РОССИЙСКАЯ ФЕДЕРАЦИЯ

(19) **RU** (11) **2 645 268**(13) **C2**

(51) МПК
G06F 21/56 (2013.01)

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

(12) ОПИСАНИЕ ИЗОБРЕТЕНИЯ К ПАТЕНТУ

(52) СПК
G06F 21/56 (2017.08); G06F 21/554 (2017.08); G06F 21/566 (2017.08); G06F 9/45558 (2017.08); H04L 63/14 (2017.08)

(21)(22) Заявка: **2016114944, 25.09.2014**

(24) Дата начала отсчета срока действия патента:
25.09.2014

Дата регистрации:
19.02.2018

Приоритет(ы):
(30) Конвенционный приоритет:
04.10.2013 US 14/046,728

(43) Дата публикации заявки: **13.11.2017** Бюл. № **32**

(45) Опубликовано: **19.02.2018** Бюл. № **5**

(85) Дата начала рассмотрения заявки PCT на национальной фазе: **04.05.2016**

(86) Заявка PCT:
RO 2014/000027 (25.09.2014)

(87) Публикация заявки PCT:
WO 2015/050469 (09.04.2015)

Адрес для переписки:
191002, Санкт-Петербург, а/я 5, ООО "Ляпунов и партнёры"

(72) Автор(ы):
ЛУКАКС Сандор (RO),
ТОША Раул-Василе (RO),
БОКА Паул-Даниэл (RO),
ХАЖМАШАН Георге-Флорин (RO),
ЛУЦАС Андрей-Влад (RO)

(73) Патентообладатель(и):
БИТДЕФЕНДЕР АйПиАр МЕНЕДЖМЕНТ ЛТД (CY)

(56) Список документов, цитированных в отчете о поиске: **US 2012/0324575 A1, 20.12.2012. WO 2010/023557 A2, 04.03.2010. US 2004/0230835 A1, 18.11.2004. WO 2012/135192 A2, 04.10.2012. RU 2454714 C1, 27.06.2012.**

(54) СЛОЖНОЕ КЛАССИФИЦИРОВАНИЕ ДЛЯ ВЫЯВЛЕНИЯ ВРЕДОНОСНЫХ ПРОГРАММ

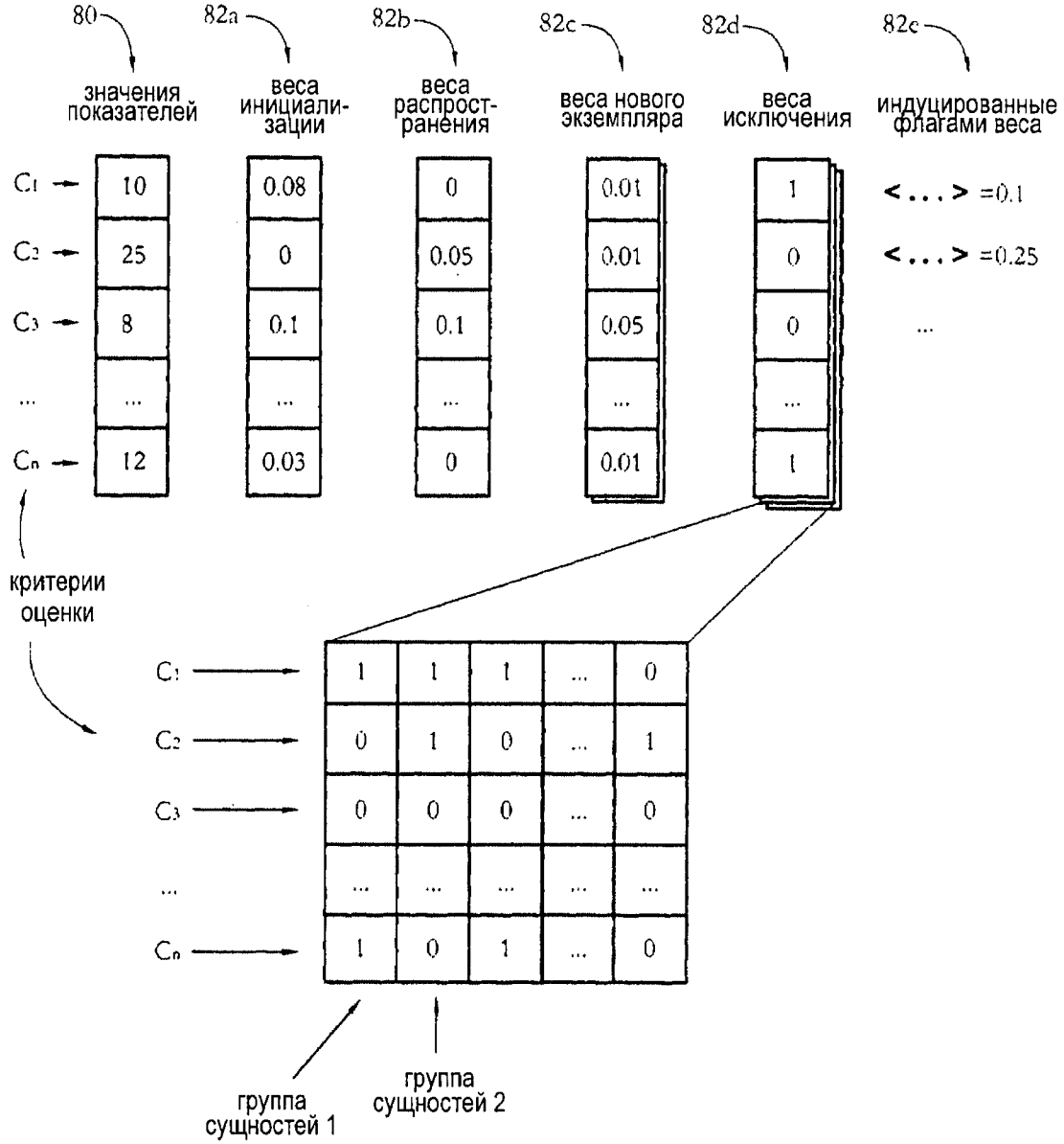
(57) Реферат:

Изобретение относится к области защиты компьютерных систем от вредоносных программ. Техническим результатом является определение, является ли программная сущность вредоносной, на основе множества показателей оценки соответствующей сущности, что позволяет создать более надежное антивредоносное решение по сравнению с аналогичными традиционными решениями. Раскрыта хостовая система для определения вредоносной программной сущности, содержащая блок памяти, хранящий инструкции, при исполнении которых по меньшей мере одним аппаратным процессором хостовой системы хостовая система выполняет модуль управления сущностями, средство оценки сущностей и классифицирующий механизм, при этом: модуль управления сущностями конфигурирован с возможностью управлять коллекцией оцениваемых программных сущностей, причем управление коллекцией содержит: идентификацию набора сущностей-потомков первой сущности коллекции; определение, завершена ли первая сущность; в ответ, когда первая сущность завершена,

RU 2645268 C2

RU 2645268 C2

определение, завершены ли все члены набора сущностей-потомков; и в ответ, когда все члены набора сущностей-потомков завершены, удаление первой сущности из коллекции; средство оценки сущностей конфигурировано с возможностью: оценивать первую сущность согласно критерию оценки; и в ответ, когда первая сущность удовлетворяет критерию оценки, передавать индикатор оценки в классифицирующий механизм; классифицирующий механизм конфигурирован с возможностью: записывать первый показатель, определенный для первой сущности, и второй показатель, определенный для второй сущности коллекции, причем первый и второй показатели определены согласно критерию оценки; в ответ на запись первого и второго показателей и в ответ на получение индикатора оценки, обновлять второй показатель согласно индикатору оценки; в ответ определять, является ли вторая сущность вредоносной согласно обновленному второму показателю, в ответ на получение индикатора оценки, обновлять первый показатель согласно индикатору оценки; и в ответ определять, является ли первая сущность вредоносной согласно обновленному первому показателю. 3 н. и 12 з.п. ф-лы, 13 ил.



Фиг.9

RU 2645268 C2

RU 2645268 C2

RUSSIAN FEDERATION

FEDERAL SERVICE
FOR INTELLECTUAL PROPERTY

(19) **RU** (11) **2 645 268**(13) **C2**

(51) Int. Cl.
G06F 21/56 (2013.01)

(12) ABSTRACT OF INVENTION

(52) CPC
G06F 21/56 *(2017.08);* ***G06F 21/554*** *(2017.08);* ***G06F 21/566*** *(2017.08);* ***G06F 9/45558*** *(2017.08);* ***H04L 63/14*** *(2017.08)*

(21)(22) Application: **2016114944, 25.09.2014**

(24) Effective date for property rights:
25.09.2014

Registration date:
19.02.2018

Priority:
(30) Convention priority:
04.10.2013 US 14/046,728

(43) Application published: **13.11.2017** Bull. № **32**

(45) Date of publication: **19.02.2018** Bull. № **5**

(85) Commencement of national phase: **04.05.2016**

(86) PCT application:
RO 2014/000027 (25.09.2014)

(87) PCT publication:
WO 2015/050469 (09.04.2015)

Mail address:
191002, Sankt-Peterburg, a/ya 5, OOO "Lyapunov i partnery"

(72) Inventor(s):
LUKAKS Sandor (RO),
TOSHA Raul-Vasile (RO),
BOKA Paul-Daniel (RO),
KHAZHMASHAN George-Florin (RO),
LUTSAS Andrej-Vlad (RO)

(73) Proprietor(s):
BITDEFENDER AjPiAr MENEDZHMENT LTD (CY)

RU 2645268 C2

(54) COMPLEX CLASSIFICATION FOR DETECTING MALWARE

(57) Abstract:

FIELD: protection of computer systems from malware.

