РОССИЙСКАЯ ФЕДЕРАЦИЯ

(19) **RU** (11) **2 667 713**(13) **C2**

(51) МПК
G06F 9/455 (2006.01)

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

(12) ОПИСАНИЕ ИЗОБРЕТЕНИЯ К ПАТЕНТУ

(52) СПК
G06F 9/455 (2006.01); G06F 12/08 (2006.01)

(21)(22) Заявка: **2016109436, 12.09.2014**

(24) Дата начала отсчета срока действия патента:
12.09.2014

Дата регистрации:
24.09.2018

Приоритет(ы):
(30) Конвенционный приоритет:
17.09.2013 US 61/879,068;
12.02.2014 US 14/179,378

(43) Дата публикации заявки: **21.09.2017** Бюл. № **27**

(45) Опубликовано: **24.09.2018** Бюл. № **27**

(85) Дата начала рассмотрения заявки PCT на национальной фазе: **16.03.2016**

(86) Заявка PCT:
US 2014/055290 (12.09.2014)

(87) Публикация заявки PCT:
WO 2015/041930 (26.03.2015)

Адрес для переписки:
129090, Москва, ул. Б.Спасская, 25, строение 3, ООО "Юридическая фирма Городисский и Партнеры"

(72) Автор(ы):
ХЕПКИН Дэвид А. (US),
ДЖОНСОН Кеннет Д. (US)

(73) Патентообладатель(и):
МАЙКРОСОФТ ТЕКНОЛОДЖИ ЛАЙСЕНСИНГ, ЭлЭлСи (US)

(56) Список документов, цитированных в отчете о поиске: **US 2008/0235534 A1, 25.09.2008. US 2013/0055391 A1, 28.02.2013. RU 2390836 C2, 27.05.2010. RU 2530691 C1, 10.10.2014.**

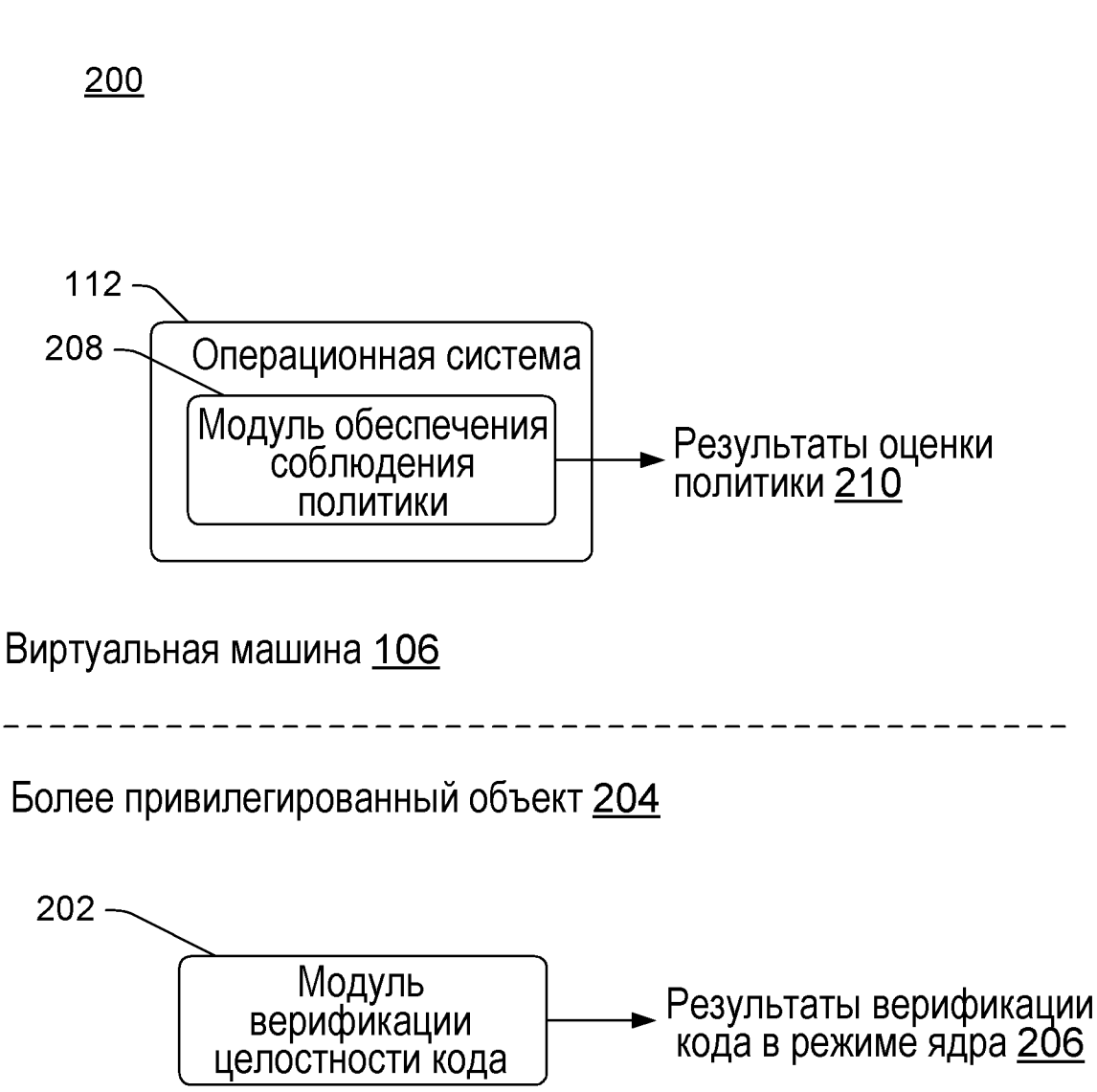
(54) СЕЛЕКТИВНОЕ ОБЕСПЕЧЕНИЕ СОБЛЮДЕНИЯ ЦЕЛОСТНОСТИ КОДА, ОБЕСПЕЧИВАЕМОЕ МЕНЕДЖЕРОМ ВИРТУАЛЬНОЙ МАШИНЫ

(57) Реферат:

Изобретение относится к вычислительной технике. Технический результат заключается в защите от вредоносных программ. Способ, реализованный в вычислительном устройстве, для защиты от вредоносных программ включает в себя исполняемый код, подлежащий исполнению виртуальным процессором виртуальной машины, причем виртуальная машина управляется менеджером виртуальной машины, определение, должна ли страница памяти быть исполняемой в режиме ядра или в пользовательском режиме, предоставление возможности в ответ на определение, что страница памяти не должна быть исполняемой в режиме ядра, определения операционной системой виртуальной машины, предоставить ли возможность исполнения исполняемого кода в пользовательском режиме. 2 н. и 8 з.п. ф-лы, 5 ил.

RU 2667713 C2

RU 2667713 C2



ФИГ. 2

RU 2667713 C2

RU 2667713 C2

RUSSIAN FEDERATION

FEDERAL SERVICE
FOR INTELLECTUAL PROPERTY

(19) **RU** (11) **2 667 713**(13) **C2**

(51) Int. Cl.
G06F 9/455 (2006.01)

(12) ABSTRACT OF INVENTION

(52) CPC
G06F 9/455 (2006.01); G06F 12/08 (2006.01)

(21)(22) Application: **2016109436, 12.09.2014**

(24) Effective date for property rights:
12.09.2014

Registration date:
24.09.2018

Priority:
(30) Convention priority:
17.09.2013 US 61/879,068;
12.02.2014 US 14/179,378

(43) Application published: **21.09.2017** Bull. № **27**

(45) Date of publication: **24.09.2018** Bull. № **27**

(85) Commencement of national phase: **16.03.2016**

(86) PCT application:
US 2014/055290 (12.09.2014)

(87) PCT publication:
WO 2015/041930 (26.03.2015)

Mail address:
129090, Moskva, ul. B.Spasskaya, 25, stroenie 3, OOO "Yuridicheskaya firma Gorodisskij i Partnery"

(72) Inventor(s):
KHEPKIN Devid A. (US),
DZHONSON Kennet D. (US)

(73) Proprietor(s):
MAJKROSOFT TEKNOLODZHI LAJSENSING, ElElSi (US)

RU 2667713 C2

RU 2667713 C2

(54) VIRTUAL MACHINE MANAGER FACILITATED SELECTIVE CODE INTEGRITY ENFORCEMENT

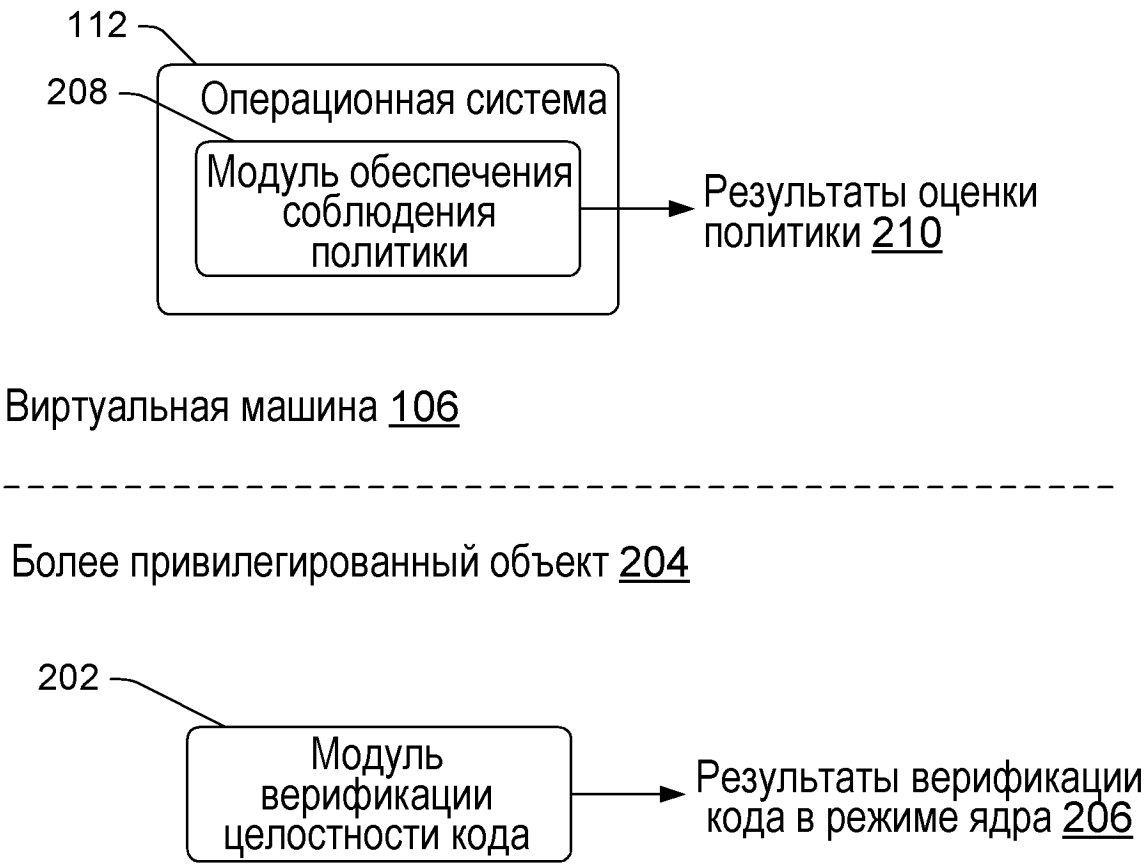
(57) Abstract:

FIELD: computer equipment.

