas asociadas con el avatar y similares. Dichos bloques de código están generalmente en una posición inicial del código ejecutable cuando el usuario inicia la aplicación de software.

El método comprende recibir información sobre uno o más espacios de dirección accedidos. En otras palabras, el espacio de direcciones se refiere a alertas del código ejecutable en el dispositivo cliente. El término *"espacio de dirección"* se refiere a una dirección que apunta a bibliotecas autorizadas del código ejecutable. Las bibliotecas autorizadas son utilizadas por el uno o más bloques de código para realizar una o más tareas de las tareas previstas del código ejecutable. En general, tras la manipulación del uno o más bloques de código, el uno o más bloques de código tienen un espacio de dirección que no apunta a la biblioteca autorizada pero a una biblioteca no autorizada proporcionada por el usuario no autorizado. La memoria está configurada para recibir y almacenar información relacionada con uno o más espacios de dirección mientras recibe el uno o más bloques de código. La información acerca de uno o más espacios de dirección se recibe desde la estructura de datos de pila de llamadas. Opcionalmente, la memoria también está configurada para recibir las bibliotecas que son señaladas por el espacio de direcciones del uno o más bloques de código. Generalmente, el usuario no autorizado inserta la biblioteca no autorizada en la memoria del dispositivo cliente y almacena el espacio de direcciones de uno o más bloques de código para apuntar a la biblioteca no autorizada. En un ejemplo, en un juego de estrategia de combate, un usuario no autorizado cambia el espacio de direcciones del bloque de código asociado con un número de municiones con el avatar. En tal ejemplo, el usuario no autorizado cambia el espacio de direcciones del bloque de código a una biblioteca que aumenta el número de municiones con el avatar. Opcionalmente, la memoria recibe la información mencionada anteriormente sobre uno o más espacios de dirección desde el servidor en el momento de recibir el código ejecutable. El procesador se configura para ejecutar el código de detección de manipulación para recibir información sobre uno o más espacios de dirección, a los que se accede. En un ejemplo, en un juego de estrategia de combate, el código de detección de manipulación se configura para recibir información sobre uno o más espacios de dirección de uno o más bloques de código de la estructura de datos de pila de llamadas.

Opcionalmente, el método comprende además recibir información sobre un tamaño asociado con cada uno de los uno o más bloques de código de manera que el tamaño corresponda a un tamaño computacional de la biblioteca a la que cada bloque de código está asociado. Opcionalmente, el tamaño asociado con cada uno de los uno o más bloques de código está predeterminado y almacenado. Además, el procesador se configura para ejecutar el código de detección de manipulación para recibir información sobre el tamaño asociado con cada uno de los uno o más bloques de código. Generalmente, tras la manipulación del uno o más bloques de código, el tamaño asociado con cada uno de los uno o más bloques de código se cambia. En un ejemplo, en un juego de estrategia de combate, un tamaño asociado con un bloque de código para una cantidad de dinero con el avatar se cambia de 10 kilobytes a 12 kilobytes. Opcionalmente, el procesador se configura para ejecutar el código de detección de manipulación para recibir información sobre el tamaño asociado con cada uno de los uno o más bloques de código.

El método comprende comparar la información recibida sobre uno o más espacios de dirección accedidos con información acerca de uno o más espacios de dirección permitidos definidos en la estructura de datos de pila de llamadas del código ejecutable. El término *"espacio de direcciones permitido"* se refiere al espacio de dirección de un bloque de código que está predeterminado que es el espacio de direcciones conocido por los desarrolladores de aplicaciones del código ejecutable y cuya ubicación real de tiempo de ejecución en la memoria del dispositivo cliente está definida por el sistema operativo del dispositivo cliente mientras se ejecuta el código ejecutable. En otras palabras, el término *"espacio de direcciones permitido"* se refiere a un espacio de direcciones que apunta a bibliotecas autorizadas del código ejecutable. Cada uno de los uno o más bloques de código de un código ejecutable no manipulado tiene el espacio de direcciones que apunta a las bibliotecas autorizadas. Cada una de las bibliotecas autorizadas tiene un espacio de dirección en la memoria y, por lo tanto, uno o más bloques de código de código ejecutable no manipulado tienen el espacio de dirección que apunta a las bibliotecas autorizadas. En un ejemplo, cada uno de los uno o más espacios de dirección accedidos y el uno o más espacios de dirección permitidos están representados como una cadena alfanumérica. En tal ejemplo, la cadena alfanumérica de cada uno de los uno o más espacios de dirección accedidos puede compararse con la cadena alfanumérica de cada uno de uno o más espacios de dirección permitidos para identificar una variación en uno o más espacios de dirección accedidos. En tal ejemplo, cada bit de la cadena alfanumérica de cada uno de los uno o más espacios de dirección accedidos puede compararse con cada bit correspondiente de la cadena alfanumérica de cada uno de uno o más espacios de dirección permitidos para identificar una variación en uno o más espacios de dirección accedidos en función de la variación en bits de las cadenas. El procesador está configurado para ejecutar el código de detección de manipulación para comparar la información recibida sobre uno o más espacios de dirección accedidos con información sobre uno o más espacios de dirección permitidos definidos en la estructura de datos de pila de llamadas del código ejecutable.