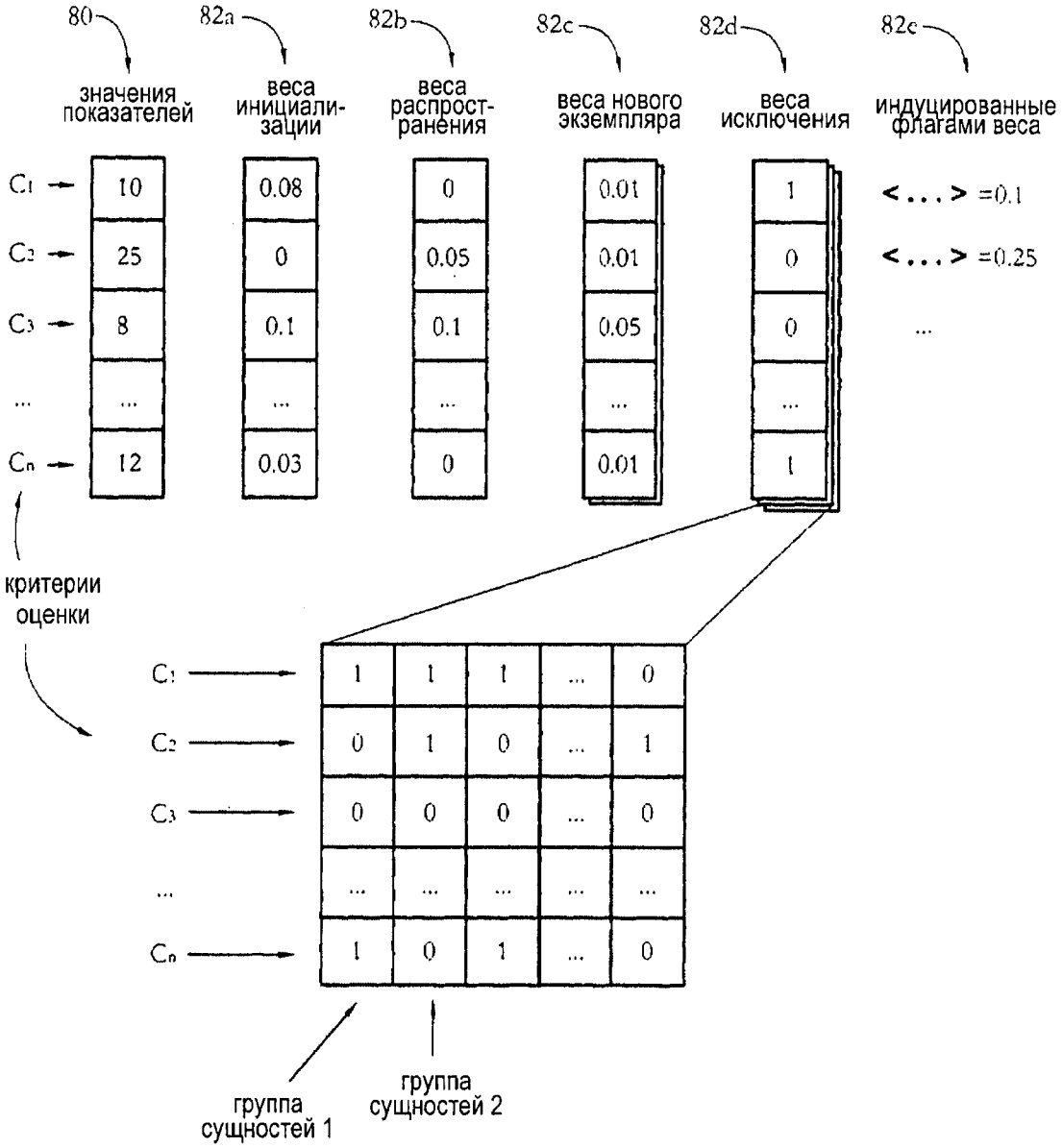
SUBSTANCE: invention relates to protection of computer systems from malware. Host system for detecting malware entity is discovered comprising a memory unit that stores instructions, using which the host system, when performed by at least one hardware processor of the host system, executes the entity management module, the entity assessment engine and the classification engine, wherein: the entity management module is configured to manage a collection of assessed software entities, the collection management comprising: identifying a set of daughter entities of the first collection entity; determining if the first entity is complete; in response, when the first entity is completed, determining if all members of the set of daughter entities are completed; and in response, when all members of the set of daughter entities are completed, removing the first entity from the collection; entity assessment engine is configured with the ability to: assess the first entity according to the assessment criterion; and in response, when the first entity satisfies the assessment criterion, transmit the assessment indicator to the classification engine; classification engine is configured to: record the first indicator determined for the first entity and the second indicator

RU 2645268 C2

determined for the second entity of the collection, wherein the first and second indicators are determined according to the assessment criterion; in response to the recording of the first and second indicators and in response to the receipt of the assessment indicator, update the second indicator according to the assessment indicator; in response, to determine whether the second entity is malicious according to the updated second indicator, in response to the recording of the first and second indicators and in response to the receipt of the assessment indicator, update the second indicator according to the assessment indicator; in response, to determine whether the second entity is malicious according to the updated second indicator.

EFFECT: technical result is the determination whether the software entity is malicious, based on a variety of indicators for the assessment of the relevant entity, which makes it possible to create a more robust anti-malware solution in comparison with similar traditional solutions.

15 cl, 13 dwg



Фиг.9

RU 2645268 C2

RU 2645268 C2

ОБЛАСТЬ ТЕХНИКИ, К КОТОРОЙ ОТНОСИТСЯ ИЗОБРЕТЕНИЕ

[0001] Изобретение относится к системам и способам защиты компьютерных систем от вредоносных программ.

УРОВЕНЬ ТЕХНИКИ

[0002] Вредоносное программное обеспечение, также известное как вредоносные программы, воздействует на большое количество компьютерных систем по всему миру. В своих многочисленных формах, таких как компьютерные вирусы, черви, руткиты и шпионские программы, вредоносные программы представляют собой серьезную угрозу для миллионов пользователей компьютеров, делая их уязвимыми, помимо прочего, к потере данных и конфиденциальной информации, краже личных данных и потери производительности.

[0003] Для выявления вредоносных программ, заражающих компьютерную систему пользователя, и, дополнительно, для удаления или остановки выполнения подобных вредоносных программ могут использоваться программные средства безопасности. В данной области техники известно несколько способов выявления вредоносных программ. Некоторые из них полагаются на соответствие фрагмента кода вредоносного агента библиотеке указывающих на вредоносность сигнатур. Другие традиционные способы выявляют набор указывающих на вредоносность поведений вредоносного агента.

[0004] Для предотвращения выявления и/или подрыва работы программных средств безопасности, некоторые вредоносные агенты используют способы запутывания, такие как шифрование своего кода, или используют несколько отличные версии кода на каждой зараженной компьютерной системе (полиморфизм). Другие примерные способы предотвращения выявления разделяют вредоносные действия на несколько действий, причем каждое действие выполняются отдельным агентом, возможно с задержкой по времени. В других примерах вредоносные программы могут попытаться активно атаковать и отключить программное средство безопасности, например, с использованием привилегий и/или путем переписывания кода программного средства безопасности.

[0005] Для того чтобы не отставать от быстро меняющегося ряда угроз от вредоносных программ, существует большой интерес к разработке надежных и масштабируемых антивредоносных решений.

РАСКРЫТИЕ ИЗОБРЕТЕНИЯ

[0006] Согласно одному аспекту хостовая система содержит по меньшей мере один процессор, конфигурированный с возможностью выполнять модуль управления сущностями, средство оценки сущностей и классифицирующий механизм. Модуль управления сущностями конфигурирован с возможностью управлять коллекцией оцениваемых программных сущностей, причем управление коллекцией содержит: идентификацию набора сущностей-потомков первой сущности коллекции; определение, завершена ли первая сущность; в ответ, когда первая сущность завершена, определение, завершены ли все члены набора сущностей-потомков; и в ответ, когда все члены набора сущностей-потомков завершены, удаление первой сущности из коллекции. Средство оценки сущностей конфигурировано с возможностью: оценивать первую сущность согласно критерию оценки; и в ответ, когда первая сущность удовлетворяет критерию оценки, передавать индикатор оценки в классифицирующий механизм. Классифицирующий механизм конфигурирован с возможностью: записывать первый показатель, определенный для первой сущности, и второй показатель, определенный для второй сущности коллекции, причем первый и второй показатели определены

согласно критерию оценки; в ответ на запись первого и второго показателей и в ответ на получение индикатора оценки, обновлять второй показатель согласно индикатору оценки; и в ответ, определять, является ли вторая сущность вредоносной согласно обновленному второму показателю.

[0007] Согласно другому аспекту долговременный машиночитаемый носитель хранит инструкции, которые, при выполнении, конфигурируют по меньшей мере один процессор хостовой системы с возможностью: управлять коллекцией оцениваемых программных сущностей, причем управление коллекцией содержит: идентификацию набора сущностей-потомков первой сущности коллекции; определение, завершена ли первая сущность; в ответ, когда первая сущность завершена, определение, завершены ли все члены набора сущностей-потомков; и в ответ, когда все члены набора сущностей-потомков завершены, удаление выбранной сущности из коллекции. Инструкции дополнительно конфигурируют указанный по меньшей мере один процессор с возможностью записывать первый показатель, определенный для первой сущности, и второй показатель, определенный для второй сущности коллекции, при этом первый и второй показатели определены согласно критерию оценки. Инструкции дополнительно конфигурируют указанный по меньшей мере один процессор с возможностью, в ответ на запись первого и второго показателей, оценивать первую сущность согласно критерию оценки. Инструкции дополнительно конфигурируют указанный по меньшей мере один процессор с возможностью, в ответ на оценку первой сущности, когда первая сущность удовлетворяет критерию оценки, обновлять второй показатель, и в ответ на обновление второго показателя, определять, является ли вторая сущность вредоносной согласно обновленному второму показателю.

[0008] Согласно другому аспекту хостовая система содержит по меньшей мере один процессор, конфигурированный с возможностью выполнять средство оценки сущностей, и классифицирующий механизм. Средство оценки сущностей конфигурировано с возможностью: оценивать первую программную сущность согласно критерию оценки, при этом первая программная сущность выполняется на клиентской системе; и в ответ, когда первая программная сущность удовлетворяет критерию оценки, передавать индикатор оценки в классифицирующий механизм. Классифицирующий механизм конфигурирован с возможностью, в ответ на получение индикатора оценки, обновлять показатель согласно индикатору оценки, причем показатель определен для второй программной сущности, прежде выполнявшейся на хостовой системе, при этом вторая программная сущность завершена в момент обновления показателя. Классифицирующий механизм дополнительно конфигурирован с возможностью, в ответ на обновление второго показателя, определять, является ли вторая программная сущность вредоносной согласно обновленному второму показателю.

[0009] Согласно другому аспекту способ содержит применение по меньшей мере одного процессора хостовой системы для определения, удовлетворяет ли первая программная сущность, выполняющаяся на хостовой системе, критерию оценки. Способ дополнительно содержит, когда первая программная сущность удовлетворяет критерию оценки, применение указанного по меньшей мере одного процессора для обновления показателя, определенного для второй программной сущности, выполнявшейся прежде на хостовой системе, причем вторая программная сущности завершена в момент обновления показателя, при этом показатель определен согласно критерию оценки. Способ дополнительно содержит, в ответ на обновление второго показателя, применение указанного по меньшей мере одного процессора для определения, является ли вторая программная сущность вредоносной согласно обновленному второму показателю.

КРАТКОЕ ОПИСАНИЕ ЧЕРТЕЖЕЙ

[0010] Вышеизложенные аспекты и преимущества настоящего изобретения будут более понятными при прочтении нижеследующего подробного описания со ссылками на чертежи, на которых:

[0011] На фиг. 1 показана примерная конфигурация аппаратного обеспечения хостовой компьютерной системы, защищенной от вредоносных программ, в соответствии с некоторыми вариантами осуществления настоящего изобретения.

[0012] На фиг. 2-A показан примерный набор программных объектов, включающих в себя приложение безопасности, которое выполняется на хостовой системе, в соответствии с некоторыми вариантами осуществления настоящего изобретения.

[0013] На фиг. 2-B показан примерный набор программных объектов, включающих в себя приложение безопасности, которое выполняется в виртуальной машине, в хостовой системе, конфигурированной для поддержки виртуализации.

[0014] На фиг. 3 показана примерная иерархия программных объектов, выполняющихся на хостовой системе на различных уровнях привилегий процессора, включая набор антивредоносных объектов в соответствии с некоторыми вариантами осуществления настоящего изобретения.