SUBSTANCE: invention relates to the computer equipment. Method implemented in a computing device for protecting against malicious programs comprises executable code to be executed by a virtual processor of a virtual machine, the virtual machine being managed by a virtual machine manager, determining whether the memory page is to be executable in a kernel mode or in an user mode, providing an opportunity in response to determining that the memory page is not to be executable in kernel mode, the operating system of the virtual machine to determine whether to allow execution of the executable code in user mode.

EFFECT: technical result consists in protecting against malicious programs.

10 cl, 5 dwg



ФИГ. 2

RU 2667713 C2

RU 2667713 C2

УРОВЕНЬ ТЕХНИКИ

[0001] Так как вычислительная техника совершенствуется, вычислительные устройства становятся во все возрастающей степени взаимосвязанными. Хотя эта взаимосвязанность обеспечивает многочисленные преимущества, она не без проблем. Одной такой проблемой является то, что вычислительные устройства во все возрастающей степени подвергаются воздействию вредоносных программ. Вредоносные программы могут действовать разными путями, такими как похищение информации с вычислительного устройства, блокирование вычислительного устройства, использование вычислительного устройства для запуска атак против других вычислительных устройств и т.п. Хотя были разработаны некоторые методы для защиты вычислительного устройства от вредоносных программ, такие вредоносные программы остаются и могут привести к разочарованию в опыте взаимодействия пользователя, когда они инфицируют компьютер пользователя.

СУЩНОСТЬ ИЗОБРЕТЕНИЯ

[0002] Этот раздел «Сущность изобретения» предусматривается для выбора идей в упрощенном виде, которые дополнительно описываются ниже в разделе «Подробное описание». Этот раздел «Сущность изобретения» не предназначен ни для определения ключевых признаков или существенных признаков заявленного объекта, ни для использования для ограничения объема заявленного объекта.

[0003] В соответствии с одним или несколькими аспектами, идентифицируется страница памяти, включающая в себя исполняемый код, подлежащий исполнению виртуальным процессором виртуальной машины, причем виртуальная машина управляется менеджером виртуальной машины. Выполняется определение, должна ли страница памяти быть исполняемой в режиме ядра. В ответ на определение, что страница памяти должна быть исполняемой в режиме ядра, выполняется проверка целостности кода исполняемого кода, и предусматривается возможность исполнения исполняемого кода для режима ядра, если только проверка целостности кода верифицирует исполняемый код. В ответ на определение, что страница памяти не должна быть исполняемой в режиме ядра, операционной системе виртуальной машины предоставляется возможность определения, предоставлять ли возможность исполнения исполняемого кода.

[0004] В соответствии с одним или несколькими аспектами, вычислительное устройство включает в себя операционную систему, менеджер виртуальной машины и процессор. Процессор выполнен с возможностью предоставления возможности менеджеру виртуальной машины ограничивать исполнение в режиме ядра страниц памяти страницами памяти, имеющими код, целостность которого была верифицирована более привилегированным объектом, который является более привилегированным, чем операционная система, но предоставления возможности исполнения в пользовательском режиме страниц памяти, не учитывая, была ли верифицирована целостность кода на страницах памяти более привилегированным объектом.

КРАТКОЕ ОПИСАНИЕ ЧЕРТЕЖЕЙ

[0005] Одинаковые позиции используются на чертежах для ссылки на подобные признаки.

[0006] Фиг. 1 представляет собой блок-схему, иллюстрирующую примерное вычислительное устройство, реализующее методы, описанные в данном документе, согласно одному или нескольким вариантам осуществления.

[0007] Фиг. 2 иллюстрирует примерную систему, реализующую методы селективного обеспечения соблюдения целостности кода, обеспечиваемого менеджером виртуальной

машины, описанные в данном документе, согласно одному или нескольким вариантам осуществления.

[0008] Фиг. 3 представляет собой блок-схему последовательности операций, иллюстрирующую примерный процесс для реализации селективного обеспечения соблюдения целостности кода, обеспечиваемого менеджером виртуальной машины, согласно одному или нескольким вариантам осуществления.

[0009] Фиг. 4 иллюстрирует другую примерную систему, реализующую селективное обеспечение соблюдения целостности кода, обеспечиваемое менеджером виртуальной машины, согласно одному или нескольким вариантам осуществления.

[0010] Фиг. 5 иллюстрирует примерную систему, которая включает в себя примерное вычислительное устройство, которое является представителем одной или нескольких систем и/или устройств, которые могут реализовать различные методы, описанные в данном документе.

ПОДРОБНОЕ ОПИСАНИЕ

[0011] В данном документе описывается селективное обеспечение соблюдения целостности кода, обеспечиваемое менеджером виртуальной машины. Виртуальная машина представляет собой программную реализацию физического устройства, которая может выполнять программы, аналогично физическому устройству. Виртуальная машина, и доступ к аппаратным средствам физического устройства, управляется менеджером виртуальной машины на физическом устройстве. Виртуальная машина и менеджер виртуальной машины обращаются к памяти, которая состоит из многочисленных блоков или участков, упоминаемых как страницы памяти (или просто страницы). Целостность кода используется для того, чтобы способствовать защите от вредоносного кода на физическом устройстве. Целостность кода ссылается на целостность кода (например, двоичного), верифицируемого на основе политики целостности кода. Если код верифицируется на основе политики целостности кода, тогда целостность кода верифицируется, и предоставляется возможность исполнения кода; в противном случае, целостность кода не верифицируется, и не предоставляется возможность исполнения кода.

[0012] Процессор может исполнять код в режиме ядра или в пользовательском режиме. Когда виртуальный процессор виртуальной машины выполняет в режиме ядра, виртуальный процессор выполняет только код, целостность которого верифицируется менеджером виртуальной машины (или другим объектом, более привилегированным, чем операционная система, выполняющаяся на виртуальном процессоре). Целостность кода страниц памяти, включающих код, верифицируется, и виртуальный процессор, выполняющий в режиме ядра, может исполнять код на странице памяти, если только менеджер виртуальной машины (или другой более привилегированный объект) верифицировал целостность кода у кода на странице памяти. Однако когда виртуальный процессор выполняет в пользовательском режиме, операционная система, выполняющаяся на виртуальном процессоре, определяет, может ли код исполняться. Операционная система может применять любую из разнообразных политик (например, может выполнять любую из всевозможных разных проверок или верификаций кода) для определения, может ли код исполняться в пользовательском режиме, включая, необязательно, исполнение кода в пользовательском режиме без выполнения каких-либо проверок или верификаций кода.

[0013] Фиг. 1 представляет собой блок-схему, иллюстрирующую примерное вычислительное устройство 100, реализующее методы, описанные в данном документе, согласно одному или нескольким вариантам осуществления. Вычислительным

устройством 100 может быть любое из всевозможных разных типов устройств. Например, вычислительным устройством 100 может быть настольный компьютер, серверный компьютер, портативный компьютер или компьютер-нетбук, планшет или компьютер в виде записной книжки, мобильная станция, развлекательное устройство, телевизионная приставка, соединенная с возможностью связи с устройством отображения, телевизор или другое устройство отображения, сотовый или другой беспроводный телефон, игровая консоль, автомобильный компьютер, носимый компьютер и т.п.

[0014] Вычислительное устройство 100 включает в себя менеджер 102 виртуальной машины, также упоминаемый как гипервизор, и один или несколько компонентов 104. Менеджер 102 виртуальной машины управляет доступом к функциональной возможности, обеспечиваемой компонентами 104. Альтернативно, менеджер 102 виртуальной машины может выполняться на главной операционной системе (не показана), в этом случае главная операционная система управляет доступом к функциональной возможности, обеспечиваемой компонентами 104.

[0015] Компоненты 104 могут представлять собой всевозможные разные процессорные компоненты, компоненты ввода/вывода (I/O) и/или другие компоненты или устройства. Например, компоненты 104 могут включать в себя один или несколько процессоров или процессорных ядер, один или несколько компонентов памяти (например, энергозависимую и/или энергонезависимую память), одно или несколько запоминающих устройств (например, оптических и/или магнитных дисков, накопителей флэш-памяти), один или несколько компонентов связи (например, адаптеры проводной и/или беспроводной сети), их комбинации и т.п. Хотя изображены как часть вычислительного устройства 100, один или несколько из компонентов 104 (например, одно или несколько запоминающих устройств) могут быть реализованы внешними для вычислительного устройства 100. Различные компоненты или модули, выполняющиеся на вычислительном устройстве 100, включая менеджер 102 виртуальной машины, могут выполнять доступ к этой функциональной возможности, обеспечиваемой компонентами 104 прямо и/или косвенно посредством других компонентов или модулей.

[0016] Менеджер 102 виртуальной машины предоставляет возможность виртуальной машине 106 выполняться на вычислительном устройстве 100. Единственная виртуальная машина 106 изображена в вычислительном устройстве 100, хотя, альтернативно, многочисленные виртуальные машины могут выполняться на вычислительном устройстве 100. Виртуальная машина ссылается на программную реализацию физического вычислительного устройства (или другой машины или системы), которое может выполнять программы, аналогично физическому вычислительному устройству. Виртуальная машина включает в себя один или несколько виртуальных компонентов, которые подобны компонентам 104 (но являются их программными реализациями). Операционная система, а также другие приложения, могут исполнять использование виртуальных компонентов, так как они будут использовать компоненты 104, включая выполнение на виртуальных процессорах или виртуальных процессорных ядрах, обращение к виртуальной памяти и т.п. Операционной системе и другим приложениям, исполняющимся в виртуальной машине 106, нет необходимости иметь сведения, и обычно они не имеют сведения, что они исполняются в виртуальной машине.

[0017] Виртуальная машина 106 включает в себя операционную систему 112, одно или несколько приложений 114 и один или несколько виртуальных компонентов 116. Операционная система 112 выполняется или исполняется на одном или нескольких виртуальных процессорах или процессорных ядрах, включенных в виде одного или

нескольких компонентов 116, и управляет исполнением приложений 114.

[0018] Менеджер 102 виртуальной машины включает в себя модуль 122 управления виртуальной машиной (VM) и модуль 124 управления страницами. Модуль 122 управления виртуальной машиной управляет отображением виртуальных компонентов 116 на компоненты 104, включая планирование виртуальных процессоров или процессорных ядер для исполнения на физических процессорах или процессорных ядрах. Модуль 124 управления страницами идентифицирует, какие страницы являются исполняемыми в режиме ядра, и может, необязательно, выполнять проверки целостности кода на коде для страниц памяти, подлежащих исполнению в режиме ядра, как описано более подробно ниже. Хотя они изображены в виде двух отдельных модулей, следует отметить, что функциональная возможность модулей 122 и 124 может быть объединена в единственный модуль (например, функциональная возможность модуля 124 управления страницами может быть включена в модуль 122 управления VM).

[0019] Операционная система 112 и менеджер 102 виртуальной машины управляют хранением и доступом к памяти, которая состоит из многочисленных блоков или участков, которые упоминаются как страницы памяти (или просто страницы). Память, например, может быть памятью любого типа, адресуемой центральным блоком обработки (CPU), такой как энергозависимая память (например, оперативное запоминающее устройство (RAM)) или энергонезависимая память (например, флэш-память). Разные программы могут распределяться страницам памяти, и этими программами могут быть приложения 114, программы операционной системы 112 или другие компоненты или модули.

[0020] Операционная система 112 и менеджер 102 виртуальной машины могут предоставлять возможность выполнения разного типа доступа к страницам памяти программой, такого как доступ для чтения, доступ для записи и доступ для исполнения. Если доступ для чтения (также упоминаемый как разрешение чтения) предоставляется странице памяти, тогда предоставляется возможность считывания содержимого страницы памяти (например, конкретной одной или нескольким программам). Если доступ для записи (также упоминаемый как разрешение записи) предоставляется странице памяти, тогда предоставляется возможность запись содержимого в страницу памяти (например, конкретной одной или несколькими программами). Если доступ для исполнения (также упоминаемый как разрешение исполнения) предоставляется странице памяти, предоставляется возможность исполнение кода, хранимого в (также упоминаемого как хранимого на) странице памяти.