Opcionalmente, el método comprende comparar el tamaño asociado con cada uno de los uno o más bloques de código con un tamaño permitido de uno o más bloques de código. El término *"tamaño permitido"* se refiere al tamaño computacional de la biblioteca, cada bloque de código no manipulado se asocia con y cuyo tamaño computacional es conocido por el(los) desarrollador(es) de la aplicación del código ejecutable. Opcionalmente, el método comprende determinar si el tamaño asociado con cada uno de los uno o más bloques de código es igual al tamaño permitido de uno o más bloques de código. El procesador está configurado para ejecutar el código de detección de manipulación para comparar el tamaño asociado con cada uno de los uno o más bloques de código con el tamaño permitido de uno o más bloques de código.

En una realización, el código ejecutable contiene al menos un segmento de solo lectura que contiene información relacionada con uno o más espacios de dirección permitidos del mismo, y en donde el procesador está configurado para ejecutar el código de detección de manipulación para cargar la información relacionada con uno o más espacios de dirección permitidos del al menos un segmento de solo lectura. Opcionalmente, el al menos un segmento de solo lectura contiene también información acerca del tamaño permitido de uno o más bloques de código del código ejecutable y en el que el procesador está configurado para ejecutar el código de detección de manipulación para cargar la información relacionada con el tamaño permitido del al menos un segmento de solo lectura. Opcionalmente, el al menos un segmento de solo lectura se añade al código ejecutable por el desarrollador en el momento del desarrollo del código ejecutable. Opcionalmente, el al menos un segmento de solo lectura se mapea directamente del código ejecutable en la memoria solo una vez por, por ejemplo, un núcleo. Opcionalmente, tras la ejecución del código ejecutable, el procesador mapea el al menos un segmento de solo lectura en la memoria. Opcionalmente, el al menos un segmento de solo lectura no puede ser manipulado por los usuarios no autorizados. Opcionalmente, el código de detección de manipulación se configura para recuperar el tamaño y la dirección de inicio del al menos un segmento de solo lectura mientras el programa de aplicación del código ejecutable comienza a ejecutar y almacena además esas direcciones como direcciones seguras o espacios de dirección permitidos.

El método comprende activar la alerta tras la detección de que uno o más espacios de dirección accedidos son diferentes del uno o más espacios de dirección permitidos, basándose en la comparación. El término *"alerta"* se refiere a una señal que se genera para indicar que una determinada condición se determina en una posición donde se genera la alerta. En la presente descripción, *"alerta"* se refiere a la señal que se genera para indicar que el espacio de dirección accedido asociado con uno o más bloques de código es diferente de uno o más de los espacios de dirección permitido. Opcionalmente, la alerta puede ser una alerta binaria. Opcionalmente, la alerta puede ser una alerta no binaria. Opcionalmente, una alerta puede tener valor real o valor falso, en donde cuando la alerta tiene el valor verdadero, esto indica que el bloque de código dado tiene un espacio de dirección accedido diferente del uno o más espacio de dirección permitido, y cuando la alerta tiene el valor falso, indica que el bloque de código dado tiene una distancia de dirección a la que se accede igual que uno del uno o más espacios de dirección permitidos. El procesador se configura para ejecutar el código de detección de manipulación para activar la alerta tras la detección de que uno o más espacios de dirección accedidos son diferentes del uno o más espacios de dirección permitidos, en base a la comparación. En un ejemplo, la identificación del valor de las alertas se realiza cada 10 segundos por el método.

En una realización, el método y el sistema comprenden además añadir la alerta como una variable global al código ejecutable. Opcionalmente, la alerta puede añadirse como una variable global a cada uno de los uno o más bloques de código del código ejecutable. Opcionalmente, la alerta de cada uno de los uno o más bloques de código tiene generalmente un valor falso. Además, el valor de la alerta de un bloque de código dado se cambia a un valor verdadero cuando el espacio de dirección accedido del bloque de código dado es diferente del uno o más espacios de dirección permitidos.

Opcionalmente, añadir la alerta como una variable global permite usar el valor de la alerta como indicador para realizar ciertas funciones. En un ejemplo, si el valor de la alerta de un bloque de código dado es falso, el espacio de dirección accedido de uno o más bloques de código posteriores no se compara con el uno o más espacio de dirección permitido.

Opcionalmente, el código de detección de manipulación está configurado para añadir información de contexto tal como un segmento de código corto después de que uno o más bloques de código que tienen uno o más espacios de dirección accedidos que son diferentes del uno o más espacios de dirección permitidos. En un ejemplo, la información de contexto proporciona información tal como beneficios que el usuario no autorizado puede buscar al manipular el bloque de código específico, los efectos de manipulación del bloque de código específico en otros usuarios y similares.

El método comprende ejecutar la acción basándose en la alerta activada. El método está configurado para reaccionar a una actividad sospechosa de la manipulación del código ejecutable mediante la ejecución de la acción para evitar que el usuario no autorizado logre los beneficios de manipular el código ejecutable y proteger además otros usuarios de pérdidas debido a la manipulación del código ejecutable. El procesador está configurado para ejecutar el código de detección de manipulación para ejecutar la acción basándose en la alerta activada. En una realización, el método comprende además terminar la ejecución del código ejecutable tras activarse la alerta. Además, tras la terminación de la ejecución del código ejecutable, el usuario no autorizado no puede utilizar el programa de aplicación que tenga el código ejecutable. Opcionalmente, la terminación de la ejecución del código ejecutable es útil para el bloqueo inmediato del usuario no autorizado del uso del programa de aplicación que tiene el código ejecutable. Opcionalmente, el desarrollador del código ejecutable puede no ser consciente de la manipulación del código ejecutable y del bloqueo posterior del usuario no autorizado. El procesador está configurado para ejecutar el código de detección de manipulación para terminar la ejecución del código ejecutable tras activarse la alerta.

