(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 685 662**

(51) Int. CI.:

G06F 21/56 (2013.01)
H04L 29/06 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **05.09.2012 PCT/RO2012/000020**

(87) Fecha y número de publicación internacional: **20.06.2013 WO13089576**

(96) Fecha de presentación y número de la solicitud europea: **05.09.2012 E 12832748 (3)**

(97) Fecha y número de publicación de la concesión europea: **20.06.2018 EP 2774076**

(54) Título: **Sistemas y métodos anti-software maligno de inclusión en lista blanca imprecisa**

(30) Prioridad:

02.11.2011 US 201161554859 P
06.12.2011 US 201113312686

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
10.10.2018

(73) Titular/es:

BITDEFENDER IPR MANAGEMENT LTD. (100.0%)
Kreontos 12
1076 Nicosa, CY

(72) Inventor/es:

TOFAN, I. VLAD;
DUDEA, V. SORIN y
CANJA, D. VIOREL

(74) Agente/Representante:

ISERN JARA, Jorge

DESCRIPCIÓN

Sistemas y métodos anti-software maligno de inclusión en lista blanca imprecisa

Antecedentes

La invención se refiere a sistemas y métodos para proteger a los usuarios frente al software malicioso y, en particular, a la inclusión en lista blanca de software.

El software malicioso, que también se conoce como software maligno, afecta a un número grande de sistemas informáticos por todo el mundo. En sus muchas formas tales como virus informáticos, gusanos, caballos de Troya y rootkits, el software maligno presenta un grave riesgo para millones de usuarios de ordenadores, lo que les hace vulnerables a la pérdida de datos, sustracción de identidad y pérdida de productividad, entre otros.

Los programas informáticos dedicados a la exploración de software maligno emplean diversos métodos de detección y eliminación de software maligno de los sistemas informáticos de los usuarios. Tales métodos incluyen técnicas basadas en comportamiento y técnicas basadas en contenido. Los métodos basados en comportamiento pueden implicar permitir que un programa sospechoso se ejecute en un entorno virtual aislado, identificar comportamiento malicioso y bloquear la ejecución del programa infractor. En los métodos basados en contenido, los contenidos de un archivo sospechoso se comparan habitualmente con una base de datos de firmas identificativas de software maligno conocidas. Si se halla una firma de software maligno conocida en el archivo sospechoso, el archivo se etiqueta como malicioso. Un método a modo de ejemplo de uso de firmas para la detección de software maligno se presenta en la publicación de solicitud de patente de EE. UU. con n.º 2008/0040804 A1, de Oliver et al, titulada “Malicious Software Detection”.

Otros métodos de hacer frente al software maligno emplean la inclusión en lista blanca de aplicaciones, que comprende mantener una lista de software y comportamientos que se permiten en el sistema informático de un usuario, y bloquear la ejecución de todas las otras aplicaciones. Tales métodos son particularmente eficaces contra el software maligno polimórfico, que es capaz de modificar aleatoriamente su firma identificativa de software maligno, volviendo ineficaces los métodos basados en contenido convencionales. Un ejemplo de tales métodos se describe en la publicación de solicitud de patente de EE. UU. con n.º 2008/0209557 A1 de Herley et al, titulada “Spyware Detection Mechanism”, en donde una aplicación de seguridad genera una lista de todos los módulos cargados, identifica de entre la lista un conjunto de módulos comunes a más de un umbral número de procesos, y elimina de la lista aquellos módulos incluidos en una lista de autenticación.

Algunas aplicaciones de inclusión en lista blanca emplean valores de troceo para identificar y asegurar la integridad del software incluido en lista blanca. Se puede crear un troceo criptográfico para un archivo o grupo de archivos afiliados a una aplicación incluida en lista blanca y almacenarse para su consulta. La aplicación respectiva se autentica entonces al comparar el troceo almacenado con un nuevo troceo generado en tiempo de ejecución.

Se han propuesto muchos algoritmos de cálculo de troceo para su uso en la seguridad informática. Un ejemplo, descrito en la publicación de solicitud de patente de EE. UU. con n.º 2011/0093426 A1, de Hoglund, titulada “Fuzzy Hash Algorithm”, proporciona un método para clasificar un objeto de datos usando un troceo impreciso. El troceo impreciso se calcula mediante el empalme de múltiples valores de troceo calculados para el objeto de datos respectivo. Otro método de seguridad a modo de ejemplo usando troceos se describe en la publicación de solicitud de patente de EE. UU. con n.º 2008/0263669 A1, de Alme, titulada “Systems, Apparatus, and Methods for Detecting Malware”, comprendiendo el método generar una primera huella imprecisa de un archivo de software maligno conocido, generar una segunda huella imprecisa de un archivo desconocido, y comparar la segunda huella imprecisa con la primera huella imprecisa para determinar si el archivo desconocido es malicioso.

El desempeño de los métodos de inclusión en lista blanca anti-software maligno puede depender de la capacidad de mantener y actualizar las bases de datos de lista blanca de una forma eficiente y flexible.

Sumario

De acuerdo con un aspecto, un método comprende realizar, en un sistema informático de cliente, una exploración de software maligno inicial de una pluralidad de objetos objetivo del sistema informático de cliente; y, en respuesta a una determinación tentativa por la exploración de software maligno inicial de que el objeto objetivo es sospechoso de ser malicioso: generar, en el sistema informático de cliente, una pluralidad de troceos objetivo del objeto objetivo, representando cada troceo objetivo un bloque de código distinto del objeto objetivo, consistiendo cada bloque de código distinto en una secuencia de instrucciones de procesador del objeto objetivo; enviar desde el sistema informático de cliente la pluralidad de troceos objetivo a un sistema informático de servidor conectado con el sistema informático de cliente a través de una red de área extensa; y recibir, en el sistema informático de cliente, a partir del sistema informático de servidor, un indicador de lado de servidor de si el objeto objetivo es malicioso. El indicador de lado de servidor es generado por el sistema informático de servidor al: para al menos un troceo objetivo de la pluralidad de troceos objetivo, recuperar una pluralidad de troceos de referencia de un objeto de referencia,

seleccionado el objeto de referencia de entre un conjunto de objetos incluidos en lista blanca de acuerdo con el troceo objetivo, y cuando la pluralidad de troceos objetivo no es idéntica a la pluralidad de troceos de referencia, determinar una puntuación de similitud de acuerdo con un recuento de troceos comunes tanto a la pluralidad de troceos objetivo como a la pluralidad de troceos de referencia; y cuando la puntuación de similitud supera un umbral previamente determinado, designar el objeto objetivo como no malicioso.

De acuerdo con otro aspecto, un método comprende recibir, en un sistema informático de servidor, una pluralidad de troceos objetivo de un objeto objetivo de un sistema informático de cliente conectado con el sistema informático de servidor a través de una red de área extensa; generar, en el sistema informático de servidor, un indicador de lado de servidor de si el objeto objetivo es malicioso; y enviar al sistema informático de cliente el indicador de lado de servidor de si el objeto objetivo es malicioso. La pluralidad de troceos objetivo se generan en el sistema informático de cliente en respuesta a una determinación tentativa por el sistema informático de cliente de que el objeto objetivo es sospechoso de ser malicioso, siendo la determinación tentativa el resultado de una exploración de software maligno inicial de una pluralidad de objetos objetivo del sistema informático de cliente. Generar, en el sistema informático de servidor, un indicador de lado de servidor de si el objeto objetivo es malicioso comprende: para al menos un troceo objetivo de la pluralidad de troceos objetivo, recuperar una pluralidad de troceos de referencia de un objeto de referencia, seleccionado el objeto de referencia de entre un conjunto de objetos incluidos en lista blanca de acuerdo con el troceo objetivo, y cuando la pluralidad de troceos objetivo no es idéntica a la pluralidad de troceos de referencia, determinar una puntuación de similitud de acuerdo con un recuento de troceos comunes tanto a la pluralidad de troceos objetivo como a la pluralidad de troceos de referencia, y cuando la puntuación de similitud supera un umbral previamente determinado, designar el objeto objetivo como no malicioso.

De acuerdo con otro aspecto, un método comprende recibir, en un sistema informático de servidor, una pluralidad de troceos objetivo de un objeto objetivo, representando cada troceo objetivo un bloque de código distinto del objeto objetivo, consistiendo cada bloque de código distinto en una secuencia de instrucciones de procesador del objeto objetivo; para al menos un troceo objetivo de la pluralidad de troceos objetivo, emplear el sistema informático de servidor para: recuperar una pluralidad de troceos de referencia de un objeto de referencia, seleccionado el objeto de referencia de entre un conjunto de objetos incluidos en lista blanca de acuerdo con el troceo objetivo, y cuando la pluralidad de troceos objetivo no es idéntica a la pluralidad de troceos de referencia, determinar una puntuación de similitud de acuerdo con un recuento de troceos comunes tanto a la pluralidad de troceos objetivo como a la pluralidad de troceos de referencia; y cuando la puntuación de similitud supera un umbral previamente determinado, emplear el sistema informático de servidor para etiquetar el objeto objetivo como no malicioso.

De acuerdo con otro aspecto, un sistema informático comprende al menos un procesador programado para recibir una pluralidad de troceos objetivo, representando cada troceo objetivo un bloque de código distinto de un objeto objetivo, consistiendo cada bloque de código distinto en una secuencia de instrucciones de procesador del objeto objetivo; para al menos un troceo objetivo de la pluralidad de troceos objetivo: recuperar una pluralidad de troceos de referencia de un objeto de referencia, seleccionado el objeto de referencia de entre un conjunto de objetos incluidos en lista blanca de acuerdo con el troceo objetivo, y cuando la pluralidad de troceos objetivo no es idéntica a la pluralidad de troceos de referencia, determinar una puntuación de similitud de acuerdo con un recuento de troceos comunes tanto a la pluralidad de troceos objetivo como a la pluralidad de troceos de referencia; y cuando la puntuación de similitud supera un umbral previamente determinado, etiquetar el objeto objetivo como no malicioso.

De acuerdo con otro aspecto, un medio de almacenamiento legible por ordenador no transitorio codifica unas instrucciones que, cuando se ejecutan en un procesador, dan lugar a que procesador realice las etapas de: recibir una pluralidad de troceos objetivo, representando cada troceo objetivo un bloque de código distinto de un objeto objetivo, consistiendo cada bloque de código distinto en una secuencia de instrucciones de procesador del objeto objetivo; para al menos un troceo objetivo de la pluralidad de troceos objetivo, recuperar una pluralidad de troceos de referencia de un objeto de referencia, seleccionado el objeto de referencia de entre un conjunto de objetos incluidos en lista blanca de acuerdo con el troceo objetivo; cuando la pluralidad de troceos objetivo no es idéntica a la pluralidad de troceos de referencia, determinar una puntuación de similitud de acuerdo con un recuento de troceos comunes tanto a la pluralidad de troceos objetivo como a la pluralidad de troceos de referencia. Cuando la puntuación de similitud supera un umbral previamente determinado, el objeto objetivo es no malicioso.

De acuerdo con otro aspecto, un sistema informático comprende: unos medios para recibir una pluralidad de troceos objetivo, representando cada troceo objetivo un bloque de código distinto de un objeto objetivo, consistiendo cada bloque de código distinto en una secuencia de instrucciones de procesador del objeto objetivo; unos medios para recuperar una pluralidad de troceos de referencia de un objeto de referencia, seleccionado el objeto de referencia de entre un conjunto de objetos incluidos en lista blanca de acuerdo con un troceo objetivo seleccionado de la pluralidad de troceos objetivo; unos medios para determinar una puntuación de similitud de acuerdo con un recuento de troceos comunes tanto a la pluralidad de troceos objetivo como a la pluralidad de troceos de referencia; y unos medios para etiquetar el objeto objetivo como no malicioso de acuerdo con la puntuación de similitud.

De acuerdo con otro aspecto, un método comprende recibir, en un sistema informático de servidor, una pluralidad de troceos objetivo, representando cada troceo objetivo un bloque de datos distinto de un objeto objetivo, consistiendo cada bloque de código distinto en una secuencia de instrucciones de procesador del objeto objetivo; en respuesta a

recibir la pluralidad de troceos objetivo, recuperar una pluralidad de troceos de referencia que representan un objeto de datos incluido en lista blanca, y cuando la pluralidad de troceos objetivo no es idéntica a la pluralidad de troceos de referencia, cuando la pluralidad de troceos objetivo y la pluralidad de troceos de referencia comparten una mayoría de elementos, etiquetar el objeto objetivo como no malicioso.

Breve descripción de los dibujos

Los aspectos y ventajas anteriores de la presente invención se entenderán mejor tras la lectura de la siguiente descripción detallada y tras consultar los dibujos en los que:

La figura 1 muestra un sistema anti-software maligno a modo de ejemplo de acuerdo con algunas realizaciones de la presente invención.

La figura 2 ilustra una configuración de hardware a modo de ejemplo de un sistema informático de cliente de acuerdo con algunas realizaciones de la presente invención.

La figura 3 muestra una configuración de hardware a modo de ejemplo de un sistema de servidor anti-software maligno de acuerdo con algunas realizaciones de la presente invención.

La figura 4 muestra un diagrama de una aplicación anti-software maligno a modo de ejemplo que se está ejecutando en el sistema informático de cliente de acuerdo con algunas realizaciones de la presente invención.

La figura 5 muestra aplicaciones a modo de ejemplo que se están ejecutando en el sistema de servidor anti-software maligno de acuerdo con algunas realizaciones de la presente invención.

La figura 6 ilustra una secuencia a modo de ejemplo de etapas realizadas por la aplicación anti-software maligno de cliente de la figura 4 de acuerdo con algunas realizaciones de la presente invención.

La figura 7 muestra un ejemplo de normalización de código de acuerdo con algunas realizaciones de la presente invención.