[0015] На фиг. 4 показана примерная последовательность этапов, выполняемых с помощью модуля управления сущностями на фиг. 3, в соответствии с некоторыми вариантами осуществления настоящего изобретения.

[0016] На фиг. 5 показан примерный классифицирующий механизм, получающий множество индикаторов оценки сущностей, которые определены для программной сущности с помощью множества модулей средства оценки сущностей, в соответствии с некоторыми вариантами осуществления настоящего изобретения.

[0017] На фиг. 6 показан примерный поток выполнения набора процессов в среде Windows®. Сплошные стрелки указывают на примерный поток выполнения при отсутствии антивредоносной системы. Пунктирные стрелки указывают на модификации в потоке выполнения, причем модификации введены множеством средств оценки сущностей, которые работают в соответствие с некоторыми вариантами осуществления настоящего изобретения.

[0018] На фиг. 7 показана примерная последовательность этапов, выполняемых модулем средства оценки сущностей в соответствии с некоторыми вариантами осуществления настоящего изобретения.

[0019] На фиг. 8 показано множество примерных классифицирующих сущности объектов (ESO), причем каждый ESO определен для соответствующей программной сущности в соответствии с некоторыми вариантами осуществления настоящего изобретения. Примерные поля данных ESO включают в себя, помимо прочего, индикатор идентичности сущности EID, множество показателей S_i и суммарный показатель A, которые определены для соответствующей сущности.

[0020] На фиг. 9 показан примерный набор значений показателя и различные примерные наборы весов, используемых классифицирующим механизмом для классификации программных сущностей в соответствии с некоторыми вариантами осуществления настоящего изобретения.

[0021] На фиг. 10 показана примерная последовательность этапов, выполняемых классифицирующим механизмом (фиг. 3-4) в соответствии с некоторыми вариантами осуществления настоящего изобретения.

[0022] На фиг. 11 показана примерная конфигурация, включающая в себя множество хостовых систем, соединенных с сервером безопасности через компьютерную сеть.

[0023] На фиг. 12 показана примерная антивредоносная транзакция между хостовой системой и сервером безопасности в соответствии с некоторыми вариантами осуществления настоящего изобретения.

ОСУЩЕСТВЛЕНИЕ ИЗОБРЕТЕНИЯ

[0024] В нижеследующем описании следует понимать, что все упоминаемые соединения между конструкциями могут быть непосредственными функциональными соединениями или опосредованными функциональными соединениями через промежуточные конструкции. Множество элементов содержит один или более элементов. Любое упоминание элемента понимается как относящееся к, по меньшей мере, одному элементу. Множество элементов содержит по меньшей мере два элемента. Если не требуется иное, любые описанные этапы способа не должны обязательно выполняться в конкретном показанном порядке. Первый элемент (например, данные), полученный из второго элемента, охватывает первый элемент, эквивалентный второму элементу, а также первый элемент, генерированный посредством обработки второго элемента и, опционально, другие данные. Выполнение определения или решения в соответствии с параметром охватывает выполнение определения или решения в соответствии с этим параметром и, опционально, согласно другим данным. Если не указано иное, индикатор некоторой величины/данных может сам представлять собой величину/данные, или сам индикатор, отличный от величины/данных. Если не указано иное, процесс представляет собой копию компьютерной программы, причем компьютерная программа представляет собой последовательность инструкций, определяющих компьютерную систему для выполнения указанной задачи. Машиночитаемые носители включают в себя долговременные носители, такие как магнитные, оптические и полупроводниковых носители (например, жесткие диски, оптические диски, флэш-память, DRAM), и линии связи, такие как токопроводящие кабели и волоконно-оптические линии связи. Согласно некоторым вариантам осуществления, настоящее изобретение обеспечивает, в частности, компьютерные системы, содержащие аппаратные средства (например, один или несколько процессоров), программированные для выполнения способов, описанных в настоящем документе, и машиночитаемые носители, кодирующие инструкции для выполнения способов, описанных в настоящем документе.

[0025] Нижеследующее описание иллюстрирует варианты осуществления изобретения посредством примеров и не обязательно посредством ограничения.

[0026] На фиг. 1 показана примерная конфигурация аппаратного оборудования хостовой системы 10, выполняющей антивредоносные операции в соответствии с некоторыми вариантами осуществления настоящего изобретения. Хостовая система 10 может представлять собой вычислительное устройство профессионального применения, такое как корпоративный сервер, или устройство конечного пользователя, такое как персональный компьютер или смартфон помимо прочего. Другие хостовые системы включают в себя развлекательные устройства, такие как телевизоры и игровые приставки, или любое другое устройство, имеющее память и процессор с поддержкой виртуализации и требующее защиты от вредоносных программ. Фиг. 1 показывает компьютерную систему в иллюстративных целях; другие клиентские устройства, такие как мобильные телефоны или планшеты, могут иметь отличную конфигурацию. В некоторых вариантах осуществления, система 10 содержит набор физических устройств, в том числе процессор 12, блок 14 памяти, набор устройств 16 ввода, набор устройств 18 вывода, набор устройств 20 хранения и набор сетевых адаптеров 22, которые все соединены набором шин 24.

[0027] В некоторых вариантах осуществления процессор 12 содержит физическое

устройство (например, многоядерную интегральную схему), конфигурированную с возможностью выполнять вычислительные и/или логические операции с набором сигналов и/или данных. В некоторых вариантах осуществления такие логические операции поступают в процессор 12 в виде последовательности инструкций процессора (например, машинного кода или программного обеспечения другого типа). Блок 14 памяти может включать в себя долговременный машиночитаемый носитель (например, оперативное запоминающее устройство (RAM)), хранящий данные/сигналы, к которым обращается процессор 12 или которые генерируются им в процессе выполнения инструкций. Устройства 16 ввода могут включать в себя компьютерные клавиатуры, мыши и микрофоны, помимо прочего, в том числе соответствующие аппаратные интерфейсы и/или адаптеры, которые позволяют пользователю вводить данные и/или инструкции в систему 10. Устройства 18 вывода могут включать в себя устройства отображения, такие как мониторы и громкоговорители, помимо прочего, а также аппаратные интерфейсы/адаптеры, такие как графические карты, что позволяет системе 10 передавать данные пользователю. В некоторых вариантах осуществления устройства 16 ввода и устройства 18 вывода могут совместно использовать общую часть аппаратного оборудования, как в случае устройств с сенсорным экраном. Устройства 20 хранения включают в себя машиночитаемые носители, обеспечивающие энергонезависимое хранение, чтение и запись программных инструкций и/или данных. Примерные устройства 20 хранения включают в себя магнитные и оптические диски и устройства флэш-памяти, а также съемные носители, такие как CD и/или DVD-диски и дисковые накопители. Набор сетевых адаптеров 22 позволяет системе 10 присоединяться к компьютерной сети и/или к другим устройствам/компьютерным системам. Шины 24 в совокупности представляют собой множество шин системы и микросхем, периферийных шин и/или все другие схемы, обеспечивающие внутреннюю связь устройств 12-22 хостовой системы 10. Например, шины 24 могут включать в себя северный мост, соединяющий процессор 12 с памятью 14, и/или южный мост, соединяющий процессор 12 с устройствами 16-22, помимо прочего.

[0028] На фиг. 2-A показан примерный набор программных объектов, выполняющихся на хостовой системе 10 в конфигурации, которая не использует аппаратную виртуализацию. В некоторых вариантах осуществления гостевая операционная система (OS) 34 содержит программное обеспечение, которое обеспечивает интерфейс для аппаратного оборудования хостовой системы 10, а также действует в качестве хоста для набора программных приложений 42a-c и 44. OS 34 может включать в себя любую широко доступную операционную систему, такую как Windows®, MacOS®, Linux®, iOS® или Android™, помимо прочего. Приложения 42a-c могут включать в себя приложения обработки текстов, обработки изображений, приложения базы данных, браузера и электронной связи, помимо прочего.

[0029] На фиг. 2-B показан примерный набор программных объектов, выполняющихся на хостовой системе 10 в одном варианте осуществления с использованием аппаратной виртуализации. Набор гостевых виртуальных машин 32a-b предоставлен гипервизором 30. Виртуальные машины (VM) широко известны в данной области как программная эмуляция реальных физических машин/компьютерных систем, каждая из которых может выполнять свои собственные операционную систему и программное обеспечение независимо от других виртуальных машин. Гипервизор 30 содержит программное обеспечение, обеспечивающее мультиплексирование (совместное использование) несколькими виртуальными машинами ресурсов аппаратного обеспечения хостовой системы 10, таких как операции процессора, память, хранение,

ввод/вывод и сетевые устройства. В некоторых вариантах осуществления гипервизор 30 обеспечивает возможность одновременного выполнения нескольких виртуальных машин и/или операционных систем (OS) на хостовой системе 10, с различными степенями изоляции. Для обеспечения возможности таких конфигураций, программное обеспечение, образующее часть гипервизора 30, может создавать множество виртуализированных, т.е. программно эмулированных устройств, причем каждое виртуализированное устройство эмулирует физическое аппаратное устройство в системе 10, такое как процессор 12 и память 14, помимо прочего. Гипервизор 30 может дополнительно назначать набор виртуальных устройств для каждой VM, работающей на хостовой системе 10. Таким образом, каждая виртуальная машина 32a-b работает, как если бы она обладала своим собственным набором физических устройств, то есть, как более или менее полная компьютерная система. Создание и назначение виртуальных устройств виртуальной машине, как правило, известно в данной области техники как предоставление соответствующей VM. Примеры популярных гипервизоров включают в себя VMware vSphere™ от VMware Inc. и гипервизор Xen с открытым исходным кодом, помимо прочего.

[0030] В некоторых вариантах осуществления гипервизор 30 содержит механизм 40 самоанализа памяти, конфигурированный с возможностью выполнения антивредоносных операций, как описано ниже. Механизм 40 может быть встроен в гипервизор 30 или может поставляться в качестве программного компонента отдельно и независимо от гипервизора 30, однако он выполняется по существу на сходном уровне привилегий процессора, как и гипервизор 30. Один механизм 40 может быть конфигурирован для защиты от вредоносных программ множества виртуальных машин, выполняющихся на хостовой системе 10.

[0031] Хотя фиг. 2-B показывает для простоты только две виртуальные машины 32a-b, хостовая система 10 может параллельно выполнять большое количество, например сотни, виртуальных машин, причем количество таких виртуальных машин может изменяться во время работы хостовой системы 10. В некоторых вариантах осуществления каждая виртуальная машина 32a-b выполняет гостевую операционную систему 34a-b и/или набор программных приложений 42d-e и 42f, соответственно, параллельно и независимо от других виртуальных машин, работающих на хостовой системе 10. Каждая OS 34a-b содержит программное обеспечение, которое обеспечивает интерфейс (виртуализированный) аппаратному обеспечению соответствующей VM 32a-b и действует в качестве хоста для вычислений приложений, выполняющихся на соответствующей OS.

