(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 732 876**

(51) Int. CI.:

G06F 12/1009 (2006.01)
G06F 9/455 (2008.01)
G06F 21/55 (2013.01)
G06F 21/56 (2013.01)
H04L 29/06 (2006.01)
G06F 21/53 (2013.01)
G06F 12/14 (2006.01)
G06F 12/109 (2006.01)
G06F 12/08 (2006.01)
G06F 12/1027 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **11.08.2015 PCT/RO2015/050009**

(87) Fecha y número de publicación internacional: **28.07.2016 WO16118033**

(96) Fecha de presentación y número de la solicitud europea: **11.08.2015 E 15879117 (8)**

(97) Fecha y número de publicación de la concesión europea: **03.04.2019 EP 3183682**

(54) Título: **Sistemas y métodos para proporcionar como salida un resultado de una instrucción de procesador vigente tras su salida de una máquina virtual**

(30) Prioridad:

18.08.2014 US 201462038476 P
18.09.2014 US 201414489829

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
26.11.2019

(73) Titular/es:

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

(72) Inventor/es:

LUKACS, SANDOR y
LUTAS, ANDREI-VLAD

(74) Agente/Representante:

ELZABURU, S.L.P

DESCRIPCIÓN

Sistemas y métodos para proporcionar como salida un resultado de una instrucción de procesador vigente tras su salida de una máquina virtual

Solicitudes Relacionadas

Esta solicitud reivindica el beneficio de la fecha de presentación de la solicitud de patente provisional de EE.UU. Nº 62/038.476, presentada el 18 de agosto de 2014, titulada “Systems And Methods for Exposing A Current Processor Instruction Upon Exiting A Virtual Machine”.

Antecedentes

La invención está relacionada con seguridad informática, y en particular con la realización de operaciones de seguridad informática en configuraciones de virtualización de hardware.

El documento D1 (US 2007/0106986 A1) describe monitores de máquina virtual seguros y una base segura que emplea monitores de sistema operativo de invitado residentes de monitor de máquina virtual, compartimentación de memoria, y llamadas autentificadas para aislar de manera segura entidades computacionales unas de otras dentro del sistema informático.

El software malicioso, también conocido como malware, afecta a un gran número de sistemas informáticos a nivel mundial. En sus muchas formas tales como virus informáticos, gusanos, rootkits, y software espía (*spyware*), el malware representa un grave riesgo para millones de usuarios informáticos, haciéndolos vulnerables a pérdida de datos e información sensible, robo de identidad, y pérdida de productividad, entre otros.

Las aplicaciones informáticas modernas a menudo emplean tecnología de virtualización de hardware para crear entornos informáticos simulados conocidos como máquinas virtuales (VM – Virtual Machines), los cuales se comportan en muchos sentidos como sistemas informáticos físicos. En aplicaciones tales como consolidación de servidores e infraestructura-como-servicio, varias máquinas virtuales pueden ejecutarse de manera simultánea en el mismo sistema informático, compartiendo los recursos hardware entre ellas, reduciendo de esta forma la inversión y los costes de funcionamiento. Cada máquina virtual puede ejecutar su propio sistema operativo y/o software, independientemente de otras máquinas virtuales. Debido a la constante proliferación de amenazas de seguridad informática tales como software malicioso y software espía (*malware* y *spyware*), cada máquina virtual de este tipo requiere potencialmente protección.

Algunas soluciones de seguridad protegen a una máquina virtual monitorizando la manera en la que procesos de invitado que se ejecutan dentro de la VM protegida acceden a la memoria, para identificar potencial actividad maliciosa. En un ejemplo, un programa de seguridad informática puede configurar el procesador para generar un evento interno (por ejemplo, una excepción o un evento de salida de VM) cuando se realiza un intento de escribir en, o de ejecutar código desde, una zona específica de memoria, por ejemplo una zona de memoria utilizada por un proceso de invitado. Estos eventos de procesador típicamente suspenden la ejecución del hilo vigente y conmutan el procesador a la ejecución de una rutina manejadora de eventos, la cual puede formar parte del programa de seguridad informática. De esta forma el programa de seguridad informática puede detectar un intento de acceder a la memoria de una manera que puede ser indicativa de malware. Después de analizar el evento, el programa de seguridad informática puede emular la instrucción de procesador que estaba en ejecución cuando se produjo el evento, y puede devolver la ejecución al hilo original. Estos métodos se conocen de manera general en la técnica como atrapar y emular.

Los métodos de atrapar y emular convencionales pueden colocar una carga computacional substancial sobre el sistema informático anfitrión, afectando potencialmente a la satisfacción y la productividad del usuario. Por lo tanto, existe considerable interés en desarrollar sistemas y métodos de seguridad informática eficientes apropiados para entornos de virtualización.

Compendio

De acuerdo con un aspecto, un sistema anfitrión comprende al menos un procesador hardware configurado para ejecutar una máquina virtual y un programa de seguridad informática. El al menos un procesador hardware está además configurado para, en respuesta a la recepción de una instrucción de invitado para ejecución, determinar si la ejecución de la instrucción de invitado dentro de la máquina virtual provoca o no una violación de un permiso de acceso a memoria. El al menos un procesador hardware está configurado además para, en respuesta a la determinación de si la ejecución de la instrucción de invitado provoca la violación, cuando la ejecución de la instrucción de invitado provoca o no la violación, determinar un resultado de aplicar un operador de la instrucción de invitado a un operando de la instrucción de invitado, escribir el resultado en una posición predeterminada accesible para el programa de seguridad informática, y suspender la ejecución de la instrucción de invitado. El al menos un procesador hardware está configurado además para, en respuesta a la suspensión de la ejecución de la instrucción de invitado, conmutar a ejecutar el programa de seguridad informática, en donde el programa de seguridad

informática está configurado para determinar si la violación es indicativa o no de una amenaza de seguridad informática.

De acuerdo con otro aspecto, un método de proteger un sistema anfitrión de amenazas de seguridad informática comprende, en respuesta a la recepción de una instrucción de invitado para ejecución, emplear al menos un procesador del sistema anfitrión para determinar si la ejecución de la instrucción de invitado provoca o no una violación de un permiso de acceso a memoria, en donde la instrucción de invitado se ejecuta dentro de una máquina virtual invitada expuesta por el sistema anfitrión. El método comprende además, en respuesta a la determinación de si la instrucción de invitado provoca o no la violación, cuando la ejecución de la instrucción de invitado provoca la violación, emplear el al menos un procesador hardware para determinar un resultado de aplicar un operador de la instrucción de invitado a un operando de la instrucción de invitado, emplear el al menos un procesador hardware para escribir el resultado en una posición predeterminada accesible para el programa de seguridad informática, y suspender la ejecución de la instrucción de invitado. El método comprende además, en respuesta a la suspensión de la ejecución de la instrucción de invitado, conmutar a ejecutar el programa de seguridad informática, en donde el programa de seguridad informática está configurado para determinar si la violación es o no indicativa de una amenaza de seguridad informática.

De acuerdo con otro aspecto, al menos un procesador hardware de un sistema anfitrión es configurable, en respuesta a la recepción de una instrucción de invitado para ejecución, para determinar si la ejecución de la instrucción de invitado provoca o no una violación de un permiso de acceso a memoria, en donde la instrucción de invitado se ejecuta dentro de una máquina virtual invitada expuesta por el sistema anfitrión. El al menos un procesador hardware es además configurable, en respuesta a la determinación de si la instrucción de invitado provoca la violación, cuando la ejecución de la instrucción de invitado provoca la violación, para determinar un resultado de aplicar un operador de la instrucción de invitado a un operando de la instrucción de invitado, escribir el resultado en una posición predeterminada accesible para el programa de seguridad informática, y suspender la ejecución de la instrucción de invitado. El al menos un procesador hardware es además configurable para, en respuesta a la suspensión de la ejecución de la instrucción de invitado, conmutar a ejecutar un programa de seguridad informática, en donde el programa de seguridad informática está configurado para determinar si la violación es indicativa o no de una amenaza de seguridad informática.

De acuerdo con otro aspecto, un medio legible por un ordenador no transitorio almacena instrucciones que, cuando son ejecutadas por al menos un procesador hardware de un sistema anfitrión, provocan que el sistema anfitrión conforme un programa de seguridad informática configurado para determinar si una violación de un permiso de acceso a memoria es indicativa o no de una amenaza de seguridad informática. El al menos un procesador hardware es configurable para, en respuesta a la recepción de una instrucción de invitado para ejecución, determinar si la ejecución de la instrucción de invitado provoca o no la violación, en donde la instrucción de invitado se ejecuta dentro de una máquina virtual invitada expuesta por el sistema anfitrión. El al menos un procesador hardware es además configurable para, en respuesta a la determinación de si la instrucción de invitado provoca o no la violación, cuando la ejecución de la instrucción de invitado provoca la violación, determinar un resultado de aplicar un operador de la instrucción de invitado a un operando de la instrucción de invitado, escribir el resultado en una posición predeterminada accesible para el programa de seguridad informática, y suspender la ejecución de la instrucción de invitado. El al menos un procesador hardware es además configurable para, en respuesta a la suspensión de la ejecución de la instrucción de invitado, conmutar a ejecutar el programa de seguridad informática.

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 referencia a los dibujos, en los cuales:

La Figura 1 muestra una configuración hardware ejemplar de un sistema informático anfitrión de acuerdo con algunas realizaciones de la presente invención.

La Figura 2-A muestra un conjunto ejemplar de máquinas virtuales expuestas por un hipervisor que se ejecuta en el sistema anfitrión, y un módulo de seguridad informática (CSM – Computer Security Module) que protege al conjunto de máquinas virtuales de acuerdo con algunas realizaciones de la presente invención.

La Figura 2-B muestra una realización alternativa de la presente invención, en donde un CSM se ejecuta por debajo de una máquina virtual, y en donde un manejador de excepciones se ejecuta dentro de la máquina virtual protegida.

La Figura 2-C muestra otra realización adicional de la presente invención, en donde tanto el CSM como el manejador de excepciones se ejecutan dentro de la máquina virtual protegida.

La Figura 3 muestra una configuración ejemplar de hardware virtualizado expuesto como una máquina virtual invitada de acuerdo con algunas realizaciones de la presente invención.

La Figura 4 muestra un conjunto de traducciones de direcciones de memoria ejemplares en una configuración de virtualización de hardware como se muestra en la Figura 2-A, de acuerdo con algunas realizaciones de la presente invención.

La Figura 5 muestra componentes ejemplares de un procesador de acuerdo con algunas realizaciones de la presente invención.

La Figura 6 muestra un registro de eventos de suspensión ejemplar del procesador de acuerdo con algunas realizaciones de la presente invención.

La Figura 7 muestra una representación en lenguaje ensamblador de una instrucción de procesador ejemplar del conjunto de instrucciones x86, y su correspondiente representación en código máquina.

La Figura 8 muestra una secuencia ejemplar de pasos realizados por el procesador para ejecutar una instrucción de procesador de acuerdo con algunas realizaciones de la presente invención.

La Figura 9 ilustra una secuencia ejemplar de pasos realizados por un módulo de seguridad informática para proteger a una máquina virtual invitada 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 puede 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 se refiere a al menos un elemento. Una pluralidad de elementos incluye al menos dos elementos. A menos que se requiera algo diferente, cualquier paso del método descrito no tiene por qué ser realizado necesariamente en un orden ilustrado particular. Un primer elemento (por ejemplo datos) derivado 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. Tomar una determinación o decisión de acuerdo con un parámetro abarca tomar la determinación o decisión de acuerdo con el parámetro y opcionalmente de acuerdo con otros datos. A menos que se especifique algo diferente, un indicador de alguna cantidad o de algunos datos puede ser la propia cantidad o los propios datos, o un indicador diferente a la propia cantidad o a los propios datos. Un programa informático es una secuencia de instrucciones de procesador que llevan a cabo una tarea. Los programas informáticos descritos en algunas realizaciones de la presente invención pueden ser entidades o sub-entidades software autónomas (por ejemplo, subrutinas, bibliotecas) de otros programas informáticos. A menos que se especifique algo diferente, un programa de seguridad informática es un programa informático que protege a equipos y datos contra acceso, modificación o destrucción no deseados o no autorizados. A menos que se especifique algo diferente, un proceso es un ejemplo de un programa informático, tal como una aplicación o una parte de un sistema operativo, y está caracterizado por tener al menos un hilo de ejecución y un espacio de memoria virtual asociados a él, en donde un contenido del respectivo espacio de memoria virtual incluye código ejecutable. A menos que se especifique algo diferente, una página representa la unidad de memoria virtual más pequeña que se puede mapear de forma individual a una memoria física de un sistema anfitrión. El término “lógica” abarca circuitos hardware que tienen una funcionalidad fija o una funcionalidad reconfigurable (por ejemplo circuitos de matrices de puertas programables in situ), pero no abarca software que emula dicha funcionalidad en un ordenador de propósito general. A menos que se especifique algo diferente, un registro representa un componente de almacenamiento integrado con o que forma parte de un procesador, y distinto a memoria de acceso aleatorio (RAM – Random-Access Memory). Medios legibles por un ordenador abarcan medios no transitorios tales como medios de almacenamiento magnéticos, ópticos y de semiconductores (por ejemplo discos duros, discos ópticos, memoria flash, DRAM), así como enlaces de comunicación tales como cables conductores de la electricidad 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) programados para realizar los métodos descritos en esta memoria, así como medios legibles por un ordenador que codifican instrucciones para realizar los métodos descritos en esta memoria.