[0021] Операционная система 112 и/или объект, более привилегированный, чем операционная система 112 (например, менеджер 102 виртуальной машины), могут определить, предоставлять ли разрешение исполнения странице памяти, основываясь, по меньшей мере частично, на верификации целостности кода для кода на странице памяти. Верификация целостности кода ссылается на верификацию целостности кода (например, двоичного кода или его частей), основываясь на политике целостности кода. Могут использоваться всевозможные разные политики целостности кода, и целостность кода, таким образом, может верифицироваться всевозможными разными способами. Если целостность кода верифицируется на основе политики целостности кода, тогда целостность кода верифицируется и коду предоставляется возможность исполнения. Однако если целостность кода не верифицируется на основе политики целостности кода, тогда целостность кода не верифицируется и коду не предоставляется возможность исполнения.

[0022] В одном или нескольких вариантах осуществления политика целостности кода

указывает, что целостность кода верифицируется на основе кода, подписанного с использованием цифровых сертификатов, которые идентифицируют источник кода (например, объект, который цифровым образом подписал код) и устанавливает цепочку сертификатов для кода. Код подписывается посредством генерирования цифровой подписи на основе кода и криптографического ключа. Без криптографического ключа (или соответствующего ключа, такого как секретный ключ из пары открытого/секретного ключа) очень трудно из-за громадного объема вычислений создать подпись, которая может быть верифицирована с использованием криптографического ключа. Однако любой объект с криптографическим ключом (или соответствующим ключом, таким как открытый ключ из пары открытого/секретного ключа) может использовать ключ для верификации цифровой подписи посредством исполнения подходящего алгоритма верификации цифровой подписи по ключу, подписи и коду, который был подписан. Так как цифровая подпись основывается на коде, любое изменение кода приводит к неверифицируемой цифровой подписи. Таким образом, цифровой сертификат предоставляет возможность объекту, верифицирующему код, верифицировать, что код не был изменен после того, как код был подписан цифровым образом.

[0023] Объект, верифицирующий код (например, операционная система 112 или менеджер 102 виртуальной машины), идентифицирует один или несколько доверяемых объектов. Эти доверяемые объекты могут идентифицироваться разным образом, таким как предварительное конфигурирование в верифицирующем объекте, предоставление администратором вычислительного устройства 100 или получение иным образом. Может устанавливаться цепочка сертификатов, которая идентифицирует доверяемый объект, а также один или несколько других объектов. Цепочка сертификатов ссылается на ряд объектов, начинающийся с объекта, который подписал цифровым образом код, и заканчивающийся объектом, который доверяется объектом, верифицирующим код. Любое количество дополнительных объектов может быть включено в цепочку, причем каждый объект верифицирует, что он доверяет предыдущему объекту в цепочке. Например, изобретатели предполагают, что код подписывается объектом A, который не доверяется объектом, верифицирующим код, но что объект D доверяется объектом, верифицирующим код. Цепочка сертификатов может включать в себя объект A, который подписал цифровым образом код, объект B, обеспечивающий цифровой сертификат, верифицирующий, что объект B доверяет объекту A, объект C, обеспечивающий цифровой сертификат, верифицирующий, что объект Cдоверяет объекту B, и объект D, обеспечивающий цифровой сертификат, верифицирующий, что объект D доверяет объекту C.

[0024] Если цепочка сертификатов верифицируется, и код не был модифицирован, тогда проверка целостности кода завершается успешно – целостность кода верифицируется и коду предоставляется возможность исполнения. Однако если цепочка сертификатов не верифицируется, и/или код был модифицирован, тогда проверка целостности кода завершается неуспешно – целостность кода не верифицируется и коду не предоставляется возможность исполнения.

[0025] Альтернативно, целостность кода может верифицироваться другими методами. Например, код может генерироваться посредством объекта, или под его управлением, который верифицирует код. Политика целостности кода может указывать, что такой код автоматически рассматривается как верифицированный объектом, и проверка целостности кода для такого кода завершается успешно (целостность кода верифицируется и коду предоставляется возможность исполнения). В качестве другого примера, код может верифицироваться посредством анализирования в соответствии с

различными другими правилами или критериями, указанными политикой целостности кода. Если анализ кода определяет, что была выполнена политика целостности кода, тогда код верифицируется, и проверка целостности кода завершается успешно (целостность кода верифицируется, и коду предоставляется возможность исполнения). Однако если анализ кода определяет, что политика целостности кода не была выполнена, тогда код не верифицируется, и проверка целостности кода завершается неуспешно (целостность кода не верифицируется, и коду не предоставляется возможность исполнения).

[0026] В одном или нескольких вариантах осуществления код программы может храниться на многочисленных страницах памяти, и проверка целостности кода для таких многочисленных страниц выполняется в целом. Проверка целостности кода для кода программы выполняется, и, если проверка целостности кода завершается успешно, тогда разрешение исполнения предоставляется всем многочисленным страницам памяти, в которых хранится код. Однако если проверка целостности кода завершается неуспешно, тогда разрешение исполнения не предоставляется никакой из многочисленных страниц памяти, в которых хранится код.

[0027] Альтернативно, проверка целостности кода для каждой из многочисленных страниц, в которой хранится код программы, может выполняться индивидуально и независимо от проверки целостности кода для кода, хранимого в других из многочисленных страниц. Например, в ответ на попытку исполнения кода в одной из многочисленных страниц памяти, проверка целостности кода выполняется для по меньшей мере кода на этой странице памяти, и разрешение исполнения предоставляется или не предоставляется этой странице памяти, основываясь на том, завершается ли проверка целостности кода неуспешно или успешно.

[0028] Один или несколько процессоров вычислительного устройства 100 поддерживают исполнение кода в многочисленных разных режимах, упоминаемых как режим ядра (также упоминаемый как режим ядра, супервизорский режим или режим супервизора) и пользовательский режим (также упоминаемый как режим пользователя). Приложения 114 обычно выполняются в пользовательском режиме, и операционная система 112 может включать в себя некоторые компоненты ядра, которые выполняются в режиме ядра, и другие компоненты, которые выполняются в пользовательском режиме. Пользовательский режим является менее привилегированным (т.е. более ограниченным), чем режим ядра. Драйверы, установленные в или иным образом включенные в операционную систему 112, чтобы способствовать выполнению связи с виртуальными компонентами 116, могут выполняться в пользовательском режиме или в режиме ядра. Использование режима ядра и пользовательского режима обеспечивает дополнительную защиту для кода, который выполняется в режиме ядра, например, процессором, исполняющим код, предотвращающий обращение кода, выполняющегося в пользовательском режиме, к памяти, используемой кодом, выполняющимся в режиме ядра.

[0029] Хотя упоминаемые в данном документе как режим ядра и пользовательский режим, альтернативно, один или несколько дополнительных режимов могут поддерживаться процессорами вычислительного устройства 100. В таких ситуациях, режимы, описанные в данном документе, могут упоминаться как режим ядра и режим не ядра, причем режимы, кроме режима ядра, рассматриваются аналогично пользовательскому режиму, описанному в данном документе. Альтернативно, режимы, описанные в данном документе, могут упоминаться как пользовательский режим и непользовательский режим, причем режимы, кроме пользовательского режима,

рассматриваются аналогично режиму ядра, описанному в данном документе.

[0030] Фиг. 2 иллюстрирует примерную систему 200, реализующую методы селективного обеспечения соблюдения целостности кода, обеспечиваемого менеджером виртуальной машины, описанные в данном документе. Система 200 включает в себя операционную систему 112, выполняющуюся в виртуальной машине 106, и модуль 202 верификации целостности кода, который является частью более привилегированного объекта 204. Модуль 202 верификации целостности кода может, необязательно, быть включен в модуль 124 управления страницами, выполняющийся в менеджере 102 виртуальной машины по фиг. 1. Модуль 202 верификации целостности кода выполняет верификацию целостности кода для кода, который виртуальный процессор собирается исполнить, когда виртуальный процессор работает в режиме ядра, обеспечивая результаты 206 верификации кода в режиме ядра, указывающие, завершается ли верификация целостности кода для кода успешно или неуспешно. Если верификация целостности кода завершается успешно, тогда код может исполняться в режиме ядра, и, если верификация целостности кода завершается неуспешно, тогда код не может исполняться с режиме ядра.

[0031] Более привилегированный объект 204 ссылается на объект, который является более привилегированным (менее ограниченным), чем операционная система 112. Более привилегированным объектом 204 может быть менеджер 102 виртуальной машины по фиг. 1. Более привилегированным объектом 204, альтернативно, может быть один или несколько других объектов, таких как виртуальный процессор, работающий в защищенном режиме, который имеет более высокую привилегию, чем режим ядра операционной системы 112 (например, защищенный режим, управляемый менеджером 102 виртуальной машины). Более привилегированный объект может быть реализован в виде программного обеспечения, аппаратно-программного обеспечения и/или аппаратного обеспечения. В ситуациях, в которых более привилегированным объектом 204 является объект, кроме менеджера виртуальной машины, результаты 206 верификации кода в режиме ядра могут предоставляться менеджеру виртуальной машины, позволяющие менеджеру виртуальной машины сохранять запись, идентифицирующую код, целостность которого была верифицирована, и, таким образом, может исполняться в режиме ядра.

[0032] Операционная система 112 включает в себя модуль 208 обеспечения соблюдения политики, который реализует одну или несколько политик для определения, может ли исполняться код, который виртуальный процессор собирается исполнить, когда виртуальный процессор работает в пользовательском режиме. Всевозможные разные политики могут быть реализованы модулем 208 обеспечения соблюдения политики, такие как выполнение верификации целостности кода, аналогично модулю 202 верификации целостности кода, проверка других характеристик кода или операционной системы 112 и т.п. Политикой, реализуемой модулем 208 обеспечения соблюдения политики, также может быть исполнение всего кода в пользовательском режиме (например, исполнение кода в пользовательском режиме без выполнения каких-либо проверок целостности кода или других верификаций). Модуль 208 обеспечения соблюдения политики применяет политику для кода, когда виртуальный процессор работает в пользовательском режиме, обеспечивая результаты 210 оценки политики, указывающие, выполняется ли политика (проверка политики завершается успешно) или не выполняется (проверка политики завершается неуспешно). Если проверка политики завершается успешно, тогда код может исполняться в пользовательском режиме, и, если проверка политики завершается неуспешно, тогда код не может

исполняться в пользовательском режиме.

0033] Таким образом, более привилегированный объект 204 выполняет верификацию целостности кода (обеспечивает соблюдение целостности кода) для страниц в режиме ядра, тогда как операционная система 112 в виртуальной машине 106 выполняет проверки политики (обеспечивает соблюдение политики) для страниц в пользовательском режиме. В одном или нескольких вариантах осуществления модуль 208 обеспечения соблюдения политики реализуется в коде в режиме ядра операционной системы 112. Таким образом, проверки политики для кода, выполняющегося в пользовательском режиме, выполняются кодом, выполняющимся в режиме ядра, причем целостность кода для кода, выполняющегося в режиме ядра, верифицировалась менеджером 102 виртуальной машины или другим более привилегированным объектом.