[0032] В некоторых вариантах осуществления приложение 44 безопасности конфигурировано с возможностью выполнения антивредоносных операций, как описано ниже, для защиты хостовой системы 10 от вредоносных программ. В примере на фиг. 2-B, экземпляр приложения 44 может выполняться на каждой VM 32a-b, при этом каждый такой экземпляр конфигурирован для защиты соответствующей виртуальной машины. Приложение 44 безопасности может быть отдельной программой или может образовывать часть программного пакета, включающую в себя, помимо прочего, антивредоносные, антиспамовые и антишпионские компоненты.

[0033] На фиг. 3 показана иерархия программных объектов, выполняющихся на хостовой системе 10 в соответствии с некоторыми вариантами осуществления настоящего изобретения. На фиг. 3 показан примерный вариант осуществления, конфигурированный с возможностью выполняться в среде виртуализации; специалисту в этой области техники может быть понятно, что проиллюстрированный вариант

осуществления может быть модифицирован, чтобы выполняться непосредственно на хостовой системе 10, а не внутри виртуальной машины 32. Фиг. 3 представлена с точки зрения уровней привилегий процессора, которые также известны в данной области техники как слои или защитные кольца. В некоторых вариантах осуществления, каждый такой слой или защитное кольцо характеризуется набором инструкций, который разрешено выполнять программному объекту, выполняющемуся на соответствующем уровне привилегий процессора. Когда программный объект пытается выполнить инструкцию, которая не разрешена в пределах соответствующего уровня привилегий, эта попытка может вызвать событие процессора, такое как исключение, ошибка или событие выхода виртуальной машины. В некоторых вариантах осуществления переключение между уровнями привилегий может быть достигнуто с помощью набора выделенных инструкций. Такие примерные инструкции включают в себя SYSCALL/SYSENTER, которые переключают из уровня пользователя в уровень ядра, SYSRET/SYSEXIT, которые переключают из уровня ядра в уровень пользователя, VMCALL, который переключает из уровня пользователя или ядра в корневой уровень, и VMRESUME, который переключает из корневого уровня в либо уровень, ядра, либо уровень пользователя.

[0034] Большинство компонентов операционной системы 34 выполняется на уровне привилегий процессора, известном в данной области техники как уровень или режим ядра (например, кольцо 0 на платформах Intel). Приложение 42g выполняется с меньшей привилегией процессора, чем OS 34 (например, кольцо 3, или режим пользователя). В одном из вариантов осуществления, поддерживающем виртуализацию, гипервизор 30 берет на себя управление процессора 12 на наиболее привилегированном уровне, также известном как корневой уровень или корневой режим (например, кольцо -1 или VMXroot на платформах Intel®), предоставляя виртуальную машину 32 для OS 34 и других программных объектов, таких как приложения 42g.

[0035] В некоторых вариантах осуществления части приложения 44 безопасности могут выполняться с привилегией процессора на уровне пользователя, т.е. на том же уровне, что и приложение 42g. Например, такие части могут включать в себя графический пользовательский интерфейс, информирующий пользователя о любых угрозах вредоносных программ или безопасности, обнаруженных на соответствующей виртуальной машине, и получающий входные данные от пользователя с указанием, например, на требуемую опцию конфигурации для приложения 44. Другим примером компонента, выполняющегося на уровне пользователя, является средство 50a оценки сущностей на уровне пользователя, которое работает, как описано ниже. В некоторых вариантах осуществления часть средства 50a оценки сущностей на уровне пользователя может работать в приложении 44 безопасности, в то время как другая часть (например, зацепляющий модуль) может работать в оцениваемом приложении, таком как приложение 42g. Другие части приложения 44 могут выполняться на уровне привилегий ядра. Например, приложение 44 может установить антивредоносный драйвер 36, модуль 37 управления сущностями и классифицирующий механизм 38, причем все они работают в режиме ядра. Драйвер 36 обеспечивает функциональные возможности для антивредоносного приложения 44, например, для сканирования памяти на вредоносные сигнатуры и/или для выявления указывающего на вредоносность поведения процессов и/или других программных объектов, выполняющихся на OS 34. В некоторых вариантах осуществления антивредоносный драйвер 36 содержит средство 50b оценки сущностей на уровне ядра, которое работает, как описано ниже.

[0036] В некоторых вариантах осуществления модуль 37 управления сущностями

управляет коллекцией программных сущностей, выполняющихся в хостовой системе 10 (или VM 32). В некоторых вариантах осуществления коллекция содержит все сущности, оцениваемые на вредоносные программы с помощью модулей оценки сущностей, таких как 55a-b. Для управления коллекции, модуль 37 может добавлять и/или удалять сущности из коллекции в ответ на выявление возникновения событий жизненного цикла, таких как события запуска и/или завершения сущностей, как показано более подробно ниже. Модуль 37 может дополнительно определять взаимотношения между сущностями, например, определять порожденные сущности (например, порожденные процессы) родительской сущности, и/или определять, внедрила ли выбранная сущность программный объект, например, библиотеку, в другую сущность, или является ли выбранная сущность объектом внедрения другой программной сущности. Порожденная сущность представляет собой исполняемую сущность, созданную другой исполняемой сущностью, называемой родительской сущностью, при этом порожденная сущность выполняется независимо от родительской сущности. Примерными порожденными сущностями являются порожденные процессы, например, созданные с помощью функции CreateProcess в Windows® OS, или с помощью вилочного механизма в Linux®. Внедрение кода - это общий термин, используемый в данной области техники для указания на семейство способов введения последовательности кода, например, динамически связываемой библиотеки (DLL), в область памяти существующего процесса, чтобы изменить первоначальную функциональность соответствующего процесса. Для выполнения таких задач, как выявление запуска процесса и/или выявление внедрения кода, модуль 37 может использовать любой способ, известный в данной области техники, например, вызывать или перехватывать определенные функции операционной системы. Например, в системе под управлением Windows® OS, модуль 37 может регистрировать обратный вызов PsSetCreateProcessNotifyRoutine для выявления запуска нового процесса, и/или перехватывать функцию CreateRemoteThread для выявления выполнения внедренного кода.

[0037] На фиг. 4 показана примерная последовательность этапов, выполняемых модулем 37 управления сущностями, в соответствии с некоторыми вариантами осуществления настоящего изобретения. В последовательности этапов 250-252, модуль 37 перехватывает событие жизненного цикла сущности с использованием, например, описанных выше способов. Если такое событие произошло, этап 254 идентифицирует сущность, инициирующую соответствующее событие. Этап 258 может включать в себя определение уникального индикатора идентификации сущности (EID) соответствующей сущности; такой индикатор может быть использован при квалификации соответствующей сущности, как показано ниже. Этап 256 определяет, содержит ли событие запуск новой сущности (например, нового процесса), и если нет, модуль 37 переходит к этапу 260. Если событие содержит запуск, на этапе 258, модуль 37 может добавлять инициирующую сущность в коллекцию оцениваемых сущностей. Этап 260 содержит определение, содержит ли событие родительскую сущность, создающую порождаемую сущность, и если нет, модуль 37 может перейти к этапу 264. Если да, на этапе 262, модуль 37 может добавить соответствующую порождаемую сущность в коллекцию оцениваемых сущностей. Этап 262 может дополнительно включать в себя определение EID порождаемой сущности и регистрацию отношения между инициирующей сущностью и порождаемой сущностью как отношение филиации (родитель-порождение).

[0038] В некоторых вариантах осуществления этап 264 определяет, содержит ли событие внедрение кода, и если нет, модуль 37 может перейти к этапу 268. Если да,

модуль 37 может идентифицировать исходную сущность и целевую сущность внедрения кода, причем исходная сущность внедряет код в целевую сущность. На этапе 266, модуль 37 может зарегистрировать отношение типа кода внедрения между исходной сущностью и целевой сущностью.

[0039] На этапе 268, модуль 37 управления сущностями определяет, содержит ли событие завершение инициирующей сущности, и если нет, модуль 37 возвращается к этапу 250. В некоторых вариантах осуществления, сущность считается завершенной, если все компоненты соответствующей сущности закончили выполнение. Например, процесс завершен, когда все потоки соответствующего процесса закончили выполнение. Если событие содержало завершение инициирующей сущности, на этапе 270, модуль 37 может определять набор сущностей-потомков инициирующей сущности. В некоторых вариантах осуществления сущности-потомки инициирующей сущности содержат порожденные сущности соответствующей сущности, а также порожденные сущности порожденных сущностей, через множество поколений. В некоторых вариантах сущности-потомки могут включать в себя целевые сущности, содержащие код, внедренный инициирующей сущностью, а также сущности, целевые для целевых сущностей, рекурсивно. На этапе 272, модуль 37 может определять, завершены ли все сущности набора сущностей-потомков, и если нет, то выполнение возвращается к этапу 250. Если все потомки завершены, на этапе 274, модуль 37 управления сущностями может удалить инициирующую сущность из коллекции оцениваемых сущностей.

[0040] В некоторых вариантах осуществления, классифицирующий механизм 38 конфигурирован с возможностью получать данные из множества модулей средства оценки сущностей, таких как средства 50a-b оценки сущностей, причем данные определены для оцениваемой программной сущности, и определять, является ли соответствующая сущность вредоносной в соответствии с соответствующими данными. В некоторых вариантах осуществления, программные сущности, анализируемые классифицирующим механизмом 38, включают в себя, помимо прочего, исполняемые объекты, такие как процессы и потоки выполнения. Процесс является экземпляром компьютерной программы, например, приложения или части операционной системы, и характеризуется тем, что он имеет, по меньшей мере, поток выполнения и секцию виртуальной памяти, назначенной для него операционной системой, при этом соответствующая секция содержит исполняемый код. В некоторых вариантах осуществления, оцениваемые программные сущности могут существенно различаться по объему и сложности, например, от отдельных потоков до отдельных приложений, до целых экземпляров операционных систем и/или виртуальных машин.

[0041] На фиг. 5 показан примерный классифицирующий механизм 38, получающий множество индикаторов 52a-d оценки, причем каждый индикатор 52a-d определен средством оценки сущностей. На фиг. 5 такие средства оценки включают в себя средство 50a оценки сущностей на уровне пользователя, средство 50b оценки сущностей на уровне ядра и средство 50c оценки системных вызовов, помимо прочего. Каждый такой модуль средства оценки может выполняться независимо от других средств оценки, при этом каждый из них может определять множество отдельных индикаторов оценки сущности оцениваемой программной сущности. В системах, использующих аппаратную виртуализацию, некоторые индикаторы оценки, такие как индикаторы 52a-c на фиг. 5, определяются компонентами, выполняющимися в VM 32, в то время как другие индикаторы оценки, такие как 52d, определяются компонентами, выполняющихся вне VM 32 (например, механизмом 40 самоанализа памяти). В некоторых вариантах осуществления каждый индикатор 52a-d оценки содержит индикатор идентификации

сущности, позволяя механизму 38 однозначно ассоциировать соответствующий индикатор оценки с программной сущностью, для которой он был определен.

[0042] Некоторые индикаторы оценки могут быть указывающими вредоносность, то есть, могут указывать на то, что оцениваемая сущность является вредоносной. Некоторые индикаторы оценки сами могут не быть указывающими на вредоносность, но могут указывать на вредоносность в сочетании с другими индикаторами оценки. Каждый индикатор 52a-d оценки может быть определен в соответствии с отдельным способом и/или критерием. Примерные критерии оценки включают в себя поведенческие критерии, такие, как определение, выполняет ли оцениваемая сущность определенное действие, например, запись в файл на диске, редактирование ключа системного реестра VM 32 или запись в страницу памяти, принадлежащую защищаемому программному объекту. Другой примерный критерий может включать в себя определение того, содержит ли секция памяти, принадлежащая оцениваемой сущности, указывающую на вредоносность сигнатуру.