En una realización, el método comprende además transmitir a una disposición de servidor, al activarse la alerta, información relacionada con uno o más de: uno o más programas que modifican los espacios de dirección accedidos para la ejecución del código ejecutable; información relacionada con una dirección de Protocolo de Internet (IP) de un dispositivo de usuario que emplea el código ejecutable; e información sobre un perfil de usuario asociado al dispositivo de usuario. El término "*disposición de servidor*" se refiere a una estructura y/o módulo que incluye componentes programables y/o no programables configurados para almacenar, procesar y/o compartir información. Específicamente, la disposición de servidor está configurada para proporcionar el código ejecutable al dispositivo de usuario y recibir la información mencionada anteriormente del dispositivo de usuario al activarse la alerta. Opcionalmente, la disposición de servidor incluye cualquier disposición de entidades computacionales físicas o virtuales capaces de mejorar la información para realizar diversas tareas computacionales. En un ejemplo, la disposición de servidor puede incluir componentes tales como memoria, un procesador, un adaptador de red y similares. Se apreciará que los términos disposición de servidor y servidor se usan de manera intercambiable en la presente descripción. Opcionalmente, la información revela detalles sobre el intento de manipulación realizado por el usuario no autorizado. Opcionalmente, el usuario no autorizado puede añadir uno o más programas que modifican los espacios de dirección al código ejecutable. Opcionalmente, el uno o más programas que modifican los espacios de dirección pueden ser externos al código ejecutable. De forma ventajosa, la información acerca de uno o más programas que modifican los espacios de dirección permite la terminación inmediata del código ejecutable tras la detección de dichos uno o más programas. Opcionalmente, la información también puede incluir una ubicación del Sistema de Posicionamiento Global (G.P.S.) y una identidad internacional de equipo móvil (I.M.E.I.) del dispositivo de usuario. Se apreciará que los términos dispositivo de usuario y dispositivo cliente se han usado indistintamente en la presente descripción. Opcionalmente, la información sobre el perfil de usuario incluye, pero no se limita a, nombre de usuario, número de identificación de usuario, imagen de perfil de usuario, nacionalidad de usuario, dirección residencial de usuario, edad de usuario, número de contacto de usuario, dirección de correo electrónico de usuario y similares. Opcionalmente, la información también puede incluir detalles sobre cualquier intento de manipulación previa por parte del usuario. Opcionalmente, la información sobre el perfil de usuario se almacena previamente en la aplicación de juego. De forma ventajosa, dicha información permite identificar usuarios no autorizados que han estado realizando repetidamente la manipulación del código ejecutable. Además, dichos usuarios no autorizados pueden estar prohibidos de forma permanente del uso del código ejecutable. En una realización, el procesador está configurado para transmitir la información mencionada anteriormente a la disposición de servidor. Opcionalmente, el método, al activarse la alerta, está configurado para corregir el uno o más espacios de dirección accedidos diferentes del uno o más espacios de dirección permitidos.

En una realización, el método comprende además implementar una función de suma de control para evitar la manipulación del código de detección de manipulación. El término *"función de suma de control"* se refiere a una función que cuando se ejecuta en el código de detección de manipulación da como resultado una primera salida. Opcionalmente, el desarrollador conoce la primera salida del programa de aplicación que tiene el código ejecutable. Más opcionalmente, la primera salida puede añadirse al código ejecutable por el desarrollador. La función de suma de control se ejecuta de nuevo en el código de detección de manipulación en el momento de la ejecución del código ejecutable en el dispositivo cliente para generar la segunda salida. Opcionalmente, la segunda salida puede añadirse al código ejecutable. Opcionalmente, el código ejecutable está configurado para comparar la primera salida con la segunda salida. Además, basándose en la comparación, si la primera salida y la segunda salida son diferentes, se puede activar una alerta y se puede finalizar la ejecución adicional del código ejecutable. En una realización, el

procesador está configurado para implementar la función de suma de control para evitar la manipulación del código de detección de manipulación.

En una implementación ilustrativa de una aplicación de juego, la aplicación de juego comprende función1, función2, función3 y función4. En tal caso, la ejecución de código ejecutable no manipulado de la aplicación de juego comprende la función 1 que llama a la función2 y la función2 tras la ejecución devuelve un valor a la función 1. Sin embargo, tras la manipulación del código ejecutable, la ejecución del código ejecutable manipulado comprende la función1 que llama a la función3 y la función3 puede llamar a la función4 y la función2. En tal caso, se encuentra que el espacio de dirección de retorno en las funciones es manipulado por el código de detección de manipulación. Además, una alerta se activa y una acción es ejecutada por el código de detección de manipulación.

La presente descripción proporciona además un producto de programa informático que comprende un código de detección de manipulación legible por ordenador que, cuando se ejecuta en un ordenador, está dispuesto para hacer que el ordenador realice el método de detección de manipulación en el código ejecutable que comprende uno o más bloques de código. El producto de programa informático hace que el ordenador monitorice la ejecución del código ejecutable, reciba información sobre uno o más espacios de dirección, compare la información recibida sobre uno o más espacios de dirección accedidos con información sobre uno o más espacios de dirección permitidos, active una alerta tras la detección de que uno o más espacios de dirección accedidos son diferentes del uno o más espacios de dirección permitidos, y ejecute la acción basada en la alerta activada. Opcionalmente, el producto de programa informático se almacena en un medio de almacenamiento no transitorio.