[0034] Таким образом, менеджер 102 виртуальной машины способен ограничить исполнение кода в режиме ядра до кода, который был верифицирован более привилегированным объектом 204, но исполнение кода в пользовательском режиме управляется отдельно операционной системой 112, выполняющейся на виртуальной машине 106. Операционная система 112 выполняет любую верификацию кода и/или другие проверки политики, основываясь на конфигурации модуля 208 обеспечения соблюдения политики и независимо от верификации целостности кода, выполняемой более привилегированным объектом 204.

[0035] Методы, описанные в данном документе, таким образом, обеспечивают дополнительный уровень безопасности вследствие менеджера виртуальной машины, предотвращающего исполнение скомпрометированного кода операционной системы 112 в режиме ядра. Одновременно, однако, методы, описанные в данном документе, позволяют верифицировать целостность кода для кода, исполняющегося в пользовательском режиме, и/или других политик, реализованных операционной системой 112 при необходимости. Операционная система 112 устанавливает политики для кода, исполняющегося в пользовательском режиме, позволяя операционной системе 112 поддерживать ситуации, где не выполняется верификация кода. Например, операционной системе 112 может потребоваться предоставить возможность исполнения некоторого кода в пользовательском режиме без верификации, или предоставить возможность выполнения динамического генерирования кода для некоторого кода, исполняющегося в пользовательском режиме. Такие определения могут быть сделаны операционной системой 112, при этом все это время пользователь вычислительного устройства 100 уверен, что операционная система 112 не была скомпрометирована, так как целостность кода операционной системы 112 верифицировалась менеджером 102 виртуальной машины (или другим более привилегированным объектом).

[0036] Фиг. 3 представляет собой блок-схему последовательности операций, иллюстрирующую примерный процесс 300 для реализации селективного обеспечения соблюдения целостности кода, обеспечиваемого менеджером виртуальной машины, согласно одному или нескольким вариантам осуществления. Процесс 300 выполняется, по меньшей мере частично, менеджером виртуальной машины, таким как менеджер 102 виртуальной машины по фиг. 1 и 2, и может быть реализован программным обеспечением, аппаратно-программным обеспечением, аппаратным обеспечением или их комбинацией. Процесс 300 показан как набор действий и не ограничивается порядком, показанным для выполнения операций различных действий. Процесс 300 представляет собой примерный процесс для реализации селективного обеспечения соблюдения целостности кода, обеспечиваемого менеджером виртуальной машины; дополнительные описания реализации селективного обеспечения соблюдения целостности кода,

обеспечиваемого менеджером виртуальной машины, включены в данный документ с ссылкой на разные фигуры.

[0037] В процессе 300 идентифицируется страница памяти, включающая в себя исполняемый код, подлежащий исполнению виртуальным процессором (действие 302). Страница памяти может идентифицироваться в разные моменты времени и в ответ на разные события, такие как запрос виртуальной машиной или операционной системой, управляемой виртуальной машиной, чтобы сделать страницу памяти исполняемой, запрос виртуальной машиной или операционной системой, управляемой виртуальной машиной, чтобы предоставить возможность исполнения программы виртуальным процессором, и т.п.

[0038] Определение выполняется в отношение того, должна ли страница памяти быть исполняемой в режиме ядра (действие 304). Это определение может выполняться различным образом, таким как идентифицирование виртуальной машиной или операционной системой при выполнении запроса на то, чтобы сделать страницу памяти исполняемой.

[0039] В ответ на определение, что страница памяти должна быть исполняемой в режиме ядра, выполняется проверка целостности кода исполняемого кода более привилегированным объектом (действие 306). Более привилегированным объектом является объект, более привилегированный, чем операционная система, управляемая виртуальной машиной, как описано выше. Более привилегированным объектом может быть менеджер виртуальной машины, или другой объект, как описано выше. Проверка целостности кода выполняется посредством верификации целостности кода, основываясь на политике целостности кода, различным образом, как описано выше, например, посредством верификации кода, основываясь на верифицируемой цепочке сертификатов и цифровой подписи, верифицирующей, что код не был модифицирован. Предоставляется возможность исполнения исполняемого кода в странице памяти, если только проверка целостности кода верифицирует исполняемый код (действие 308). Может предоставляться возможность исполнения исполняемого кода в странице памяти, например, менеджером виртуальной машины, выдающим разрешение исполнения странице памяти, идентифицированной в действии 302.

[0040] Возвращаясь к действию 304, в ответ на определение, что страница памяти не должна быть исполняемой в режиме ядра, операционной системе виртуальной машины, управляемой менеджером виртуальной машины, предоставляется возможность определения, предоставить ли возможность исполнения исполняемого кода на основе политики операционной системы (действие 310). Операционная система может реализовать всевозможные разные политики для определения, исполнять ли код, включая проверку целостности кода, как описано выше.

[0041] Запрос на то, чтобы сделать страницу памяти исполняемой, в действии 302, может направляться менеджеру виртуальной машины, или, альтернативно, более привилегированному объекту (например, более привилегированному объекту 204 на фиг. 2). В одном или нескольких вариантах осуществления, если менеджер виртуальной машины является более привилегированным объектом, тогда запрос направляется менеджеру виртуальной машины, и менеджер виртуальной машины выполняет проверку целостности кода в действии 306. Однако если другой объект является более привилегированным объектом, тогда запрос направляется более привилегированному объекту (например, непосредственно или посредством менеджера виртуальной машины), более привилегированный объект выполняет проверку целостности кода в действии 306, и, если проверка целостности кода в действии 306 верифицирует исполняемый код,

тогда более привилегированный объект уведомляет менеджера виртуальной машины сделать страницу памяти исполняемой в режиме ядра. Менеджер виртуальной машины может сделать страницу памяти исполняемой в режиме ядра, например, посредством обновления таблицы преобразования адресов второго уровня, как описано более подробно ниже.

[0042] Таким образом, менеджер виртуальной машины способствует селективному обеспечению соблюдения целостности кода. Соблюдение целостности кода для кода в режиме ядра обеспечивается менеджером виртуальной машины (или другим более привилегированным объектом, который по-новому применяет менеджер виртуальной машины), тогда как обеспечение соблюдения целостности кода для кода в пользовательском режиме остается для операционной системы, выполняющейся в виртуальной машине, управляемой менеджером виртуальной машины.

[0043] Возвращаясь к фиг. 1, в одном или нескольких вариантах осуществления вычислительное устройство 100 применяет виртуальную память. Виртуальная память ссылается на адресное пространство, которое отображается на другое адресное пространство (например, физическую память). Приложению назначается пространство виртуальной памяти, в котором исполняется код приложения, и хранятся данные. Менеджер памяти (например, процессор) управляет отображением адресов виртуальной памяти в пространство виртуальной памяти для адресования в пространстве другой памяти. При отображении адресов виртуальной памяти из адресного пространства виртуальной памяти в пространство другой памяти, выполняется преобразование адресов. Таблица преобразования адресов используется для выполнения этого отображения и может применяться по-новому для реализации методов, описанных в данном документе.

[0044] В одном или нескольких вариантах осуществления таблица преобразования адресов реализуется аппаратным обеспечением, таким как физический процессор, т.е. компонент 104 вычислительного устройства 100. Таблица преобразования адресов предоставляет возможность менеджеру 102 виртуальной машины селективно обеспечивать соблюдение целостности кода, как описано более подробно ниже. Альтернативно, аппаратное обеспечение вычислительного устройства 100 (например, физический процессор, т.е. компонент 104) может использовать различные другие таблицы, списки, записи, структуры и т.п., чтобы предоставить возможность менеджеру 102 виртуальной машины селективно обеспечивать соблюдение целостности кода. Альтернативно, различные другие таблицы, списки, записи, структуры и т.п., реализованные программным обеспечением (например, как часть менеджера 102 виртуальной машины, или как часть другого компонента или модуля вычислительного устройства 100) могут использоваться для того, чтобы предоставить возможность менеджеру 102 виртуальной машины селективно обеспечивать соблюдение целостности кода.

[0045] Фиг. 4 иллюстрирует примерную систему 400, реализующую селективное обеспечение соблюдения целостности кода, обеспечиваемое менеджером виртуальной машины, согласно одному или нескольким вариантам осуществления. Системой 400, например, может быть вычислительное устройство 100 по фиг. 1. Система 400 включает в себя физический процессор 402, пространство 404 физической памяти, виртуальный процессор 406 и программу 408. Физическим процессором 402 может быть компонент 104 на фиг. 1, пространством 404 физической памяти может быть компонент 104 на фиг. 1, виртуальным процессором 406 может быть виртуальный компонент 116 на фиг. 1, и программой 408 может быть приложение 114 или часть операционной системы 112

на фиг. 1. Физический процессор 402 включает в себя менеджер 410 памяти, который управляет доступом к пространству 404 физической памяти. Пространством 404 физической памяти могут быть различные энергозависимые и/или энергонезависимые памяти, такие как RAM, флэш-память и т.п.

[0046] Физический процессор 402 назначает пространство 412 памяти виртуальной машины виртуальному процессору 406 и поддерживает таблицу 414 преобразования адресов второго уровня. Таблица 414 преобразования адресов второго уровня отображает адреса в пространстве 412 памяти виртуальной машины в адреса в пространстве 404 физической памяти. На какой адрес пространства 404 физической памяти отображается конкретный адрес в пространстве 412 памяти виртуальной машины в любой данный момент времени может меняться и управляется менеджером 410 памяти. Менеджер 410 памяти может изменять отображения, предоставляя возможность многочисленным разным виртуальным процессорам совместно использовать пространство 404 физической памяти и/или предоставляя возможность пространству 412 памяти виртуальной машины быть больше пространства 404 физической памяти, используя любой из различных общедоступных и/или проприетарных методов.

[0047] Виртуальный процессор 406 включает в себя менеджер 416 памяти, который управляет доступом к пространству 412 памяти виртуальной машины. Виртуальный процессор 406 назначает пространство 418 памяти программ для программы 408 и поддерживает таблицу 420 преобразования адресов первого уровня. Таблица 420 преобразования адресов первого уровня отображает адреса в пространстве 418 памяти программ в адреса в пространстве 412 памяти виртуальной машины. На какой адрес пространства 412 памяти виртуальной машины отображается конкретный адрес в пространстве 418 памяти программ в любой данный момент времени может меняться и управляется менеджером 416 памяти. Менеджер 416 памяти может менять отображения, предоставляя возможность многочисленным разным программам совместно использовать пространство 412 памяти виртуальной машины и/или предоставляя возможность пространству 418 памяти программ быть больше пространства 412 памяти виртуальной машины, используя любой из различных общедоступных и/или проприетарных методов.

[0048] В ответ на доступ к адресу в пространстве 418 памяти программ, менеджер 416 памяти использует таблицу 420 преобразования адресов первого уровня для преобразования адреса памяти в пространстве 418 памяти программ в адрес в пространстве 412 памяти виртуальной машины. Доступ может принимать всевозможные разные формы, такие как адрес кода программы 408, подлежащего исполнению операционной системой, адрес, где данные, подлежащие считыванию, сохраняются программой 408, адрес, где данные, подлежащие записи программой 408, должны сохраняться, и т.п.

[0049] Аналогично, в ответ на доступ к адресу в пространстве 412 памяти виртуальной машины, менеджер 410 памяти использует таблицу 414 преобразования адресов второго уровня для преобразования адреса памяти в пространстве 412 памяти виртуальной машины в адрес в пространстве 404 физической памяти. Доступ может принимать всевозможные разные формы, такие как адрес кода центральной части операционной системы (например, ядра), управляемой виртуальным процессором 406, подлежащей выполнению, адрес программы 408, подлежащий исполнению операционной системой, адрес, где данные, подлежащие считыванию, сохраняются программой 408 или центральной частью операционной системы, адрес, где данные, подлежащие записи программой 408 или центральной частью операционной системы, должны сохраняться,

и т.п.