[0043] Для иллюстрации работы средств 50a-c оценки сущностей, на фиг. 6 показан примерный поток выполнения набора программных сущностей 70a-b в соответствии с некоторыми вариантами осуществления настоящего изобретения. Для простоты, выбранные сущности 70a-b являются процессами, выполняющимися в экземпляре Windows® OS; сходные схемы могут быть представлены для других операционных систем, таких как Linux, например. Сплошные стрелки представляют поток выполнения при отсутствии средств оценки сущностей (например, при отсутствии приложения 44 безопасности). Пунктирные стрелки представляют модификации в потоке из-за наличия средств 50a-c оценки сущностей, выполняющихся в соответствии с некоторыми вариантами осуществления настоящего изобретения.

[0044] Процесс 70a загружает множество динамически связываемых библиотек (DLL) 72a-c; в примере на фиг. 6, DLL 72c внедряется в процесс 70a посредством (возможно вредоносного) процесса 70b. Когда процесс 70a (или одна из его загруженных DLL) выполняет инструкцию, вызывающую некоторую функциональность системы, например, чтобы написать что-нибудь в файл на диске или изменить раздел реестра, соответствующие инструкции вызывают API (программный интерфейс приложения) режима пользователя, такой как KERNEL32.DLL или NTDLL.DLL. В примере на фиг. 6, соответствующий вызов API в режиме пользователя перехватывается и анализируется с помощью поведенческого фильтра 50a на уровне пользователя. Такие перехваты могут быть обеспечены с помощью такого способа, как внедрение DLL или зацепление, помимо прочего. Зацепление - это общий термин, используемый в данной области техники для способа перехвата вызовов функций или сообщений, или событий, передаваемых между программными компонентами. Один примерный способ зацепления содержит изменение точки входа целевой функции путем вставки инструкции, перенаправляющей выполнение на вторую функцию. После такого зацепления может быть выполнена вторая функция вместо или до целевой функции. В примере на фиг. 6, антивредоносный драйвер 36 может зацепиться с определенными функциями KERNEL32.DLL или NTDLL.DLL, чтобы проинструктировать соответствующие функции для перенаправления выполнения на фильтр 50a. Таким образом, фильтр 50a может выявить, что процесс 70a пытается выполнять определенное поведение, идентифицированное в соответствии с зацепленной функцией. Когда фильтр 50a выявляет такое поведение, фильтр 50 может сформулировать индикатор 52a оценки и передать индикатор 52a в классифицирующий механизм 38 (см., например, фиг. 5).

[0045] В типичном потоке выполнения, функция API в режиме пользователя,

вызванная сущностью 70a, может запросить услугу от ядра операционной системы. В некоторых вариантах осуществления такие операции осуществляются путем выдачи системного вызова, например, SYSCALL и SYSENTER на платформах x86. В примере на фиг. 6, такие системные вызовы перехватываются средством 50c оценки системных вызовов. В некоторых вариантах осуществления такой перехват включает в себя, например, модификацию подпрограммы обработчика системных вызовов путем изменения величины, хранящейся в модельно зависимом реестре (MSR) процессора 12, который эффективно перенаправляет выполнение на фильтр 50c. Такие способы известны в данной области техники как зацепление MSR, при этом они могут обеспечивать возможность средству 50c оценки системных вызовов выявлять, что оцениваемый процесс пытается выполнять определенные системные вызовы. Когда такие системные вызовы перехватываются, фильтр 50c системных вызовов может формулировать индикатор 52c оценки сущностей и передавать этот индикатор 52c в классифицирующий механизм 38.

[0046] После системного вызова, управление процессора обычно передается ядру OS 34. В некоторых вариантах осуществления средство 50b оценки сущностей на уровне ядра конфигурировано для перехвата определенных операций ядра OS, и, таким образом, определения, что оцениваемый процесс пытается выполнять определенные операции, которые могут быть вредоносными. Для перехвата таких операций, некоторые варианты осуществления могут применять набор фильтрующих механизмов, встроенных в OS 34 и предоставляемых ею. Например, в Windows OS, FltRegisterFilter может использоваться для перехвата таких операций, как создание, открытие, запись и удаление файла. В другом примере средство 50b оценки может использовать ObRegisterCallback для перехвата создания или дублирования операций описателей объектов или PsSetCreateProcessNotifyRoutine для перехвата создания новых процессов. В еще одном примере операции реестра Windows, такие как создание и настройка ключей/значений реестра, могут быть перехвачены с помощью CmRegisterCallbackEx. Подобные механизмы фильтрации известны в данной области техники для других операционных систем, таких как Linux®. Когда средство 50b оценки сущностей режима ядра перехватывает такие операции, средство 50b оценки сущностей может формулировать индикатор 52b оценки сущностей и передавать индикатор 52b в классифицирующий механизм 38.

[0047] Для передачи данных, таких как индикаторы 52a-c оценки сущностей, из средств 50a-c оценки в классифицирующий механизм 38, специалист в данной области техники может использовать любой способ связи между процессами. Например, для связи между компонентами режима пользователя и режима ядра, средства 50a-c оценки и механизм 38 могут быть конфигурированы с возможностью использовать общую секцию памяти. Когда необходим обмен данными между компонентами, выполняющимися в VM 32, и компонентами, выполняющимися вне соответствующей VM, такая связь может быть осуществлена с использованием любого способа, известного в области виртуализации. Например, для передачи индикатора 52d оценки из механизма 40 самоанализа памяти в классифицирующий механизм 38, некоторые варианты осуществления используют механизм прерывания внедрения, чтобы сигнализировать классифицирующему механизму 38, что данные передаются снаружи соответствующей VM. Фактические данные могут быть переданы, например, через общую секцию памяти, описанную выше.

[0048] На фиг. 7 показана примерная последовательность этапов, выполняемых средством оценки сущностей, таким как средства 50a-c оценки и/или механизм 40 самоанализа памяти на фиг. 4-5, в соответствии с некоторыми вариантами осуществления

настоящего изобретения. В последовательности этапов 302-304, средство оценки сущностей ожидает возникновения события триггера в хостовой системе 10 и/или виртуальной машине 32. Примерные события триггера включают в себя, помимо прочего, программную сущность, выполняющую определенное поведение, например, выдачу конкретной инструкции процессора, попытка использования конкретной части аппаратного оборудования, такого как устройства 20 или сетевого адаптера (сетевых адаптеров) 22, или попытка записи на защищенную страницу памяти. Например, событие триггера для средства 50c оценки может включать в себя программную сущность, выдающую системный вызов (например, SYSENTER). Другой пример события триггера для средства 50d оценки может включать в себя приложение, вызывающее функцию UrlDownloadToFile API. Для выявления возникновения события триггера, соответствующей соответствующее средство оценки сущностей может использовать любой способ, известный в данной области, такой как внедрение кода и зацепление MSR, помимо прочего. Некоторые примеры перехвата события триггера описаны выше по отношению к фиг. 6.

[0049] При выявлении события запуска, на этапе 306, средство оценки сущностей может идентифицировать программную сущность (например, процесс), вызывающую соответствующее событие триггера. В некоторых вариантах осуществления настоящего изобретения средство оценки сущностей может определять идентичность программной сущности из структуры данных, используемой OS 34 для представления каждого процесса и/или потока, выполняющегося в данный момент. Например, в Windows, каждый процесс представляется как блок исполнительного процесса (EPROCESS), который содержит, помимо прочего, описатели для каждого из потоков соответствующего процесса, а также уникальный идентификатор (ID) процесса, который позволяет OS 34 идентифицировать соответствующий процесс из множества выполняющихся процессов. Сходные представления процесса/потока доступны для других операционных систем, таких как Linux.

[0050] На этапе 308, средство оценки сущностей может формулировать индикатор оценки, в том числе идентификатор (например, ID процесса) соответствующей программной сущности, и индикатор типа действия/события, выполняемого соответствующей программной сущностью и перехваченного на этапах 302-304. В некоторых вариантах осуществления изобретения средство оценки сущностей может определять тип действия и/или поведения соответствующей программной сущности, на основе параметров перехваченного события триггера. В качестве примера работы, когда процесс пытается загрузить файл из Интернета, средство 50a оценки сущностей на уровне пользователя может перехватить эту попытку. Помимо идентификации, какой процесс выполняет это действие, средство 50a оценки может также определять, помимо прочего, тип действия (загрузка файла), IP-адрес, с которого загружается файл, и расположение загруженного файла на диске. Такие данные могут быть селективно включены в индикатор оценки, что позволяет классифицирующему механизму 38 определять, что сущность X выполнила действие Y с параметрами Z. На этапе 310, средство оценки сущностей передает индикатор оценки в классифицирующий механизм 38.

[0051] В некоторых вариантах осуществления классифицирующий механизм 38 и/или модуль 37 управления сущностями поддерживает централизованную базу данных оцениваемых программных сущностей, таких как процессы и потоки, которые выполняются на хостовой системе 10 (или VM 32). На фиг. 8 показан набор оцениваемых сущностей 70c-e, каждая из которых представлена в виде примерного оценивающего

сущности объекта (ESO) 74a-c, соответственно. Каждый ESO содержит множество полей данных, некоторые из которых показаны на фиг. 8. Такие поля могут включать в себя уникальный идентификатор сущности (EID) 76a, множество показателей 76b оценки и совокупный показатель 76d. В некоторых вариантах осуществления показатели 76b оценки определяются механизмом 38 в соответствии с индикаторами 52a-d оценки, полученными от отдельных средств оценки сущностей. Каждый такой показатель может быть определен согласно критерию оценки, идентифированному индикаторами 76c. В некоторых вариантах осуществления показатели 76b оценки имеют взаимно однозначное соответствие с критериями 76c оценки, так что каждый показатель приписывается согласно соответствующему критерию. Например, конкретный критерий C_k может включать в себя определение того, загружает ли оцениваемая сущность файл из компьютерной сети, такой как Интернет. В одном таком примере, соответствующий показатель S_k может быть предоставлен, только если оцениваемая сущность пытается выполнить загрузку.

[0052] В некоторых вариантах осуществления ESO 74a может дополнительно содержать набор флагов 76e. Некоторые флаги 76e могут представлять собой бинарные индикаторы (например, 0/1, да/нет). В одном таком примере, флаг указывает на то, удовлетворяет ли соответствующая оцениваемая сущность E_1 конкретному критерию оценки (например, является ли E_1 исполняемым файлом, загруженным из Интернета, выполняется ли E_1 в режиме командной строки и т.д.). Другой примерный флаг указывает на классификацию сущности E_1, например, индикатор того, что E_1 принадлежит к определенной категории объектов, таких как троянские вредоносные программы, объекты браузера, приложения-программы чтения PDF и т.д. Пример использования флагов включает в себя ситуацию, в которой обновление показателя S_i оценки сущности E_1 инициирует обновление другого показателя S_j оценки сущности E_1 (см. ниже). Флаги могут использоваться для включения и выключения таких механизмов совместного обновления. Например, может быть известно, что, когда E_1 удовлетворяет критерию C_i оценки (например, если сущность выполняет определенное действие), сущность E_1, вероятно, также удовлетворяет критерию C_j. Таким образом, флаг F_i, указывающий на соединение $<C_i, C_j>$, может быть установлен для сущности E_1, инициируя обновление показателя S_j, когда показатель S_i обновляется.