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

La Figura 1 muestra una configuración hardware ejemplar de un sistema anfitrión 10 de acuerdo con algunas realizaciones de la presente invención. El sistema anfitrión 10 puede representar un dispositivo informático corporativo tal como un servidor de empresa, o un dispositivo de usuario final tal como un ordenador personal, una tableta, o un teléfono inteligente. Otros sistemas anfitriones ejemplares incluyen TVs, consolas de videojuegos, dispositivos informáticos portátiles, o cualquier otro dispositivo electrónico que tenga una memoria y un procesador. El sistema anfitrión 10 se puede utilizar para ejecutar un conjunto de aplicaciones software, tales como un navegador, una aplicación de procesamiento de textos, y una aplicación de comunicación electrónica (por ejemplo, correo electrónico, mensajería instantánea), entre otras. En algunas realizaciones, el sistema anfitrión 10 está configurado para soportar virtualización de hardware y para exponer un conjunto de máquinas virtuales, como se muestra más adelante.

La Figura 1 ilustra un sistema informático; la configuración hardware de otros sistemas anfitriones, tales como teléfonos inteligentes y tabletas, puede ser diferente. El sistema 10 comprende un conjunto de dispositivos físicos, incluidos un procesador 12, una unidad de memoria 14, un conjunto de dispositivos de entrada 16, un conjunto de dispositivos de salida 18, un conjunto de dispositivos de almacenamiento 20, y un conjunto de adaptadores de red

22, todos ellos conectados por un concentrador controlador 24. En algunas realizaciones, el procesador 12 comprende un dispositivo físico (por ejemplo un circuito integrado de múltiples núcleos conformado sobre un substrato semiconductor) configurado para ejecutar operaciones computacionales y/o lógicas con un conjunto de señales y/o datos. En algunas realizaciones, dichas operaciones lógicas son proporcionadas al procesador 12 en forma de una secuencia de instrucciones de procesador (por ejemplo código máquina u otro tipo de software). Algunas realizaciones de la presente invención introducen cambios en la estructura y funcionalidad de un procesador convencional, permitiendo los respectivos cambios que el procesador 12 pueda operar de manera más eficiente en configuraciones de virtualización de hardware.

La unidad de memoria 14 puede comprender medios legibles por un ordenador volátiles (por ejemplo RAM) que almacenan datos/señales a los que ha accedido el procesador 12 o que han sido generados por él mientras se llevaban a cabo instrucciones. Los dispositivos de entrada 16 pueden incluir teclados de ordenador, ratones, y micrófonos, entre otros, incluidos las respectivas interfaces y/o adaptadores hardware que permiten que un usuario pueda introducir datos y/o instrucciones en el sistema anfitrión 10. Los dispositivos de salida 18 pueden incluir dispositivos de visualización tales como monitores y altavoces, entre otros, así como interfaces/adaptadores hardware tales como tarjetas gráficas, que permiten que el sistema anfitrión 10 comunique datos a un usuario. En algunas realizaciones, los dispositivos de entrada 16 y los dispositivos de salida 18 pueden compartir un hardware común, como en el caso de dispositivos de pantalla táctil. Los dispositivos de almacenamiento 20 incluyen medios legibles por un ordenador que permiten el almacenamiento no volátil, la lectura y la escritura de instrucciones de procesador y/o datos. Dispositivos de almacenamiento 20 ejemplares incluyen discos magnéticos y ópticos y dispositivos de memoria flash, así como medios extraíbles tales como discos y lectores de CD y/o de DVD. El conjunto de adaptadores de red 22 permite que el sistema anfitrión 10 se pueda conectar a una red informática y/o a otros dispositivos/sistemas informáticos. El concentrador controlador 24 representa de forma genérica la pluralidad de buses del sistema, periféricos, y/o de grupos de chips, y/o cualquier otro circuito que permiten la comunicación entre el procesador 12 y los dispositivos 14, 16, 18, 20 y 22. Por ejemplo, el concentrador controlador 24 puede incluir una unidad de gestión de memoria (MMU – Memory Management Unit), un controlador de entrada/salida (I/O), y un controlador de interrupciones, entre otros. En otro ejemplo, el concentrador controlador 24 puede comprender un procesador 12 de conexión de puente norte a la memoria 14 y/o un procesador 12 de conexión de puente sur a los dispositivos 16, 18, 20 y 22. En algunas realizaciones, partes del concentrador controlador (tales como la MMU) pueden estar integradas con el procesador 12, es decir, pueden compartir un substrato común con el procesador 12.

La Figura 2-A muestra una configuración funcional ejemplar de acuerdo con algunas realizaciones de la presente invención, en la cual el sistema anfitrión 10 utiliza tecnología de virtualización de hardware para operar un conjunto de máquinas virtuales invitadas 52a-b expuestas por un hipervisor 50. Estas configuraciones son habituales en aplicaciones tales como computación en la nube y consolidación de servidores, entre otras. Una máquina virtual (VM) es conocida en la técnica como una abstracción, por ejemplo, una emulación software, de una máquina física/sistema informático real, siendo la VM capaz de ejecutar un sistema operativo y otro software. En algunas realizaciones, el hipervisor 50 incluye software configurado para crear o habilitar una pluralidad de dispositivos virtualizados, tales como un procesador virtual y un concentrador controlador virtual, y presentar dichos dispositivos virtualizados a software en lugar de los dispositivos físicos, reales, del sistema anfitrión 10. Estas operaciones del hipervisor 50 se conocen habitualmente en la técnica como exponer una máquina virtual. En algunas realizaciones, el hipervisor 50 permite un multiplexado (una compartición) por parte de múltiples máquinas virtuales de recursos hardware del sistema anfitrión 10. El hipervisor 50 puede gestionar además dicho multiplexado de modo que cada VM invitada 52a-b opera independientemente y no es consciente de otras VMs en ejecución que se ejecutan de manera concurrente en el sistema anfitrión 10. Ejemplos de hipervisores populares incluyen el VMware vSphere™ de VMware Inc. y el hipervisor de código abierto Xen, entre otros.

Cada VM 52a-b puede ejecutar un sistema operativo (OS – Operating System) de invitado 54a-b, respectivamente. Un conjunto de aplicaciones 56a-d ejemplares representan genéricamente cualquier aplicación software, tal como procesamiento de textos, procesamiento de imágenes, reproductor multimedia, base de datos, calendario, gestión de contactos personales, navegador, videojuegos, comunicación de voz, comunicación de datos, y aplicaciones anti-malware, entre otros. Los sistemas operativos 54a-b pueden comprender cualquier sistema operativo ampliamente disponible como por ejemplo Microsoft Windows®, MacOS®, Linux®, iOS®, o Android™, entre otros. Cada OS 54a-b proporciona una interfaz entre aplicaciones que se ejecutan dentro de la respectiva VM y los dispositivos hardware virtualizados de la respectiva VM. En la siguiente descripción, software que se ejecuta en un procesador virtual de una máquina virtual se dice que se ejecuta dentro de la respectiva máquina virtual. Por ejemplo, en el ejemplo de la Figura 2-A, se dice que las aplicaciones 56a-b se ejecutan dentro de la VM invitada 52a, mientras que se dice que las aplicaciones 56c-d se ejecutan dentro de la VM invitada 52b. Por el contrario, se dice que el hipervisor 50 se ejecuta fuera, o debajo, de las VMs invitadas 52a-b.

La Figura 3 muestra una configuración ejemplar de una máquina virtual 52, tal como es expuesta por el hipervisor 50. La VM 52 puede representar cualquiera de las VMs 52a-b de la Figura 2-A. La VM 52 incluye un procesador 112 virtualizado, una unidad de memoria 114 virtualizada, dispositivos de entrada 116 virtualizados, dispositivos de salida 118 virtualizados, almacenamiento 120 virtualizado, adaptadores de red 122 virtualizados, y un concentrador controlador 124 virtualizado. El procesador 112 virtualizado comprende una emulación de al menos parte de la funcionalidad del procesador 12, y está configurado para recibir para ejecución instrucciones de procesador que

forman parte de software tal como un sistema operativo y otras aplicaciones. El software que utiliza el procesador 112 para ejecución se considera que se ejecuta dentro de la máquina virtual 52. En algunas realizaciones, la unidad de memoria 114 virtualizada comprende espacios direccionables para almacenar y recuperar datos utilizados por el procesador 112 virtualizado. Otros dispositivos virtualizados (por ejemplo, entrada, salida, almacenamiento, etc., virtualizados) emulan al menos parte de la funcionalidad de los respectivos dispositivos físicos del sistema anfitrión 10. El procesador 112 virtualizado puede estar configurado para interactuar con dichos dispositivos virtualizados como lo haría con los correspondientes dispositivos físicos. Por ejemplo, software que se ejecuta dentro de la VM 52 puede enviar y/o recibir tráfico de red a través de uno o más adaptadores de red 122 virtualizados. En algunas realizaciones, el hipervisor 50 puede exponer sólo un subconjunto de dispositivos virtualizados a la VM 52 (por ejemplo, sólo el procesador 112 virtualizado, la memoria 114 virtualizada, y partes del concentrador 124). El hipervisor 50 también puede proporcionar un uso exclusivo de VM seleccionada de algunos dispositivos hardware del sistema anfitrión 10. En un ejemplo de este tipo, la VM 52a (Figura 2-A) puede tener uso exclusivo de los dispositivos de entrada 16 y de los dispositivos de salida 18, pero carecer de un adaptador de red virtualizado. Mientras tanto, la VM 52b puede tener uso exclusivo de uno o más adaptadores de red 22. Estas configuraciones se pueden implementar, por ejemplo, utilizando tecnología VT-d® de Intel®.

Los procesadores modernos implementan una jerarquía de niveles de privilegio de procesador, también conocidos en la técnica como anillos de protección. Cada anillo o nivel de este tipo está caracterizado por un conjunto de acciones y/o instrucciones de procesador que se permite que lleve a cabo software que se ejecuta dentro del respectivo anillo. Los niveles/anillos de privilegio ejemplares incluyen modo usuario (anillo 3) y modo kernel (anillo 0). Algunos sistemas anfitriones configurados para soportar virtualización de hardware pueden incluir un anillo adicional con los mayores privilegios de procesador (por ejemplo, anillo -1, modo root, o VMXroot en plataformas Intel®). En algunas realizaciones, el hipervisor 50 toma control del procesador 12 en el nivel de máximo privilegio (anillo -1), creando de esta forma una plataforma de virtualización de hardware expuesta como una máquina virtual a otro software que se ejecuta en el sistema anfitrión 10. Un sistema operativo, tal como el OS de invitado 54a en la Figura 2-A, se ejecuta dentro del entorno virtual de la respectiva VM, típicamente con menor privilegio de procesador que el hipervisor 50 (por ejemplo, en anillo 0 o modo kernel). Las aplicaciones de usuario habituales, tales como 56a-b, típicamente se ejecutan a privilegio de procesador menor que el OS 34a (por ejemplo en anillo 3 o modo usuario). Algunas partes de las aplicaciones 56a-b se pueden ejecutar a nivel de privilegio kernel, mientras que algunas partes del OS 34a se pueden ejecutar en modo usuario (anillo 3). Cuando un objeto software intenta ejecutar una acción o instrucción que requiere privilegios de procesador mayores de los permitidos por su anillo de protección asignado, el intento típicamente genera un evento de procesador, tal como una excepción o un fallo, que transfiere el control del procesador 12 a una entidad (por ejemplo, rutina de manejador) que se ejecuta en un anillo con suficientes privilegios para llevar a cabo la respectiva acción.

En particular, un intento de realizar ciertas acciones o de ejecutar ciertas instrucciones desde dentro de una VM invitada puede disparar una categoría especial de eventos de procesador, denominados de manera general en esta memoria eventos de suspensión de VM. En algunas realizaciones, un evento de suspensión de VM suspende la ejecución del hilo vigente dentro de una VM invitada y conmuta el procesador 12 a ejecutar una rutina manejadora. Eventos de suspensión de VM ejemplares incluyen, entre otros, un evento de salida de VM (por ejemplo, VMExit en plataformas Intel®) y una excepción de virtualización (por ejemplo #VE en plataformas Intel®). Los eventos de salida de VM conmutan el procesador 12 a ejecutar una rutina manejadora fuera de la respectiva VM invitada, típicamente al nivel del hipervisor 50. La excepción de virtualización puede conmutar el procesador 12 a ejecutar una rutina manejadora dentro de la respectiva VM invitada, en lugar de a salir de la respectiva VM.