La figura 8 muestra una representación en memoria a modo de ejemplo de una instrucción de procesador de acuerdo con algunas realizaciones de la presente invención.

La figura 9 muestra un bloque de código a modo de ejemplo y un patrón de código de operación a modo de ejemplo que se corresponde con el bloque de código de acuerdo con algunas realizaciones de la presente invención.

La figura 10 ilustra un fragmento de código a modo de ejemplo que comprende una pluralidad de bloques de código y un indicador de datos de objeto (ODI) a modo de ejemplo que se corresponde con el fragmento de código, de acuerdo con algunas realizaciones de la presente invención.

La figura 11 muestra una secuencia a modo de ejemplo de etapas realizadas por la aplicación anti-software maligno de servidor de la figura 5 de acuerdo con algunas realizaciones de la presente invención.

Descripción detallada de realizaciones preferidas

En la siguiente descripción, se entiende que todas las conexiones mencionadas entre estructuras pueden ser conexiones operativas directas o conexiones operativas indirectas a través de estructuras intermedias. Un conjunto de elementos incluye uno o más elementos. Se entiende que cualquier mención de un elemento hace referencia a al menos un elemento. Una pluralidad de elementos incluye al menos dos elementos. A menos que se requiera lo contrario, no es necesario que etapa de método alguna descrita se realice necesariamente en un orden particular ilustrado. Un primer elemento (por ejemplo, datos) derivados de un segundo elemento abarca un primer elemento igual al segundo elemento, así como un primer elemento generado al procesar el segundo elemento y, opcionalmente, otros datos. Realizar una determinación o decisión de acuerdo con un parámetro abarca realizar la determinación o decisión de acuerdo con el parámetro y, opcionalmente, de acuerdo con otros datos. A menos que se especifique lo contrario, un indicador de una cierta cantidad/unos ciertos datos puede ser la propia cantidad/los propios datos, o un indicador diferente de la propia cantidad/los propios datos. Los programas informáticos descritos en algunas realizaciones de la presente invención pueden ser entidades o entidades secundarias de software autónomas (por ejemplo, subrutinas, objetos de código) de otros programas informáticos. A menos que se especifique lo contrario, un objeto objetivo es un archivo o un proceso residente en un sistema informático de cliente. Un identificador de un objeto objetivo comprende datos que permiten la identificación y recuperación selectivas del propio objeto objetivo, no meramente como parte de una estructura de datos más grande tal como la totalidad de la memoria de un sistema informático de cliente. A menos que se especifique lo contrario, un indicador de datos de objeto (ODI) de un objeto objetivo comprende características de los datos de objeto objetivo (por ejemplo, un bloque de código, un patrón de código de operación, un troceo) propicias para determinar si el objeto objetivo es malicioso,

por ejemplo, está infectado con software maligno. A menos que se especifique lo contrario, un troceo es una salida de una función de troceo. Las funciones de troceo son transformaciones matemáticas que mapean secuencias de símbolos (por ejemplo, caracteres, bits) a unas secuencias más cortas de números o cadenas de bits. Un troceo objetivo es un troceo computado sobre los datos de un objeto objetivo. A menos que se especifique lo contrario, se entiende que la expresión “incluido en lista blanca” quiere decir que se tiene confianza en que está limpio, es decir, que no contiene software maligno. Un primer conjunto es idéntico a un segundo conjunto cuando todos los elementos del primer conjunto están contenidos en el segundo conjunto y todos los elementos del segundo conjunto están contenidos en el primer conjunto. Los medios legibles por ordenador abarcan medios no transitorios tales como medios de almacenamiento magnéticos, ópticos y de semiconductores (por ejemplo, unidades de disco duro, discos ópticos, memoria flash, DRAM), así como enlaces de comunicaciones tales como cables conductores y enlaces de fibra óptica. De acuerdo con algunas realizaciones, la presente invención proporciona, entre otros, sistemas informáticos que comprenden hardware (por ejemplo, uno o más procesadores) programado para realizar los métodos descritos en el presente documento, así como medios legibles por ordenador que codifican instrucciones para realizar los métodos descritos en el presente documento.

La siguiente descripción ilustra realizaciones de la invención a modo de ejemplo y no necesariamente a modo de limitación.

La figura 1 muestra un sistema de detección de software maligno 10 a modo de ejemplo de acuerdo con algunas realizaciones de la presente invención. El sistema 10 comprende un conjunto de sistemas de servidor anti-software maligno (AM) 20a-c y un conjunto de sistemas informáticos de cliente 30a-b. Los sistemas informáticos de cliente 30a-b pueden representar ordenadores de usuario final que tienen, cada uno, un procesador, memoria y almacenamiento, y que ejecutan un sistema operativo tal como Windows®, MacOS® o Linux. Algunos sistemas informáticos de cliente 30a-b pueden representar dispositivos informáticos y/o de telecomunicaciones móviles tales como PC de tipo tableta y teléfonos móviles. En algunas realizaciones, los sistemas informáticos de cliente 30a-b pueden representar clientes individuales, o varios sistemas informáticos de cliente pueden pertenecer al mismo cliente. En algunas realizaciones, uno de los sistemas 30a-b puede ser un ordenador de servidor tal como un servidor de correo, caso en el cual se pueden usar servicios de detección de software maligno para identificar software maligno presentes en correos electrónicos u otros mensajes enviados a múltiples clientes, y para emprender una acción apropiada (por ejemplo, eliminar o poner en cuarentena elementos infectados con software maligno) antes de que los mensajes se entreguen a los clientes. Una red 12 conecta los sistemas informáticos de cliente 30a-c y los sistemas de servidor anti-software maligno 20a-c. La red 12 puede ser una red de área extensa tal como Internet. Partes de la red 12, por ejemplo, una parte de la red 12 que interconecta los sistemas informáticos de cliente 30a-b, también pueden incluir una red de área local (LAN).

La figura 2 muestra una configuración de hardware a modo de ejemplo de un sistema informático de cliente 30. En algunas realizaciones, el sistema 30 comprende un procesador 24, una unidad de memoria 26, un conjunto de dispositivos de entrada 28, un conjunto de dispositivos de salida 32, un conjunto de dispositivos de almacenamiento 34 y un controlador de interfaz de comunicación 36, todos ellos conectados por un conjunto de buses 38.

En algunas realizaciones, el procesador 24 comprende un dispositivo físico (por ejemplo, un circuito integrado de múltiples núcleos) configurado para ejecutar operaciones de cómputo y/o lógicas con un conjunto de señales y/o datos. En algunas realizaciones, tales operaciones lógicas se entregan al procesador 24 en forma de secuencia de instrucciones de procesador (por ejemplo, código máquina u otro tipo de software). La unidad de memoria 26 puede comprender medios legibles por ordenador volátiles (por ejemplo, RAM) que almacenan datos/señales a los que se accede o que sean generados por el procesador 24 en el curso de la puesta en práctica de instrucciones. Los dispositivos de entrada 28 pueden incluir teclados de ordenador y ratones, entre otros, permitiendo que un usuario introduzca datos y/o instrucciones en el sistema 30. Los dispositivos de salida 32 pueden incluir dispositivos de representación tales como monitores. En algunas realizaciones, los dispositivos de entrada 28 y los dispositivos de salida 32 pueden compartir un fragmento común de hardware, como en el caso de los dispositivos de pantalla táctil. Los dispositivos de almacenamiento 34 incluyen medios legibles por ordenador que posibilitan el almacenamiento no volátil, la lectura y la escritura de datos y/o instrucciones de software. Los dispositivos de almacenamiento 34 a modo de ejemplo incluyen discos magnéticos y ópticos y dispositivos de memoria flash, así como medios extraíbles tales como discos CD y/o DVD y unidades de disco. El controlador de interfaz de comunicación 36 posibilita que el sistema 30 se conecte con una red informática y/o con otras máquinas/sistemas informáticos. Los controladores de interfaz de comunicación 36 típicos incluyen adaptadores de red. Los buses 38 representan colectivamente la pluralidad de buses de sistema, de periféricos y de conjuntos de chips, y/o todos los otros conjuntos de circuitos que posibilitan la intercomunicación de los dispositivos 24-36 del sistema informático 30. Por ejemplo, los buses 38 pueden comprender el bus en puente norte que conecta el procesador 24 con la memoria 26, y/o el bus en puente sur que conecta el procesador 24 con los dispositivos 28-36, entre otros.

La figura 3 muestra una configuración de hardware de un sistema de servidor AM 20 a modo de ejemplo de los sistemas 20a-c, de acuerdo con algunas realizaciones de la presente invención. El sistema de servidor AM 20 puede ser un sistema informático que comprende un procesador de servidor 124, una memoria de servidor 126, un conjunto de dispositivos de almacenamiento de servidor 134 y un controlador de interfaz de comunicación de servidor 136, todos ellos conectados entre sí por medio de un conjunto de buses de servidor 138. Aunque algunos

detalles de configuración de hardware pueden diferir entre el sistema de servidor 20 y el sistema informático de cliente 30, el alcance de los dispositivos 124, 126, 134, 136 y 138 puede ser similar al de los dispositivos 24, 26, 34, 36 y 38 descritos anteriormente, respectivamente.

El sistema informático de cliente 30 puede incluir una aplicación anti-software maligno (AM) de cliente 40 y una memoria caché de lado de cliente 56, tal como se muestra en la figura 4. En algunas realizaciones, la aplicación AM de cliente 40 puede ser una aplicación autónoma, o puede ser un módulo anti-software maligno de un conjunto de aplicaciones de seguridad que tiene antivirus, cortafuegos, filtro de correo no deseado y otros módulos. La aplicación AM de cliente puede comprender una unidad de exploración AM activa 42, una unidad de exploración AM estática 44, un emulador 46 conectado con la unidad de exploración AM estática 44, un motor de normalización de código 48 conectado con las unidades de exploración 42 y 44, un gestor de comunicación AM de cliente 52 y un motor de cálculo de troceo 54 conectado con el gestor de comunicación 52 y el motor de normalización de código 48.

En algunas realizaciones, la aplicación AM de cliente 40 está configurada para realizar una parte de lado de cliente de una exploración colaborativa cliente-servidor para detectar software maligno almacenado en medios legibles por ordenador que forman parte del sistema informático de cliente 30 (por ejemplo, memoria, unidad de disco duro) o en medios legibles por ordenador conectados con el sistema 30 (por ejemplo, lápiz de memoria, unidad de disco duro externa, dispositivos de red, etc.) Como parte de una exploración colaborativa cliente-servidor, la aplicación AM de cliente 40 está configurada para enviar un indicador de datos de objeto (ODI) objetivo 100 a los sistemas de servidor AM 20a-c y para recibir un informe de exploración 50 a partir de los sistemas 20a-c.

Los objetos objetivo explorados por la aplicación AM 40 incluyen procesos y archivos informáticos. Cada proceso puede incluir un conjunto de módulos de memoria cargados (es decir, imágenes cargadas de un archivo ejecutable objetivo y sus biblioteca de vínculos dinámicos a las que se hace referencia), así como cualquier archivo adicional que se corresponda con los módulos de memoria cargados. Un objeto objetivo se puede considerar software maligno si el mismo contiene al menos una parte de una entidad de software malicioso (por ejemplo, virus, gusano, troyano).

En algunas realizaciones, el ODI 100 comprende una pluralidad de indicadores de bloque de código, cada indicador de bloque de código indicativo de un bloque de código distinto del objeto objetivo. Los contenidos y formatos a modo de ejemplo del ODI 100 se analizarán con detalle en relación con las figuras 7-9.

En algunas realizaciones, el informe de exploración 50 incluye un identificador (por ejemplo, etiqueta, ID de archivo) del objeto objetivo, un indicador de estatus de software maligno (por ejemplo, infectado, limpio, desconocido) del objeto objetivo, y/o un conjunto de identificadores de agentes de software maligno que infectan el objeto objetivo, tales como nombres de agentes de software maligno individuales (por ejemplo, Win32.Worm.Downadup.Gen), indicadores de clase de software maligno (virus, rootkit, etc.) o punteros a los agentes respectivos en una base de conocimiento de software maligno. En algunas realizaciones, se puede compilar un único informe de exploración para un lote de objetos objetivo.

En algunas realizaciones, el gestor de comunicación de servidor 52 está configurado para gestionar la comunicación con los sistemas AM de servidor 20a-c. Por ejemplo, el gestor 52 puede establecer conexiones a través de la red 12, enviar y recibir datos a/desde los servidores AM 20a-c, mantener una lista de transacciones de exploración en curso y asociar los ODI objetivo 100 con los servidores AM que llevan a cabo la exploración de lado de servidor.

La unidad de exploración AM activa 42 y la unidad de exploración AM estática 44 posibilitan que la aplicación AM de cliente 40 ejecute una exploración anti-software maligno preliminar del objeto objetivo, tal como se muestra con más detalle en lo sucesivo. Si la exploración preliminar detecta un contenido malicioso, el objeto objetivo infractor se notifica al usuario directamente, sin tener que pasar por una exploración cliente-servidor, ahorrando de ese modo tiempo y recursos informáticos. En algunas realizaciones, los objetos objetivo de archivo son manejados por la unidad de exploración AM estática 44, mientras que los objetos objetivo de proceso son manejados por la unidad de exploración AM activa 42. En algunas realizaciones, la unidad de exploración AM estática 44 puede usar el emulador 46 para desempaquetar un archivo y ejecutarlo en un entorno protegido, aparte de la memoria principal. Las unidades de exploración 42, 44 pueden usar métodos basados en comportamiento, diversas heurísticas, métodos basados en contenido (por ejemplo, coincidencia de firma) o una combinación de los mismos, para determinar si el objeto objetivo es software maligno. Los ejemplos de criterios heurísticos para determinar si un objeto objetivo es malicioso comprenden, entre otros, los tamaños relativos de diversas secciones en el archivo ejecutable portátil (PE) del objeto objetivo, la densidad de información en cada sección, la presencia de marcas y grupos de marcas específicos en el encabezado de PE, información acerca del empaquetador/protector (de haber alguno) y la presencia de determinados patrones de texto en el interior del ejecutable.