[0053] ESO 74a может дополнительно включать в себя индикатор 76f завершения, указывающий на то, активна ли или завершена в данный момент соответствующая сущность. Такие индикаторы завершения могут позволять классифицирующему механизму 38 вести записи и/или обновлять показатели завершенных сущностей. ESO 74a может дополнительно включать в себя набор идентификаторов программных сущностей, связанных с соответствующей сущностью E_1, Примеры таких связанных программных сущностей могут включать в себя родительскую сущность E_1, обозначаемую идентификатором 76g, и набор порожденных сущностей E_1, обозначаемых идентификатором 76H. ESO 74a может дополнительно включать в себя набор индикаторов целевых сущностей внедрения (элементы 76j), которые идентифицируют программные сущности, в которые E_1 внедрила код, и дополнительно набор индикаторов исходных сущностей внедрения (элементы 76k), которые идентифицируют программные сущности, которые внедрили код в E_1.

[0054] В некоторых вариантах осуществления классификация оцениваемых программных сущностей протекает в соответствии с набором значений показателя и далее в соответствии с дополнительными параметрами. Фиг. 9 иллюстрирует такие данные, причем множество значений показателя обозначено элементом 80. Значения показателя индексируются своими соответствующими критериями оценки C_1, … C_n. Каждое такое значение может представлять, например, предварительно определенное количество пунктов, которые получает оцениваемая сущность, если она удовлетворяет соответствующему критерию оценки (например, если она загружает файл из Интернета, если она записывает в документ MS Word® и т.д.).

[0055] Примерные параметры, управляющие квалификацией, включают в себя набор весов 82a инициализации, набор весов 82b распространения, набор весов 82c нового экземпляра, набор весов 82d исключения, а также набор индуцированных флагами весов 82e. Веса 82a-e индексируются критериями оценки C_1, … C_n. Некоторые типы весов находятся во взаимно однозначном соответствии с критериями оценки, так что имеется одно значение веса w_i для каждого C_i. Другие типы весов находятся в соответствии «один ко многим» с критериями оценки. Одним из таких примеров является веса 82d исключения на фиг. 9, где может иметься множество весов w_{ij}, соответствующих определенному критерию оценки C_i. Веса могут быть сгруппированы по классам или категориям сущностей, как это показано посредством примера на фиг. 9; например, может иметься первое значение веса, применимое к приложениям обработки текстов (например, MS Word®), второе значение веса (возможно, отличное от первого), применимое к веб-браузерам (например, Firefox® и MS Internet Expolrer®), и третье значение веса, применимое к приложениям файлового менеджера (например, Windows Explorer®). Различие среди различных категорий сущностей может быть полезным, поскольку некоторые критерии оценки могут быть в большей степени указывать на вредоносность для одной категории сущностей, чем для других. В более общем случае, каждый классифицирующий вес может индексироваться набором $<C_i, E_k, \ldots>$, где C_i обозначает конкретный критерий оценки и где E_k обозначает конкретную оцениваемую сущность. Фактический формат данных для хранения и доступа к классифицирующим весам 82a-e может быть разным в зависимости от вариантов осуществления. Веса 82a-e могут храниться, помимо прочего, в виде матриц, списков, реляционных баз данных (RDB) или расширяемого языка разметки (XML) структур. Примерное использование весов для квалификации обсуждается ниже.

[0056] Значения 80 показателя и/или веса 82a-e определены предварительно, например, оператором. В некоторых вариантах осуществления такие значения могут изменяться во времени, при этом они могут быть скорректированы с целью оптимизации выявления вредоносных программ. Обновленные значения показателя и/или значения веса могут передаваться с сервера безопасности в хостовую систему 10 в виде периодических обновлений и/или обновлений по требованию программного обеспечения (см. ниже, по отношению к фиг. 10-11).

[0057] На фиг. 10 показана примерная последовательность этапов, выполняемых классифицирующим механизмом 38 в соответствии с некоторыми вариантами осуществления настоящего изобретения. На этапе 302, механизм 38 получает индикатор оценки сущностей от средства оценки сущностей, например, от одного из средств 50a-c оценки на фиг. 5. В некоторых вариантах осуществления, реализующих аппаратную виртуализацию, механизм 38 может получать соответствующий индикатор оценки сущностей от компонента, выполняющегося вне соответствующей виртуальной машины

(например, механизма 40 самоанализа памяти на фиг. 5). На этапе 304, механизм 38 может идентифицировать программную сущность, для которой был определен соответствующий индикатор оценки сущностей, например, в соответствии с ID сущности, встроенной в соответствующий индикатор оценки (см. выше, по отношению к фиг. 7).

[0058] Затем классифицирующий механизм 38 выполняет блок этапов 318-332 для сущности E, идентифицированой на этапе 304, а также для других сущностей, связанных с E. Такие связанные сущности могут включать в себя родительские и порожденные сущности сущности E, целевые сущности внедрения, в которые E внедрила код, и исходные сущности внедрения, которые внедрили код в E, помимо прочего. Таким образом, каждый раз, когда механизм 38 получает индикатор оценки (с указанием, например, на то, что сущность E выполнила определенное действие), блок 318-332 может выполняться несколько раз, обновляя не только показатели оценки сущности E, но и показатели оценки сущностей, связанных с E. В некоторых вариантах осуществления блок 318-332 выполняется один раз для E и один раз для каждой сущности E*, связанной с E. В альтернативных вариантах осуществления блок 318-332 выполняется рекурсивно, пока не удовлетворяется какой-нибудь критерий сходимости. Примерный критерий сходимости содержит проверку того, изменяются ли показатели оценки E и/или E * между последовательными выполнениями блока 318-332, и выход, если такого изменения нет. В примерном алгоритме на фиг. 10, переменная X используется для указания на то, что сущность в данный момент совершает обновления показателя. На этапе 316, X устанавливается для сущности E, идентифицированной на этапе 304.

[0059] На этапе 318, механизм 38 обновляет показатели оценки сущности X (например, сущностей 76b на фиг. 8). В некоторых вариантах осуществления, обновление показателя оценки содержит замену записанного значения соответствующего показателя оценки на новое значение:

$$S_k^{(X)} \rightarrow S_k^{(X)} + \Delta S_k, \qquad [1]$$

где $S_k^{(X)}$ обозначает показатель оценки, определенный для сущности X согласно критерию оценки C_k, и ΔS_k обозначает приращение, которое может быть положительным или отрицательным (в некоторых вариантах показатели оценки могут уменьшаться при обновлении).

[0060] В некоторых вариантах осуществления приращение показателя ΔS_k определяется классифицирующим механизмом 38 согласно индикатору оценки, полученному на этапе 312. Соответствующий индикатор может включать в себя показатель и/или указывать на критерий оценки, использованный при определении соответствующего индикатора. В некоторых вариантах осуществления классифицирующий механизм 38 определяет приращение показателя ΔS_k согласно значению 80 показателя, который соответствует соответствующему критерию оценки C_k (см. фиг. 9), например:

$$\Delta S_k = V_k, \qquad [2]$$

где V_k обозначает значение 80 показателя, назначенного критерию C_k. В одном таком примере, в котором критерий C_k содержит определение того, загружает ли сущность X объект из сети, и в котором V_k=20, показатель оценки $S_k^{(X)}$ будет увеличен на 20 пунктов каждый раз, когда объект X выполняет загрузку. В некоторых вариантах

осуществления, $\Delta S_k = \varepsilon V_k$, где ε представляет собой вес бинарного исключения (см., например, элементы 82d на фиг. 9), что вызывает обновление показателя S_k только для подмножества оцениваемых сущностей. Такие веса исключения являются полезными, например, чтобы различать между различными типами оцениваемых сущностей. Например, браузеру должен быть разрешен доступ к неограниченному количеству IP-адресов без вызова подозрений в отношении вредоносности; критерий оценки, содержащий выявление доступа в Интернет, может быть эффективно выключен для объектов браузера путем установки веса исключения, равным 0, для сущностей типа браузера, сохраняя при этом его активным (ε=1) для других типов сущностей.

[0061] В некоторых вариантах осуществления приращение показателя ΔS_k, используемое при обновлении показателя оценки сущности X, определяется в соответствии с показателем оценки, определенным для сущности X*, связанной с X, то есть показатели могут распространяться от одной сущности к связанной сущности, например, от порождения к родителю, или от цели внедрения к источнику внедрения. В одном таком примере, действие, выполняемое порождаемым процессом, может инициировать обновление не только показателя сущности, выполняющей действие (порожденный процесс), но и показателя родительского процесса соответствующего порождаемого процесса. Такие обновления показателя могут включать в себя вычисление приращения показателя согласно:

$$\Delta S_k = w_k S_k^{(X^*)}, \qquad [3]$$

где w_k обозначает численный зависимый от критерия вес, указывающий на степень, с которой показатель сущности X* влияет на показатель сущности X. Веса могут включать в себя веса 82b распространения (фиг. 9). Некоторые варианты осуществления различают среди разнообразия таких весов распространения, например веса, используемые для распространения показателей от порождаемого сущности к родительской сущности, могут отличаться по величине от весов, используемых для распространения показателей от родительской сущности к порождаемой сущности. Аналогичным образом, веса, используемые для распространения показателей от порождаемой сущности к родительской сущности, могут отличаться по величине от весов, используемых для распространения показателей от сущности, подлежащей внедрению кода, к сущности, выполняющей внедрение кода. В некоторых вариантах осуществления показатели могут распространяться от активных сущностей к завершенным сущностям. Например, действие порождаемого процесса может увеличивать показатель родительского процесса, даже если родительский процесс завершен.

[0062] В некоторых вариантах осуществления сущность X* в уравнении [3] является другим экземпляром сущности X. Например, X и X* могут быть копиями того же самого процесса или потока, которые выполняются параллельно. В таких случаях вес w_k может быть новым весом экземпляра (например, элемент 82c на фиг. 9), или весом инициализации (например, элемент 82a). В некоторых вариантах осуществления, когда запускается новый экземпляр X' сущности X, механизм 38 может обновлять некоторые или все показатели оценки существующей сущности X, используя новые веса экземпляра w_k для распространения показателей от X к X'. Аналогичным образом, когда X' запускается, механизм 38 может обновлять некоторые или все показатели оценки X', используя веса инициализации w_k для распространения показателей от уже выполняющейся сущности X к новой сущности X'.

[0063] В некоторых вариантах осуществления обновление показателя оценки S_k может инициировать обновление отдельного показателя оценки S_m соответствующей сущности. Например,

$$S_k^{(X)} \rightarrow S_k^{(X)} + V_k \text{ инициирует } S_m^{(X)} \rightarrow S_m^{(X)} + F^{(X)} f_{km} V_m, \qquad [4]$$

где $F^{(X)}$ представляет собой флаг, установленный для сущности X (см., например, элементы 76e на фиг. 8), при этом флаг указывает на связь между критериями оценки C_k и C_m, и где f_{km} представляет собой вызванный флагом вес (см., например, элемент 82e на фиг. 9), указывающий на степень, с которой обновление S_k влияет на обновление S_m сущности X.