La presente descripción proporciona además un sistema para la detección de manipulación en un código ejecutable. El sistema comprende una memoria configurada para almacenar el código ejecutable que comprende uno o más bloques de código y una estructura de datos de pila de llamadas asociada con el mismo, y el producto de programa informático mencionado anteriormente; y un procesador configurado para ejecutar el código de detección de manipulación. El procesador está configurado para monitorizar la ejecución del código ejecutable, recibir información sobre uno o más espacios de dirección, comparar la información recibida sobre uno o más espacios de dirección accedidos con información sobre uno o más espacios de dirección permitidos, generar una alerta tras la detección de que uno o más espacios de dirección accedidos son diferentes del uno o más espacios de dirección permitidos, y ejecutar la acción basada en la alerta activada.

Descripción detallada de los dibujos

Con referencia a la Figura 1A, se muestra un sistema **100A** para la detección de manipulación en un código ejecutable, según una realización de la presente descripción. Como se muestra, el sistema **100A** comprende un dispositivo de usuario **102,** El dispositivo de usuario **102** comprende una memoria **104** y un procesador **106,** La memoria **104** se configura para almacenar el código ejecutable que comprende uno o más bloques de código y una estructura de datos de pila de llamadas asociada con el mismo, y un código de detección de manipulación. El procesador **106** se configura para ejecutar el código de detección de manipulación.

Con referencia a la Figura 1B, se muestra un entorno de red de un sistema **100B** para la detección de manipulación en un código ejecutable, según una realización de la presente descripción. El sistema **100B** comprende el dispositivo de usuario **102** acoplado de forma comunicativa a una disposición de servidor **108** a través de una red **110.** El dispositivo de usuario **102** está configurado para recibir desde la disposición de servidor **108,** el código ejecutable que comprende uno o más bloques de código y una estructura de datos de pila de llamadas asociada con el mismo, y un código de detección de manipulación.

Con referencia a la Figura 1C, se muestra un entorno de red de un sistema **100C** para la detección de manipulación en un código ejecutable **112**, según otra realización de la presente descripción. El sistema **100C** comprende el dispositivo de usuario **102** acoplado de forma comunicativa a la disposición 108 de servidor a través de la red **110**. La memoria **104** del dispositivo de usuario **102** está configurado para recibir desde la disposición de servidor **108**, código ejecutable **112** y un código de detección de manipulación **114**. El procesador **106** del dispositivo de usuario **102** está configurado para transmitir a la disposición de servidor **108**, tras activar una alerta, la información relacionada con uno o más de: uno o más programas que modifican espacios de dirección para la ejecución del código ejecutable **112** información relacionada con una dirección de protocolo de Internet (I.P.) del dispositivo de usuario **102** empleando el código ejecutable **112;** e información sobre un perfil de usuario asociado al dispositivo de usuario **102**.

Con referencia a la Figura 2, se muestra un diagrama de bloques que representa detalles de la memoria **104** del sistema para la detección de manipulación en el código ejecutable **112**, según una realización de la presente descripción. Como se muestra, la memoria **104** comprende el código ejecutable **112**, una estructura **200** de datos de pila de llamadas y el código de detección de manipulación **114**. El código ejecutable **112** comprende uno o más bloques de código representados como bloques de código **202A, 202B, 202C** a **202N**. La memoria **104** comprende uno o más espacios de dirección representados como espacios de dirección **204A, 204B, 204C** a **204N** a la que se accede mediante uno o más bloques de código **202A, 202B, 202C** a **202N** para la ejecución del código ejecutable **112**.

Con referencia a la Figura 3, se muestra un diagrama de flujo de un método **300** para la detección de la manipulación en un código ejecutable que comprende uno o más bloques de código, según una realización de la presente descripción. En una etapa **302,** se monitoriza la ejecución del código ejecutable con una estructura de datos de pila de llamadas asociada con el mismo. En la presente descripción, la ejecución implica acceder a uno o más espacios de dirección. En una etapa **304,** se recibe información sobre uno o más espacios de dirección, a los que se accede. En una etapa **306,** la información recibida acerca de uno o más espacios de dirección accedidos se compara con la información acerca de uno o más espacios de dirección permitidos definidos en la estructura de datos de pila de llamadas del código ejecutable. En una etapa **308,** se activa una alerta tras la detección de que uno o más espacios de dirección accedidos son diferentes del uno o más espacios de dirección permitidos, en base a la comparación. En una etapa **310,** una acción se ejecuta en base a la alerta activada.

Las etapas **302** a **310** son únicamente ilustrativas y también pueden proporcionarse otras alternativas donde se añaden una o más etapas, se eliminan una o más etapas o se proporcionan una o más etapas en una secuencia distinta sin apartarse del ámbito de las reivindicaciones de la presente memoria.

Es posible realizar modificaciones a las realizaciones de la presente descripción descritas anteriormente sin salirse del ámbito de la presente descripción definido por las reivindicaciones adjuntas. Se prevé que expresiones tales como “que incluye”, “que comprende”, “que incorpora”, “tienen”, “es”, utilizadas para describir y reivindicar la presente descripción, se interpreten de un modo no exclusivo, a saber, permitiendo que partes, componentes o elementos no descritos explícitamente también estén presentes. También debe interpretarse que el singular se refiere al plural.