La aplicación AM de cliente 40 puede emplear el motor de normalización de código 48 y el motor de cálculo de troceo 54 para producir el ODI objetivo 100. El funcionamiento del motor de normalización de código 48 se analizará en lo sucesivo en relación con la figura 7. El motor de cálculo de troceo 54 está configurado para recibir un patrón de código de operación y para generar un troceo del patrón de código de operación respectivo, tal como se muestra en relación con las figuras 8-9. En algunas realizaciones, un troceo es la salida de una función de troceo, una

transformación matemática que mapea una secuencia de símbolos (por ejemplo, caracteres, bits) a una secuencia de números o una cadena de bits. Las funciones de troceo a modo de ejemplo empleadas por el motor de cálculo de troceo 54 incluyen comprobación de redundancia cíclica (CRC), resumen de mensaje (MD) o cálculo de troceo seguro (SHA), entre otros. Un troceo a modo de ejemplo es el CRC32 de 4 bytes.

Algunas realizaciones de la memoria caché de lado de cliente 56 comprenden, en cualquier instante dado, un repositorio de ODI que se corresponden con objetos objetivo residentes en el sistema de cliente 30 respectivo, objetos que ya se han explorado en busca de software maligno. En algunas realizaciones, la memoria caché 56 puede comprender un conjunto de troceos de los ODI de objeto objetivo; cada ODI recibido de los sistemas de cliente 30 se puede someter a cálculo de troceo, con los troceos duplicados eliminados, y los troceos resultantes almacenados como indicadores únicos de los ODI respectivos. La memoria caché 56 prevé un aumento en la velocidad de la exploración de software maligno. Si el ODI o el troceo del mismo de un objeto objetivo se halla en la memoria caché de cliente 56, indicando que el objeto objetivo respectivo ya se ha explorado al menos una vez, el estatus de software maligno del objeto objetivo se puede recuperar directamente de la memoria caché 56 y notificarse al usuario, un proceso considerablemente más rápido que realizar una nueva exploración del objeto objetivo. Para cada ODI, algunas realizaciones de la memoria caché 56 pueden comprender un identificador de objeto (por ejemplo, etiqueta, ID de archivo) y un indicador de estatus de software maligno del objeto objetivo respectivo.

La figura 5 muestra aplicaciones a modo de ejemplo que se están ejecutando en el sistema de servidor AM 20 de acuerdo con algunas realizaciones de la presente invención. En algunas realizaciones, el sistema 20 comprende una aplicación AM de servidor 60, una memoria caché de lado de servidor 68, una base de datos de lista blanca 65, una base de datos de software maligno 66 y una base de datos de irrupciones 67b, todas ellas conectadas con la aplicación de servidor AM 60.

En algunas realizaciones, la aplicación de servidor AM 60 está configurada para realizar una pluralidad de transacciones de detección de software maligno con los sistemas informáticos de cliente 30a-b. Para cada una de tales transacciones, la aplicación AM de servidor 60 está configurada para realizar una parte de lado de servidor de una exploración colaborativa para detectar software maligno residente en el sistema informático de cliente respectivo, tal como se describe con detalle en lo sucesivo. Como parte de una transacción cliente-servidor, la aplicación 60 recibe el ODI objetivo 100 a partir del sistema informático de cliente, y transmite el informe de exploración 50 al sistema informático de cliente respectivo. La aplicación AM de servidor 60 puede comprender un gestor de comunicación AM de servidor 62 y un comparador de código 64 conectado con el gestor de comunicación 62.

En algunas realizaciones, el gestor de comunicación de servidor 62 está configurado para gestionar la comunicación con los sistemas informáticos de cliente 30a-b. Por ejemplo, el gestor 62 puede establecer conexiones a través de la red 12, enviar y recibir datos a/desde clientes, mantener una lista de transacciones de exploración en curso y asociar los ODI objetivo 100 con los sistemas informáticos de cliente 30a-b que se originan. El comparador de código 64 está configurado para computar una puntuación de similitud que indica un grado de similitud entre un objeto objetivo y un conjunto de objetos de referencia almacenados en las bases de datos 65-67, tal como se describe con detalle en lo sucesivo.

En algunas realizaciones, la memoria caché de lado de servidor 68 comprende un repositorio de ODI de objetos objetivo que ya se han explorado en busca de software maligno, ODI recibidos de diversos sistemas informáticos de cliente 30a-b en el curso de las exploraciones colaborativas cliente-servidor previas. Tal como se analiza adicionalmente en lo sucesivo, si el ODI de un objeto objetivo se halla en la memoria caché de servidor 68, indicando que el objeto objetivo respectivo ya se ha explorado al menos una vez, el estatus de software maligno (por ejemplo, limpio, infectado, etc.) del objeto objetivo se puede recuperar de la memoria caché 68 sin realizar una nueva exploración del objeto objetivo. Junto con los ODI objetivo, algunas realizaciones de la memoria caché de servidor 68 pueden almacenar el estatus de software maligno (por ejemplo, limpio, infectado) del objeto objetivo respectivo.

Las bases de datos 65-67 se mantienen como repositorios de conocimiento relacionado con software maligno actual. En algunas realizaciones, cada base de datos 65-67 comprende un conjunto de indicadores de datos que se corresponden con una colección de objetos de referencia (archivos y procesos) de estatus de software maligno conocido. En algunas realizaciones, las bases de datos 65-67 almacenan datos en forma de troceos de patrón de código de operación (que se describen adicionalmente en lo sucesivo en relación con las figuras 7-10). La base de datos de lista blanca 65 incluye un conjunto de troceos recuperados de objetos de los que se tiene confianza en que están limpios (es decir, elementos incluidos en lista blanca). La base de datos de software maligno 66 comprende troceos identificativos de software maligno recuperados de objetos conocidos como software maligno. En algunas realizaciones, la base de datos de irrupciones 67 comprende troceos computados para objetos que son de estatus de software maligno desconocido (aún no reconocidos como software maligno o limpio).

En algunas realizaciones, todos los troceos de patrón de código de operación almacenados en las bases de datos 65-67 tiene el mismo tamaño (por ejemplo, 4 bytes). Estos se pueden almacenar de forma secuencial en la memoria y/o medios legibles por ordenador de los sistemas de servidor 20a-c. En algunas realizaciones, una segunda

estructura de datos que comprende identificadores de objeto (por ejemplo, ID de archivo también representados como números de 4 bytes) se almacena junto al conjunto de troceos de referencia. Un mapeo bidireccional almacenado en la memoria del servidor AM respectivo se usa para relacionar cada troceo con el ID de archivo del objeto del que se recuperó el mismo. Esto permite que la aplicación AM de servidor recupere de forma selectiva troceos de referencia, para determinar si los objetos objetivo recibidos de los sistemas informáticos de cliente son similares a cualquier objeto de referencia almacenado en las bases de datos 65-67. Las bases de datos 65-67 se están manteniendo actualizadas mediante la adición de datos de objeto objetivo recibidos de los sistemas informáticos de cliente 30a-b, tal como se describe adicionalmente en lo sucesivo.

La figura 6 muestra una secuencia a modo de ejemplo de etapas realizadas por la aplicación AM de cliente 40 de acuerdo con algunas realizaciones de la presente invención. En una etapa 202, la aplicación 40 selecciona un objeto objetivo para realizar una exploración en busca de software maligno. En algunas realizaciones, los objetos objetivo pueden ser especificados directa o indirectamente por un usuario (exploración a petición). Por ejemplo, el usuario puede dar instrucciones a la aplicación AM 40 de explorar un determinado archivo, o los contenidos de una determinada carpeta, o los contenidos almacenados en un determinado medio legible por ordenador (por ejemplo, CDROM, dispositivo de memoria flash). Otros objetos objetivo a modo de ejemplo se seleccionan durante la exploración en el acceso, en donde la aplicación 40 está configurada para explorar determinados tipos de archivos o procesos antes de leer/cargar/iniciar los mismos. En algunas realizaciones, se puede compilar un conjunto de objetos objetivo para el fin de una exploración programada del sistema informático de cliente que ejecuta la aplicación 40. Un conjunto de ese tipo, a modo de ejemplo, de objetos objetivo residentes en un sistema de cliente que ejecuta Windows@ de Microsoft, puede incluir archivos ejecutables procedentes de la carpeta WINDIR, ejecutables procedentes de la carpeta WINDIR/system32, ejecutables de los procesos actualmente en ejecución, biblioteca de vínculos dinámicos (DLL) importada por los procesos actualmente en ejecución y ejecutables de todos los servicios de sistema instalados, entre otros. En algunas realizaciones, los objetos objetivo también pueden incluir archivos/procesos seleccionados como objetivo por programas de software maligno de interés, por ejemplo, programas de software maligno considerados los más extendidos y activos en el momento del inicio de la exploración de software maligno respectiva.

En algunas realizaciones, un identificador (por ejemplo, ID de archivo) se usa para etiquetar de forma única el objeto objetivo respectivo. El identificador comprende datos que permiten una identificación selectiva del propio objeto objetivo (por ejemplo, un archivo o proceso) y no como parte de una estructura más grande tal como, por ejemplo, la totalidad de la memoria del sistema informático de cliente respectivo. Los identificadores de objeto objetivo a modo de ejemplo comprenden rutas de archivo y direcciones de memoria, entre otros. El identificador también permite que la aplicación AM de cliente 40 recupere de forma selectiva el objeto objetivo, con el fin de computar el ODI objetivo 100, así como realizar, de forma no ambigua, transacciones de exploración cliente-servidor con múltiples objetos objetivo.

En una etapa 204 (la figura 6), la aplicación AM de cliente 40 puede ejecutar una exploración anti-software maligno preliminar del objeto objetivo. En algunas realizaciones, los objetos objetivo de archivo son manejados por la unidad de exploración AM estática 44, mientras que los objetos objetivo de proceso son manejados por la unidad de exploración AM activa 42. Las unidades de exploración 42, 44 pueden usar métodos de comportamiento (por ejemplo, emulación), diversas heurísticas (por ejemplo, la geometría de un encabezado de ejecutable portátil del objeto objetivo), métodos basados en contenido (por ejemplo, coincidencia de firma) o una combinación de los mismos, para determinar si el objeto objetivo es software maligno. En algunas realizaciones, las unidades de exploración 42, 44 pueden producir un indicador del estatus de software maligno del objeto objetivo. Los indicadores de estatus a modo de ejemplo incluyen malicioso, sospechoso de ser malicioso y limpio, entre otros.

En algunas realizaciones, un objeto objetivo puede ser sospechoso de ser malicioso cuando el objeto objetivo tiene algunas características en común con objetos maliciosos conocidos, pero no suficientes para que se considere software maligno. Las características sospechosas a modo de ejemplo incluyen la presencia dentro del encabezado de PE del objeto objetivo de determinados valores/pares de valores, la presencia dentro del objeto objetivo de determinadas secuencias de código (por ejemplo, código que comprueba si el objeto objetivo se está ejecutando dentro de un entorno virtual) y la presencia de patrones de texto (firmas) identificativos de software maligno tales como contraseñas comunes y nombres y/o indicadores de ruta de software anti-software maligno, entre otros. Otras características sospechosas pueden comprender determinados patrones de comportamiento identificativos de software maligno del objeto objetivo.

En algunas realizaciones, las unidades de exploración 42, 44 computan una puntuación de software maligno para el objeto objetivo respectivo, en donde a cada característica identificativa de software maligno se le puede dar un peso específico. Cuando la puntuación de software maligno supera un primer umbral, el objeto objetivo respectivo puede ser sospechoso de ser malicioso; cuando la puntuación supera un segundo umbral más alto, el objeto objetivo se puede etiquetar como software maligno. Un objeto objetivo a modo de ejemplo que contiene cadenas específicas del protocolo de IRC, nombres de programas antivirus, contraseñas comunes de Windows® y secuencias de código específicas de vulnerabilidades de seguridad puede recibir una puntuación de software maligno comparativamente alta y, por lo tanto, se puede etiquetar como software maligno, mientras que otro objeto objetivo a modo de ejemplo, que solo contiene los nombres de algunas aplicaciones anti-software maligno, puede recibir una puntuación

relativamente baja, pero se puede seguir sospechando que es malicioso.

En una etapa 206, la aplicación 40 determina si el objeto objetivo es malicioso de acuerdo con la exploración de software maligno preliminar. En caso negativo, la operación de la aplicación 40 avanza a una etapa 210 descrita en lo sucesivo. En caso afirmativo, en una etapa 208, la aplicación AM 40 etiqueta el objeto objetivo como software maligno y actualiza la memoria caché de lado de cliente 56 en consecuencia, en una etapa 230. A continuación, la aplicación AM de cliente 40 emite el resultado de la exploración de software maligno en una etapa 232.

En algunas realizaciones, la etapa 232 puede comprender emitir una alerta (por ejemplo, una ventana emergente) para informar al usuario de que el sistema informático de cliente respectivo puede estar infectado. Como alternativa, la aplicación 40 puede documentar la exploración de software maligno en un registro de sistema. Algunas realizaciones de la aplicación AM 40 pueden presentar un informe de exploración al usuario, comprendiendo el informe, entre otros, el nombre (o identificador de objeto) del objeto objetivo, un indicador del tipo de software maligno detectado e información adicional con respecto al software maligno respectivo (por ejemplo, métodos de limpieza posibles).