[0064] На этапе 320, классифицирующий механизм 38 может обновлять флаги сущности X (см. обсуждение флагов выше, по отношению к фиг. 8), согласно индикатору оценки, полученному на этапе 312. Флаги могут быть установлены, чтобы активировать и/или дезактивировать механизмы совместного обновления показателя, такие, как описано выше по отношению к уравнению [4]. В одном таком примере оцениваемая сущность может быть идентифицирована как приложение веб-браузера согласно индикатору оценки (этап 312); такая идентификация должна указывать классифицирующему механизму 38 не классифицировать соответствующую сущность для будущих загрузок из Интернета. Это может быть достигнуто путем установки значения конкретного флага F равным 0 для соответствующей сущности, причем флаг F указывает классифицирующему механизму 38 обновить показатель оценки соответствующей сущности, когда сущность загружает объект из Интернета.

[0065] На этапе 322, классифицирующий механизм 38 может определять совокупный показатель сущности X путем объединения отдельных показателей оценки, определенных для соответствующего процесса, например, в виде суммы:

$$A^{(X)} = \sum_k S_k^{(X)}. \qquad [5]$$

[0066] На этапе 324, механизм 38 может сравнивать совокупный показатель с предварительно определенным пороговым значением. Когда совокупный показатель не превышает пороговое значение, классифицирующий механизм 38 может перейти к этапу 326, описанному ниже. В некоторых вариантах осуществления изобретения пороговое значение может быть установлено равным значению, определенному в соответствии с вводом пользователя. Такие пороговые значения могут отражать соответствующие предпочтения безопасности пользователя. Например, когда пользователь предпочитает строгий режим безопасности, пороговое значение может быть установлено равным относительно низкому значению; когда пользователь предпочитает более толерантную настройку безопасности, пороговое значение может быть установлено равным относительно высокому значению. В некоторых вариантах осуществления изобретения пороговое значение может быть получено от удаленного сервера безопасности, как описано ниже со ссылками на фиг. 10-11.

[0067] В некоторых вариантах осуществления, на этапах 322-324, классифицирующий механизм 38 может определять множество совокупных показателей и сравнивать каждый совокупный показатель с (возможно, отдельным) пороговым значением. Каждый такой совокупный показатель может быть определен согласно отдельному подмножеству показателей оценки. В примерном варианте осуществления каждое такое подмножество показателей, и их соответствующее подмножество критериев оценки,

может представлять конкретный класс или тип вредоносных программ (например, троянов, руткитов и т.д.). Это может позволить механизму 38 выполнять классификацию выявленных вредоносных программ. В другом варианте осуществления, классифицирующий механизм 38 применяет множество пороговых значений для классификации сущностей выполнения в соответствии с различными степенями вредоносности (например, чистые, подозрительные, опасные и критические).

[0068] Когда совокупный показатель превышает пороговое значение, на этапе 326, механизм 38 может принять решение о том, что оцениваемый процесс является вредоносным и может предпринять антивредоносное действие. В некоторых вариантах осуществления, такое антивредоносное действие может включать в себя, помимо прочего, завершение оцениваемого процесса, помещение оцениваемого процесса в карантин и удаление или дезактивирование ресурса (например, файла или секции памяти) оцениваемого процесса. В некоторых вариантах осуществления антивредоносное действие может дополнительно включать в себя предупреждение пользователя хостовой системы 10 и/или предупреждение системного администратора, например, путем отправки сообщения системному администратору через компьютерную сеть, соединенную с хостовой системой 10 через сетевой адаптер (адаптеры) 22. В некоторых вариантах осуществления антивредоносное действие может также содержать отправку отчета о безопасности в удаленный сервер безопасности, как описано ниже со ссылками на фиг. 10-11.

[0069] В последовательности этапов 328-330, механизм 38 может идентифицировать сущность X*, связанную с X, при этом показателям X* необходимо обновление вслед за текущими обновлениями показателя сущности X. Например, X* может быть родительской или порожденной сущностью X. В некоторых вариантах осуществления сущности X* могут быть идентифицированы в соответствии с полями 76g-k ESO сущности X (см., например, фиг. 8). Когда такие сущности X* не существуют или когда все такие сущности X* уже были рассмотрены для обновления показателя, механизм 38 возвращается к этапу 312. Когда имеется, по меньшей мере, сущность X*, на этапе 332 классифицирующий механизм делает X* текущей сущностью и возвращается к этапу 318.

[0070] Примерный классифицирующий механизм 38, изображенный на фиг. 3-4, работает в VM 32 на уровне привилегий процессора OS (например, в режиме ядра). В альтернативных вариантах осуществления, классифицирующий механизм 38 может выполняться в VM 32 в режиме пользователя, или даже вне VM 32, на уровне привилегий процессора гипервизора 30.

[0071] В некоторых вариантах осуществления механизм 40 самоанализа выполняется, по существу, на том же уровне привилегий, что и гипервизор 30, и конфигурирован с возможностью выполнять самоанализ виртуальных машин, таких как VM 32 (фиг. 3). Самоанализ виртуальной машины, или программной сущности, выполняющейся на соответствующей VM, может включать в себя анализ поведения программной сущности, определение адресов памяти таких сущностей и/или доступ к этим адресам, ограничение доступа некоторых процессов к содержимому памяти, расположенной на таких адресах, анализ этого содержания и определение индикаторов оценки соответствующих сущностей (например, индикатор 52d на фиг. 5), помимо прочего.

[0072] В некоторых вариантах осуществления хостовая система 10 может быть конфигурирована для обмена информацией безопасности, такой как детали о событиях выявления вредоносных программ, с удаленным сервером безопасности. Фиг. 11 иллюстрирует такую примерную конфигурацию, в которой множество хостовых систем

10a-c, таких как вышеописанная система 10, соединены с сервером 110 безопасности через компьютерную сеть 26. В примерном варианте осуществления, хостовые системы 10a-c представляют собой отдельные компьютеры, используемые сотрудниками фирмы, в то время как сервер 110 безопасности может включать в себя компьютерную систему, конфигурированную сетевым администратором соответствующей фирмы для мониторинга вредоносных угроз или событий безопасности, происходящих на системах 10a-c. В другом варианте осуществления, например, в системе Инфраструктура как услуга (IAAS), в которой каждая хостовая система 10a-c представляет собой сервер, осуществляющий хостинг десятков или сотен виртуальных машин, сервер 110 безопасности может включать в себя компьютерную систему, конфигурированную с возможностью управлять антивредоносными операциями для всех таких виртуальных машин из центрального местоположения. В еще одном варианте осуществления сервер 110 безопасности может включать в себя компьютерную систему, конфигурированную поставщиком антивредоносного программного обеспечения (например, поставщиком приложения 44 безопасности, помимо прочего) с возможностью получать статистические и/или поведенческие данные о вредоносных программах, выявленных на различных системах в сети 26. Сеть 26 может включать в себя глобальную сеть, такую как Интернет, в то время как части сети 26 могут включать в себя локальные сети (LAN).

[0073] На фиг. 12 показан примерный обмен данными между хостовой системой 10 и сервером 110 безопасности в одном варианте осуществления, как показано на фиг. 11. Хостовая система 10 может быть конфигурирована с возможностью отправлять отчет 80 безопасности в сервер 110 и получать набор настроек 82 безопасности от сервера 110. В некоторых вариантах осуществления отчет 80 безопасности содержит индикаторы 52a-d оценки сущностей и/или показатели, определенные средствами 50a-c и/или 40 оценки сущностей, которые выполняются на хостовой системе 10, и/или совокупные показатели, определенные классифицирующим механизмом 38, помимо прочего. Отчет 80 безопасности может также включать в себя данные, идентифицирующие соответствующую систему 10 и оцениваемые сущности (например, ID сущностей, имена, пути, хэши или идентификаторы сущностей других видов), а также индикаторы, связывающие индикатор/показатель оценки сущностей с хостовой системой и сущностью, для которой он был определен. В некоторых вариантах осуществления отчет 80 может дополнительно включать в себя статистические и/или поведенческие данные в отношении сущностей, выполняющихся на хостовой системе 10. Система 10 может быть конфигурирована с возможностью посылать отчет 80 при выявлении вредоносных программ, и/или в соответствии с расписанием (например, каждые несколько минут, каждый час и т.д.).

[0074] В некоторых вариантах осуществления настройки 82 безопасности могут включать в себя рабочие параметры средств оценки сущностей (например, параметры фильтров 50a-c на фиг. 5) и/или параметры классифицирующего механизма 38. Примерные параметры механизма 38 включают в себя пороговое значение для принятия решения, является ли оцениваемый процесс вредоносным, а также значения 80 показателя и веса 82a-e, помимо прочего.

[0075] В некоторых вариантах осуществления сервер 110 выполняет алгоритм оптимизации для динамического регулирования таких параметров, чтобы максимально увеличить эффективность выявления вредоносных программ, например, для увеличения процента выявления при сведении ложных срабатываний к минимуму. Алгоритмы оптимизации могут получать статистические и/или поведенческие данные о различных сущностях, выполняющихся на множестве хостовых систем 10a-c, в том числе

индикаторы/показатели оценки сущностей, отчет о которых был передан классифицирующему механизму 38 различными средствами оценки сущностей, и определять оптимальные значения этих параметров. Эти значения затем передаются в соответствующие хостовые системы через сеть 26.

[0076] В одном таком примере оптимизации, изменение значений 80 показателя может эффективно изменять релевантность соответствующих критериев оценки по отношению друг к другу. Угрозы от вредоносных программ, как правило, происходят волнами, при которых большое количество компьютерных систем во всем мире поражаются тем же самым вредоносным агентом в течение короткого промежутка времени. Путем получения отчетов 80 безопасности в режиме реального времени от множества хостовых систем, сервер 110 безопасности может быть осведомлен о текущих угрозах вредоносных программ и может оперативно предоставлять оптимальные настройки 82 безопасности соответствующим хостовым системам, причем настройки 82 включают в себя, например, набор значений 80 показателя, оптимизированных для выявления текущих угроз вредоносных программ.

[0077] Примерные системы и способы, описанные выше, позволяют защитить хостовую систему, например, компьютерную систему, от вредоносных программ, таких как вирусы, троянские программы и программы-шпионы. Для каждой сущности из множества исполняемых сущностей, таких как процессы и потоки, выполняющиеся в данный момент на хостовой системе, классифицирующий механизм записывает множество показателей оценки, причем каждый показатель определяется согласно со своим собственным критерием оценки. В некоторых вариантах осуществления, оцениваемые программные сущности могут существенно различаться по своим объему и сложности, например, от отдельных потоков выполнения, до отдельных приложений, до целых экземпляров операционных систем и/или виртуальных машин.