[0050] В одном или нескольких вариантах осуществления таблица 420 преобразования адресов первого уровня и таблица 414 преобразования адресов второго уровня отображают страницы адресов, а не индивидуальные адреса. Например, таблица 420 преобразования адресов первого уровня отображает страницы пространства 418 памяти программ на страницы пространства 412 памяти виртуальной машины, и таблица 414 преобразования адресов второго уровня отображает страницы пространства 412 памяти виртуальной машины на страницы пространства 404 физической памяти. Альтернативно, таблица 414 и/или таблица 420 могут отображать адреса на основе других группирований, таких как индивидуально или другие коллекции субстраниц адресов, в коллекциях многочисленных страниц и т.п.

[0051] Виртуальный процессор 406 может выполняться в режиме ядра или пользовательском режиме. Менеджер 416 памяти может поддерживать запись страниц памяти, которые могут исполняться в режиме ядра. Эта запись может быть включена как часть таблицы 420 преобразования адресов первого уровня или, альтернативно, может поддерживаться другим образом. Менеджер 410 памяти поддерживает запись страниц памяти в пространстве 412 памяти виртуальной машины, которые могут исполняться в режиме ядра. Эта запись может быть включена как часть таблицы 414 преобразования адресов второго уровня или, альтернативно, может поддерживаться другим образом.

[0052] Когда доступ выполняется для исполнения кода на виртуальном процессоре 406, адрес кода, подлежащего исполнению, преобразуется таблицей 420 преобразования адресов первого уровня для получения преобразованного адреса в пространстве 412 памяти виртуальной машины, и преобразованный адрес затем преобразуется таблицей 414 преобразования адресов второго уровня для получения адреса в пространстве 404 физической памяти. Если доступ выполняется, когда виртуальный процессор 406 работает в пользовательском режиме, тогда операционная система, выполняющаяся на виртуальном процессоре 406, применяет соответствующую политику при определении, может ли код исполняться.

[0053] Однако если доступ выполняется, когда виртуальный процессор работает в режиме ядра, тогда выполняется проверка в отношении того, было ли странице предоставлено разрешение исполнения для режима ядра. Запись, было ли уже предоставлено странице разрешение исполнения для режима ядра, может, необязательно, поддерживаться в таблице 414 преобразования адресов второго уровня, или, альтернативно, где-то в другом месте физическим процессором 402. В таких ситуациях, если разрешение исполнения для режима ядра уже было предоставлено странице, тогда указание, что разрешение исполнения для режима ядра было предоставлено странице, может быть возвращено виртуальному процессору 406, и операционная система, выполняющаяся на виртуальном процессоре 406, предоставляет возможность исполнения кода, когда виртуальный процессор 406 находится в режиме ядра.

[0054] Если разрешение исполнения для режима ядра еще не было предоставлено странице, тогда операционная система, выполняющаяся на виртуальном процессоре 406, может, необязательно, запросить, чтобы страница была сделана исполняемой для режима ядра. Виртуальный процессор 406 передает запрос менеджеру виртуальной машины (или другому более привилегированному объекту), который выполняет проверку целостности кода, как описано выше. Если проверка целостности кода завершается успешно, тогда странице предоставляется разрешение исполнения для режима ядра, и, если проверка целостности кода завершается неуспешно, тогда странице

не предоставляется разрешение исполнения для режима ядра.

[0055] В одном или нескольких вариантах осуществления атрибут включается в таблицу 414 преобразования адресов второго уровня, чтобы предоставить возможность задания разрешения исполнения для режима ядра отдельно от разрешения исполнения или политики для пользовательского режима. Этот атрибут может кодироваться в таблице 414 преобразования адресов второго уровня всевозможными разными способами, но включает в себя следующие три характеристики. Во-первых, атрибут реализуется в таблице преобразования адресов второго уровня, так что атрибут может управляться независимо для каждого преобразования второго уровня (каждого преобразования, выполняемого менеджером 410 памяти, используя таблицу 414 преобразования адресов второго уровня). Во-вторых, атрибут применяется к разрешению исполнения в режиме ядра – атрибут не должен оказывать влияние (но, альтернативно, может оказывать влияние) на разрешение считывания и/или записи в режиме ядра (и в пользовательском режиме). В-третьих, атрибут предоставляет возможность запрещения исполнения в режиме ядра независимо от исполнения в пользовательском режиме, разрешая по меньшей мере следующие две комбинации исполнения: 1) непредоставление возможности исполнения в режиме ядра, но предоставление возможности исполнения в пользовательском режиме; 2) непредоставление возможности исполнения в режиме ядра и непредоставление возможности исполнения в пользовательском режиме.

[0056] По меньшей мере один атрибут может быть включен в таблицу 414 преобразования адресов второго уровня всевозможными разными способами. Например, каждый элемент в таблице преобразования адресов второго уровня может включать в себя два бита, один бит соответствует исполнению в пользовательском режиме, и один бит соответствует исполнению в режиме ядра. Биту, соответствующему страницам пользовательского режима, может назначаться одно значение (например, назначаться значение «1», также упоминаемое как «бит установлен») для указания, что предоставляется разрешение для исполнения кода на странице в пользовательском режиме, и может назначаться другое значение (например, назначаться значение «0», также упоминаемое как «бит сброшен») для указания, что не предоставляется разрешение для исполнения кода на странице в пользовательском режиме. Аналогично, биту, соответствующему исполнению в режиме ядра, может назначаться одно значение (например, назначаться значение «1», также упоминаемое как «бит установлен») для указания, что предоставляется разрешение для исполнения кода на странице в режиме ядра, и может назначаться другое значение (например, назначаться значение «0», также упоминаемое как «бит сброшен») для указания, что не предоставляется разрешение для исполнения кода на странице в режиме ядра.

[0057] В одном или нескольких вариантах осуществления в таблице 414 преобразования адресов второго уровня начальное значение или значение по умолчанию для страниц памяти предназначено для непредоставления возможности исполнения (разрешение исполнения не предоставляется) для режима ядра, но предоставления возможности исполнения (разрешение исполнения предоставляется) для пользовательского режима. Если целостность кода для кода в странице памяти была верифицирована менеджером виртуальной машины (или другим более привилегированным объектом), тогда значения для страницы памяти могут быть изменены на предоставление возможности исполнения (разрешение исполнения предоставляется) для режима ядра для страницы, а также для пользовательского режима. Однако если целостность кода для кода в странице памяти неуспешное в результате

проверки целостности кода операционной системой, операционная система может, необязательно, изменить (или запросить, чтобы менеджер виртуальной машины изменил) значения для страницы памяти на непредоставление возможности исполнения (разрешение исполнения не предоставляется) для пользовательского режима для страницы.

[0058] Следует отметить, что таблица 414 преобразования адресов второго уровня находится под контролем менеджера виртуальной машины, такого как менеджер 102 виртуальной машины на фиг. 1. Таблица 414 преобразования адресов второго уровня не является доступной для программного обеспечения, выполняющегося в виртуальной машине (например, приложений 114 или операционной системы 112 виртуальной машины 106 на фиг. 1). Операционная система в виртуальной машине имеет непосредственный контроль над своей таблицей преобразования адресов первого уровня, и менеджер виртуальной машины имеет исключительный контроль над таблицей преобразования адресов второго уровня. Таким образом, имея контроль над исполнением в режиме ядра в таблице преобразования адресов второго уровня, менеджер виртуальной машины может эффективно и легко использовать этот контроль для предоставления или отказа в доступе к исполнению в режиме ядра независимо от управления операционной системой таблицей преобразования адресов первого уровня. Имея возможность ограничения исполнения кода в режиме ядра в таблице преобразования адресов второго уровня, менеджер виртуальной машины может обеспечивать соблюдение своих политик целостности кода без непосредственного вовлечения себя в обновления таблицы преобразования адресов первого уровня, выполняемые операционной системой в виртуальной машине.

[0059] Также следует отметить, что, хотя выше описаны таблицы преобразования адресов первого и второго уровня, альтернативно, целостность кода, обеспечиваемая менеджером виртуальной машины, может быть реализована программным обеспечением, используя единственную таблицу преобразования адресов (например, таблицу преобразования адресов первого уровня). Например, менеджер виртуальной машины может получить контроль над таблицей преобразования адресов первого уровня (например, таблицей 420 на фиг. 4) и предотвратить непосредственный доступ операционной системой из виртуальной машины к таблице преобразования адресов первого уровня. Затем менеджер виртуальной машины может обеспечить соблюдение политик, описанных выше (например, менеджер виртуальной машины может гарантировать, что код, исполняемый ядром, представляет собой код, который верифицировал менеджер виртуальной машины, но менеджер виртуальной машины не обеспечивает соблюдение ограничений на исполнение кода в пользовательском режиме).

[0060] Хотя конкретная функциональная возможность описывается в данном документе с ссылкой на конкретные модули, следует отметить, что функциональная возможность индивидуальных модулей, описанных в данном документе, может быть разделена на многочисленные модули, и/или по меньшей мере некоторые функциональные возможности многочисленных модулей могут быть объединены в единственный модуль. Кроме того, конкретный модуль, описанный в данном документе как выполняющий действие, включает в себя сам этот конкретный модуль, выполняющий действие, или, альтернативно, этот конкретный модуль, вызывающий или иным образом обращающийся к другому компоненту или модулю, который выполняет действие (или выполняет действие совместно с этим конкретным модулем). Таким образом, конкретный модуль, выполняющий действие, включает в себя сам этот конкретный модуль, выполняющий действие, и/или другой модуль, который вызывается

или к которому иным образом выполняется обращение этим конкретным модулем, выполняющим действие.

[0061] Фиг. 5 иллюстрирует примерную систему, обозначенную в целом позицией 500, которая включает в себя примерное вычислительное устройство 502, которое представляет одну или несколько систем и/или устройств, которые могут реализовать различные методы, описанные в данном документе. Вычислительным устройством 502, например, может быть сервер провайдера услуг, устройство, ассоциированное с клиентом (например, клиентское устройство), система на кристалле и/или любое другое подходящее вычислительное устройство или вычислительная система.

[0062] Примерное вычислительное устройство 502, как изображено, включает в себя систему 504 обработки, один или несколько считываемых компьютером носителей 506 и один или несколько интерфейсов 508 ввода/вывода (I/O), которые соединены, с возможностью связи, друг с другом. Хотя это не показано, вычислительное устройство 502 дополнительно может включать в себя системную шину или другую систему передачи данных и команд, которая соединяет различные компоненты друг с другом. Системная шина может включать в себя любую одну или комбинацию разных шинных структур, таких как шина памяти или контролер памяти, периферийная шина, универсальная последовательная шина и/или процессорная или локальная шина, которая использует любую из множества шинных архитектур. Также рассматривается множество других примеров, таких как линии управления и передачи данных.

[0063] Система 504 обработки представляет функциональную возможность для выполнения одной или нескольких операций, использующих аппаратное обеспечение. Следовательно, система 504 обработки изображена как включающая в себя аппаратные элементы 510, которые могут быть выполнены в виде процессоров, функциональных блоков и т.п. Она может включать в себя реализацию аппаратными средствами в виде специализированной интегральной микросхемы или другого логического устройства, образованного с использованием одного или нескольких полупроводниковых приборов. Аппаратные элементы 510 не ограничиваются материалами, из которых они образованы, или механизмами обработки, примененными в них. Например, процессоры могут состоять из полупроводникового прибора(-ов) и/или транзисторов (например, электронных интегральных схем (IC)). В таком контексте, исполняемые процессором инструкции могут быть инструкциями, исполняемыми электронным способом.