REIVINDICACIONES

1. Un método para la detección de manipulación en un código ejecutable (112) que comprende uno o más bloques de código (202A, 202B, 202C, 202N), que comprende:

-monitorizar la ejecución del código ejecutable con una estructura (200) de datos de pila de llamadas asociada con el mismo, cuya ejecución implica acceder a uno o más espacios de dirección (202A, 204B, 204C, 204N);
-recibir información sobre uno o más espacios de dirección accedidos;
-comparar la información recibida sobre uno o más espacios de dirección accedidos con información acerca de sobre uno o más espacios de dirección permitidos definidos en la estructura de datos de pila de llamadas del código ejecutable;
-activar una alerta tras la detección de que uno o más espacios de dirección accedidos son diferentes del uno o más espacios de dirección permitidos, en base a la comparación;
-ejecutar una acción basada en la alerta; y
-añadir la alerta como una variable global a cada uno de los uno o más bloques de código del código ejecutable, en donde el indicador de cada uno de los uno o más bloques de código tiene un valor falso, y un valor de la alerta de un bloque de código dado cambia a un valor verdadero cuando un espacio de dirección accedido del bloque de código dado es diferente del uno o más espacios de dirección permitidos,

además en donde, si el valor del indicador del bloque de código dado es un valor falso, un espacio de dirección accedido de uno o más bloques de código posteriores no se compara con uno o más espacios de dirección permitidos.

2. Un método de la reivindicación 1, que comprende además monitorizar la ejecución de cada uno de los uno o más bloques de código del código ejecutable.

3. Un método de la reivindicación 1 o 2, que comprende además monitorizar la ejecución de uno o más bloques de código predeterminados del código ejecutable.

4. Un método de una cualquiera de las reivindicaciones anteriores, que comprende además monitorizar la ejecución de uno a tres bloques de código inicial del código ejecutable.

5. Un método de una cualquiera de las reivindicaciones anteriores que comprende además terminar la ejecución del código ejecutable tras activarse la alerta.

6. Un método de una cualquiera de las reivindicaciones anteriores que comprende además transmitir a una disposición (108) de servidor, al activarse la alerta, información relacionada con uno o más de:

-uno o más programas que modifican los espacios de dirección accedidos para la ejecución del código ejecutable;
-información relacionada con una dirección de protocolo de Internet (IP) de un dispositivo (102) de usuario que emplea el código ejecutable; y
-información sobre un perfil de usuario asociado al dispositivo de usuario.

7. Un sistema (100A, 100B) para la detección de manipulación en un código ejecutable, que comprende:

-una memoria (104) configurada para almacenar el código ejecutable que comprende uno o más bloques de código y una estructura de datos de pila de llamadas asociada con el mismo, y un código (114) de detección de manipulación; y
-un procesador (106) configurado para ejecutar el código de detección de manipulación:

-monitorizar la ejecución del código ejecutable, implicando la ejecución acceder a uno o más espacios de dirección;
-recibir información sobre uno o más espacios de dirección accedidos;
-comparar la información recibida sobre uno o más espacios de dirección accedidos con información sobre uno o más espacios de dirección permitidos definidos en la estructura de datos de pila de llamadas del código ejecutable;
-activar una alerta tras la detección de que uno o más espacios de dirección accedidos son diferentes del uno o más espacios de dirección permitidos, en base a la comparación;
-ejecutar una acción basada en la alerta; y
-añadir la alerta como una variable global a cada uno de los uno o más bloques de código del código ejecutable, en donde la alerta de cada uno de los uno o más bloques de código tiene un valor falso, y un valor de la alerta de un bloque de código dado cambia a un valor

verdadero cuando un espacio de dirección accedido del bloque de código dado es diferente del uno o más espacios de dirección permitidos,

además en donde, si el valor del indicador del bloque de código dado es un valor falso, un espacio de dirección accedido de uno o más bloques de código posteriores no se compara con uno o más espacios de dirección permitidos.

8. Un sistema de la reivindicación 7, en donde el procesador está configurado para ejecutar el código de detección de manipulación para monitorizar la ejecución de cada uno de los uno o más bloques de código del código ejecutable.

9. Un sistema de la reivindicación 7 u 8, en donde el procesador está configurado para ejecutar el código de detección de manipulación para monitorizar la ejecución de uno o más bloques de código predeterminados del código ejecutable.

10. Un sistema de una cualquiera de las reivindicaciones 7 a 9, en donde el procesador está configurado para ejecutar el código de detección de manipulación para monitorizar la ejecución de uno a tres bloques de código inicial del código ejecutable.

11. Un sistema de una cualquiera de las reivindicaciones 7 a 10, en donde procesador está configurado para ejecutar el código de detección de manipulación para terminar la ejecución del código ejecutable tras activarse la alerta.

12. Un sistema de una cualquiera de las reivindicaciones 7 a 11, en donde el procesador está configurado para transmitir a una disposición de servidor, al activarse la alerta, información relacionada con uno o más de:

-uno o más programas que modifican los espacios de dirección accedidos para la ejecución del código ejecutable;
-información relacionada con una dirección de protocolo de Internet (IP) de un dispositivo de usuario que emplea el código ejecutable; y
-información sobre un perfil de usuario asociado al dispositivo de usuario.