En la etapa 210, la aplicación AM de cliente 40 puede determinar si el objeto objetivo es sospechoso de ser malicioso de acuerdo con un resultado de la exploración preliminar (véase la etapa 204 en lo que antecede). En caso afirmativo, la operación avanza a una etapa 212 analizada en lo sucesivo. En caso negativo, en una etapa 228, la aplicación 40 puede etiquetar el objeto objetivo como no malicioso (limpio) y avanzar a la etapa 230.

En la etapa 212, cuando el objeto objetivo es un archivo, la aplicación 40 puede cargar el archivo objetivo en un entorno protegido proporcionado por el emulador 46, para eliminar cualquier capa de empaquetamiento y/o cifrado que proteja el código del objeto objetivo. Cuando el objeto objetivo es un proceso, la operación de la aplicación 40 puede omitir la etapa 212, debido a que el objeto objetivo ya se habrá cargado en memoria de sistema.

En una etapa 214, el motor de normalización de código 48 realiza una normalización de código del objeto objetivo. Los compiladores pueden generar un código máquina diferente del mismo bloque de código fuente dependiendo de los parámetros de compilación usados, en particular debido a la optimización de código. Algunas variaciones de código adicionales pueden ser introducidas por un protector/software maligno polimórfico. En algunas realizaciones, la normalización de código comprende transformar el conjunto de instrucciones de procesador que forma el objeto objetivo en un conjunto normalizado de instrucciones de procesador, para eliminar variaciones de código informático introducidas por compilación y/u otros polimorfismos. Una operación de normalización de código a modo de ejemplo puede proceder tal como sigue:

1. El compilador usado para construir el objeto objetivo se detecta de acuerdo con determinadas características del objeto objetivo. Cuando el compilador es conocido, se determina una ubicación del código específico de objeto en el interior de la imagen en memoria del objeto objetivo. Cuando el compilador no se puede determinar, las áreas objetivo para la extracción de código se seleccionan con el fin de cubrir tantas ubicaciones de código específicas de objeto potenciales como sea posible (por ejemplo, punto de entrada, comienzo de la primera sección, comienzo de todas las secciones, etc.)

2. El desensamblado de código comienza en la ubicación hallada en la etapa previa. En algunas realizaciones, el desensamblado de código sigue unas ramificaciones de código (por ejemplo, JMP/Jxx/CALL en código x86). Las instrucciones desensambladas se procesan en secuencia. Como parte del proceso de normalización, algunas instrucciones se mantienen sin cambios y otras se alteran. Las alteraciones a modo de ejemplo incluyen:

a. se sustituyen los ID de registro, basándose en el orden en el que aparecen los mismos en el interior del bloque de función;

b. se eliminan los desplazamientos y valores constantes;

c. las secuencias de PUSH seguido de POP se sustituyen con instrucciones MOV;

d. Las secuencias que establecen el valor de una dirección de variable/registro/memoria a 0 (por ejemplo, XOR <elemento>, <elemento>) se sustituyen con MOV <elemento>, 0;

e. La adición/sustracción de 1 o 2 se sustituye con una o dos instrucciones INC/DEC, respectivamente.

f. Las instrucciones JZ/JNZ se sustituyen con instrucciones JE/JNE, respectivamente;

g. Se eliminan los prólogos y epílogos de función;

h. Se eliminan las clases de instrucción CMP, MOV y TEST;

i. Se eliminan las no-operaciones (ADD y SUB con 0; NOP etc.).

La figura 7 muestra un ejemplo de normalización de código, de acuerdo con algunas realizaciones de la presente invención. Un fragmento de código desensamblado de un objeto objetivo a modo de ejemplo comprende un bloque de función 70. En algunas realizaciones, los bloques de función comienzan con una secuencia de instrucciones PUSH EBP; MOV EBP, ESP y los mismos terminan con POP EBP. Cada línea de código (instrucción de procesador) a partir del bloque de función 70 se modifica de acuerdo con la indicación que aparece a la derecha, para producir un bloque de función normalizado 72 correspondiente.

En una etapa 216 (la figura 6), la aplicación AM de cliente 40 computa un indicador de datos de objeto (ODI) del objeto objetivo. En algunas realizaciones, el ODI comprende una pluralidad de indicadores de bloque de código, cada indicador de bloque de código indicativo de un bloque de código distinto del objeto objetivo. Un indicador de bloque de código a modo de ejemplo comprende un patrón de código de operación del bloque de código respectivo.

En algunas realizaciones, un bloque de código comprende una secuencia de instrucciones de procesador consecutivas, extraída la secuencia del código normalizado del objeto objetivo. En algunas realizaciones, los bloques de código comprenden un número de instrucciones previamente determinado e independiente del código. Como alternativa, el recuento de instrucciones dentro de un bloque de código varía dentro de un intervalo previamente determinado. Los bloques de código a modo de ejemplo comprenden entre 5 y 50 instrucciones contiguas. En algunas realizaciones, el tamaño (por ejemplo, número de instrucciones) de los bloques de código es sustancialmente más pequeño que el tamaño de los bloques de función, de tal modo que un bloque de función puede comprender más de un bloque de código. En algunas realizaciones, los bloques de código comienzan o bien al comienzo de un bloque de función, o bien en una instrucción CALL. Un bloque de código 74 a modo de ejemplo se muestra en la figura 7.

En algunas realizaciones, la etapa 216 comprende separar el objeto objetivo en bloques de código y extraer un conjunto de indicadores de código de operación a partir de cada uno de tales bloques de código. La figura 8 muestra una representación en memoria binaria a modo de ejemplo de una instrucción de procesador 80 (ilustrada para la familia de procesadores de 32 bits x86 de Intel®). En algunas realizaciones, cada instrucción de procesador se almacena en memoria como una secuencia de bytes, comprendiendo la secuencia un conjunto de campos de instrucción, tales como un campo de Prefijo 82a, una pareja de campos de Código de operación 82b-c, un campo de Mod/Reg/R/M 82d y un campo de Desplazamiento/Datos 82e. En algunas realizaciones, los campos de Código de operación 82b-c codifican el tipo de instrucción (por ejemplo, MOV, PUSH, etc.), mientras que los campos 82a, 82d-e codifican diversos parámetros de instrucción (por ejemplo, nombres de registro, direcciones de memoria, etc.) En algunas realizaciones, tales como el formato x86, el tamaño en bytes y el contenido de los campos de instrucción dependen de la instrucción y, por lo tanto, las instrucciones para la arquitectura x86 son de longitudes variables. La instrucción ilustrada en la figura 8 (XOR CL, 12H) comprende solo el primer byte de Código de operación (10000000 para XOR), el byte de Mod/Reg/R/M (11110001 para CL de registro) y el byte de Desplazamiento/Datos (00010010 es el equivalente binario de 12H), mientras que otras instrucciones pueden comprender ambos campos de Código de operación, u otras combinaciones de los campos de Prefijo, Código de operación, Mod, Reg y/o Datos.

La figura 9 muestra un patrón de código de operación 90 a modo de ejemplo que se corresponde con el bloque de código 74. En algunas realizaciones, el patrón de código de operación 90 es una estructura de datos (por ejemplo, secuencia de bytes, lista, etc.) que comprende un conjunto de indicadores de código de operación 92, correspondiéndose cada indicador de código de operación con una instrucción de procesador del bloque de código normalizado 74. Los indicadores de código de operación 92 a modo de ejemplo comprenden los contenidos de los campos de código de operación de la instrucción de procesador respectiva, caso en el cual el patrón de código de operación 90 comprende una secuencia de tipos de instrucción que constituyen el bloque de código respectivo. En la realización ilustrada en la figura 9, cada indicador de código de operación 92 comprende una combinación de bytes de código de operación y bytes de parámetro (por ejemplo, el indicador de código de operación para la instrucción PUSH EDX es 52 en hexadecimal).

La figura 10 ilustra un fragmento de código normalizado y un ODI 100 a modo de ejemplo del fragmento, de acuerdo con algunas realizaciones de la presente. El ODI 100 comprende una pluralidad de indicadores de bloque de código 104a-c, proporcionando cada indicador de bloque de código un resumen (por ejemplo, huella, firma) de un bloque de código 74a-c respectivo. Un indicador de bloque de código 104a-c a modo de ejemplo comprende el patrón de código de operación 90a-c respectivo. En algunas realizaciones, los indicadores de bloque de código 104a-c comprenden troceos de los patrones de código de operación 90a-c, respectivamente, tal como se ilustra en la figura 10. Aparte de los indicadores de bloque de código 104a-c, algunas realizaciones del ODI 100 pueden comprender un identificador de objeto 102 (por ejemplo, un ID de archivo) que etiqueta el objeto objetivo respectivo, y/o un conjunto de indicadores de característica de objeto 106 del objeto objetivo. Los indicadores de característica de objeto a modo de ejemplo comprenden un tamaño de archivo (por ejemplo, 130 kB), un indicador de tipo de archivo (por ejemplo, si un archivo es un ejecutable, una DLL, etc.), una dirección de memoria del objeto objetivo y un conjunto de números que indican un resultado de un conjunto de pruebas heurísticas anti-software maligno (por ejemplo, si el objeto objetivo presenta determinados comportamientos o contenidos específicos de software maligno), entre otros. En algunas realizaciones, los indicadores de característica de objeto 106 pueden ser computados por las unidades de exploración AM 42-44, por ejemplo, durante la exploración preliminar del objeto objetivo (la etapa 202).

Por razones de simplicidad, el resto de la memoria descriptiva supondrá que los indicadores de bloque de código 104a-c comprenden troceos de los patrones de código de operación 90a-c. La ejecución de la etapa 216 (la figura 6) avanza entonces tal como sigue. La aplicación AM de cliente 40 puede separar el objeto objetivo en bloques de código distintos (ilustrados mediante los bloques de código 74a-c en la figura 10). Para cada bloque de código 74a-c, la aplicación 40 puede proceder a calcular un patrón de código de operación 90a-c, respectivamente, tal como se muestra en la figura 9. La aplicación 40 puede invocar entonces el motor de cálculo de troceo 54 para computar un troceo del patrón de código de operación 90a-c, para producir el indicador de bloque de código (es decir, el troceo objetivo) 104a-c respectivo. El motor de cálculo de troceo 54 puede emplear un algoritmo de cálculo de troceo tal como comprobación de redundancia cíclica (CRC), resumen de mensaje (MD) o cálculo de troceo seguro (SHA), entre otros.

Después de computar el ODI objetivo 100, en una etapa 218 (la figura 6), la aplicación AM de cliente 40 realiza una consulta del ODI en la memoria caché de lado de cliente 56. Si el ODI coincide con un registro de memoria caché (coincidencia de memoria caché), indicando que el objeto objetivo respectivo ya se ha explorado en busca de software maligno al menos una vez, la aplicación 40 avanza a una etapa 220, para etiquetar el objeto objetivo de acuerdo con el registro de memoria caché (por ejemplo, limpio o software maligno) y avanza a la etapa 232 analizada en lo que antecede.

Si no se encuentra una coincidencia con el ODI objetivo 100 en la memoria caché de lado de cliente 56, en una etapa 222, la aplicación 40 puede invocar el gestor de comunicación AM de cliente 52 para iniciar una transacción de exploración cliente-servidor. El gestor de comunicación 52 transmite el ODI objetivo 100 a los servidores AM 20a-c y, en una etapa 224, recibe el informe de exploración 50 a partir de los servidores 20a-c. En algunas realizaciones, cada ODI puede formar parte de una transacción de exploración cliente-servidor distinta, o múltiples ODI se pueden transmitir de forma simultánea, dentro de la misma transacción (procesamiento por lotes).

En una etapa 226, la aplicación 40 determina si el objeto objetivo está incluido en lista blanca (limpio) de acuerdo con el informe de exploración 50. En caso afirmativo, el objeto objetivo se etiqueta como no malicioso (la etapa 228). Si el objeto objetivo es malicioso de acuerdo con el informe de exploración 50, la aplicación 40 etiqueta el objeto objetivo como software maligno (la etapa 208).

La figura 11 muestra una secuencia a modo de ejemplo de etapas realizadas por la aplicación AM de servidor 60 (la figura 5) de acuerdo con algunas realizaciones de la presente invención. En una etapa 302, el gestor de comunicación AM de servidor 62 recibe el ODI objetivo 100 a partir del sistema informático de cliente 30. En una etapa 304, la aplicación 60 realiza una consulta del ODI 100 en la memoria caché de lado de servidor 68. Si el ODI coincide con un registro de memoria caché (coincidencia de memoria caché), indicando que el objeto objetivo respectivo ya se ha explorado en busca de software maligno al menos una vez, la aplicación 60 avanza a una etapa 306, para etiquetar el objeto objetivo de acuerdo con el registro de memoria caché (por ejemplo, limpio o software maligno). En una etapa 308, el gestor de comunicación 62 compila el informe de exploración 50 y transmite el informe 50 al sistema informático de cliente 30 respectivo.

Si no se halla registro alguno del ODI 100 en la memoria caché de lado de servidor 68, en una etapa 310, la aplicación AM de servidor 60 filtra los troceos del ODI 100 para producir un subconjunto relevante de troceos. En algunas realizaciones, los troceos de los patrones de código de operación que no son específicos de objeto se pueden descartar del ODI 100 para mejorar el desempeño de la exploración de software maligno. Tales patrones de código de operación no específicos se corresponden, por ejemplo, con un desempaquetador de código (por ejemplo, instalador, auto-extractor) y/o código de biblioteca, o se encuentran presentes en objetos tanto limpios como de software maligno.

En una etapa 312, para cada troceo del ODI 100, la aplicación AM de servidor 60 puede consultar la base de datos de lista blanca 65 para recuperar un conjunto de objetos de referencia incluidos en lista blanca que contienen el troceo respectivo. En algunas realizaciones, se usa un algoritmo basado en montón para clasificar los objetos de referencia recuperados de acuerdo con su similitud con el objeto objetivo.