Instrucciones ejemplares que disparan un evento de suspensión de VM incluyen VMCALL en plataformas Intel®. Los eventos de suspensión de VM también pueden ser disparados por otros eventos, tales como violaciones de acceso a memoria. En un ejemplo de este tipo, cuando un objeto software que se ejecuta dentro de una VM intenta escribir en una sección de memoria marcada como de no escritura, o ejecutar código desde una sección de memoria marcada como no ejecutable, el procesador 12 puede generar un evento de salida de VM. Dichos mecanismos de conmutación de VM permiten, por ejemplo, que un programa de seguridad informática proteja a una máquina virtual desde el exterior de la respectiva VM. El programa de seguridad informática puede interceptar eventos de salida de VM que se producen en respuesta a ciertas acciones realizadas por software que se ejecuta en el interior de la VM, acciones que pueden ser indicativas de una amenaza de seguridad. El programa de seguridad informática puede a continuación bloquear y/o analizar con mayor detalle dichas acciones, potencialmente sin el conocimiento de software de dentro de la VM. Dichas configuraciones pueden reforzar substancialmente la seguridad informática.

En algunas realizaciones (por ejemplo, Figura 2-A), el hipervisor 50 incluye un módulo de seguridad informática (CSM) 60, configurado para realizar dichas operaciones de seguridad informática, entre otras. El módulo 60 puede estar incorporado en el interior del hipervisor 50 (por ejemplo como una biblioteca), o se puede proporcionar como un programa informático distinto e independiente del hipervisor 50, pero que se ejecuta al nivel de privilegio del hipervisor 50. Un único módulo 60 puede estar configurado para proteger múltiples VMs invitadas que se ejecutan en el sistema anfitrión 10. Operaciones de seguridad llevadas a cabo por el módulo 60 pueden incluir detectar una acción realizada por un proceso que se ejecuta dentro de una VM invitada (por ejemplo, llamar a ciertas funciones del OS, acceder a registros del OS, descargar un fichero desde una posición remota, escribir datos en un fichero, etc.). Otras operaciones de seguridad del módulo 60 pueden comprender determinar una dirección de una sección de memoria que contiene una parte de un objeto software que se ejecuta dentro de una VM invitada, acceder a la

respectiva sección de memoria, y analizar un contenido almacenado dentro de la respectiva sección de memoria. Otros ejemplos de operaciones de seguridad incluyen interceptar y/o restringir el acceso a dichas secciones de memoria, por ejemplo, impedir la sobreescritura de código o datos pertenecientes a un proceso protegido, e impedir la ejecución de código almacenado en ciertas páginas de memoria. En algunas realizaciones, el CSM 60 incluye un manejador 61 de eventos de salida de VM configurado para interceptar eventos de salida de VM que se producen dentro de las VMs invitadas 52a-b. En una realización alternativa, el manejador 61 puede ser un módulo distinto (por ejemplo, una biblioteca) al hipervisor 50, independiente del CSM 60, que intercepta eventos de salida de VM y que transfiere control selectivamente al CSM 60 después de determinar una razón y/o un tipo de cada salida de VM que se produjo.

La Figura 2-B ilustra una realización alternativa en la cual el módulo de seguridad informática 60 protege a una VM invitada 52c desde fuera de la respectiva VM. En estas realizaciones, el procesador 12 puede estar configurado para generar una excepción de virtualización (en vez de un evento de salida de VM, como se describió anteriormente en relación con la Figura 2-A) cuando se produce una violación de acceso a memoria. En la realización ejemplar de la Figura 2-B, un manejador 63 de excepciones de virtualización se ejecuta dentro de la VM 52c, por ejemplo al nivel de privilegio de un sistema operativo 54c, y está configurado para interceptar excepciones de virtualización e interaccionar con el CSM 60.

La comunicación entre el manejador 63 y el CSM 60 puede desarrollarse de acuerdo con cualquier método de comunicación entre procesos conocido en la técnica. Para transmitir datos desde dentro de la VM protegida al nivel del hipervisor 50, algunas realizaciones del manejador 63 utilizan una instrucción especializada (por ejemplo, VMCALL en plataformas Intel®) para transferir el control del procesador 12 de la respectiva VM al hipervisor 50. Los datos que se están transmitiendo pueden ser colocados por el manejador 63 de excepciones en una sección de memoria predeterminada compartida con el CSM 60. Para transmitir datos al manejador 63, algunas realizaciones de CSM 60 pueden inyectar una interrupción en la VM 52c, siendo manejada la interrupción por el manejador 63. Los respectivos datos se pueden transferir de nuevo a través de la sección de memoria compartida descrita anteriormente.

En otra realización adicional, ilustrada en la Figura 2-C, tanto el CSM 60 como el manejador 63 se ejecutan dentro de la VM protegida, por ejemplo en modo kernel (anillo 0). Estas realizaciones también pueden emplear excepciones de virtualización para detectar violaciones de acceso a memoria. Decidir entre las configuraciones 2-A-B-C puede comprender evaluar un compromiso entre prestaciones y seguridad. Los eventos de salida de VM son relativamente costosos en términos de computación, requiriendo típicamente carga y/o descarga de grandes estructuras de datos en/de memoria con cada ciclo de salida y reentrada. Por tanto, configuraciones tales como la 2-A pueden requerir más computación para interceptar un evento que configuraciones tales como las 2-B-C. Por otro lado, mantener componentes de seguridad críticos tales como el CSM 60 y los manejadores 61-63 fuera de la VM protegida (como en los ejemplos 2-A-B) puede reforzar la seguridad, dado que para el malware que se ejecuta dentro de la respectiva VM puede ser más difícil interferir con el funcionamiento de dichos componentes.

Para ser capaz de proteger una VM invitada en una configuración como la ilustrada en las Figuras 2-A-B (es decir, desde fuera de la respectiva VM), algunas realizaciones de CSM 60 emplean estructuras de datos de traducción de direcciones y/o mecanismos de traducción de direcciones del procesador 12. Las máquinas virtuales típicamente operan con una memoria física virtualizada (véase, por ejemplo, la memoria 114 en la Figura 3), también conocida en la técnica como memoria física-del invitado. La memoria física virtualizada comprende una representación abstracta de la memoria 14 física real, por ejemplo como un espacio contiguo de direcciones, denominadas habitualmente direcciones físicas-del invitado (GPA - Guest-Physical Addresses). Cada uno de dichos espacios de direcciones está fijado de manera única a una VM invitada, con partes de dicho espacio de direcciones mapeadas a secciones de memoria física 14 y/o dispositivos de almacenamiento físico 20. En sistemas configurados para soportar virtualización, este mapeado se consigue típicamente utilizando estructuras de datos dedicadas, aceleradas por hardware, y mecanismos controlados por el procesador 12, conocidos como traducción de direcciones de segundo nivel (SLAT - Second Level Address Translation). Implementaciones de SLAT populares incluyen tablas de páginas extendidas (EPT - Extended Page Tables) en plataformas Intel®, e indexación de virtualización rápida (RVI - Rapid Virtualization Indexing)/tablas de páginas anidadas (NPT - Nested Page tables) en plataformas AMD®. En estos sistemas, la memoria física virtualizada puede estar particionada en unidades conocidas en la técnica como páginas, representando una página la unidad más pequeña de memoria física virtualizada mapeada individualmente a memoria física a través de mecanismos tales como EPT/NPT, es decir, se realiza mapeado entre memoria física virtualizada y física con granularidad de páginas. Típicamente todas las páginas tienen un tamaño predeterminado, por ejemplo, 4 kilobytes, 2 megabytes, etc. El particionado de memoria física virtualizada en páginas suele ser configurado por el hipervisor 50. En algunas realizaciones, el hipervisor 50 también configura las estructuras SLAT, y por lo tanto configura traducción de direcciones entre memoria física y memoria física virtualizada. Dichas traducciones de direcciones se conocen en la técnica como traducciones de física del invitado a física del anfitrión (GPA-a-HPA - Guest-Physical to Host-Physical).

En algunas realizaciones, el sistema operativo que se ejecuta dentro de una VM establece un espacio de memoria virtual para cada proceso que se ejecuta dentro de la respectiva VM, representando dicho espacio de memoria virtual una abstracción de memoria física. La memoria virtual de proceso típicamente comprende un espacio de direcciones contiguo, conocida comúnmente en la técnica como direcciones virtuales de invitado (GVA - Guest-

Virtual Addresses) o direcciones lineales de invitado (GLA - Guest-Linear Addresses). En algunas realizaciones, los espacios de memoria virtual de proceso también están particionados en páginas, representando dichas páginas la unidad más pequeña de memoria virtual mapeada individualmente por el OS a la memoria física virtualizada de la respectiva VM, es decir, se realiza mapeado de memoria virtual a memoria física-virtualizada con granularidad de páginas. El OS puede configurar una estructura de datos dedicada, tal como una tabla de páginas, utilizada por el procesador virtualizado de la respectiva VM para realizar traducciones de direcciones virtuales de invitado a direcciones físicas de invitado, o traducciones GVA-a-GPA.

La Figura 4 ilustra una traducción de dirección de memoria ejemplar en la realización de la Figura 2-A. Después de la exposición por el hipervisor 50, la VM invitada 52a ve un espacio de memoria física virtualizada 114a como su propio espacio de memoria física. El OS de invitado 54a asigna a un proceso que se ejecuta dentro de la VM invitada 52a un espacio de memoria virtual 214a. Cuando el proceso intenta acceder a memoria en una dirección virtual de invitado 62, la GVA 62 es traducida por la MMU (virtualizada) de la VM invitada 52a en una dirección física de invitado 64 dentro del espacio de memoria física virtualizada 114a. La traducción GVA-a-GPA 70a puede desarrollarse, por ejemplo, de acuerdo con tablas de páginas configuradas y controladas por el OS de invitado 34a. La GPA 64 es mapeada además por la MMU a una dirección física del anfitrión (HPA – Host-Physical Address) 66 dentro de la memoria física 14 del sistema anfitrión 10. La traducción GVA-a-GPA 70b puede desarrollarse, por ejemplo, de acuerdo con estructuras SLAT configuradas por el hipervisor 50.

A cada proceso que se ejecuta por debajo de las VMs invitadas 52a-b se le asigna típicamente un espacio de memoria virtual direccionable a través de lo que se conoce en la técnica como direcciones virtuales del anfitrión (HVA – Host-Virtual Addresses). En el ejemplo de la Figura 4, el hipervisor 50 establece un espacio de memoria virtual 214b para el módulo de seguridad informática 60. El CSM 60 puede entonces referenciar la HPA 66 a través de una HVA 68. Cuando el módulo 60 está integrado dentro del hipervisor 50, por ejemplo como una biblioteca, el espacio de memoria 214b puede coincidir con el espacio de memoria virtual del hipervisor 50. Para gestionar estos espacios, el hipervisor 50 puede configurar estructuras de datos dedicadas y mecanismos (por ejemplo tablas de páginas) dedicados utilizados por la MMU para realizar traducciones HVA-a-HPA tales como la traducción 70c.

En algunas realizaciones, el hipervisor 50 y/o el CSM 60 pueden establecer permisos de acceso para cada una de un subconjunto de páginas de memoria física. Dichas páginas de memoria pueden ser utilizadas, por ejemplo, por ciertos procesos de invitado críticos que se ejecutan dentro de una VM protegida, tales como procesos del OS y/o rutinas anti-malware. Los permisos de acceso indican, por ejemplo, si se puede o no leer de la página respectiva o si se puede o no escribir en ella, y si se permite o no que software ejecute código desde la página respectiva. Los permisos de acceso se pueden indicar, por ejemplo, como parte de la entrada SLAT que representa la respectiva página de memoria. Algunos sistemas anfitriones pueden permitir establecer permisos de acceso con granularidad de sub-páginas.

El hipervisor 50 y/o el CSM 60 pueden configurar además el procesador 12 para generar un evento de suspensión de VM cuando software que se ejecuta dentro de una VM invitada intenta acceder a memoria de una manera que viola permisos de acceso (por ejemplo, intenta escribir en una página de memoria marcada como de no escritura). A un intento de este tipo se le denomina en esta memoria violación de acceso a memoria. El respectivo evento de suspensión de VM puede ser un evento de salida de VM en configuraciones tales como la Figura 2-A, y una excepción de virtualización en configuraciones tales como las Figuras 2-B-C.