13. Un sistema de una cualquiera de las reivindicaciones 7 a 12, en donde el código ejecutable comprende al menos un segmento de solo lectura que contiene información relacionada con uno o más espacios de dirección permitidos del mismo, y en el que el procesador está configurado para ejecutar el código de detección de manipulación para cargar la información relacionada con uno o más espacios de dirección permitidos del al menos un segmento de solo lectura.

14. Un sistema de una cualquiera de las reivindicaciones 7 a 13, en donde el código de detección de manipulación se añade en línea al código ejecutable.

15. Un sistema de una cualquiera de las reivindicaciones 7 a 14, en donde el código de detección de manipulación es una función llamada para la ejecución del código ejecutable.

16. Un sistema de una cualquiera de las reivindicaciones 7 a 15, en donde el código de detección de manipulación es una función de puntero de pila.

17. Un sistema de una cualquiera de las reivindicaciones 7 a 16, en donde el procesador está configurado para implementar una función de suma de control para evitar la manipulación del código de detección de manipulación.

18. Un sistema de una cualquiera de las reivindicaciones 7 a 17, en donde el código ejecutable está asociado con una aplicación de juego.

19. Un producto de programa informático, que comprende un código de detección de manipulación legible por ordenador que significa que, cuando se ejecuta en un ordenador, se dispone para hacer que el ordenador realice el método según una cualquiera de las reivindicaciones 1 – 6.

20. Un producto de programa informático según la reivindicación 19, almacenado en un medio de almacenamiento no transitorio.

21. Un sistema para la detección de manipulación en un código ejecutable, que comprende:

-una memoria configurada para almacenar el código ejecutable que comprende uno o más bloques de código y una estructura de datos de pila de llamadas asociada con el mismo, y un producto de programa informático según las reivindicaciones en la reivindicación 19 o 20; y
-un procesador configurado para ejecutar el código de detección de manipulación.