En una etapa 314, la aplicación AM de servidor 60 invoca el comparador de código 64 para computar una puntuación de similitud que caracteriza lo similar que es el objeto objetivo a cada objeto de referencia incluido en lista blanca recuperado en la etapa 312. En algunas realizaciones, la puntuación de similitud se computa de acuerdo con la fórmula:

$$S = 100 * \frac{C}{\text{máx}\,(N_T + N_R)}, \qquad [1]$$

en donde C indica el número (recuento) de troceos comunes tanto al objeto objetivo como al objeto de referencia respectivo, N_T indica el número (recuento) de troceos del ODI objetivo, filtrados tal como se analiza en la etapa 310 en lo que antecede, y en donde N_R indica el número (recuento) de troceos del objeto de referencia.

Realizaciones alternativas pueden computar la puntuación de similitud de acuerdo con fórmulas tales como:

$$S = 200 * \frac{C}{N_T + N_R} \quad [2]$$

o

$$S = 50 * (\frac{C}{N_T} + \frac{C}{N_R}). \quad [3]$$

En una etapa 316, la aplicación 60 compara la puntuación de similitud (por ejemplo, la fórmula [1]) con un umbral previamente determinado. Cuando la puntuación de similitud supera el umbral, indicando que el objeto objetivo es similar a al menos un objeto incluido en lista blanca, algunas realizaciones de la aplicación AM de servidor 60 pueden etiquetar el objeto objetivo como no malicioso (limpio) en una etapa 318. Un valor a modo de ejemplo del umbral de inclusión en lista blanca es 50, indicando que un objeto objetivo está incluido en lista blanca cuando el mismo comparte más de un 50 % de sus patrones de código de operación con un objeto incluido en lista blanca.

A continuación, una etapa 320 actualiza la base de datos de lista blanca 65 con un registro del objeto objetivo actual, y una etapa 322 actualiza la memoria caché de lado de servidor 68 con un registro del objeto objetivo y un indicador del resultado de exploración (por ejemplo, limpio).

Cuando la puntuación de similitud de inclusión en lista blanca (la etapa 318) no supera el umbral, indicando que el objeto objetivo no es suficientemente similar a objeto incluido en lista blanca conocido alguno, la aplicación AM de servidor avanza a una etapa 324, en donde el ODI objetivo 100 se compara con un conjunto de registros de objetos de software maligno. En algunas realizaciones, el conjunto de troceos del ODI 100 se filtra adicionalmente para eliminar todos los troceos que coincidieron con registros a partir de la base de datos de lista blanca 65 (véase la etapa 312 en lo que antecede), conservando por lo tanto un subconjunto de troceos que no se hallan en objeto incluido en lista blanca conocido alguno. Para cada uno de tales troceos no reconocidos del objeto objetivo, el comparador de código 64 puede consultar las bases de datos de software maligno y/o de irrupciones 66-67 para recuperar un conjunto de objetos de software maligno que contienen el troceo respectivo. En una etapa 326, el comparador de código 64 puede proceder entonces a computar una puntuación de similitud de software maligno que indica lo similar que es el objeto objetivo a cada uno de tales objetos de software maligno. En algunas realizaciones, el comparador de código 64 usa cualquiera de las fórmulas [1-3] descritas anteriormente para computar la puntuación de similitud de software maligno.

Una etapa 328 compara la puntuación de similitud de software maligno con un umbral previamente establecido. Cuando la puntuación de similitud de software maligno supera el umbral, indicando que el objeto objetivo es similar a al menos un objeto de software maligno almacenado en las bases de datos 66-67, en una etapa 330, el objeto objetivo se etiqueta como software maligno. Un umbral a modo de ejemplo para la clasificación como software maligno es 70 (es decir, el objeto objetivo comparte al menos un 70 % de los patrones de código de operación con un objeto de software maligno conocido). A continuación, las bases de datos de software maligno y/o de irrupciones 66-67 se actualizan para incluir un registro del objeto objetivo. La memoria caché de lado de servidor 68 se actualiza para incluir un registro del objeto objetivo y un indicador de su estatus de software maligno (por ejemplo, infectado), y un informe de exploración se compila y se transmite al sistema informático de cliente (la etapa 308).

Cuando la puntuación de similitud de software maligno no supera el umbral, indicando que el objeto objetivo no es similar a objetos de software maligno conocidos, algunas realizaciones de las aplicaciones AM de servidor pueden etiquetar el objeto objetivo como incluido en lista blanca/no malicioso (la etapa 318) y actualizar la base de datos de lista blanca 65 en consecuencia.

El ODI objetivo 100 también puede desencadenar una alerta de irrupción de software maligno. En algunas realizaciones, la aplicación AM de servidor 60 cuenta los objetos de referencia procedentes de la base de datos de irrupciones 67, objetos que son similares al objeto objetivo y han sido recibidos por los sistemas de servidor AM 20a-c dentro de un periodo de tiempo previamente determinado (por ejemplo, las últimas 6 horas). Cuando el recuento supera un umbral (por ejemplo, 10), se supone una irrupción de software maligno y el objeto objetivo, así como todos los objetos de referencia similares al mismo, se marcan como infectados. Las bases de datos de software maligno y/o de irrupciones 66-67 se actualizan entonces en consecuencia.

Los sistemas y métodos a modo de ejemplo descritos anteriormente permiten que un sistema anti-software maligno mantenga una base de datos de lista blanca flexible y que use la base de datos de lista blanca para mejorar el desempeño de la detección de software maligno.

En las aplicaciones de inclusión en lista blanca convencionales, un troceo de un objeto objetivo (proceso o archivo

informático) se compara con un conjunto de troceos que se corresponden con objetos incluidos en lista blanca (objetos de los que se tiene confianza en que están limpios). Si el troceo del objeto objetivo coincide con un troceo incluido en lista blanca, indicando que el objeto objetivo es idéntico a al menos uno de los objetos incluidos en lista blanca, el objeto objetivo es de confianza y, por ejemplo, se permite su ejecución. Debido a determinadas propiedades matemáticas de las funciones de troceo, la inclusión en lista blanca convencional no prevé variaciones en el código de objetos incluidos en lista blanca: si dos objetos difieren tan poco como un bit, los troceos de los dos objetos ya no coinciden. Mientras tanto, los procesos y archivos informáticos legítimos pueden presentar variaciones sustanciales, debido, por ejemplo, a diferencias entre compiladores o entre versiones sucesivas del mismo software.

Algunas realizaciones de los sistemas y métodos descritos anteriormente permiten que un sistema anti-software maligno dé cuentas de diferencias benignas entre objetos de datos, tales como diferencias introducidas por compiladores y otros polimorfismos. Un objeto objetivo se separa en una multitud de bloques de código, y se calcula un troceo para cada bloque de código. El conjunto obtenido de troceos objetivo se compara entonces con una base de datos de troceos que se corresponden con bloques de código extraídos de objetos incluidos en lista blanca. Un objeto objetivo se puede etiquetar como incluido en lista blanca (de confianza) si el mismo tiene un número sustancial de troceos en común con un objeto incluido en lista blanca. Objetos que son ligeramente diferentes de objetos incluidos en lista blanca conocidos pueden seguir recibiendo un estatus de inclusión en lista blanca. Al permitir un determinado grado de falta de coincidencia entre los conjuntos de troceos de objetos distintos, algunas realizaciones de la presente invención aumentan la eficiencia de la inclusión en lista blanca sin una disminución no aceptable en la seguridad de los datos.

El tamaño de un bloque de código se puede decidir de acuerdo con varios criterios. Los bloques de código pequeños (por ejemplo, de unas pocas de instrucciones de procesador cada uno) pueden conducir a un número grande de troceos por objeto objetivo, lo que puede aumentar la carga de almacenamiento y de procesamiento del servidor anti-software maligno y ralentizar la exploración. Por otro lado, los bloques de código pequeños ofrecen un grado significativo de flexibilidad: si dos objetos difieren solo ligeramente, las diferencias serán captadas solo por una pequeña fracción de troceos, produciendo una puntuación de similitud alta. Los bloques de código grandes (por ejemplo, varios cientos de instrucciones de procesador) producen como promedio menos (por ejemplo, varios) troceos por objeto objetivo y, por lo tanto, son ventajosos desde una perspectiva de almacenamiento y de procesamiento. No obstante, los bloques de código grandes adolecen de la misma desventaja que el cálculo de troceo convencional: diferencias pequeñas entre dos objetos pueden ser captadas por una proporción grande de troceos, produciendo una puntuación de similitud baja. La realización de pruebas reveló un tamaño de bloque de código óptimo de entre 5 y 50 instrucciones de procesador y, en particular, aproximadamente 5-15 (por ejemplo, -10) instrucciones en algunas realizaciones.

Los sistemas y métodos a modo de ejemplo descritos anteriormente permiten que un sistema anti-software maligno realice una transacción de exploración cliente-servidor colaborativa y evalúe el estatus de software maligno del objeto objetivo de acuerdo con los resultados de la exploración de lado de servidor del objeto objetivo. Llevar a cabo una parte de la exploración de software maligno en un servidor anti-software maligno remoto tiene un número de ventajas frente a la exploración local de objetos objetivo en un sistema informático de cliente.

La proliferación de agentes de software maligno y software en general ha contribuido a un aumento sostenido en el tamaño de las bases de datos de lista blanca y de troceos de software maligno, que pueden ascender de varios megabytes a varios gigabytes de datos. Los métodos y sistemas a modo de ejemplo descritos anteriormente permiten almacenar las bases de datos de troceos en el servidor anti-software maligno, evitando de ese modo la entrega de actualizaciones de software de gran volumen de datos procedentes de un servidor corporativo a un número grande de clientes de forma regular.

Al realizar una fracción significativa de la exploración de software maligno de forma central en el servidor, los sistemas y métodos descritos anteriormente prevén la incorporación oportuna de troceos de software maligno recientemente detectado y de software legítimo nuevo. En contraposición, en la detección de software maligno convencional en donde la exploración se distribuye predominantemente a los sistemas informáticos de cliente, la recopilación de información acerca de nuevas amenazas de seguridad y nuevo software incluido en lista blanca puede implicar métodos indirectos, llevando significativamente más tiempo alcanzar a los productores de software anti-software maligno.

El tamaño de los archivos intercambiados entre los sistemas de cliente y de servidor anti-software maligno descritos anteriormente se mantiene a un mínimo. En lugar de enviar objetos objetivo completos del cliente al servidor para la exploración de lado de servidor, los métodos y sistemas a modo de ejemplo descritos anteriormente están configurados para intercambiar troceos, que pueden ascender de varios bytes a varios kilobytes por objeto objetivo, reduciendo de ese modo, significativamente, el tráfico de red.

A un experto en la materia le resultará evidente que las realizaciones anteriores se pueden alterar en muchas formas sin apartarse del alcance de la invención. En consecuencia, el alcance de la invención se debería determinar mediante las siguientes reivindicaciones y sus equivalentes legales.

REIVINDICACIONES

1. Un método que comprende:

recibir, en un sistema informático de servidor, una pluralidad de troceos objetivo 104a-c de un objeto objetivo 70, representando cada troceo objetivo 104a-c un bloque de código 74a-c distinto del objeto objetivo 70, consistiendo cada bloque de código 74a-c distinto en una secuencia de instrucciones de procesador del objeto objetivo 70;
para al menos un troceo objetivo de la pluralidad de troceos objetivo 104a-c, emplear el sistema informático de servidor para:

recuperar una pluralidad de troceos de referencia de un objeto de referencia, seleccionado el objeto de referencia de entre un conjunto de objetos incluidos en lista blanca de acuerdo con el troceo objetivo, y
cuando la pluralidad de troceos objetivo 104a-c no es idéntica a la pluralidad de troceos de referencia, determinar una puntuación de similitud de acuerdo con un recuento de troceos comunes tanto a la pluralidad de troceos objetivo 104a-c como a la pluralidad de troceos de referencia; y

cuando la puntuación de similitud supera un umbral previamente determinado, emplear el sistema informático de servidor para etiquetar el objeto objetivo 70 como no malicioso.

2. El método de la reivindicación 1, en donde el troceo objetivo comprende un troceo de un patrón de código de operación 90, comprendiendo el patrón de código de operación 90 una secuencia de indicadores de instrucción, cada indicador de instrucción indicativo de una instrucción de procesador del bloque de código distinto.

3. El método de la reivindicación 1, en donde la secuencia de instrucciones de procesador consiste en entre 5 y 50 instrucciones de procesador consecutivas.

4. El método de la reivindicación 1, que comprende adicionalmente:

realizar un procedimiento de normalización de código en el objeto objetivo 70 para producir un objeto objetivo normalizado 72, y en donde cada bloque de código 74a-c distinto consiste en una secuencia de instrucciones informáticas del objeto objetivo normalizado 72; y
aplicar una función de troceo al bloque de código distinto para producir el troceo objetivo.

5. Un sistema informático que comprende al menos un procesador programado para:

recibir una pluralidad de troceos objetivo 104a-c, representando cada troceo objetivo 104a-c un bloque de código 74a-c distinto de un objeto objetivo 70, consistiendo cada bloque de código 74a-c distinto en una secuencia de instrucciones de procesador del objeto objetivo 70;
para al menos un troceo objetivo de la pluralidad de troceos objetivo 104a-c:

recuperar una pluralidad de troceos de referencia de un objeto de referencia, seleccionado el objeto de referencia de entre un conjunto de objetos incluidos en lista blanca de acuerdo con el troceo objetivo, y
cuando la pluralidad de troceos objetivo 104a-c no es idéntica a la pluralidad de troceos de referencia, determinar una puntuación de similitud de acuerdo con un recuento de troceos comunes tanto a la pluralidad de troceos objetivo 104a-c como a la pluralidad de troceos de referencia; y

cuando la puntuación de similitud supera un umbral previamente determinado, etiquetar el objeto objetivo 70 como no malicioso.