Algunas realizaciones de la presente invención introducen cambios en la estructura y funcionalidad de un procesador hardware convencional, para permitir que el procesador funcione de manera más eficiente en configuraciones de virtualización de hardware. La Figura 5 muestra componentes hardware ejemplares del procesador 12 de acuerdo con algunas realizaciones de la presente invención. Los componentes ilustrados se deben interpretar como dispositivos genéricos que realizan la funcionalidad descrita; los detalles estructurales pueden variar substancialmente entre implementaciones. Por ejemplo, cada componente ilustrado puede comprender múltiples subsistemas interconectados, no necesariamente en proximidad física unos con otros. Los componentes ilustrados no son exhaustivos; el procesador 12 puede incluir muchos otros componentes (por ejemplo, planificador, controlador de interrupciones, diversos cachés, etc.), que se omitieron de la Figura 5 para simplificar la presentación.

El procesador 12 puede incluir lógica/circuitos configurados para llevar a cabo diferentes etapas de un *pipeline* del procesador. Por ejemplo, un decodificador de instrucciones 30 realiza operaciones de decodificación de instrucciones, que pueden incluir traducir cada instrucción de procesador convirtiéndola en un conjunto de operaciones y/o micro-operaciones de procesador elementales. Un conjunto de unidades de ejecución 36 conectadas al decodificador 30 puede realizar la etapa de ejecución del pipeline. Una o más unidades de ejecución 36 ejemplares incluyen, entre otras, una unidad lógica aritmética (ALU - Arithmetic Logic Unit) y una unidad de punto flotante (FPU - Floating-Point Unit).

En algunas realizaciones, la etapa de ejecución del pipeline para una instrucción comprende determinar un resultado de aplicar un operador de la respectiva instrucción a un operando de la respectiva instrucción. Estos resultados pueden comprender, entre otros, una dirección de memoria, un valor a ser consolidado en una dirección de memoria o en un registro de procesador (por ejemplo, a un registro de propósito general tal como AX, a un registro específico

de modelo – MSR (Model-Specific Register), a un registro de control tal como EFLAGS, o a un registro oculto tal como la parte oculta de un registro de segmento x86, también conocido como un caché descriptor), un valor del puntero de instrucción (por ejemplo, RIP), y un valor del puntero de pila (por ejemplo, RSP).

Los uno o más operandos de una instrucción pueden estar explícitos en la declaración de la instrucción, o pueden estar implícitos. Una instrucción x86 ejemplar con operandos implícitos es la instrucción STC, la cual establece la bandera de arrastre del registro de control EFLAGS del procesador en el valor 1. En algunas realizaciones, el registro EFLAGS y el valor 1 se interpretan como operandos (implícitos), aunque no aparecen explícitamente en la declaración de la instrucción STC.

Una unidad de consolidación 38 (*commit unit*) puede realizar la etapa de consolidación del pipeline, es decir, almacenar la salida de las una o más unidades de ejecución 36 en la memoria 14 y/o actualizar los contenidos de ciertos registros de procesador para reflejar cambios/resultados producidos por la etapa de ejecución. La unidad de consolidación 38 puede comprender módulos lógicos conocidos en la técnica como unidades de retirada.

Un módulo 34 de acceso a memoria conectado al decodificador 30 y a las una o más unidades de ejecución 36 incluye lógica configurada para interaccionar con la memoria 14, por ejemplo, para extraer instrucciones de la memoria, para cargar datos desde memoria, y para almacenar en memoria resultados de ejecución de instrucciones de procesador. En algunas realizaciones, el módulo de acceso a memoria comprende una MMU configurada para realizar las traducciones de direcciones virtual-a-física necesarias para acceso a memoria.

Los procesadores modernos típicamente soportan ejecución fuera de orden y/o especulativa de instrucciones de procesador. En dichos sistemas, múltiples instrucciones son extraídas, decodificadas, y ejecutadas de forma concurrente por las mismas una o más unidades de ejecución 36. Los resultados de dichas ejecuciones son entonces consolidados en orden, para preservar el flujo deseado del respectivo programa informático. Dichas configuraciones se utilizan, por ejemplo, en conjunto con algoritmos de predicción de bifurcación, para mejorar las prestaciones del procesador 12. En algunas realizaciones configuradas para ejecución fuera de orden, el procesador 12 puede comprender además una unidad despachadora 32 acoplada al decodificador 30 y a unidades 36 de ejecución, y un fichero 40 de registro acoplado a una o más unidades de ejecución 36 y a la unidad de consolidación 38. La unidad despachadora 36 puede planificar micro-operaciones individuales para ejecución, y mantener un mapeado que asocie cada micro-operación con su respectiva instrucción, para controlar el orden de ejecución y consolidación. El fichero 40 de registros comprende una matriz de registros de procesador internos, organizada, por ejemplo, como un búfer de reordenación. El fichero 40 de registros puede comprender además lógica que permita que la unidad despachadora 36 pueda asociar una fila de registros del fichero 40 a cada micro-operación planificada, una operación conocida en la técnica como renombrado de registros. En estas configuraciones, cada fila de registros de este tipo puede contener, por ejemplo, los valores de los registros de propósito general y/o de estado del procesador 12, correspondiendo dichos valores a una etapa de ejecución intermedia de una cierta instrucción de procesador.

El procesador 12 puede incluir además una unidad 38 de control de máquinas virtuales configurada para gestionar datos de estado de máquinas virtuales. En algunas realizaciones, un objeto de estado de máquina virtual (VMSO - Virtual Machine State Object) comprende una estructura de datos utilizada internamente por el procesador 12 para representar el estado en ese momento de cada procesador virtualizado expuesto en el sistema anfitrión 10. Los VMSOs ejemplares incluyen la estructura de control de máquinas virtuales (VMCS - Virtual Machine Control Structure) en plataformas Intel®, y el bloque de control de máquinas virtuales (VMCB - Virtual Machine Control Block) en plataformas AMD®. Los VMSOs son establecidos típicamente por el hipervisor 50 como parte de la exposición de cada máquina virtual. En algunas realizaciones, el procesador 12 asocia una zona en memoria con cada VMSO, de modo que el software puede referenciar un VMSO específico utilizando una dirección de memoria o un puntero (por ejemplo, puntero de VMCS en plataformas Intel®).

Cada VMSO puede comprender un área de estado del invitado y un área de estado del anfitrión, conteniendo el área de estado del invitado el estado de la CPU de la respectiva VM invitada, y almacenando el área de estado del anfitrión el estado en ese momento del hipervisor 50. En algunas realizaciones, el área de estado del invitado del VMSO incluye contenidos de los registros de control (por ejemplo, CR0, CR3, etc.), puntero de instrucción (por ejemplo, RIP), registros de propósito general (por ejemplo, EAX, ECX, etc.), y registros de estado (por ejemplo, EFLAGS) del procesador virtual de la respectiva VM invitada, entre otros. El área de estado del anfitrión del VMSO puede incluir un puntero (por ejemplo, un puntero EPT en plataformas Intel®) a una estructura de datos SLAT configurada para traducciones de direcciones GPA-a-HPA para la respectiva VM invitada.

En algunas realizaciones, el procesador 12 puede almacenar una parte de un VMSO dentro de registros/cachés internos dedicados, mientras que otras partes del respectivo VMSO pueden residir en memoria. En cualquier momento dado, como máximo un VMSO (denominado en esta memoria el VMSO vigente) puede estar cargado en el procesador, identificando la máquina virtual que tiene control en ese momento del procesador 12. Los procesadores modernos están configurados típicamente para soportar multihilos (*multithreading*). En tales configuraciones, el procesador físico 12 puede operar una pluralidad de núcleos, comprendiendo además cada núcleo múltiples procesadores lógicos, en donde cada procesador lógico puede procesar un hilo de ejecución con independencia de, y concurrentemente con, otros procesadores lógicos. Múltiples procesadores lógicos pueden compartir algunos

recursos hardware, por ejemplo, una MMU común. En una realización en multihilos (*multithreaded*), en cada procesador lógico distinto puede estar cargado un VMSO distinto.

Cuando el procesador 12 conmuta de ejecutar la respectiva VM a ejecutar el hipervisor 50 (por ejemplo, tras una salida de VM), el procesador 12 puede guardar el estado de la respectiva VM en el área de estado del invitado del VMSO vigente. Cuando el procesador 12 conmuta de ejecutar una primera VM a ejecutar una segunda VM, el VMSO asociado a la primera VM se descarga, y el VMSO asociado a la segunda VM se carga en el procesador, convirtiéndose el segundo VMSO en el VMSO vigente. En algunas realizaciones, dicha carga/descarga de datos VMSO a/desde el procesador 12 es realizada por el módulo de control 38 de máquinas virtuales. El módulo 38 puede llevar a cabo además la recuperación y/o guardado de datos VMSO de/en la memoria 14.

En algunas realizaciones, el procesador 12 comprende además un registro 44 de eventos de suspensión conectado a una o más unidades de ejecución 36 y/o a la unidad de consolidación 38, y configurado para almacenar datos específicos de la instrucción asociados con una instrucción de invitado, en donde la ejecución de dichas instrucciones de invitado ha provocado un evento de suspensión de VM (por ejemplo, una salida de VM o una excepción de virtualización). En algunas realizaciones, el registro 44 de eventos de suspensión es un registro expuesto, accesible para software que se ejecuta en el sistema anfitrión 10, es decir, datos almacenados en el registro 44 pueden ser legibles por software tal como el módulo de seguridad 60. En un ejemplo de este tipo, el registro de eventos de suspensión 44 incluye un registro específico de modelo (MSR - Model-Specific Register) del procesador 12. Algunas realizaciones pueden restringir el acceso al registro 44 a un subconjunto de objetos software, seleccionados de acuerdo con un criterio tal como privilegio del procesador (por ejemplo, sólo anillo -1 o modo root) o tipo de objeto (por ejemplo, sólo drivers). Algunas realizaciones pueden restringir el acceso software al registro 44 sólo a un subconjunto de operaciones (por ejemplo, de sólo lectura).

La Figura 6 muestra un conjunto ejemplar de campos del registro de eventos de suspensión 44 de acuerdo con algunas realizaciones de la presente invención. El registro 44 puede incluir un campo 46a de desensamblado y un campo 46b de resultado de ejecución. El campo 46a de desensamblado puede almacenar datos que se producen como resultado de desensamblar la respectiva instrucción de invitado.

La Figura 7 muestra una representación 45 en lenguaje ensamblador de una instrucción de procesador Intel® x86 ejemplar. La instrucción ilustrada da instrucciones al procesador para que incremente el contenido almacenado en memoria en la dirección (virtual) EBX + 4*ECX + 0x02, por el valor almacenado en ese momento en el registro AX. La respectiva instrucción se representa en memoria como una representación 47 en código máquina; la traducción entre las representaciones 45 y 47 es realizada típicamente por un compilador o ensamblador. La representación en código máquina de instrucciones x86 tiene una forma 48 genérica, que comprende una secuencia de campos de codificación, incluidos, entre otros, un prefijo, un codop, y un campo de desplazamiento (representaciones de instrucciones de otras ISAs pueden ser diferentes). La Figura 7 muestra el ejemplo de cada campo de codificación para la instrucción x86 ejemplar dada. En algunas ISAs, la representación en código máquina puede variar en longitud, es decir, algunos campos de codificación pueden aparecer en la representación en código máquina de ciertas instrucciones, pero pueden no aparecer en la representación de otras instrucciones.

En algunas realizaciones, desensamblar una instrucción comprende analizar la representación en código máquina de la instrucción para identificar y/o calcular un conjunto de elementos semánticos. Dichos elementos semánticos pueden incluir un operador (por ejemplo, MOV, ADD, etc.) y un operando (por ejemplo, AX, [EBX+4*ECX+0x02]) de la instrucción, entre otros. En algunas realizaciones, los elementos semánticos de una instrucción incluyen campos de codificación de instrucción individuales (tales como Prefijo, Codop, modR/M, SIB, Desplazamiento, y Valor Inmediato, en el caso de la ISA x86). Este desensamblado puede ser llevado a cabo, al menos en parte, por el decodificador de instrucciones 30 y/o por las una o más unidades de ejecución 36.

En el ejemplo de la Figura 7, desensamblar la instrucción puede comprender determinar los contenidos de campos de codificación individuales, como se muestra mediante las flechas que indican la correspondencia entre representación en código máquina 47 y forma genérica 48. En algunas realizaciones, desensamblar la instrucción ilustrada incluye identificar el operador ADD y/o determinar de acuerdo con el código máquina 47 que la respectiva instrucción tiene dos operandos, que uno de los operandos es el contenido del registro AX, que el segundo operando es un contenido de memoria, y determinar una expresión (por ejemplo, EBX+4*ECX+0x02) de la respectiva dirección de memoria. En algunas realizaciones, desensamblar la instrucción comprende además calcular una dirección de memoria indicada por un operando de la respectiva instrucción (por ejemplo, el valor de la expresión EBX+4*ECX+0x02 en el ejemplo de la Figura 7). En otro ejemplo, en el cual la instrucción desensamblada es una instrucción de salto relativo (por ejemplo, JMP $+10 en plataformas x86, representada en código máquina como 0xEB 0x08), desensamblar la instrucción puede comprender calcular una dirección de memoria absoluta del destino de acuerdo con la dirección de la instrucción, con la longitud de la instrucción, y/o con el tamaño del salto relativo.