[0064] Считываемые компьютером носители 506 изображены как включающие в себя память/запоминающее устройство 512. Память/запоминающее устройство 512 представляет возможности памяти/запоминающего устройства, ассоциированные с одним или несколькими считываемыми компьютером носителями. Память/запоминающее устройство 512 может включать в себя энергозависимые носители (такие как оперативное запоминающее устройство (RAM)) и/или энергонезависимые носители (такие как постоянное запоминающее устройство (ROM)), флэш-память, оптические диски, магнитные диски и т.п.). Память/запоминающее устройство 512 может включать в себя несъемные носители (например, RAM, ROM несъемный жесткий диск и т.п.), а также съемные носители (например, флэш-память, съемный жесткий диск, оптический диск и т.п.). Считываемые компьютером носители 506 могут быть выполнены всевозможными разными способами, как дополнительно описано ниже.

[0065] Интерфейс(-ы) 508 ввода/вывода представляют функциональную возможность, позволяющую пользователю вводить команды и информацию в вычислительное устройство 502, и также позволяющую представлять информацию пользователю и/или другим компонентам или устройствам, используя различные устройства ввода/вывода.

Примеры устройств ввода включают в себя клавиатуру, устройство управления курсором (например, мышь), микрофон (например, для ввода голоса), сканер, функциональную возможность касания (например, емкостные и другие датчики, которые выполнены с возможностью обнаружения физического касания), камеру (например, которая может применять видимые или невидимые длины волн, такие как инфракрасные частоты, для обнаружения движения, которое не включает в себя касание в виде жестов) и т.п. Примеры устройств вывода включают в себя устройство отображения (например, монитор или проектор), громкоговоритель, принтер, сетевую карту, тактильное устройство и т.п. Таким образом, вычислительное устройство 502 может быть выполнено разнообразными способами, как дополнительно описано ниже, для поддержки взаимодействия с пользователем.

[0066] Вычислительное устройство 502 также включает в себя менеджер 514 виртуальной машины (также упоминаемый как гипервизор). Менеджер 514 виртуальной машины позволяет виртуальной машине выполняться на вычислительном устройстве 502. Менеджером 514 виртуальной машины, например, может быть менеджер 102 виртуальной машины на фиг. 1 или фиг. 2.

[0067] Различные методы могут описываться в данном документе в общем контексте программного обеспечения, аппаратных элементов или программных модулей. Как правило, такие модули включают в себя подпрограммы, программы, объекты, элементы, компоненты, структуры данных и т.п., которые выполняют конкретные задачи или реализуют конкретные абстрактные типы данных. Термины «модуль», «функциональная возможность» и «компонент», как они используются в данном документе, представляют, в основном, программное обеспечение, аппаратно-программное обеспечение, аппаратное обеспечение или их комбинацию. Признаки методов, описанных в данном документе, являются платформно-независимыми, означая, что методы могут быть реализованы на различных вычислительных платформах, имеющих множество процессоров.

[0068] Реализация описанных модулей и методов может сохраняться или передаваться по считываемым компьютерам носителям некоторого вида. Считываемые компьютером носители могут включать в себя разнообразные носители, к которым может обращаться вычислительное устройство 502. В качестве примера, а не ограничения, считываемые компьютером носители могут включать в себя «считываемые компьютером носители информации» и «считываемые компьютером носители сигнала».

[0069] «Считываемые компьютером носители информации» ссылаются на носители и/или устройства, которые делают возможным долговременное запоминающее устройство для информации и/или запоминающее устройство, которое является материальным, в противоположность простой передаче сигнала, несущим волнам или сигналам как таковым. Таким образом, считываемые компьютером носители информации ссылаются на не несущие сигнал носители. Считываемые компьютером носители информации включают в себя аппаратное обеспечение, такое как энергозависимые и энергонезависимые, съемные и несъемные носители и/или запоминающие устройства, реализованные по способу или по технологии, подходящей для хранения информации, такой как считываемые компьютером инструкции, структуры данных, модули программ, логические элементы/схемы или другие данные. Примеры считываемых компьютером носителей информации могут включать в себя, но не ограничиваются ими, RAM, ROM, электрически стираемое программируемое постоянное запоминающее устройство (EEPROM), флэш-память или другую технологию памяти, компакт-диск (CD-ROM), цифровые многофункциональные диски (DVD) или другие

оптические запоминающие устройства, жесткие диски, магнитные кассеты, магнитную ленту, запоминающее устройство на магнитных дисках или другие магнитные запоминающие устройства, или другие запоминающие устройства, материальные носители или изделие, пригодное для хранения требуемой информации, и к которому может обращаться компьютер.

[0070] «Считываемые компьютером носители сигнала» ссылаются на несущий сигнал носитель, который выполнен с возможностью передачи инструкций на аппаратное обеспечение вычислительного устройства 502, например, посредством сети. Носители сигнала обычно могут воплощать считываемые компьютером инструкции, структуры данных, модули программ или другие данные в модулированном данными сигнале, таком как несущие волны, сигналы данных или другой транспортный механизм. Носители сигнала также включают в себя любые носители для доставки информации. Термин «модулированный данными сигнал» означает сигнал, который имеет одну или несколько его характеристик, устанавливаемых или изменяемых таким образом, чтобы кодировать информацию в сигнале. В качестве примера, и не ограничения, носители связи включают в себя проводные носители, такие как проводная сеть или непосредственное проводное соединение, и беспроводные носители, такие как акустические, радиочастотные (RF), инфракрасные и другие беспроводные носители.

[0071] Как описано выше, аппаратные элементы 510 и считываемые компьютером носители 506 представляют инструкции, модули, логику программируемых устройств и/или логику стационарных устройств, реализованную в аппаратном виде, которые могут применяться в некоторых вариантах осуществления для реализации по меньшей мере некоторых аспектов методов, описанных в данном документе. Аппаратные элементы могут включать в себя компоненты интегральной схемы или системы на кристалле, специализированной интегральной микросхемы (ASIC), программируемой вентильной матрицы (FPGA), сложного программируемого логического устройства (CPLD) и другие реализации в кремнии или другие аппаратные устройства. В данном контексте, аппаратный элемент может работать как устройство обработки, которое выполняет задачи программы, определяемые инструкциями, модулями и/или логикой, воплощенной аппаратным элементом, а также аппаратное устройство, используемое для хранения инструкций для исполнения, например, считываемых компьютером носителей информации, описанных ранее.

[0072] Комбинации вышеупомянутых также могут применяться для реализации различных методов и модулей, описанных в данном документе. Следовательно, программное обеспечение, аппаратное обеспечение или модули программы и другие модули программы могут быть реализованы в виде одной или нескольких инструкций и/или логики, воплощенной на считываемом компьютером носителях информации некоторого вида и/или одним или несколькими аппаратными элементами 510. Вычислительное устройство 502 может быть выполнено с возможностью реализации конкретных инструкций и/или функций, соответствующих программным и/или аппаратным модулям. Следовательно, реализация модулей в качестве модуля, который является исполняемым вычислительным устройством 502 в качестве программного обеспечения, может достигаться, по меньшей мере частично, аппаратным обеспечением, например, посредством использования считываемых компьютером носителей информации и/или аппаратных элементов 510 системы обработки. Инструкции и/или функции могут быть исполняемыми/осуществимыми одним или несколькими изделиями (например, одним или несколькими вычислительными устройствами 502 и/или системами 504 обработки) для реализации методов, модулей и примеров, описанных в данном

документе.

[0073] Как дополнительно изображено на фиг. 5, примерная система 500 делает возможными повсеместные окружения для «бесшовного» опыта взаимодействия пользователя при выполнении приложений на персональном компьютере (PC), телевизионном устройстве и/или мобильном устройстве. Услуги и приложения выполняются, по существу, аналогично во всех трех окружениях для общего опыта взаимодействия пользователя при переходе с одного устройства на другое при использовании приложения, воспроизведении видеоигры, просмотре видео и т.п.

[0074] В примерной системе 500, многочисленные устройства соединены между собой посредством центрального вычислительного устройства. Центральное вычислительное устройство может быть локальным для многочисленных устройств или может располагаться удаленным от многочисленных устройств. В одном или нескольких вариантах осуществления центральное вычислительное устройство может представлять собой облако из одного или нескольких серверных компьютеров, которые соединены с многочисленными устройствами посредством сети, Интернета или другой линии передачи данных.

[0075] В одном или нескольких вариантах осуществления эта архитектура соединений между собой позволяет выполнять доставку функциональной возможности на многочисленные устройства для обеспечения общего и «бесшовного» опыта взаимодействия для пользователя многочисленных устройств. Каждое из многочисленных устройств может иметь разные физические требования и возможности, и центральное вычислительное устройство использует платформу, чтобы сделать возможной доставку опыта взаимодействия на устройство, который подстраивается под устройство и, тем не менее, является общим для всех устройств. В одном или нескольких вариантах осуществления создается класс целевых устройств, и опыт взаимодействия подстраивается под общий класс устройств. Класс устройств может определяться физическими особенностями, типами использования или другими общими характеристиками устройств.

[0076] В различных реализациях вычислительное устройство 502 может принимать всевозможные разные конфигурации, такие как, для использования компьютера 516, мобильного устройства 518 и телевизора 520. Каждая из этих конфигураций включает в себя устройства, которые могут иметь, как правило, разные конструкции и возможности, и, таким образом, вычислительное устройство 502 может быть выполнено в соответствии с одним или несколькими разными классами устройств. Например, вычислительное устройство 502 может быть реализовано в виде компьютерного 516 класса устройства, который включает в себя персональный компьютер, настольный компьютер, многоэкранный компьютер, портативный компьютер, нетбук и т.п.

[0077] Вычислительное устройство 502 также может быть реализовано в виде мобильного 518 класса устройства, который включает в себя мобильные устройства, такие как мобильный телефон, портативный музыкальный проигрыватель, портативное игровое устройство, планшетный компьютер, многоэкранный компьютер и т.п. Вычислительное устройство 502 также может быть реализовано в виде телевизионного 520 класса устройства, который включает в себя устройства, имеющие или соединенные с, в основном, большими экранами в окружениях нерегулярного просмотра. Эти устройства включают в себя телевизоры, телевизионные приставки, игровые консоли и т.п.

[0078] Методы, описанные в данном документе, могут поддерживаться этими различными конфигурациями вычислительного устройства 502 и не ограничиваются

конкретными примерами методов, описанных в данном документе. Эта функциональная возможность также может быть реализована вся или частично посредством использования распределенной системы, такой как по «облаку» 522 посредством платформы 524, как описано ниже.

[0079] Облако 522 включает в себя и/или представляет платформу 524 для ресурсов 526. Платформа 524 абстрагирует лежащую в основе функциональную возможность аппаратных (например, серверов) и программных ресурсов облака 522. Ресурсы 526 могут включать в себя приложения и/или данные, которые могут использоваться, когда компьютерная обработка исполняется на серверах, которые являются удаленными от вычислительного устройства 502. Ресурсы 526 также могут включать в себя услуги, предоставляемые по Интернету и/или при помощи абонентской сети, такой как сотовая сеть или сеть WiFi.

[0080] Платформа 524 может абстрагировать ресурсы и функции для соединения вычислительного устройства 502 с другими вычислительными устройствами. Платформа 524 также может служить для абстрагирования масштабирования ресурсов для обеспечения соответствующего уровня масштаба встречаемых потребностей ресурсов 526, которые реализуются посредством платформы 524. Следовательно, в варианте осуществления с соединенными между собой устройствами, реализация функциональности, описанной в данном документе, может распределяться по системе 500. Например, функциональная возможность может быть реализована, частично, на вычислительном устройстве 502, а также посредством платформы 524, которая абстрагирует функциональную возможность облака 522.