6. El sistema de la reivindicación 5, en donde el troceo objetivo comprende un troceo de un patrón de código de operación 90, comprendiendo el patrón de código de operación 90 una secuencia de indicadores de instrucción, cada indicador de instrucción indicativo de una instrucción de procesador del bloque de código distinto.

7. El sistema de la reivindicación 5, en donde la secuencia de instrucciones de procesador consiste en entre 5 y 50 instrucciones de procesador consecutivas.

8. El sistema de la reivindicación 7, en donde la secuencia de instrucciones de procesador consiste en entre 5 y 15 instrucciones de procesador consecutivas.

9. El sistema de la reivindicación 5, en donde la secuencia de instrucciones de procesador comienza con una instrucción CALL.

10. El sistema de la reivindicación 5, en donde el procesador está programado adicionalmente para:

realizar un procedimiento de normalización de código en el objeto objetivo 70 para producir un objeto objetivo normalizado 72, en donde cada bloque de código 74a-c distinto consiste en una secuencia de instrucciones

informáticas del objeto objetivo normalizado 72; y
aplicar una función de troceo al bloque de código distinto para producir el troceo objetivo.

11. El sistema de la reivindicación 5, en donde la puntuación de similitud se determina como una función de:

$$C/\text{máx}\ (N_T, N_R)$$

en donde C indica el recuento de troceos comunes tanto a la pluralidad de troceos objetivo 104a-c como a la pluralidad de troceos de referencia, mientras que N_T y N_R indican la cardinalidad de la pluralidad de troceos objetivo 104a-c y la cardinalidad de la pluralidad de troceos de referencia, respectivamente.

12. El sistema de la reivindicación 5, en donde la puntuación de similitud se determina como una función de:

$$C/(N_T + N_R)$$

en donde *C* indica el recuento de troceos comunes tanto a la pluralidad de troceos objetivo 104a-c como a la pluralidad de troceos de referencia, mientras que N_T y N_R indican la cardinalidad de la pluralidad de troceos objetivo 104a-c y la cardinalidad de la pluralidad de troceos de referencia, respectivamente.

13. El sistema de la reivindicación 5, en donde la puntuación de similitud se determina como una función de:

$$C/N_T + C/N_R$$

en donde *C* indica el recuento de troceos comunes tanto a la pluralidad de troceos objetivo 104a-c como a la pluralidad de troceos de referencia, mientras que N_T y N_R indican la cardinalidad de la pluralidad de troceos objetivo 104a-c y la cardinalidad de la pluralidad de troceos de referencia, respectivamente.

14. El sistema de la reivindicación 5, en donde el objeto objetivo 70 comprende un archivo informático.

15. El sistema de la reivindicación 5, en donde el objeto objetivo 70 comprende un proceso informático.

16. Un medio de almacenamiento legible por ordenador no transitorio que codifica unas instrucciones que, cuando se ejecutan en un procesador, dan lugar a que procesador realice las etapas de:

recibir una pluralidad de troceos objetivo 104a-c, representando cada troceo objetivo 104a-c un bloque de código 74a-c distinto de un objeto objetivo 70, consistiendo cada bloque de código 74a-c distinto en una secuencia de instrucciones de procesador del objeto objetivo 70;
para al menos un troceo objetivo de la pluralidad de troceos objetivo 104a-c:

recuperar una pluralidad de troceos de referencia de un objeto de referencia, seleccionado el objeto de referencia de entre un conjunto de objetos incluidos en lista blanca de acuerdo con el troceo objetivo, y
cuando la pluralidad de troceos objetivo 104a-c no es idéntica a la pluralidad de troceos de referencia, determinar una puntuación de similitud de acuerdo con un recuento de troceos comunes tanto a la pluralidad de troceos objetivo 104a-c como a la pluralidad de troceos de referencia; y

cuando la puntuación de similitud supera un umbral previamente determinado, etiquetar el objeto objetivo 70 como no malicioso.

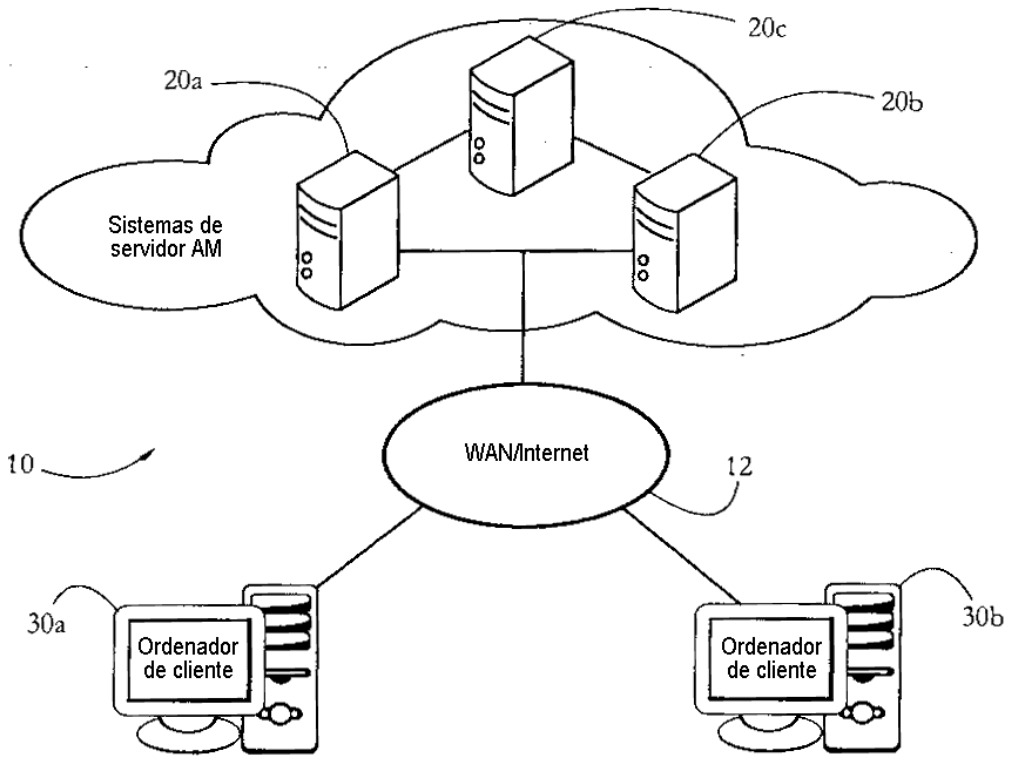

20c
20a
20b
Sistemas de servidor AM
WAN/Internet
10
12
30a
Ordenador de cliente
Ordenador de cliente
30b

FIG. 1

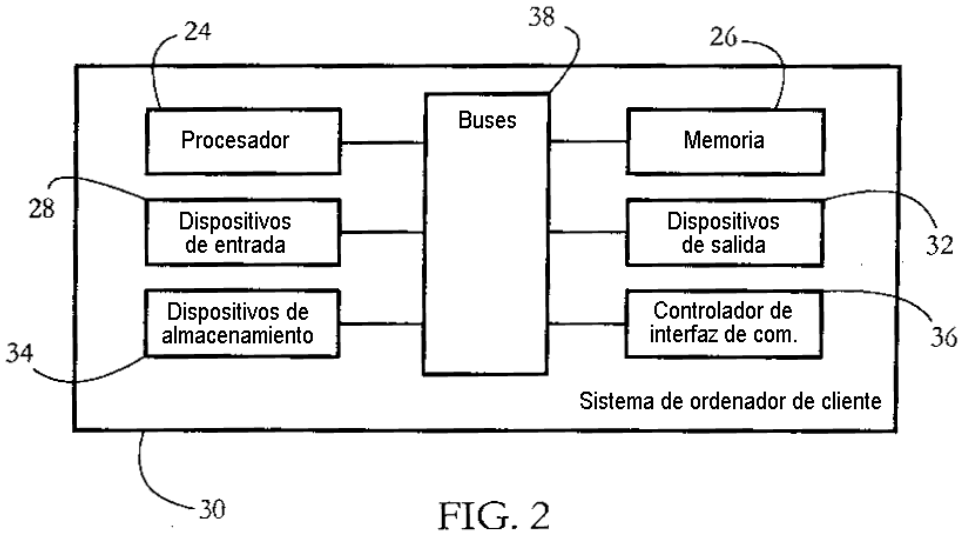

24
38
26
Procesador
Buses
Memoria
28
Dispositivos de entrada
Dispositivos de salida
32
Dispositivos de almacenamiento
Controlador de interfaz de com.
34
36
Sistema de ordenador de cliente
30

FIG. 2

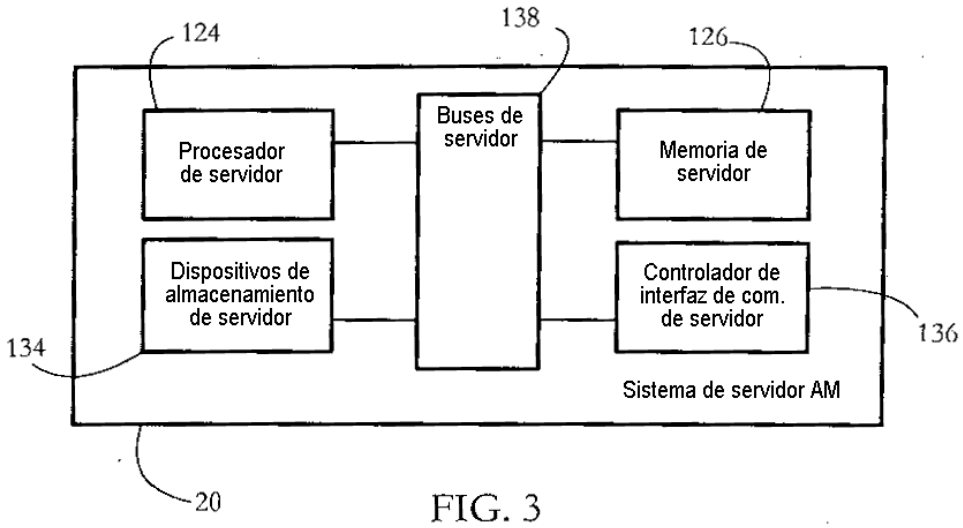

124
138
126
Procesador de servidor
Buses de servidor
Memoria de servidor
Dispositivos de almacenamiento de servidor
Controlador de interfaz de com. de servidor
136
134
Sistema de servidor AM
20

FIG. 3

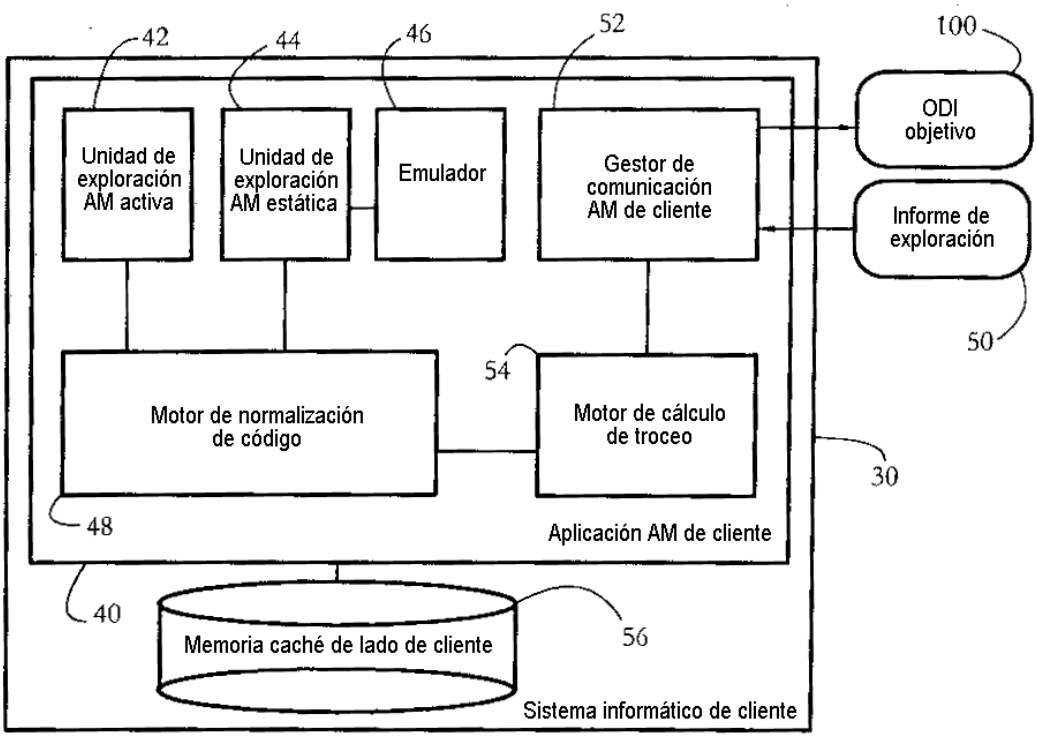

42
44
46
52
100
Unidad de exploración AM activa
Unidad de exploración AM estática
Emulador
Gestor de comunicación AM de cliente
ODI objetivo
Informe de exploración
50
54
Motor de normalización de código
Motor de cálculo de troceo
30
48
Aplicación AM de cliente
40
Memoria caché de lado de cliente
56
Sistema informático de cliente