En algunas realizaciones, el campo de desensamblado 46a del registro 44 (Figura 6) incluye un contenido de los campos de codificación de instrucción de la respectiva instrucción (véase la Figura 7). Otro contenido ejemplar del campo 46a incluye un identificador de operador que indica el operador de la respectiva instrucción, y un indicador de un operando de la respectiva instrucción. El indicador de operando puede incluir además un identificador de un

registro de procesador (por ejemplo, AX), y una bandera que indica, por ejemplo, si el respectivo operando es el contenido de un registro o un contenido de memoria. El campo de desensamblado 46a puede comprender además una dirección de memoria (por ejemplo, GVA, GPA, y/o HPA) indicada por un operando. La estructura del campo de desensamblado 46a puede ser específica de la plataforma. Por ejemplo, en plataformas Intel®, el campo de desensamblado 46a puede incluir un contenido de unos campos de codificación prefijo, codop, modR/M, SIB, desplazamiento, y valor inmediato de la instrucción de invitado vigente. En otras plataformas, el campo 46a puede almacenar otros valores de acuerdo con la arquitectura del conjunto de instrucciones (ISA - Instruction Set Architecture) de la respectiva plataforma.

En algunas realizaciones, el campo de resultado de ejecución 46b del registro de eventos de suspensión 44 puede almacenar datos indicativos de un resultado de ejecutar la respectiva instrucción de procesador. Estos resultados pueden incluir un valor de un registro de estado (por ejemplo, FLAGS), un valor de un puntero de instrucción (por ejemplo, RIP), y un valor de un registro de propósito general (por ejemplo, EAX) que se producen como resultado de ejecutar la respectiva instrucción. El campo 46b puede comprender además un valor a ser consolidado a memoria como resultado de ejecutar la respectiva instrucción, un tamaño del valor respectivo (por ejemplo, byte, palabra, etc.) y/o una dirección de memoria en la que debe ser consolidado el respectivo valor.

En algunas realizaciones, las una o más unidades de ejecución 36 y/o la unidad de consolidación 38 pueden estar configuradas para determinar si la ejecución de una instrucción de invitado provoca o no un evento de procesador de VM (tal como una salida de VM de excepción de virtualización), y cuando sí, guardar datos de desensamblado de instrucciones en el registro de eventos de suspensión 44 antes de generar el respectivo evento. El procesador 12 puede estar configurado además para retrasar la generación del evento de procesador hasta la finalización de la etapa de ejecución de la respectiva instrucción de invitado, y para guardar un resultado de ejecutar la respectiva instrucción en el registro de eventos 44 en vez de consolidar dichos resultados en memoria y/o en un registro de propósito general del procesador 12. Para evitar consolidar resultados de dichas instrucciones, el procesador 12 puede estar configurado para generar el evento de procesador de VM antes de la etapa de consolidación del pipeline para la respectiva instrucción. Más adelante se detallará más dicha funcionalidad.

La Figura 8 muestra una secuencia ejemplar, detallada, de pasos realizados por el procesador 12 para ejecutar una instrucción de invitado de acuerdo con algunas realizaciones de la presente invención. La Figura 8 muestra una realización, en la cual el procesador 12 está configurado para generar un evento de salida de VM en respuesta a una violación de acceso a memoria. Una persona con experiencia en la técnica apreciará que la presente descripción se puede modificar fácilmente para cubrir una realización, que genera otros eventos de suspensión de VM (tales como excepción de virtualización) en vez de un evento de salida de VM. “Instrucción de invitado” es un término utilizado en esta memoria para denotar una instrucción de procesador que forma parte de un programa informático que se ejecuta dentro de una VM invitada, tal como las VMs 52a-b en la Figura 2-A.

Un paso 302 intenta extraer la instrucción de invitado. Cuando falla el intento de extracción, un paso 303 puede determinar si el fallo es provocado por una violación de acceso a memoria (por ejemplo, cuando la instrucción de invitado reside en una página de memoria marcada como no ejecutable en una estructura SLAT de la VM invitada). Cuando no, en un paso 306, el procesador 12 genera un evento de salida de VM y transfiere la ejecución a un manejador de eventos, tal como el manejador 61 en la Figura 2-A. Cuando el fallo en la extracción de la instrucción de invitado es provocado por una violación de acceso a memoria, dicho fallo puede ser indicativo de que un programa de seguridad (por ejemplo, un módulo anti malware) está intentando proteger un contenido de la respectiva página de memoria. Una sección de memoria ejemplar típicamente protegida de ejecución de esta manera almacena una pila de ejecución de un proceso de invitado. Marcar la pila como no ejecutable puede proteger el proceso de invitado, por ejemplo, de un exploit de pila. En estas situaciones, algunas realizaciones pueden reintentar extraer la instrucción de invitado, ignorando los respectivos permisos de acceso a memoria (paso 305). En un paso 307, la instrucción de invitado extraída se marca con una bandera dedicada, para indicar que la respectiva instrucción fue “extraída a la fuerza”, es decir, fue extraída mientras se rompían permisos de acceso a memoria. El procesador 12 puede entonces continuar a un paso 308.

Después de la etapa de extracción, el paso 308 decodifica y despacha la instrucción de invitado. En un paso 310, se lanza la ejecución de la instrucción de invitado. Cuando la ejecución de la instrucción de invitado satisface un criterio para salida de VM, donde el criterio no está relacionado con acceso a memoria, el procesador 12 continúa a un paso 322 detallado más adelante. Dichas salidas de VM se pueden disparar en una variedad de situaciones. Por ejemplo, la instrucción de invitado puede ser una instrucción especializada, tal como VMCALL, la cual dispara automáticamente un evento de salida de VM cuando es llamada desde dentro de una VM invitada. Otra razón ejemplar para salida de VM, que no está relacionada con acceso a memoria, es que se produzca un evento hardware (por ejemplo, una interrupción) durante la ejecución de la instrucción de invitado.

Cuando la ejecución de la instrucción de invitado provoca una violación de acceso a memoria (por ejemplo, cuando la instrucción de invitado da instrucciones al procesador para que escriba un resultado en una página de memoria marcada como de no escritura), un procesador convencional típicamente suspende la ejecución de la instrucción de invitado, limpia los uno o más pipelines del procesador y genera un evento de suspensión de VM (por ejemplo, VMExit). Por el contrario, en algunas realizaciones de la presente invención, la ejecución de la instrucción de invitado no se suspende. En vez de esto, en un paso 318, el evento de salida de VM se retrasa hasta que finaliza la

etapa de ejecución del pipeline para la instrucción de invitado. Sin embargo, en algunas realizaciones, los resultados de la etapa de ejecución completada no son consolidados, como sucedería en los sistemas convencionales. En vez de esto, en un paso 320, el procesador 12 puede dar instrucciones a la unidad de consolidación 38 para que almacene los resultados de la etapa de ejecución completada de la instrucción de invitado en el registro de eventos de suspensión 44. Esta funcionalidad se puede conseguir, por ejemplo, utilizando una señal de activación para conmutar la unidad de consolidación 38 de consolidar resultados a memoria y/o registros de propósito general del procesador 12, a almacenar resultados en el registro 44 cuando se ha producido una violación de acceso a memoria. La señal de control puede indicar si la ejecución de la instrucción de invitado ha provocado una violación de acceso a memoria. La unidad de consolidación 38 puede recibir una señal de este tipo, por ejemplo, procedente de la MMU a través del módulo de acceso a memoria 34. En algunas realizaciones, el paso 320 comprende que la unidad de consolidación 38 recupere un resultado de ejecutar la instrucción de invitado desde el fichero de registros 40.

En una realización alternativa, en vez de guardar los resultados de ejecución de la instrucción de invitado en el registro 44, el paso 320 puede guardar dichos resultados en una zona de memoria dedicada, tal como el área de estado del invitado del VMSO de la respectiva VM invitada. En otra realización adicional, el procesador 12 puede transmitir dichos resultados al manejador de salida de VM 61 tras ejecutar la salida VM (paso 306).

En un paso 322, el procesador 12 puede almacenar resultados de desensamblar la instrucción de invitado en el registro de eventos de suspensión 44 (y/o en memoria como se ha descrito anteriormente). De forma alternativa, datos de desensamblado de instrucción se pueden almacenar en un área dedicada del VMSO de la VM invitada que se está ejecutando en ese momento. Los datos de desensamblado de instrucción pueden ser producidos por el decodificador de instrucciones 30 y/o una o más unidades de ejecución 36 durante la decodificación y/o ejecución de la instrucción de invitado; el paso 322 puede incluir recuperar dichos datos del respectivo módulo de procesador. Después de almacenar resultados de ejecución y/o datos de desensamblado para la instrucción de invitado, el procesador 12 puede generar un evento de salida VM (paso 306).

Cuando la ejecución de la instrucción de invitado vigente se desarrolla sin provocar violaciones de acceso a memoria (paso 314) y sin razones no relacionadas con memoria para una salida de VM (paso 312), un paso 315 puede determinar si la instrucción de invitado vigente fue extraída a la fuerza (véanse los pasos 305-307 anteriores). Cuando no, un paso 316 consolida resultados de la ejecución en memoria y/o en registros de procesador de propósito general. Cuando la instrucción de invitado vigente es extraída a la fuerza, algunas realizaciones pueden tratar la respectiva instrucción como una instrucción que provoca una violación de acceso a memoria, es decir, esperando a que la respectiva instrucción complete la etapa de ejecución del pipeline, almacenando resultados y/o datos de desensamblado de instrucción en el registro 44, antes de generar un evento de salida VM (véanse los pasos 318-320-322-306 anteriores).

La Figura 9 muestra una secuencia ejemplar de pasos realizados por una VM invitada y/o por el módulo de seguridad informática 60 (Figuras 2-A-B) de acuerdo con algunas realizaciones de la presente invención relacionadas con seguridad informática. Un proceso de invitado, tal como una aplicación (por ejemplo, 56a en la Figura 2-A) o un proceso del sistema operativo (por ejemplo, OS de invitado 54a en la Figura 2-A) se puede ejecutar dentro de la VM invitada, avanzando paso a paso a través de una secuencia de instrucciones de invitado (paso 332). La ejecución del proceso de invitado continúa hasta que se genera una salida de VM, de acuerdo, por ejemplo, con un escenario descrito anteriormente en relación con la Figura 8. Una persona con experiencia en la técnica puede apreciar cómo se puede adaptar la descripción a un sistema en el que el procesador 12 genera una excepción de virtualización en vez de un evento de salida de VM, y en el que un manejador de excepciones que se ejecuta dentro de la VM invitada (por ejemplo, manejador 63 en la Figura 2-B) está configurado para interceptar la respectiva excepción.

En un paso 336, el manejador 61 intercepta el evento de salida de VM, el cual es analizado en búsqueda de evidencia de una amenaza de seguridad. Cuando el evento indica una amenaza de seguridad (por ejemplo, una operación ejecutada con intenciones maliciosas), en un paso 340, el CSM 60 puede emprender acción protectora contra el proceso de invitado y/o contra la VM invitada. Dicha acción puede incluir, entre otros, bloquear la ejecución del proceso de invitado, devolver un mensaje de error o un conjunto de resultados ficticios al proceso de invitado, y alertar a un administrador del sistema anfitrión.

Cuando el evento de salida de VM no es indicativo de una amenaza de seguridad, un paso 342 determina si los resultados de ejecutar la instrucción de invitado están disponibles (ya sea en el registro de eventos 44 del procesador 12 o en memoria). Cuando no, el CSM 60 avanza a un paso 348 detallado más adelante. Cuando sí, un paso 344 recupera los respectivos resultados del registro 44 y/o de memoria (por ejemplo, área de estado del invitado del VMSO de la respectiva VM invitada). En un paso 346, el CSM 60 puede aplicar los resultados de ejecutar la instrucción de invitado vigente. En algunas realizaciones, el paso 346 comprende un conjunto de operaciones llevadas a cabo en sistemas convencionales en la etapa de consolidación. Por ejemplo, el paso 346 puede incluir actualizar valores de registros de procesador de propósito general, de control, y de estado del procesador virtualizado de la respectiva VM invitada. En algunas realizaciones, dichos registros están accesibles dentro del área de estado del invitado del VMSO de la respectiva VM invitada. El paso 346 puede incluir además guardar algunos resultados en direcciones de memoria indicadas por un operando de la instrucción de invitado

vigente. El paso 346 puede incluir además incrementar el puntero de instrucción (por ejemplo, RIP en plataformas x86), para mostrar que la ejecución de la instrucción de invitado vigente está completa.