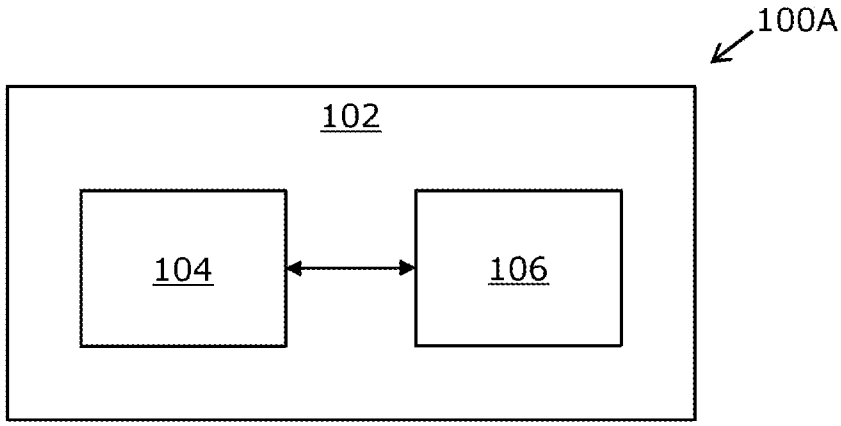

Figura 1A

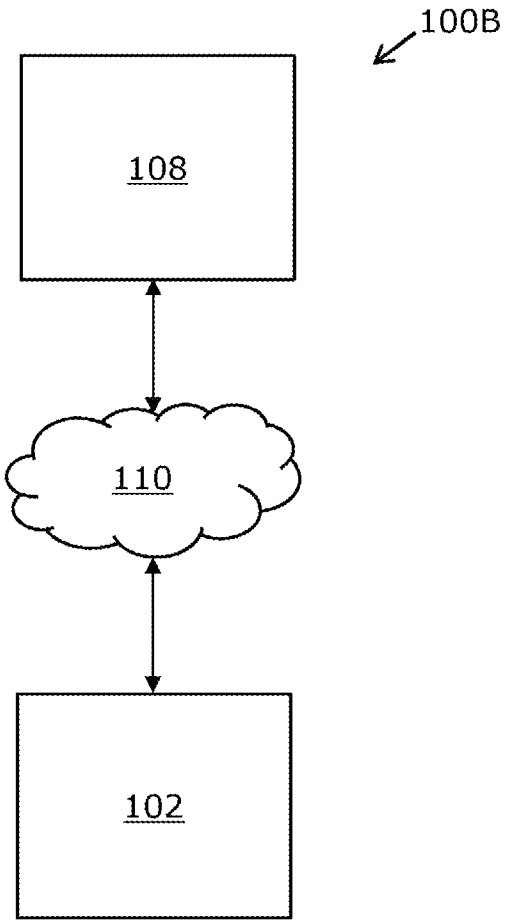

Figura 1B

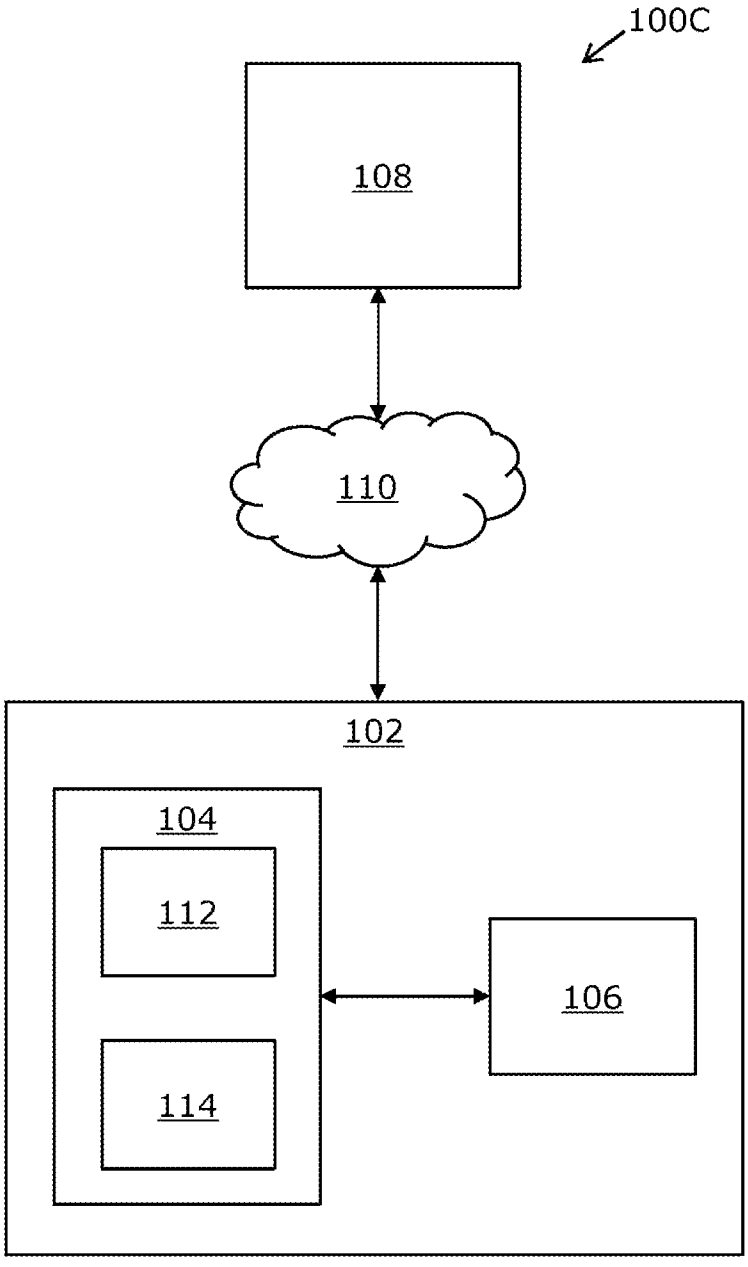

Figura 1C

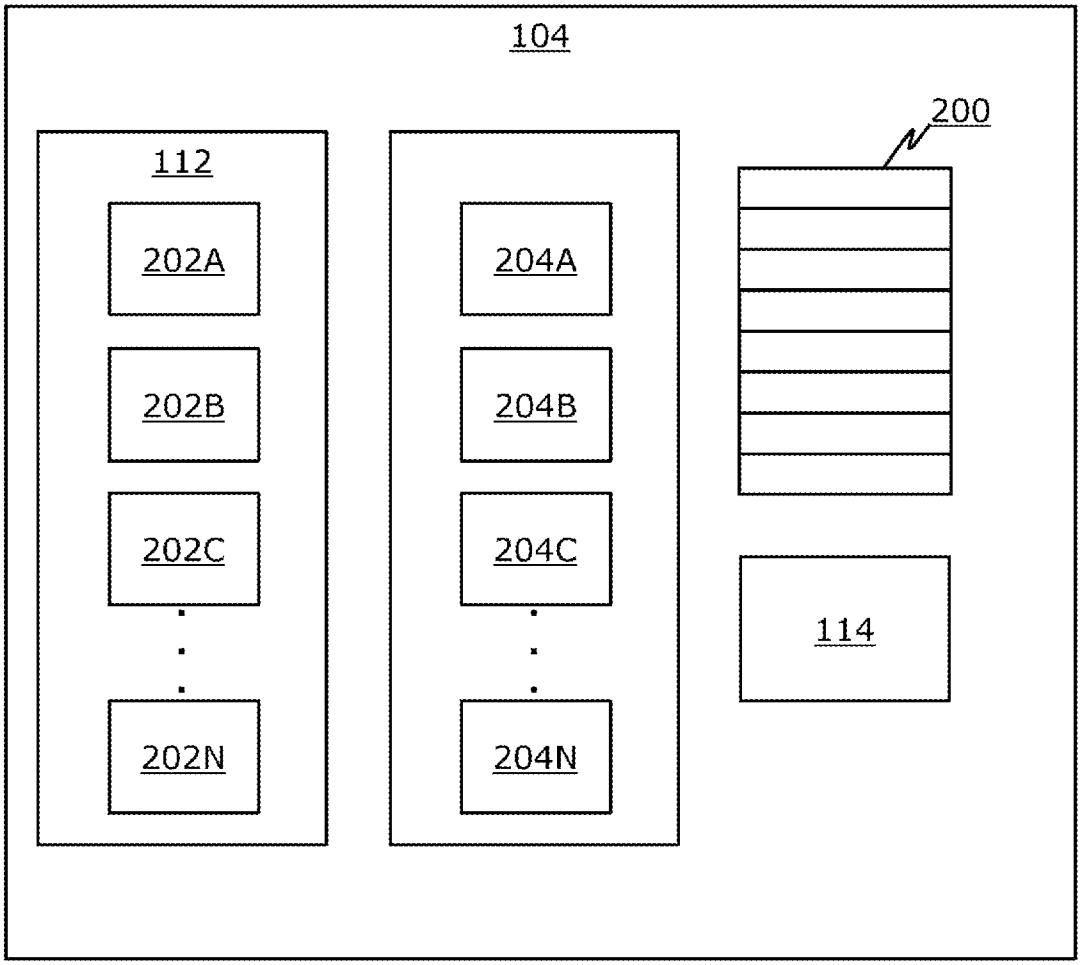

Figura 2

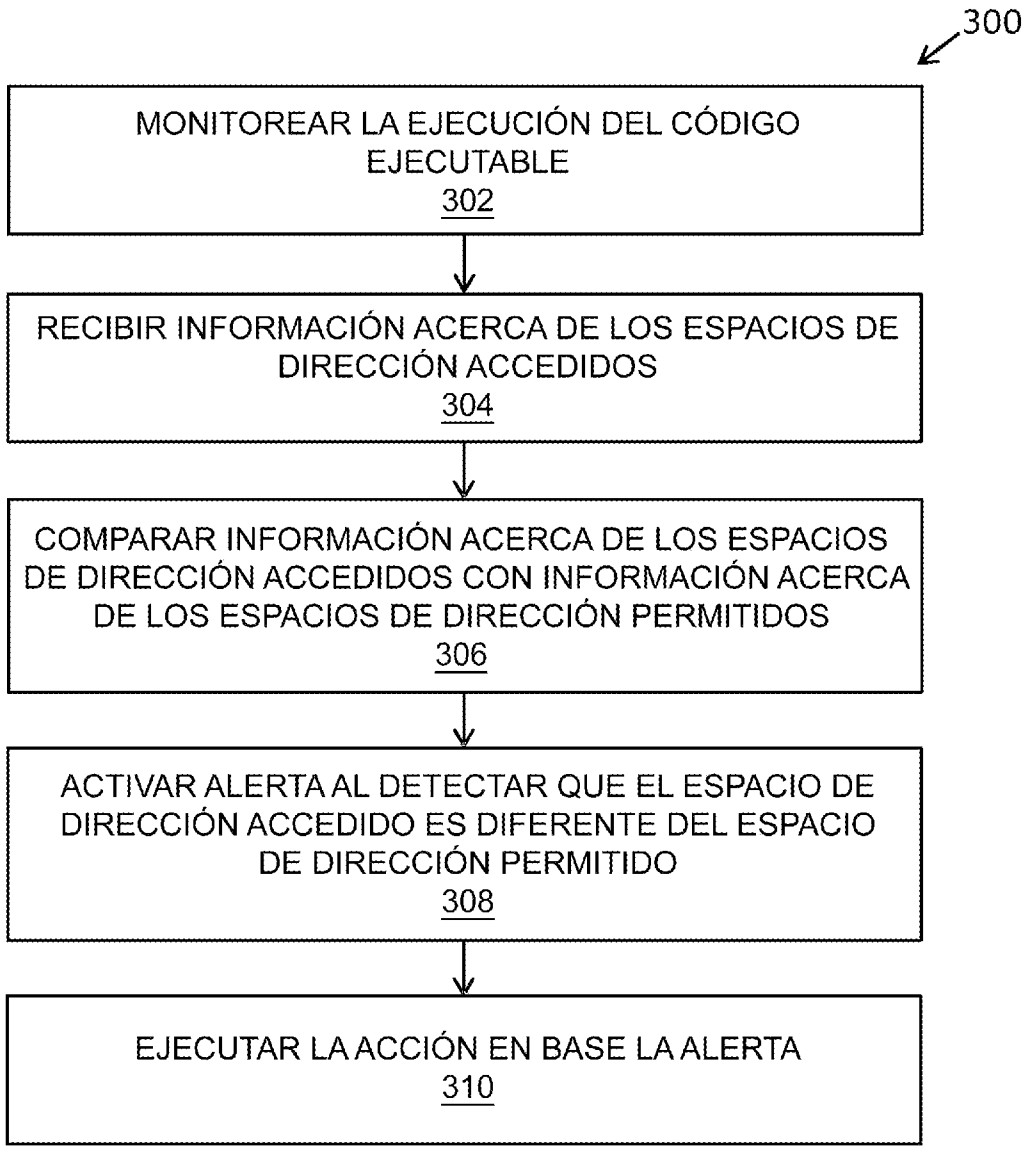
300
MONITOREAR LA EJECUCIÓN DEL CÓDIGO EJECUTABLE
302
RECIBIR INFORMACIÓN ACERCA DE LOS ESPACIOS DE DIRECCIÓN ACCEDIDOS
304
COMPARAR INFORMACIÓN ACERCA DE LOS ESPACIOS DE DIRECCIÓN ACCEDIDOS CON INFORMACIÓN ACERCA DE LOS ESPACIOS DE DIRECCIÓN PERMITIDOS
306
ACTIVAR ALERTA AL DETECTAR QUE EL ESPACIO DE DIRECCIÓN ACCEDIDO ES DIFERENTE DEL ESPACIO DE DIRECCIÓN PERMITIDO
308
EJECUTAR LA ACCIÓN EN BASE LA ALERTA
310

Figura 3