FIG. 4

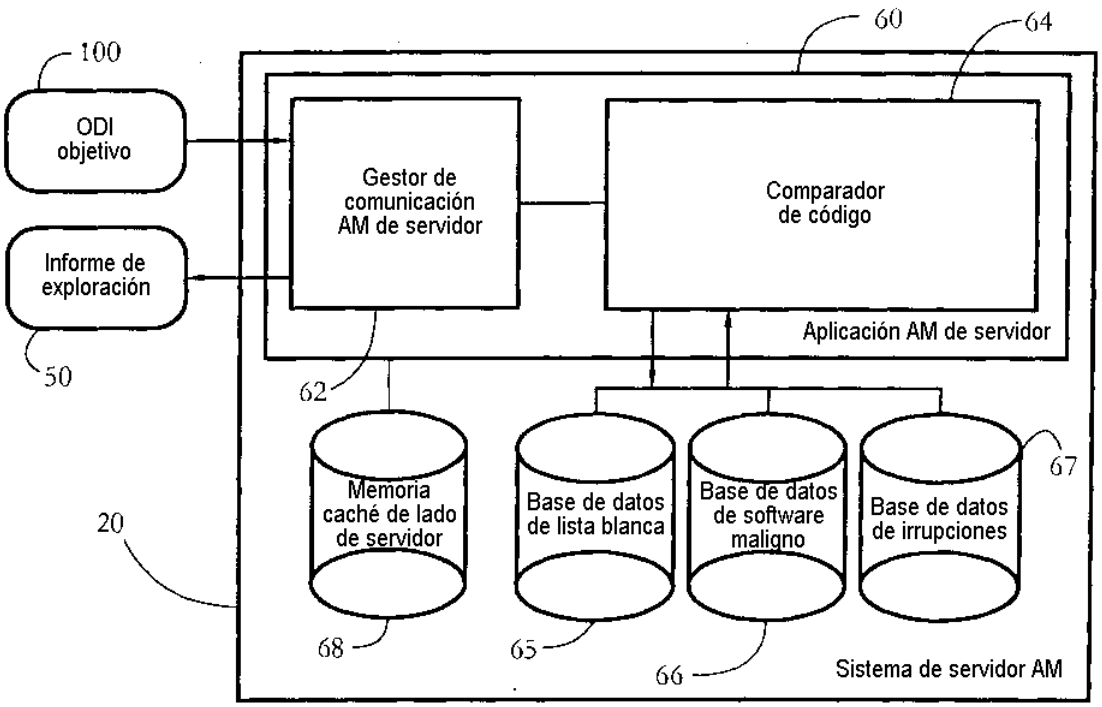
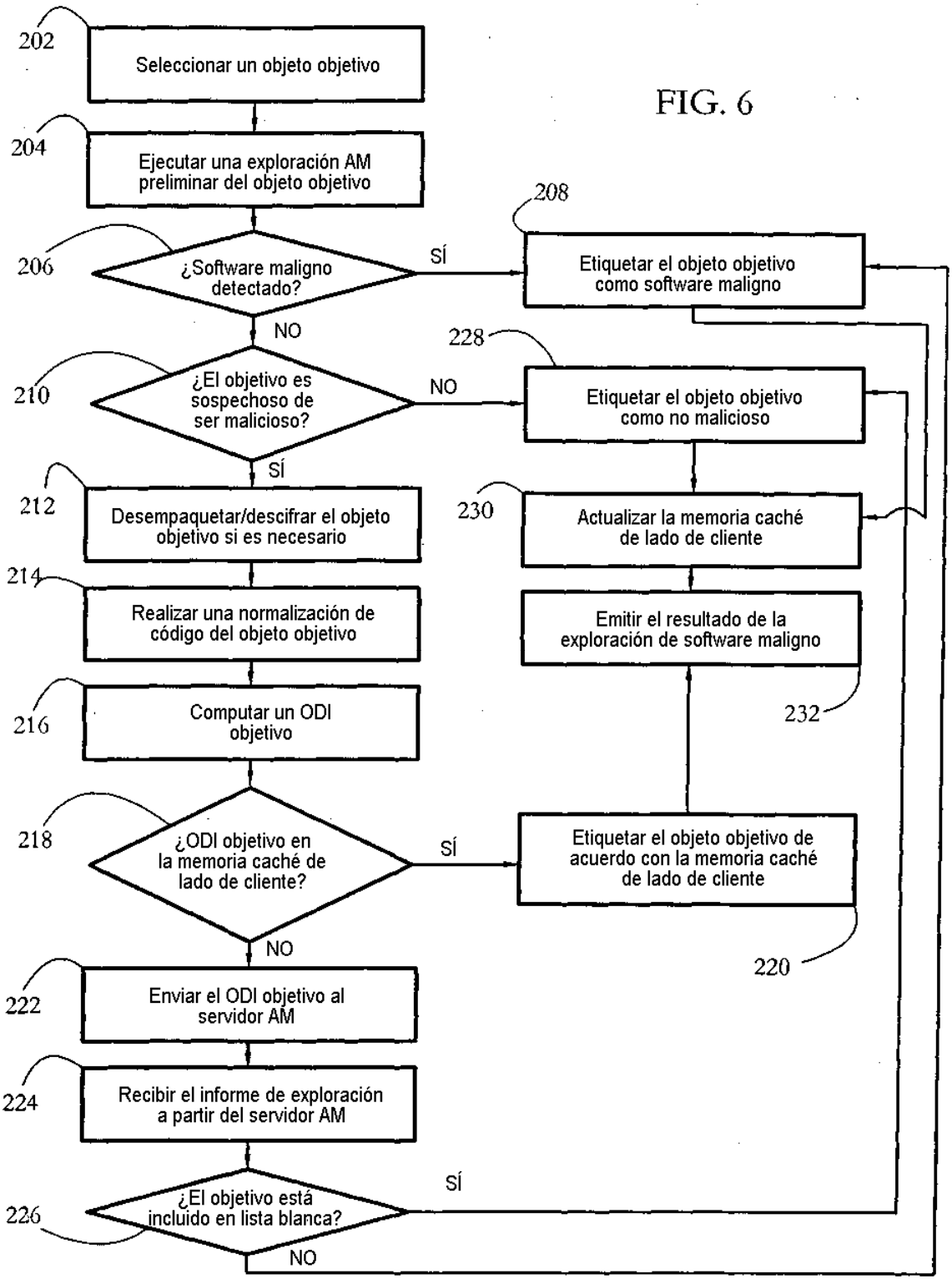

100
ODI objetivo
Informe de exploración
50
60
64
Gestor de comunicación AM de servidor
Comparador de código
Aplicación AM de servidor
62
20
Memoria caché de lado de servidor
Base de datos de lista blanca
Base de datos de software maligno
Base de datos de irrupciones
67
68
65
66
Sistema de servidor AM

FIG. 5

FIG. 6


202
Seleccionar un objeto objetivo
204
Ejecutar una exploración AM preliminar del objeto objetivo
206
¿Software maligno detectado?
SÍ
208
Etiquetar el objeto objetivo como software maligno
NO
210
¿El objetivo es sospechoso de ser malicioso?
NO
228
Etiquetar el objeto objetivo como no malicioso
SÍ
212
Desempaquetar/descifrar el objeto objetivo si es necesario
230
Actualizar la memoria caché de lado de cliente
214
Realizar una normalización de código del objeto objetivo
Emitir el resultado de la exploración de software maligno
216
Computar un ODI objetivo
232
218
¿ODI objetivo en la memoria caché de lado de cliente?
SÍ
Etiquetar el objeto objetivo de acuerdo con la memoria caché de lado de cliente
220
NO
222
Enviar el ODI objetivo al servidor AM
224
Recibir el informe de exploración a partir del servidor AM
SÍ
226
¿El objetivo está incluido en lista blanca?
NO

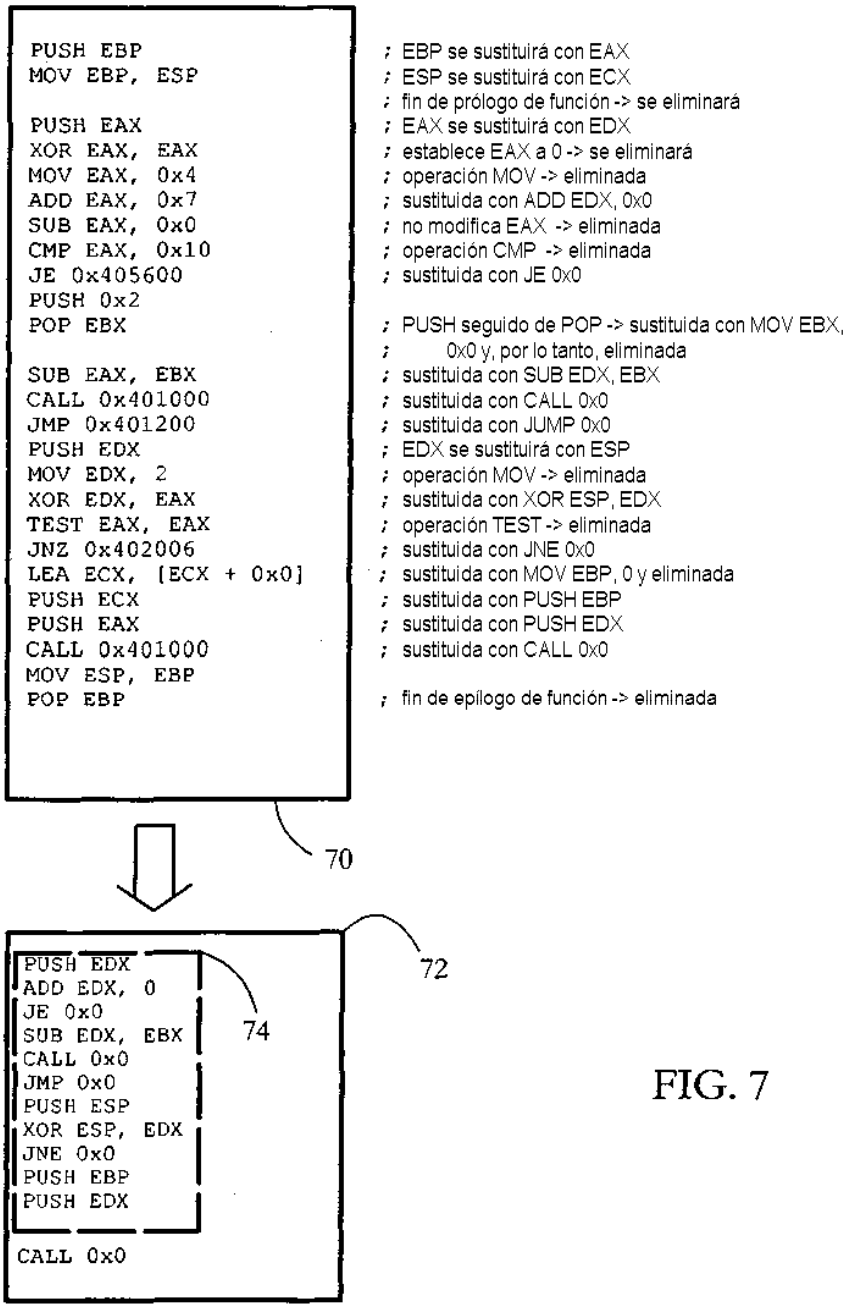

PUSH EBP ; EBP se sustituirá con EAX
MOV EBP, ESP ; ESP se sustituirá con ECX
; fin de prólogo de función -> se eliminará
PUSH EAX ; EAX se sustituirá con EDX
XOR EAX, EAX ; establece EAX a 0 -> se eliminará
MOV EAX, 0x4 ; operación MOV -> eliminada
ADD EAX, 0x7 ; sustituida con ADD EDX, 0x0
SUB EAX, 0x0 ; no modifica EAX -> eliminada
CMP EAX, 0x10 ; operación CMP -> eliminada
JE 0x405600 ; sustituida con JE 0x0
PUSH 0x2
POP EBX ; PUSH seguido de POP -> sustituida con MOV EBX,
; 0x0 y, por lo tanto, eliminada
SUB EAX, EBX ; sustituida con SUB EDX, EBX
CALL 0x401000 ; sustituida con CALL 0x0
JMP 0x401200 ; sustituida con JUMP 0x0
PUSH EDX ; EDX se sustituirá con ESP
MOV EDX, 2 ; operación MOV -> eliminada
XOR EDX, EAX ; sustituida con XOR ESP, EDX
TEST EAX, EAX ; operación TEST -> eliminada
JNZ 0x402006 ; sustituida con JNE 0x0
LEA ECX, [ECX + 0x0] ; sustituida con MOV EBP, 0 y eliminada
PUSH ECX ; sustituida con PUSH EBP
PUSH EAX ; sustituida con PUSH EDX
CALL 0x401000 ; sustituida con CALL 0x0
MOV ESP, EBP
POP EBP ; fin de epílogo de función -> eliminada
70
72
PUSH EDX
ADD EDX, 0
JE 0x0
SUB EDX, EBX
CALL 0x0
JMP 0x0
PUSH ESP
XOR ESP, EDX
JNE 0x0
PUSH EBP
PUSH EDX
74
CALL 0x0

FIG. 7

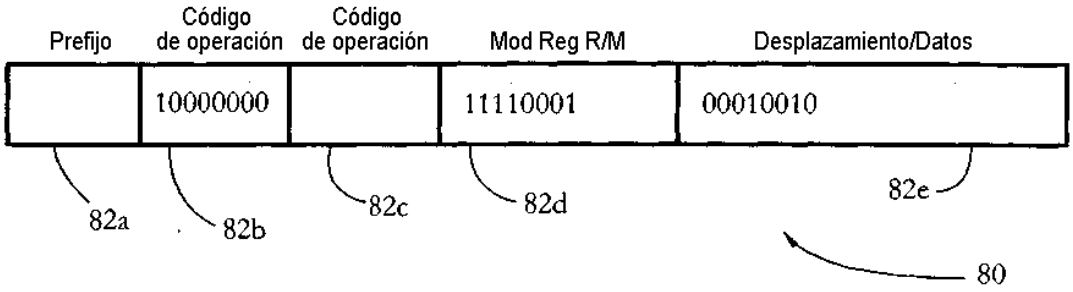

Prefijo
Código de operación
Código de operación
Mod Reg R/M
Desplazamiento/Datos
10000000
11110001
00010010
82a
82b
82c
82d
82e
80