Algunas realizaciones de la presente invención añaden una instrucción dedicada a la arquitectura del conjunto de instrucciones (ISA - Instruction Set Architecture) vigente del procesador 12, dando instrucciones la nueva instrucción al procesador 12 para que aplique un resultado de ejecución de una instrucción de invitado directamente, desde debajo de la VM invitada que ejecuta la respectiva instrucción de invitado. La nueva instrucción (una mnemotecnia ejemplar es VMAPPLY) puede llevar a cabo operaciones del paso 346 (Figura 9), por ejemplo, copiar contenidos desde el registro de eventos de suspensión 44 en registros virtuales del procesador virtualizado de la VM invitada respectiva y/o en memoria.

En algunas realizaciones, el paso 346 puede además verificar si la instrucción de invitado vigente es una instrucción atómica (por ejemplo, como se indica mediante un prefijo LOCK). Cuando sí, en vez de aplicar resultados directamente a registros del invitado y/o a memoria, el paso 346 puede forzar una reejecución de la instrucción de invitado vigente tras volver a la VM invitada (véase el paso 356 más adelante).

Cuando no están disponibles resultados de ejecución de la instrucción de invitado vigente (por ejemplo, cuando la salida de VM vigente fue provocada por una instrucción con privilegios tal como VMCALL), en un paso 348, el módulo de seguridad informática 60 determina si están disponibles datos de desensamblado para la instrucción de invitado vigente. Cuando sí, en un paso 350, el CSM 60 puede recuperar dichos datos, por ejemplo del campo de desensamblado 46a del registro 44 (véase, por ejemplo, la Figura 6). El CSM 60 puede proceder entonces a emular la instrucción de invitado vigente de acuerdo con los datos de desensamblado recuperados (paso 354).

Cuando no está disponible ningún dato de desensamblado, un paso 352 puede desensamblar la instrucción de invitado vigente antes de proceder con la emulación. En un paso 356, el CSM 60 puede relanzar la VM invitada respectiva (por ejemplo, emitiendo una instrucción VMRESUME en plataformas Intel®). En algunas realizaciones en las que el paso 346 incluye una modificación del puntero de instrucción, la ejecución del proceso de invitado empezará con la instrucción de procesador inmediatamente detrás de la instrucción de invitado vigente, o con una instrucción de procesador indicada por la instrucción de invitado vigente (por ejemplo, en el caso de instrucciones de cambio de flujo de control tales como JMP, CALL, etc.).

Los sistemas y métodos ejemplares descritos anteriormente permiten que un sistema anfitrión, tal como un ordenador o un teléfono inteligente, pueda llevar a cabo de forma eficiente tareas de seguridad informática cuando opera en una configuración de virtualización de hardware. Las tareas de seguridad pueden incluir, entre otras, proteger al sistema anfitrión contra malware tal como virus informáticos y spyware. En algunas realizaciones, el sistema anfitrión está configurado para ejecutar un sistema operativo y un conjunto de aplicaciones software dentro de una máquina virtual. Un módulo de seguridad puede ejecutarse fuera de la respectiva máquina virtual, por ejemplo al nivel de un hipervisor, y puede proteger a la respectiva máquina virtual contra malware.

En algunas realizaciones, el módulo de seguridad identifica una sección de memoria (por ejemplo, un conjunto de páginas de memoria) que contienen código y/o datos que son críticos para la seguridad de la VM protegida, y configura permisos de acceso para la respectiva sección de memoria. Dichos permisos de acceso pueden indicar, por ejemplo, que la respectiva sección de memoria es de no escritura y/o no-ejecutable. El módulo de seguridad puede además configurar el procesador del sistema anfitrión para generar un evento de suspensión de VM (tal como una salida de VM o una excepción de virtualización) en respuesta a una violación de acceso a memoria, por ejemplo, cuando software que se ejecuta dentro de la VM protegida intenta escribir en una sección de memoria marcada como de no escritura, o ejecutar código desde una sección de memoria marcada como no-ejecutable. El módulo de seguridad puede entonces interceptar estos eventos de procesador a través de un manejador de eventos, y puede determinar si dichos eventos son o no indicativos de una amenaza de seguridad informática. En configuraciones en las que el módulo de seguridad se ejecuta fuera de la VM protegida, la actividad del módulo de seguridad es potencialmente invisible para software que se ejecuta dentro de la VM protegida, incluido malware.

En los sistemas convencionales, la interceptación de eventos de suspensión de VM se desarrolla de acuerdo con métodos conocidos de manera genérica en la técnica como “atrapar y emular”. En un ejemplo de una técnica convencional, después de determinar qué instrucción provocó el respectivo evento (por ejemplo, una salida de VM), el programa anti-malware emula la respectiva instrucción antes de devolver la ejecución a la VM protegida, y modifica el puntero de instrucción para indicar que la respectiva instrucción ya ha sido ejecutada. Sin el paso de emulación, devolver la ejecución a la VM protegida dispararía típicamente la salida de VM, creando de esta manera un bucle infinito.

Por lo tanto, los sistemas y métodos atrapar y emular convencionales pueden requerir que el programa anti-malware incluya un desensamblador de instrucciones y/o un emulador de instrucciones. Estos componentes pueden ser complejos de desarrollar y de mantener y pueden no ser portátiles, por ejemplo, de un procesador a otro. Además, en los sistemas convencionales, los pasos de desensamblado y/o emulación se llevan a cabo típicamente para cada evento de suspensión de VM, colocando una considerable carga computacional sobre el sistema anfitrión. Por el contrario, algunas realizaciones de la presente invención eliminan la necesidad de un desensamblador y/o un emulador, acelerando substancialmente las operaciones de seguridad informática.

Algunas realizaciones de la presente invención introducen cambios en la configuración y funcionamiento de los procesadores convencionales, permitiendo que dichos procesadores operen de forma más eficiente en configuraciones de virtualización de hardware. En algunas realizaciones, el procesador está configurado para retrasar la generación del evento de suspensión de VM hasta que la fase de ejecución del pipeline para la instrucción vigente está completa, al menos en ciertas situaciones (por ejemplo cuando el evento de suspensión es disparado por una violación de acceso a memoria). El procesador puede estar configurado además para guardar un resultado de la etapa de ejecución de la instrucción vigente en un registro de procesador especial (distinto a los registros de propósito general o registros de control del procesador) o en un área de memoria especial (por ejemplo, un área de estado del invitado del VMSO de la respectiva VM invitada).

Estas mejoras pueden ser especialmente beneficiosas para aplicaciones de seguridad informática, permitiendo una protección eficiente de una máquina virtual desde el exterior, por ejemplo, desde el nivel de un hipervisor que expone a la VM respectiva. Cuando se comparan con una solución de seguridad informática convencional, algunas realizaciones de la presente invención permiten una reducción substancial en computación, al eliminar las etapas de desensamblado y emulación del funcionamiento del software de seguridad configurado para interceptar y analizar eventos de suspensión de VM. En vez de emular la instrucción que generó el respectivo evento, algunas realizaciones permiten que el software de seguridad pueda leer los respectivos resultados desde un registro de procesador o posición de memoria, y pueda aplicar dichos resultados directamente.

A diferencia de los sistemas convencionales, algunas realizaciones de la presente invención generan un evento de suspensión de VM sólo después de que la instrucción de invitado vigente completa la etapa de ejecución del pipeline. Estos cambios en la funcionalidad de componentes hardware pueden introducir un retraso debido a tics de reloj extra necesarios para llevar a cabo la etapa de ejecución de la respectiva instrucción. Sin embargo, dicha penalización de prestaciones es substancialmente compensada por la eliminación de la emulación de instrucciones y/u operaciones de desensamblado necesarias en software de seguridad informática convencional (que potencialmente ascienden a cientos de instrucciones adicionales para cada evento de suspensión de VM).

Para una persona con experiencia en la técnica será evidente que las realizaciones anteriores se pueden alterar de muchas maneras sin salirse del alcance de la invención. La invención se define en las reivindicaciones adjuntas.

REIVINDICACIONES

1. Un sistema anfitrión que comprende al menos un procesador hardware configurado para ejecutar una máquina virtual y un programa de seguridad informática, en donde el al menos un procesador hardware está configurado además para:

en respuesta a la recepción de una instrucción de invitado para ejecución, determinar si la ejecución de la instrucción de invitado dentro de la máquina virtual provoca una violación de un permiso de acceso a memoria; y

en respuesta a la determinación de si la ejecución de la instrucción de invitado provoca la violación, cuando la ejecución de la instrucción de invitado provoca la violación:

determinar un resultado de aplicar un operador de la instrucción de invitado a un operando de la instrucción de invitado;

escribir el resultado en una posición predeterminada accesible para el programa de seguridad informática;

suspender la ejecución de la instrucción de invitado; y

en respuesta a la suspensión de la ejecución de la instrucción de invitado, conmutar a ejecutar el programa de seguridad informática, en donde el programa de seguridad informática está configurado para determinar si la violación es indicativa de una amenaza de seguridad informática.

2. El sistema anfitrión de la reivindicación 1, en el cual la instrucción de invitado da instrucciones al al menos un procesador hardware para que escriba un valor en un registro del al menos un procesador hardware, y en el cual el resultado comprende el valor.

3. El sistema anfitrión de la reivindicación 2, en el cual el registro es un registro de propósito general del al menos un procesador hardware.

4. El sistema anfitrión de la reivindicación 2, en el cual el registro es un registro de control del al menos un procesador hardware.

5. El sistema anfitrión de la reivindicación 1, en el cual la instrucción de invitado da instrucciones al al menos un procesador hardware para que escriba un valor en una memoria del sistema anfitrión, y en el cual el resultado comprende el valor.

6. El sistema anfitrión de la reivindicación 1, en el cual el resultado comprende una dirección de memoria.

7. El sistema anfitrión de la reivindicación 1, en el cual el resultado comprende el operando de la instrucción de invitado.

8. El sistema anfitrión de la reivindicación 1, en el cual el programa de seguridad informática se ejecuta dentro de la máquina virtual.

9. El sistema anfitrión de la reivindicación 1, en el cual el programa de seguridad informática se ejecuta fuera de la máquina virtual.

10. El sistema anfitrión de la reivindicación 1, en el cual la posición predeterminada comprende un registro predeterminado del al menos un procesador hardware.

11. El sistema anfitrión de la reivindicación 1, en el cual la posición predeterminada comprende una sección predeterminada de una memoria del sistema anfitrión.

12. El sistema anfitrión de la reivindicación 1, en el cual la posición predeterminada comprende una estructura de datos indicativa de un estado en ese momento de la máquina virtual.

13. El sistema anfitrión de la reivindicación 1, en el cual el al menos un procesador hardware está configurado además para, en respuesta a la conmutación a ejecutar el programa de seguridad informática, leer el resultado desde la posición predeterminada.

14. El sistema anfitrión de la reivindicación 13, en el cual el al menos un procesador hardware está configurado además para, en respuesta a la lectura del resultado, escribir el resultado en un destino determinado de acuerdo con la instrucción de invitado.

15. El sistema anfitrión de la reivindicación 1, en el cual el programa de seguridad informática comprende una instrucción que, cuando es ejecutada por el al menos un procesador hardware, provoca que el al menos un procesador hardware lea el resultado desde la posición predeterminada y escriba el resultado en un destino determinado de acuerdo con la instrucción de invitado.

16. El sistema anfitrión de la reivindicación 1, en el cual el al menos un procesador hardware está configurado además para, en respuesta a la conmutación a ejecutar el programa de seguridad informática:

determinar si la instrucción de invitado es una instrucción atómica; y

en respuesta, cuando la instrucción de invitado es una instrucción atómica, ejecutar la instrucción de invitado dentro de la máquina virtual.

17. Un método de proteger un sistema anfitrión de amenazas de seguridad informática, comprendiendo el método:

en respuesta a la recepción de una instrucción de invitado para ejecución, emplear al menos un procesador del sistema anfitrión para determinar si la ejecución de la instrucción de invitado provoca una violación de un permiso de acceso a memoria, en donde la instrucción de invitado se ejecuta dentro de una máquina virtual invitada expuesta por el sistema anfitrión; y

en respuesta a la determinación de si la instrucción de invitado provoca la violación, cuando la ejecución de la instrucción de invitado provoca la violación:

emplear el al menos un procesador hardware para determinar un resultado de aplicar un operador de la instrucción de invitado a un operando de la instrucción de invitado;

emplear el al menos un procesador hardware para escribir el resultado en una posición predeterminada accesible para el programa de seguridad informática;

suspender la ejecución de la instrucción de invitado; y

en respuesta a la suspensión de la ejecución de la instrucción de invitado, conmutar a ejecutar el programa de seguridad informática, en donde el programa de seguridad informática está configurado para determinar si la violación es o no indicativa de una amenaza de seguridad informática.

18. El método de la reivindicación 16, que comprende además, en respuesta a la conmutación a ejecutar el programa de seguridad informática, emplear el al menos un procesador hardware para leer el resultado desde la posición predeterminada.

19. El método de la reivindicación 17, que comprende además, en respuesta a la lectura del resultado, emplear el al menos un procesador para escribir el resultado en un destino determinado de acuerdo con la instrucción de invitado.