[0081] Хотя объект изобретения был описан на языке, характерном для конструктивных особенностей и/или методологических действий, следует понимать, что объект изобретения, определенный в прилагаемой формуле изобретения, необязательно ограничивается конкретными признаками или действиями, описанными выше. Скорее, конкретные признаки и действия, описанные выше, описаны в качестве примерных форм реализации притязаний.

(57) Формула изобретения

1. Способ, реализованный в вычислительном устройстве, для защиты от вредоносных программ, причем способ содержит:

идентификацию страницы памяти, включающую в себя исполняемый код, подлежащий исполнению виртуальным процессором виртуальной машины, причем виртуальная машина управляется менеджером виртуальной машины;

определение, должна ли страница памяти быть исполняемой в режиме ядра;

выполнение более привилегированным объектом, чем операционная система виртуальной машины, и в ответ на определение, что страница памяти должна быть исполняемой в режиме ядра, проверки целостности кода исполняемого кода, основанной на политике целостности кода, и предоставление возможности исполнения исполняемого кода для режима ядра, если только проверка целостности кода верифицирует исполняемый код; и

предоставление возможности в ответ на определение, что страница памяти не должна быть исполняемой в режиме ядра, определения операционной системой виртуальной машины, предоставить ли возможность исполнения исполняемого кода в пользовательском режиме.

2. Способ по п. 1, в котором более привилегированным объектом, чем операционная система виртуальной машины, является менеджер виртуальной машины.

3. Способ по п. 1, дополнительно содержащий выполнение операционной системой и в ответ на определение, что страница памяти не должна быть исполняемой в режиме ядра, проверки целостности кода исполняемого кода и предоставление возможности исполнения исполняемого кода, если только проверка целостности кода операционной системой верифицирует исполняемый код.

4. Способ по п. 1, в котором вычислительное устройство дополнительно содержит компонент, включающий в себя атрибут, предоставляющий возможность задавать разрешение исполнения для режима ядра виртуального процессора отдельно от разрешения исполнения или политики для пользовательского режима виртуального процессора.

5. Вычислительное устройство, включающее в себя операционную систему, менеджер виртуальной машины и процессор, причем процессор выполнен с возможностью предоставления менеджеру виртуальной машины возможности ограничения исполнения в режиме ядра страниц памяти, включающих в себя исполняемый код, подлежащий исполнению виртуальным процессором виртуальной машины, страницами памяти, имеющими код, целостность которого была верифицирована проверкой целостности кода, основанной на политике целостности кода, более привилегированным объектом, который является более привилегированным, чем операционная система, но предоставления возможности исполнения в пользовательском режиме страниц памяти, не учитывая, была ли верифицирована целостность кода страниц памяти более привилегированным объектом посредством предоставления возможности, в ответ на определение, что страница памяти не должна быть исполняемой в режиме ядра, определения операционной системой виртуальной машины, предоставлять ли возможность исполнения исполняемого кода в пользовательском режиме.

6. Устройство по п. 5, в котором процессор дополнительно выполнен с возможностью установки атрибута разрешения исполнения страницы памяти, хранящей код, целостность которого была верифицирована менеджером виртуальной машины.

7. Устройство по п. 5, в котором процессор включает в себя таблицу преобразования адресов, атрибут которой предоставляет возможность задавать разрешение исполнения для исполнения в режиме ядра страниц памяти отдельно от разрешения исполнения для исполнения в пользовательском режиме страниц памяти.

8. Устройство по п. 5, в котором процессор дополнительно включает в себя аппаратный компонент, имеющий атрибут, предоставляющий возможность задавать разрешение исполнения для исполнения в режиме ядра страниц памяти отдельно от политики исполнения для исполнения в пользовательском режиме страниц памяти.

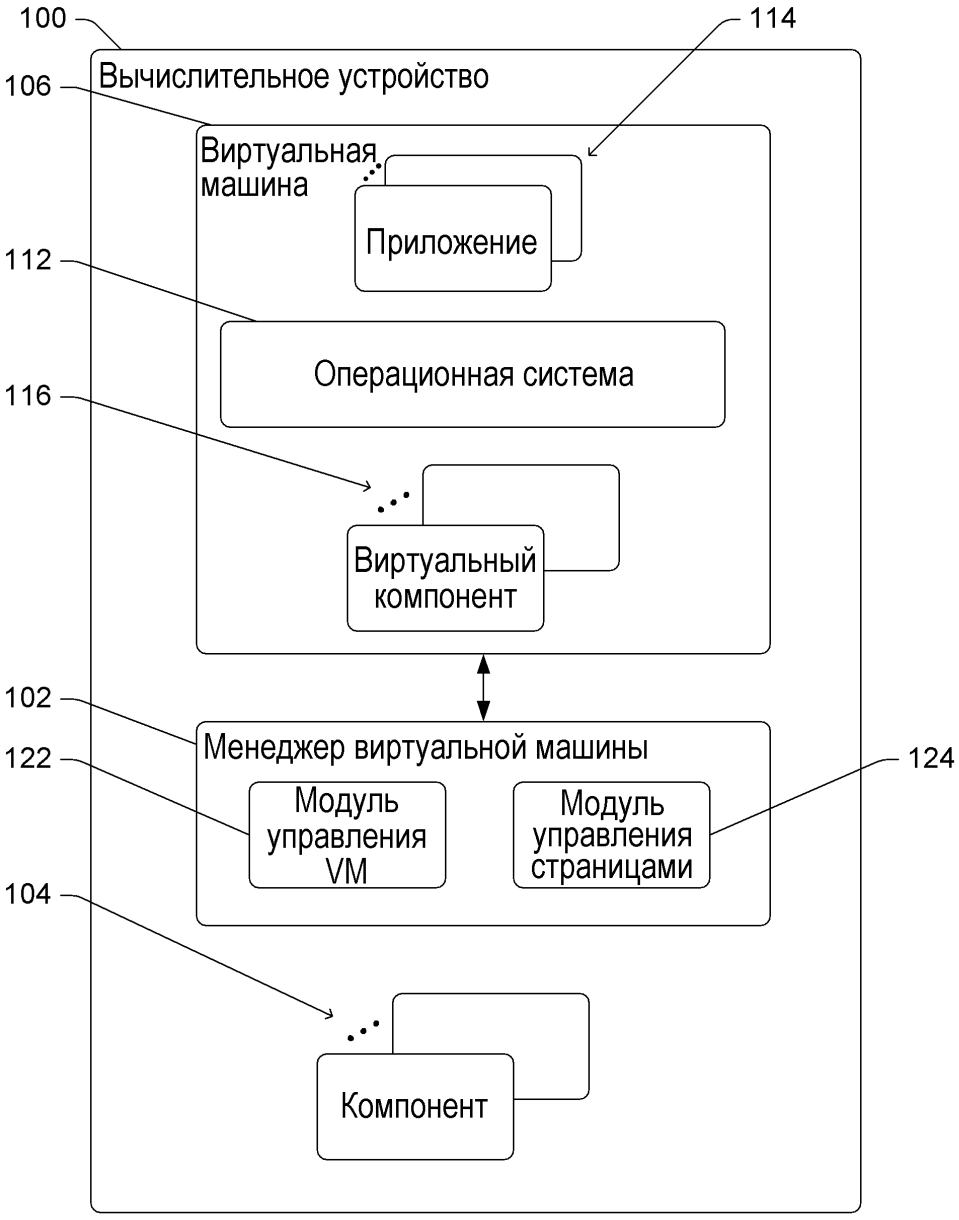
9. Устройство по п. 8, в котором атрибут предоставляет возможность задавать разрешение исполнения для исполнения в режиме ядра страниц памяти отдельно от политики исполнения для исполнения в пользовательском режиме страниц памяти для каждой страницы памяти.

10. Устройство по п. 8, в котором атрибут предоставляет возможность одной комбинации исполнения, которая не предоставляет возможность исполнения в режиме ядра кода в странице памяти, но предоставляет возможность исполнения в пользовательском режиме кода в странице памяти, и дополнительной комбинации, которая не предоставляет возможность как исполнения в режиме ядра кода в странице памяти, так и исполнения в пользовательском режиме кода в странице памяти.