FIG. 8

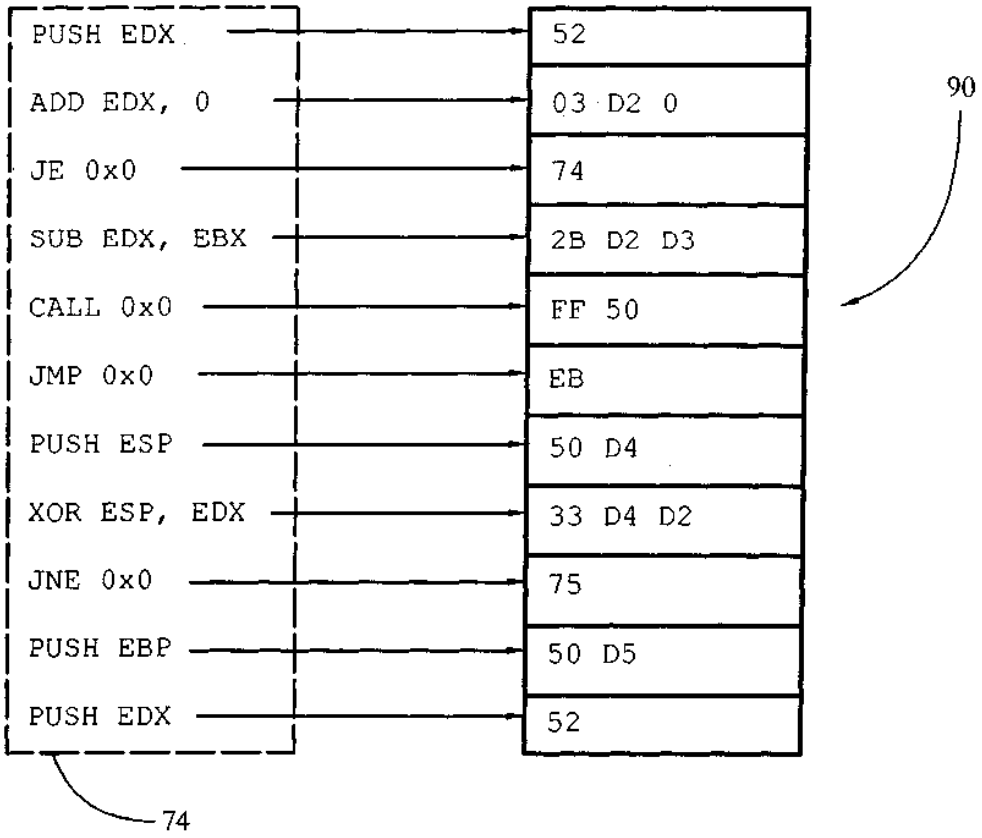

PUSH EDX
ADD EDX, 0
JE 0x0
SUB EDX, EBX
CALL 0x0
JMP 0x0
PUSH ESP
XOR ESP, EDX
JNE 0x0
PUSH EBP
PUSH EDX
52
03 D2 0
74
2B D2 D3
FF 50
EB
50 D4
33 D4 D2
75
50 D5
52
90
74

FIG. 9

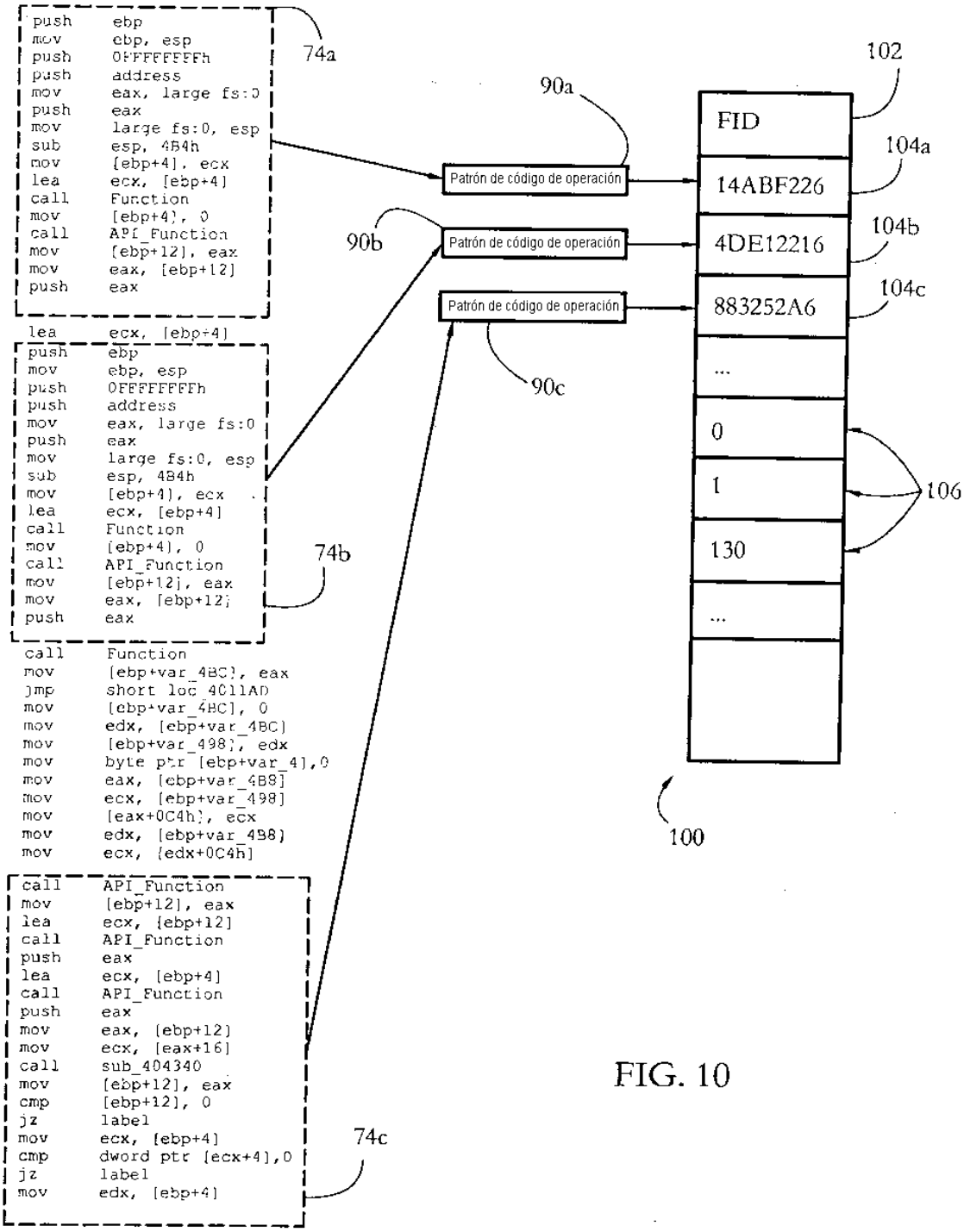
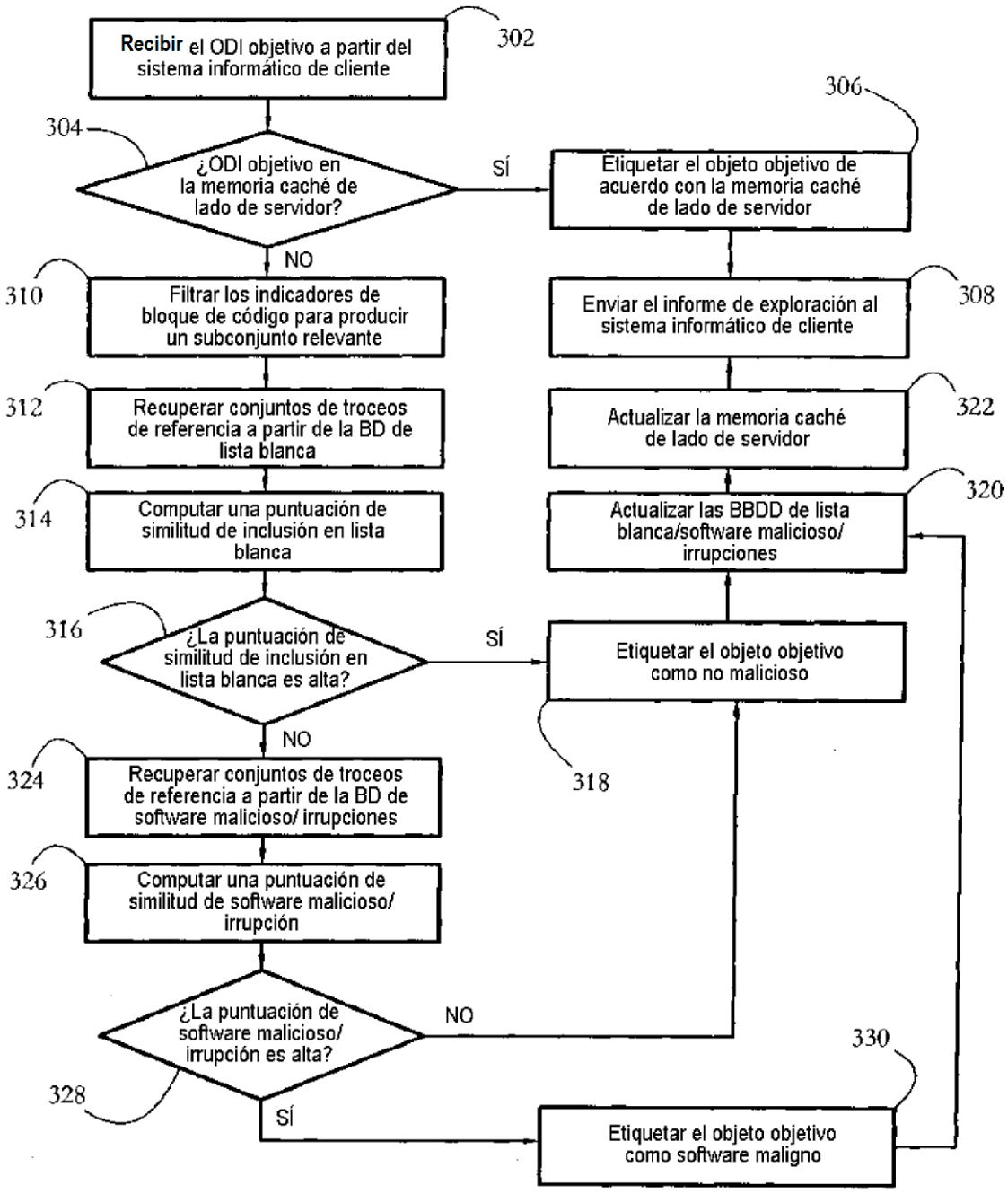

push ebp
mov ebp, esp
push 0FFFFFFFFh
push address
mov eax, large fs:0
push eax
mov large fs:0, esp
sub esp, 4B4h
mov [ebp+4], ecx
lea ecx, [ebp+4]
call Function
mov [ebp+4], 0
call API_Function
mov [ebp+12], eax
mov eax, [ebp+12]
push eax
74a
lea ecx, [ebp+4]
push ebp
mov ebp, esp
push 0FFFFFFFFh
push address
mov eax, large fs:0
push eax
mov large fs:0, esp
sub esp, 4B4h
mov [ebp+4], ecx
lea ecx, [ebp+4]
call Function
mov [ebp+4], 0
call API_Function
mov [ebp+12], eax
mov eax, [ebp+12]
push eax
74b
call Function
mov [ebp+var_4BC], eax
jmp short loc_4011AD
mov [ebp+var_4BC], 0
mov edx, [ebp+var_4BC]
mov [ebp+var_498], edx
mov byte ptr [ebp+var_4],0
mov eax, [ebp+var_4B8]
mov ecx, [ebp+var_498]
mov [eax+0C4h], ecx
mov edx, [ebp+var_4B8]
mov ecx, [edx+0C4h]
call API_Function
mov [ebp+12], eax
lea ecx, [ebp+12]
call API_Function
push eax
lea ecx, [ebp+4]
call API_Function
push eax
mov eax, [ebp+12]
mov ecx, [eax+16]
call sub_404340
mov [ebp+12], eax
cmp [ebp+12], 0
jz label
mov ecx, [ebp+4]
cmp dword ptr [ecx+4],0
jz label
mov edx, [ebp+4]
74c
90a
90b
90c
Patrón de código de operación
Patrón de código de operación
Patrón de código de operación
102
FID
14ABF226
4DE12216
883252A6
...
0
1
130
...
104a
104b
104c
106
100

FIG. 10

FIG. 11


Recibir el ODI objetivo a partir del sistema informático de cliente
302
304
¿ODI objetivo en la memoria caché de lado de servidor?
SÍ
306
Etiquetar el objeto objetivo de acuerdo con la memoria caché de lado de servidor
NO
310
Filtrar los indicadores de bloque de código para producir un subconjunto relevante
Enviar el informe de exploración al sistema informático de cliente
308
312
Recuperar conjuntos de troceos de referencia a partir de la BD de lista blanca
Actualizar la memoria caché de lado de servidor
322
314
Computar una puntuación de similitud de inclusión en lista blanca
Actualizar las BBDD de lista blanca/software malicioso/ irrupciones
320
316
¿La puntuación de similitud de inclusión en lista blanca es alta?
SÍ
Etiquetar el objeto objetivo como no malicioso
318
NO
324
Recuperar conjuntos de troceos de referencia a partir de la BD de software malicioso/ irrupciones
326
Computar una puntuación de similitud de software malicioso/ irrupción
¿La puntuación de software malicioso/ irrupción es alta?
NO
330
328
SÍ
Etiquetar el objeto objetivo como software maligno