20. Al menos un procesador hardware de un sistema anfitrión, siendo el al menos un procesador hardware configurable para:

en respuesta a la recepción de una instrucción de invitado para ejecución, determinar si ejecutar la instrucción de invitado provoca una violación de un permiso de acceso a memoria, en donde la instrucción de invitado se ejecuta dentro de una máquina virtual invitada expuesta por el sistema anfitrión; y

en respuesta a la determinación de si la instrucción de invitado provoca la violación, cuando la ejecución de la instrucción de invitado provoca la violación:

determinar un resultado de aplicar un operador de la instrucción de invitado a un operando de la instrucción de invitado;

escribir el resultado en una posición predeterminada accesible para el programa de seguridad informática;

suspender la ejecución de la instrucción de invitado; y

en respuesta a la suspensión de la ejecución de la instrucción de invitado, conmutar a ejecutar un programa de seguridad informática, en donde el programa de seguridad informática está configurado para determinar si la violación es indicativa de una amenaza de seguridad informática.

21. Un medio legible por un ordenador no transitorio que almacena instrucciones que, cuando son ejecutadas por al menos un procesador hardware de un sistema anfitrión, provocan que el sistema anfitrión conforme un programa de seguridad informática configurado para determinar si una violación de un permiso de acceso a memoria es indicativa de una amenaza de seguridad informática, y donde el al menos un procesador hardware es configurable para:

en respuesta a la recepción de una instrucción de invitado para ejecución, determinar si la ejecución de la instrucción de invitado provoca la violación, en donde la instrucción de invitado se ejecuta dentro de una máquina virtual invitada expuesta por el sistema anfitrión; y

en respuesta a la determinación de si la instrucción de invitado provoca la violación, cuando la ejecución de la instrucción de invitado provoca la violación:

determinar un resultado de aplicar un operador de la instrucción de invitado a un operando de la instrucción de invitado;

escribir el resultado en una posición predeterminada accesible para el programa de seguridad informática;

suspender la ejecución de la instrucción de invitado; y

en respuesta a la suspensión de la ejecución de la instrucción de invitado, conmutar a ejecutar el programa de seguridad informática.

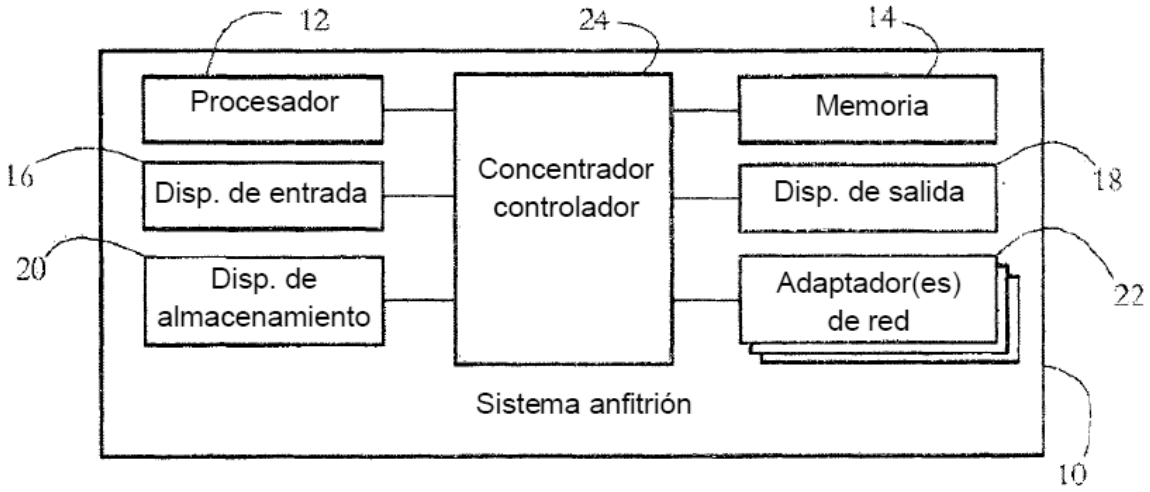
12
24
14
Procesador
Memoria
16
Disp. de entrada
Concentrador controlador
Disp. de salida
18
20
Disp. de almacenamiento
Adaptador(es) de red
22
Sistema anfitrión
10

FIG. 1

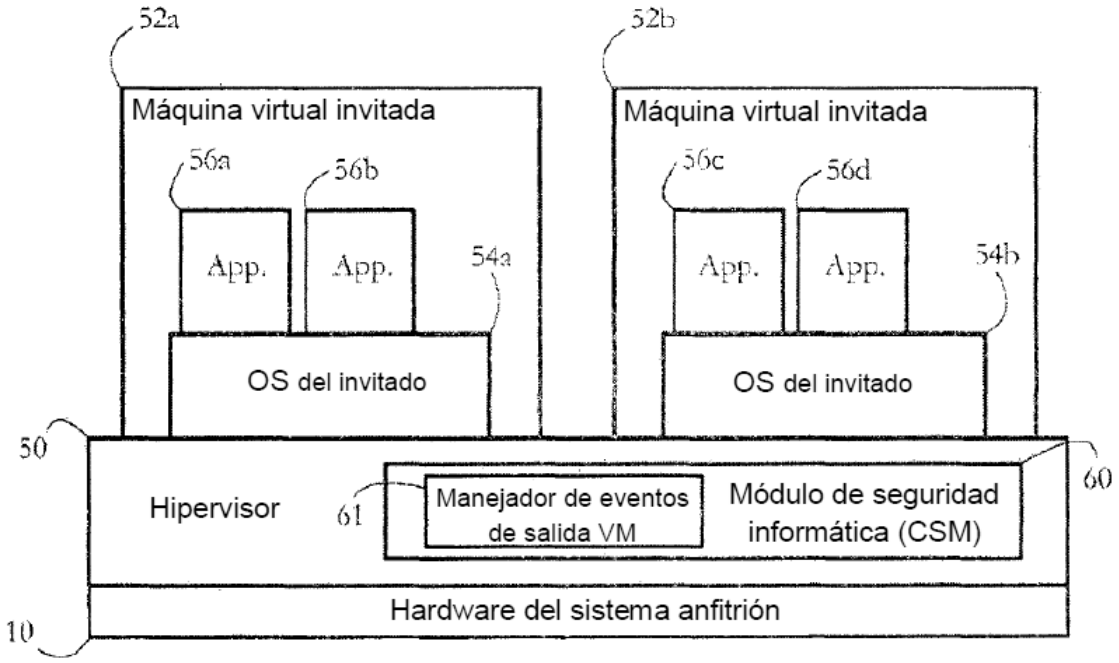
52a
52b
Máquina virtual invitada
Máquina virtual invitada
56a
56b
56c
56d
App.
App.
App.
App.
54a
54b
OS del invitado
OS del invitado
50
60
Hipervisor
61
Manejador de eventos de salida VM
Módulo de seguridad informática (CSM)
Hardware del sistema anfitrión
10

FIG. 2-A

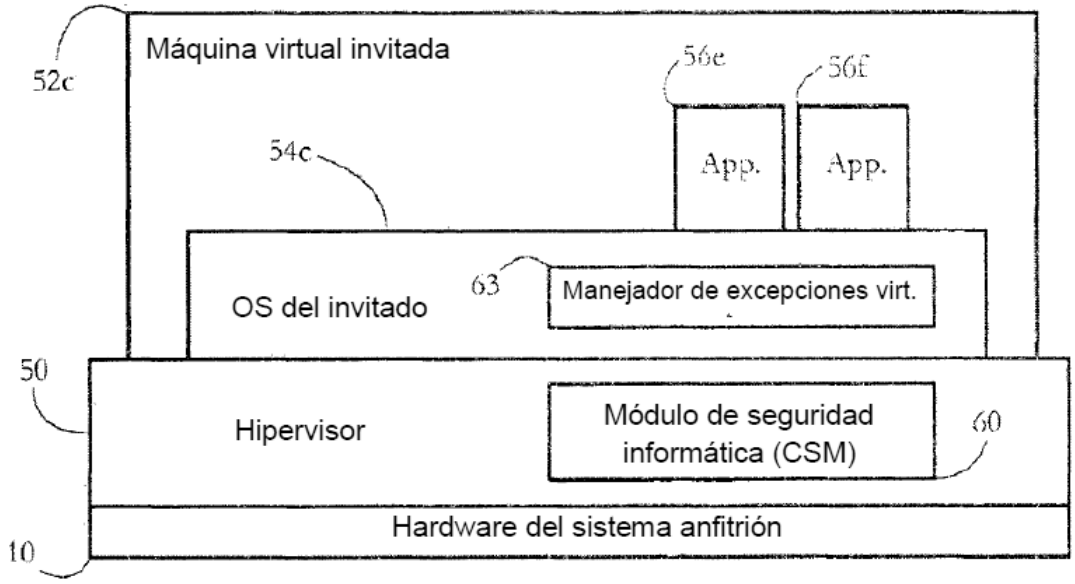
52c
Máquina virtual invitada
56e
56f
App.
App.
54c
OS del invitado
63
Manejador de excepciones virt.
50
Hipervisor
Módulo de seguridad informática (CSM)
60
Hardware del sistema anfitrión
10

FIG. 2-B

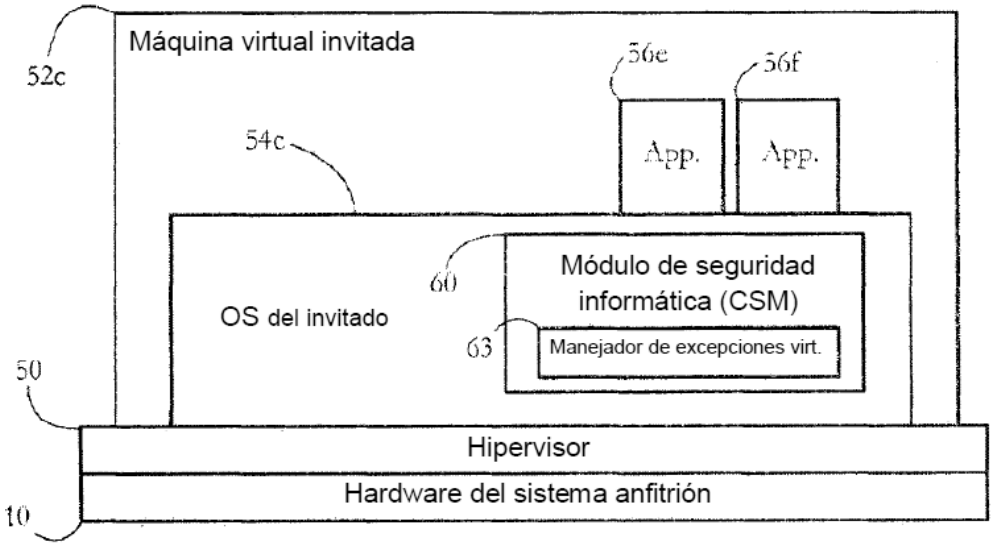
Máquina virtual invitada
52c
56e
56f
App.
App.
54c
OS del invitado
60
Módulo de seguridad informática (CSM)
63
Manejador de excepciones virt.
50
Hipervisor
Hardware del sistema anfitrión
10

FIG. 2-C

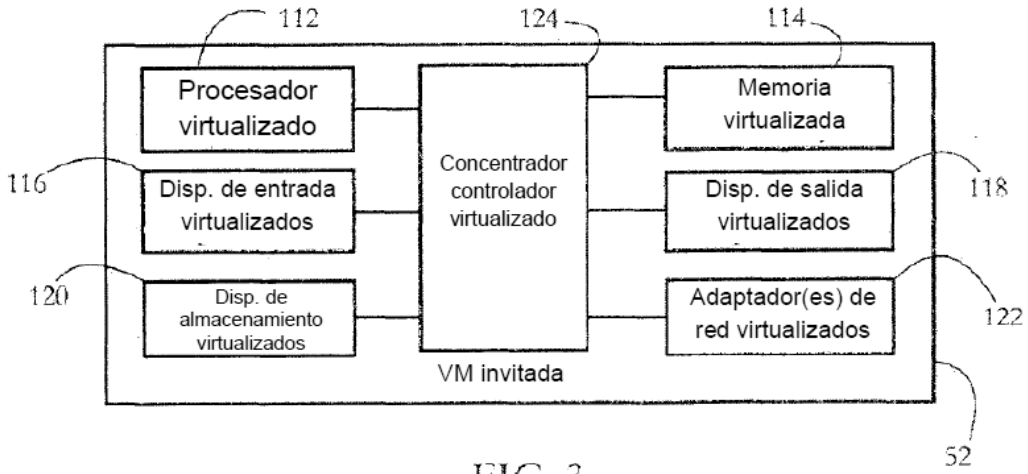
112
124
114
Procesador virtualizado
Memoria virtualizada
116
Disp. de entrada virtualizados
Concentrador controlador virtualizado
Disp. de salida virtualizados
118
120
Disp. de almacenamiento virtualizados
Adaptador(es) de red virtualizados
122
VM invitada
52