1

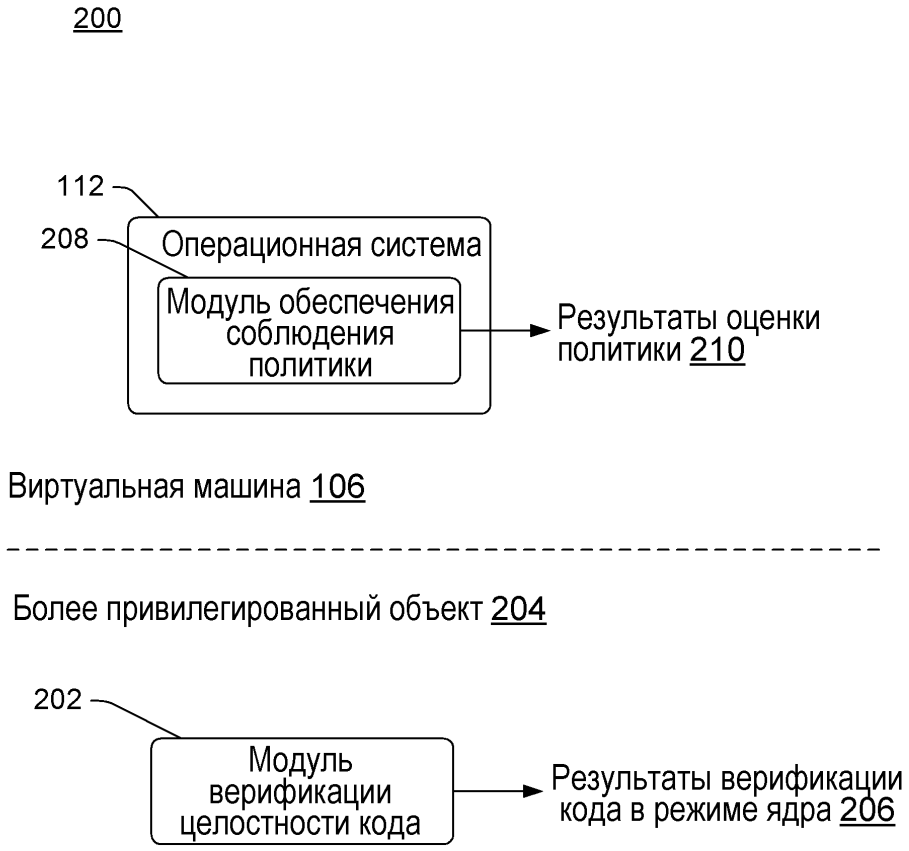
532086

1/5



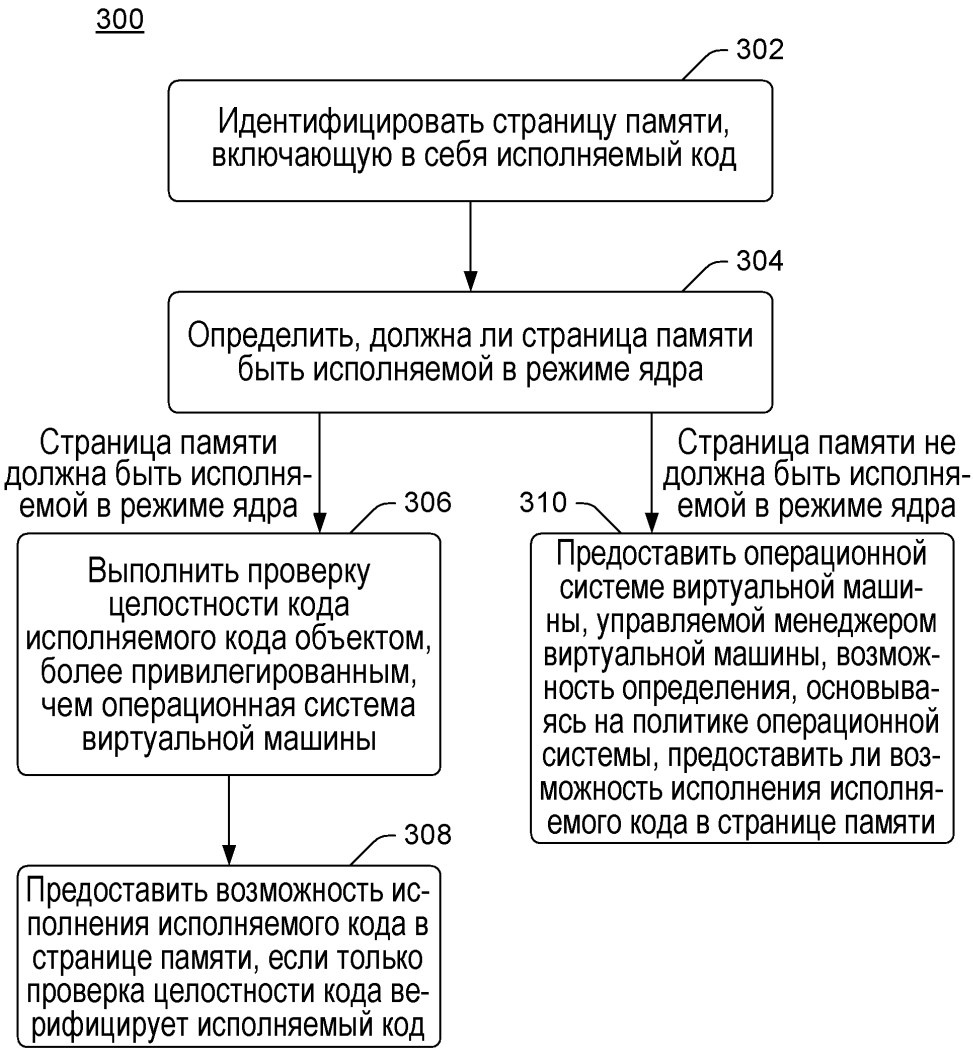
ФИГ. 1

2


200
112
208
Операционная система
Модуль обеспечения соблюдения политики
Результаты оценки политики 210
Виртуальная машина 106
Более привилегированный объект 204
202
Модуль верификации целостности кода
Результаты верификации кода в режиме ядра 206

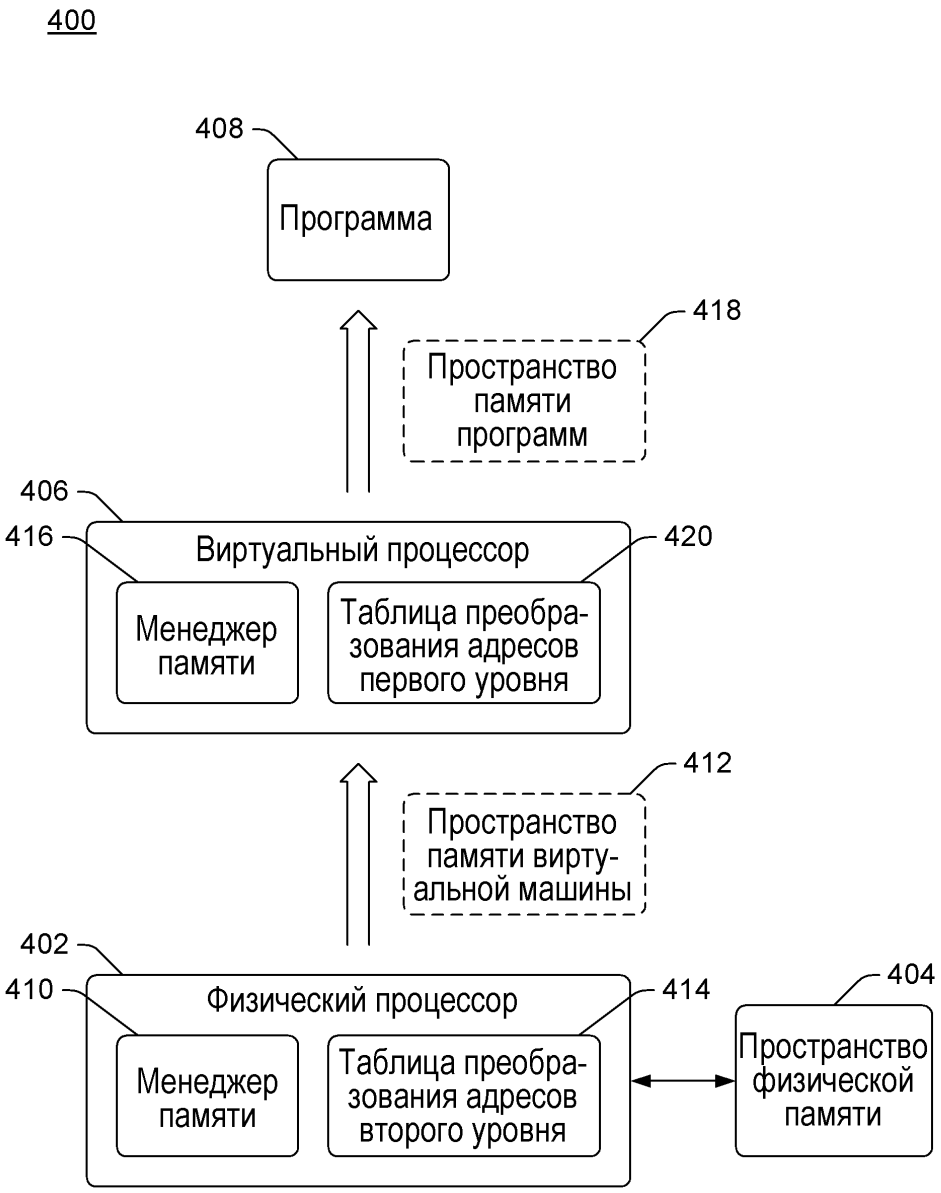
ФИГ. 2

3

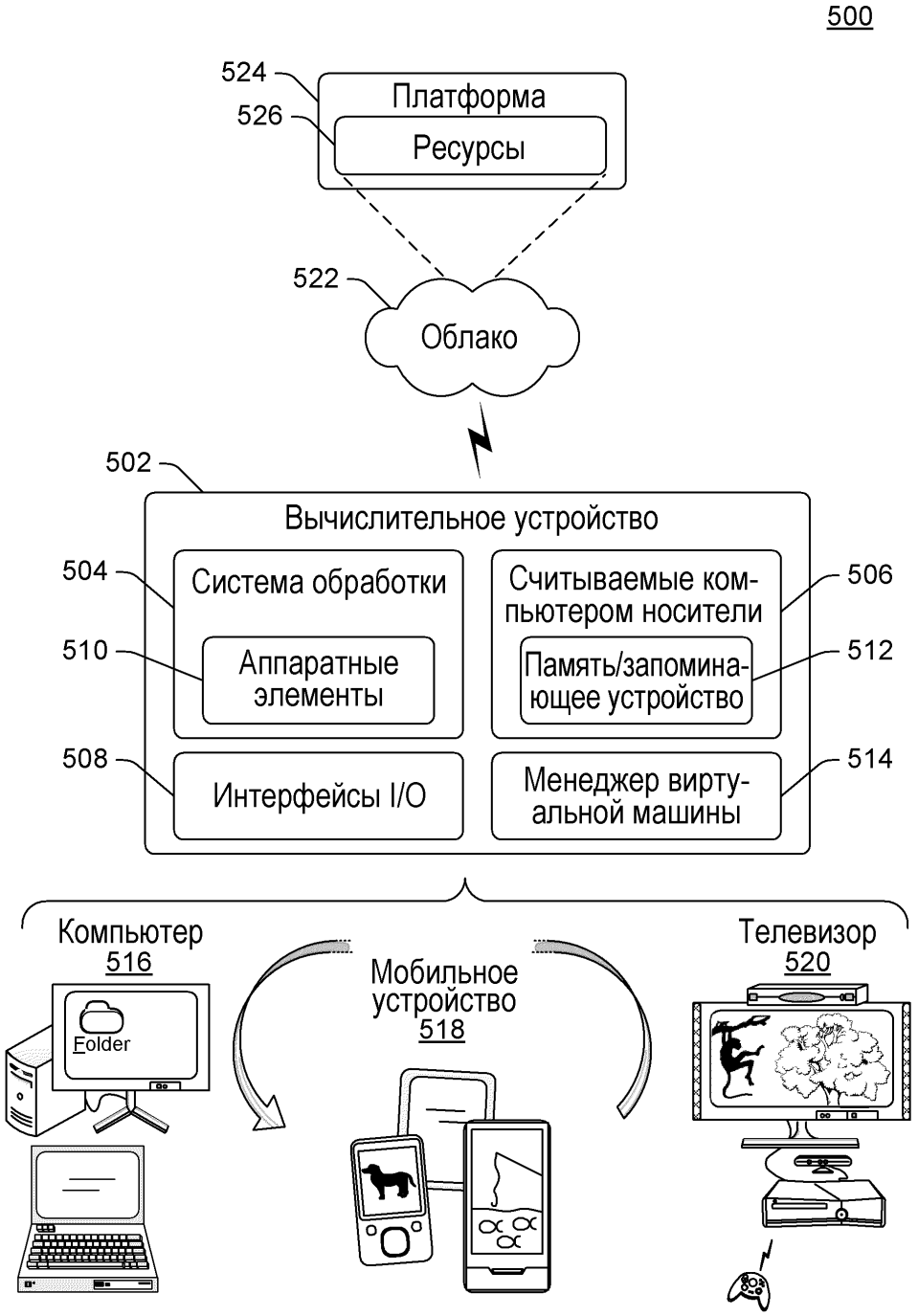

300
302
Идентифицировать страницу памяти, включающую в себя исполняемый код
304
Определить, должна ли страница памяти быть исполняемой в режиме ядра
Страница памяти должна быть исполняемой в режиме ядра
Страница памяти не должна быть исполняемой в режиме ядра
306
Выполнить проверку целостности кода исполняемого кода объектом, более привилегированным, чем операционная система виртуальной машины
308
Предоставить возможность исполнения исполняемого кода в странице памяти, если только проверка целостности кода верифицирует исполняемый код
310
Предоставить операционной системе виртуальной машины, управляемой менеджером виртуальной машины, возможность определения, основываясь на политике операционной системы, предоставить ли возможность исполнения исполняемого кода в странице памяти

ФИГ. 3

4


400
408
Программа
418
Пространство памяти программ
406
416
Виртуальный процессор
420
Менеджер памяти
Таблица преобразования адресов первого уровня
412
Пространство памяти виртуальной машины
402
410
Физический процессор
414
404
Менеджер памяти
Таблица преобразования адресов второго уровня
Пространство физической памяти

ФИГ. 4


500
524
Платформа
526
Ресурсы
522
Облако
502
Вычислительное устройство
504
Система обработки
506
Считываемые компьютером носители
510
Аппаратные элементы
512
Память/запоминающее устройство
508
Интерфейсы I/O
514
Менеджер виртуальной машины
Компьютер
516
Folder
Мобильное устройство
518
Телевизор
520

ФИГ. 5