FIG. 3

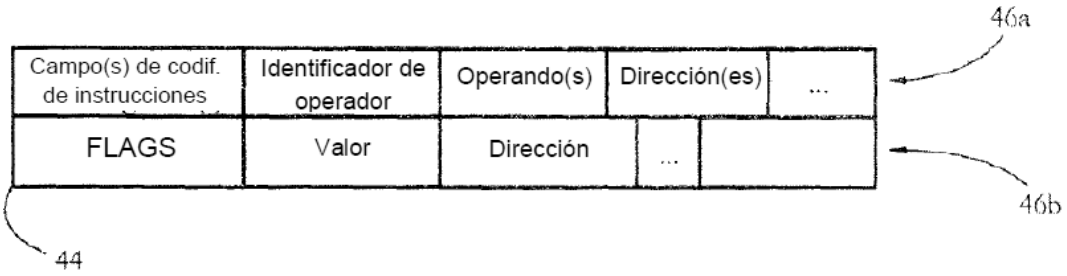
Campo(s) de codif. de instrucciones
Identificador de operador
Operando(s)
Dirección(es)
...
FLAGS
Valor
Dirección
...
46a
46b
44

FIG. 6

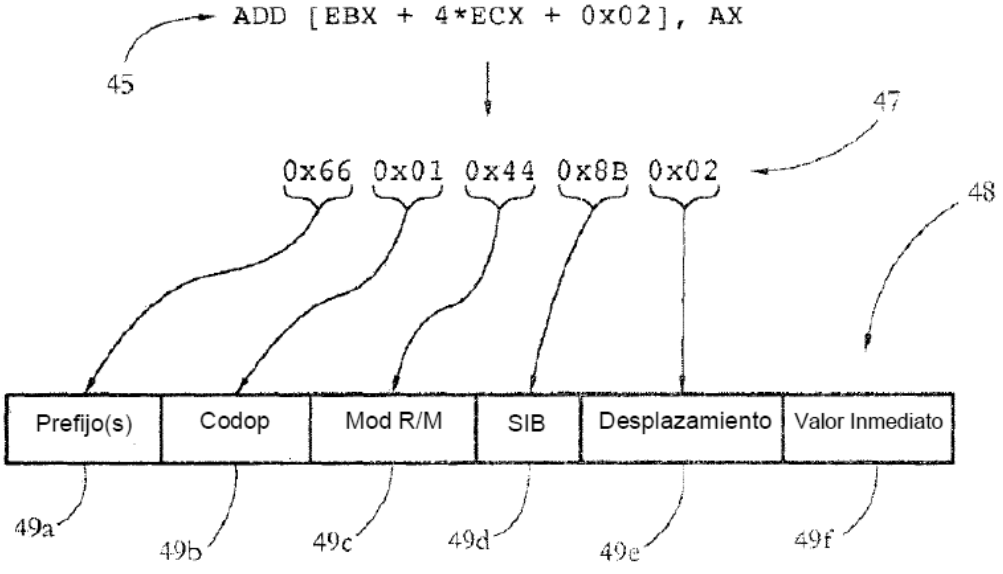
ADD [EBX + 4*ECX + 0x02], AX
45
47
0x66 0x01 0x44 0x8B 0x02
48
Prefijo(s)
Codop
Mod R/M
SIB
Desplazamiento
Valor Inmediato
49a
49b
49c
49d
49e
49f

FIG. 7

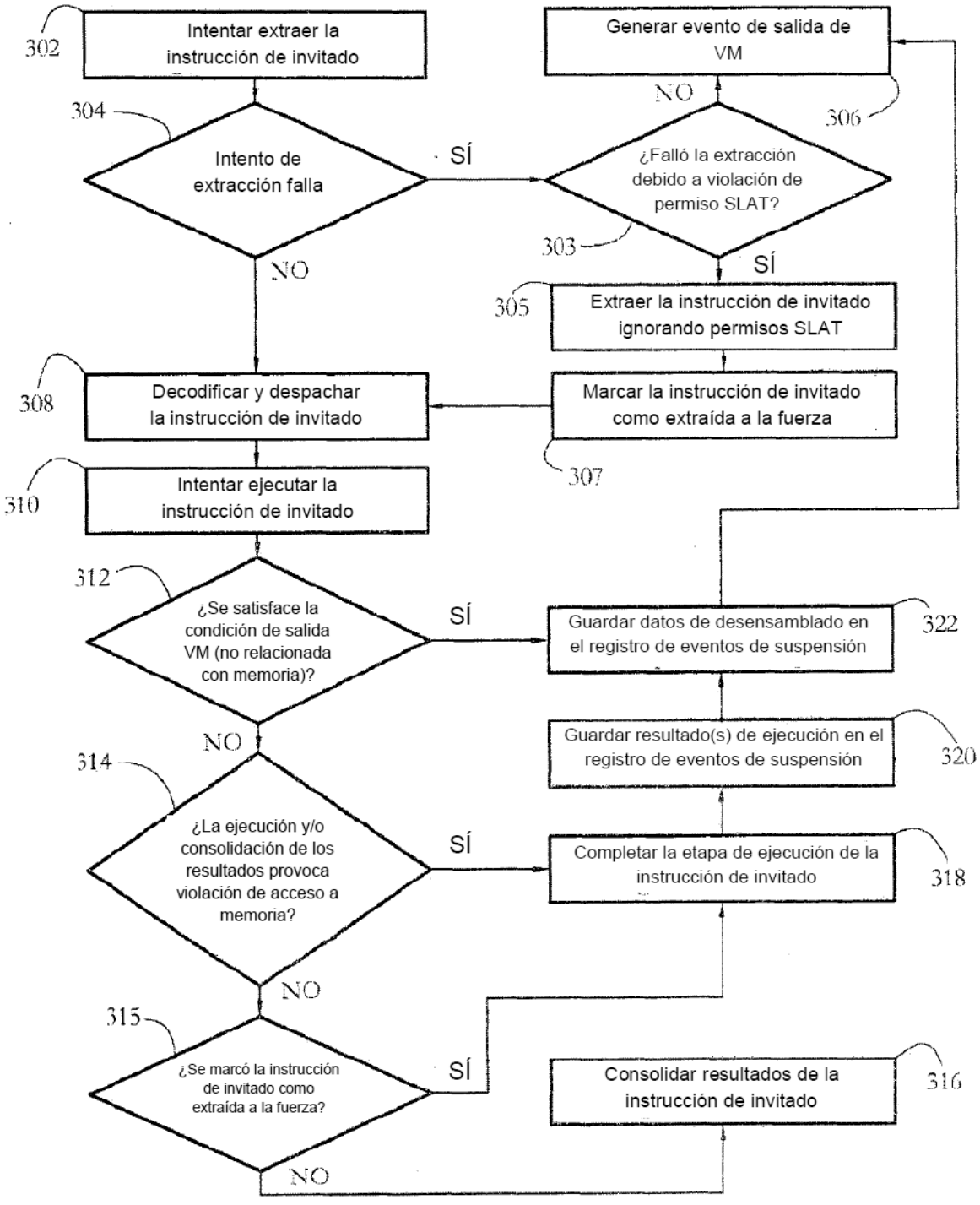
302
Intentar extraer la instrucción de invitado
304
Intento de extracción falla
SÍ
NO
Generar evento de salida de VM
306
NO
¿Falló la extracción debido a violación de permiso SLAT?
303
SÍ
305
Extraer la instrucción de invitado ignorando permisos SLAT
308
Decodificar y despachar la instrucción de invitado
Marcar la instrucción de invitado como extraída a la fuerza
307
310
Intentar ejecutar la instrucción de invitado
312
¿Se satisface la condición de salida VM (no relacionada con memoria)?
SÍ
Guardar datos de desensamblado en el registro de eventos de suspensión
322
NO
Guardar resultado(s) de ejecución en el registro de eventos de suspensión
320
314
¿La ejecución y/o consolidación de los resultados provoca violación de acceso a memoria?
SÍ
Completar la etapa de ejecución de la instrucción de invitado
318
NO
315
¿Se marcó la instrucción de invitado como extraída a la fuerza?
SÍ
Consolidar resultados de la instrucción de invitado
316
NO

FIG. 8

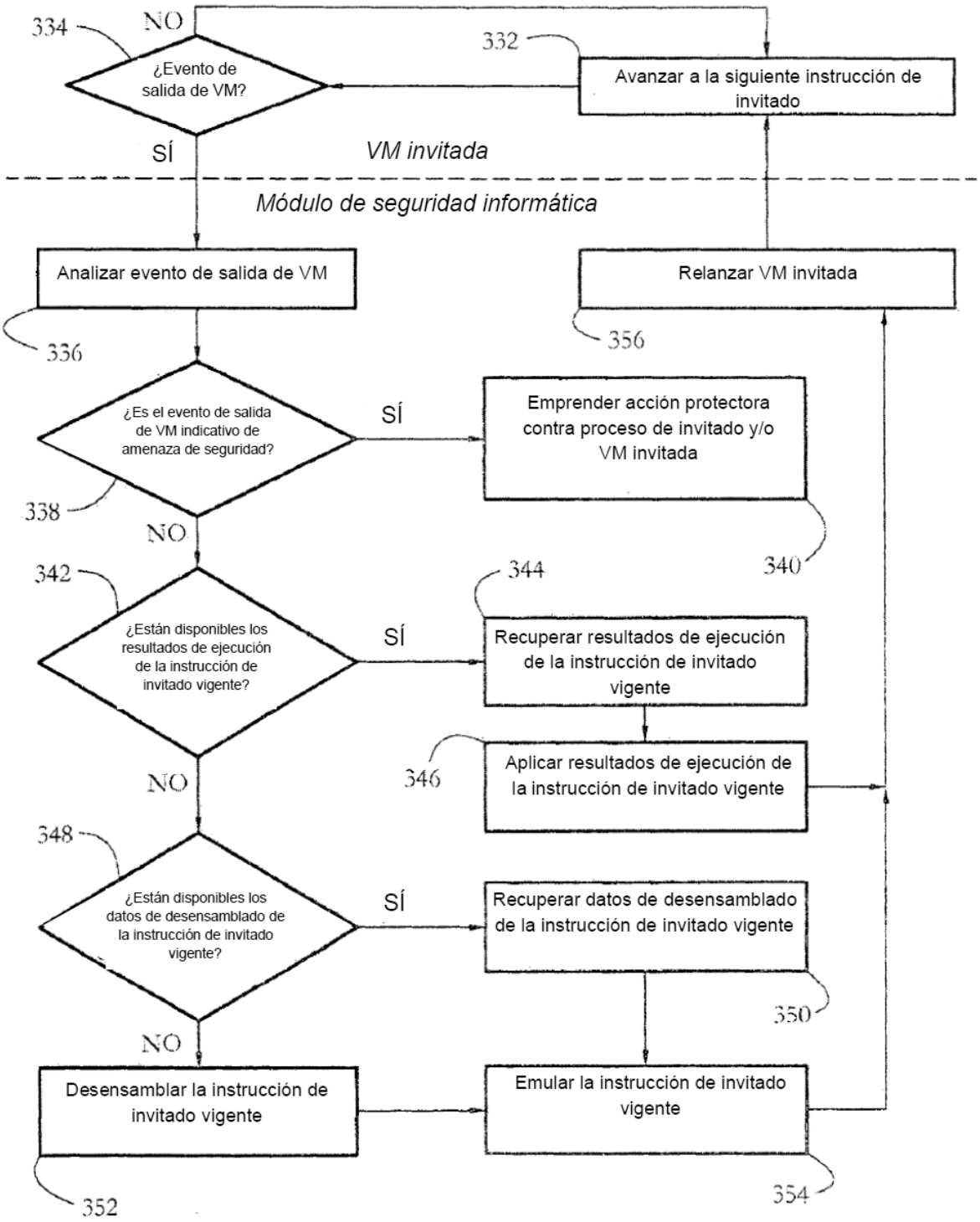
334
NO
¿Evento de salida de VM?
332
Avanzar a la siguiente instrucción de invitado
SÍ
VM invitada
Módulo de seguridad informática
Analizar evento de salida de VM
Relanzar VM invitada
356
336
¿Es el evento de salida de VM indicativo de amenaza de seguridad?
SÍ
Emprender acción protectora contra proceso de invitado y/o VM invitada
338
NO
340
342
344
¿Están disponibles los resultados de ejecución de la instrucción de invitado vigente?
SÍ
Recuperar resultados de ejecución de la instrucción de invitado vigente
346
Aplicar resultados de ejecución de la instrucción de invitado vigente
NO
348
¿Están disponibles los datos de desensamblado de la instrucción de invitado vigente?
SÍ
Recuperar datos de desensamblado de la instrucción de invitado vigente
350
NO
Desensamblar la instrucción de invitado vigente
Emular la instrucción de invitado vigente
352
354

FIG. 9