[0078] Каждый раз, когда подвергаемая мониторингу сущность удовлетворяет критерию оценки (например, выполняет действие), соответствующий показатель сущности обновляется. Обновление показателя целевой сущности может инициировать обновления показателей других сущностей, связанных с целевой сущностью. Такие связанные сущности включают в себя, помимо прочего, порождение целевой сущности, родитель целевой сущности, сущности, в которые целевая сущность внедрила код, и сущности, которые внедрили код в целевую сущность.

[0079] Традиционные антивредоносные системы обычно классифицируют каждую сущность отдельно от других сущностей. Некоторые вредоносные программы могут пытаться избежать выявления путем разделения вредоносных действий среди нескольких различных агентов, таких как порожденные процессы вредоносного процесса, так что ни один из отдельных агентов не выполняет указывающее на вредоносность действие, достаточное для ее выявления. В отличие от этого, некоторые варианты осуществления настоящего изобретения распространяют показатели от одной сущности к другим связанным сущностям, подтверждая, таким образом, данные, указывающие на вредоносность, связанным сущностям. Распространение показателя может гарантировать, что обеспечено выявление по меньшей мере одного из агентов, вовлеченных во вредоносные действия.

[0080] В одной примерной стратегии избегания, вредоносный агент может создать множество порожденных процессов и выйти. Вредоносные действия могут быть разделены между порожденными процессами, так что действия отдельного порождения сами по себе не могут вызывать тревогу в отношении вредоносных программ. В некоторых вариантах осуществления настоящего изобретения, показатели могут

распространяться от одной сущности к другой, даже когда последняя завершена. Такая конфигурация может выявлять родительский процесс как вредоносный, даже если она не может выявить вредоносность порожденных процессов. Некоторые варианты осуществления поддерживают список сущностей, оцениваемых в данный момент; список может включать в себя как активные, так и завершенные сущности. Сущность может быть исключена из списка только тогда, когда все потомки соответствующей сущности завершены.

[0081] В традиционных антивредоносных системах, как правило, только один показатель записывается для каждой сущности. Посредством сохранения множества показателей каждой сущности, каждый из которых вычисляется в соответствии с его отдельным критерием, некоторые варианты осуществления настоящего изобретения обеспечивают распространение показателей среди связанных сущностей на основе каждого критерия. Такие показатели могут либо увеличиваться, либо уменьшаться при распространении, что обеспечивает возможность более точной оценки вредоносности на протяжении всего жизненного цикла каждой сущности с меньшим количеством ложных выявлений. В некоторых вариантах осуществления, степень, с которой показатели одной сущности влияют на показатели связанной сущности, регулируется с помощью численного веса распространения. Такие веса могут отличаться от одной сущности к другой и/или от одного критерия оценки к другому, что позволяет гибко и точно настраивать распространение показателей. Значения весов могут определяться операторами и/или подвергаться автоматической оптимизации, направленной на повышение эффективности выявления вредоносных программ.

[0082] Некоторые обычные антивредоносные системы определяют, является ли оцениваемая сущность вредоносной путем определения, выполняет ли соответствующая сущность указывающее на вредоносность поведение, и/или имеет ли сущность указывающие на вредоносность функции, такие как указывающую на вредоносность последовательность кода. В противоположность этому, в некоторых вариантах осуществления настоящего изобретения критерии оценки сущностей сами по себе не обязательно являются указывающими на вредоносность. Например, некоторые критерии включают в себя определение, выполняет ли сущность неопасные действия, такие как открытие файла или доступ к IP-адресу. Тем не менее такие действия могут быть вредоносными в сочетании с другими действиями, которые сами по себе не могут быть указывающими на вредоносность самостоятельно. Посредством мониторинга большого разнообразия поведений сущностей и/или функций, последующей записи большого количества (возможно, сотен) показателей оценки и агрегирования таких показателей в режиме каждой сущности, некоторые варианты осуществления настоящего изобретения могут увеличить процент выявления при сведении к минимуму ложных срабатываний.

[0083] Некоторые варианты осуществления настоящего изобретения могут защитить виртуализированную среду. В одном варианте осуществления, конфигурированном с возможностью поддержки виртуализации, некоторые компоненты настоящего изобретения могут выполняться внутри виртуальной машины, в то время как другие компоненты могут выполняться вне соответствующей виртуальной машины, например на уровне гипервизора, предоставляющего соответствующую виртуальную машину. Такие компоненты, выполняющиеся на уровне гипервизора, могут быть конфигурированы для выполнения антивредоносных операций для множества виртуальных машин, выполняющихся параллельно на соответствующей хостовой системе.

[0084] Специалисту в данной области техники понятно, что вышеуказанные варианты

осуществления изобретения могут быть изменены различными способами без выхода за рамки объема изобретения. Соответственно, объем изобретения должен определяться согласно нижеследующей формуле изобретения и ее законных эквивалентов.

(57) Формула изобретения

1. Хостовая система для определения вредоносной программной сущности, содержащая блок памяти, хранящий инструкции, при исполнении которых по меньшей мере одним аппаратным процессором хостовой системы хостовая система выполняет модуль управления сущностями, средство оценки сущностей и классифицирующий механизм, при этом:

модуль управления сущностями конфигурирован с возможностью управлять коллекцией оцениваемых программных сущностей, причем управление коллекцией содержит:

идентификацию набора сущностей-потомков первой сущности коллекции;

определение, завершена ли первая сущность;

в ответ, когда первая сущность завершена, определение, завершены ли все члены набора сущностей-потомков; и

в ответ, когда все члены набора сущностей-потомков завершены, удаление первой сущности из коллекции;

средство оценки сущностей конфигурировано с возможностью:

оценивать первую сущность согласно критерию оценки; и

в ответ, когда первая сущность удовлетворяет критерию оценки, передавать индикатор оценки в классифицирующий механизм;

классифицирующий механизм конфигурирован с возможностью:

записывать первый показатель, определенный для первой сущности, и второй показатель, определенный для второй сущности коллекции, причем первый и второй показатели определены согласно критерию оценки;

в ответ на запись первого и второго показателей и в ответ на получение индикатора оценки, обновлять второй показатель согласно индикатору оценки;

в ответ определять, является ли вторая сущность вредоносной согласно обновленному второму показателю,

в ответ на получение индикатора оценки, обновлять первый показатель согласно индикатору оценки; и

в ответ определять, является ли первая сущность вредоносной согласно обновленному первому показателю.

2. Хостовая система по п. 1, в которой первая сущность является порождением второй сущности.

3. Хостовая система по п. 1, в которой вторая сущность является порождением первой сущности.

4. Хостовая система по п. 1, в которой первая сущность содержит секцию кода, внедренного второй сущностью.

5. Хостовая система по п. 1, в которой вторая сущность содержит секцию кода, внедренного первой сущностью.

6. Хостовая система по п. 1, в которой обновление второго показателя содержит изменение второго показателя на величину, определенную согласно $w \cdot S$, где S является первым показателем и w представляет собой численный вес.

7. Хостовая система по п. 1, в которой управление коллекцией оцениваемых программных сущностей дополнительно содержит:

перехват запуска новой программной сущности; и

в ответ добавление новой программной сущности в коллекцию.

8. Долговременный машиночитаемый носитель для определения вредоносной программной сущности, хранящий инструкции, которые при их исполнении конфигурируют по меньшей мере один процессор хостовой системы, чтобы:

управлять коллекцией оцениваемых программных сущностей, причем управление коллекцией содержит:

идентификацию набора сущностей-потомков первой сущности коллекции;

определение, завершена ли первая сущность;

в ответ, когда первая сущность завершена, определение, завершены ли все члены набора сущностей-потомков; и

в ответ, когда все члены набора сущностей-потомков завершены, удаление первой сущности из коллекции;

записывать первый показатель, определенный для первой сущности, и второй показатель, определенный для второй сущности коллекции, при этом первый и второй показатели определены согласно критерию оценки;

в ответ на запись первого и второго показателей, оценивать первую сущность согласно критерию оценки;

в ответ на оценку первой сущности, когда первая сущность удовлетворяет критерию оценки, обновлять второй показатель;

в ответ на обновление второго показателя определять, является ли вторая сущность вредоносной согласно обновленному второму показателю;

в ответ на оценку первой сущности, когда первая сущность удовлетворяет критерию оценки, обновлять первый показатель; и

в ответ определять, является ли первая сущность вредоносной согласно обновленному первому показателю.

9. Машиночитаемый носитель по п. 8, в котором первая сущность является порождением второй сущности.

10. Машиночитаемый носитель по п. 8, в котором вторая сущность является порождением первой сущности.

11. Машиночитаемый носитель по п. 8, в котором первая сущность содержит секцию кода, внедренного второй сущностью.

12. Машиночитаемый носитель по п. 8, в котором вторая сущность содержит секцию кода, внедренного первой сущностью.

13. Машиночитаемый носитель по п. 8, в котором обновление второго показателя содержит изменение второго показателя на величину, определенную согласно $w \cdot S$, где S является первым показателем и w представляет собой численный вес.

14. Машиночитаемый носитель по п. 8, в котором управление коллекцией оцениваемых программных сущностей дополнительно содержит:

перехват запуска новой программной сущности; и

в ответ, добавление новой программной сущности в коллекцию.

15. Способ для определения вредоносной программной сущности, включающий использование по меньшей мере одного процессора хостовой системы для выполнения следующих этапов:

управление коллекцией оцениваемых программных сущностей, причем управление коллекцией содержит:

идентификацию набора сущностей-потомков первой сущности коллекции;

определение, завершена ли первая сущность;

в ответ, когда первая сущность завершена, определение, завершены ли все члены набора сущностей-потомков; и

в ответ, когда все члены набора сущностей-потомков завершены, удаление первой сущности из коллекции;

запись первого показателя, определенного для первой сущности, и второго показателя, определенного для второй сущности коллекции, при этом первый и второй показатели определяют согласно критерию оценки;

в ответ на запись первого и второго показателей, оценка первой сущности согласно критерию оценки;

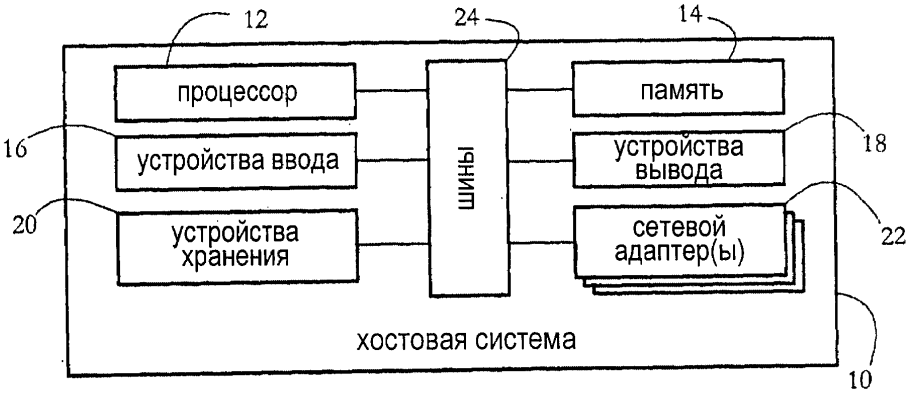
в ответ на оценку первой сущности, когда первая сущность удовлетворяет критерию оценки, обновление второго показателя;

в ответ на обновление второго показателя, определение, является ли вторая сущность вредоносной согласно обновленному второму показателю;

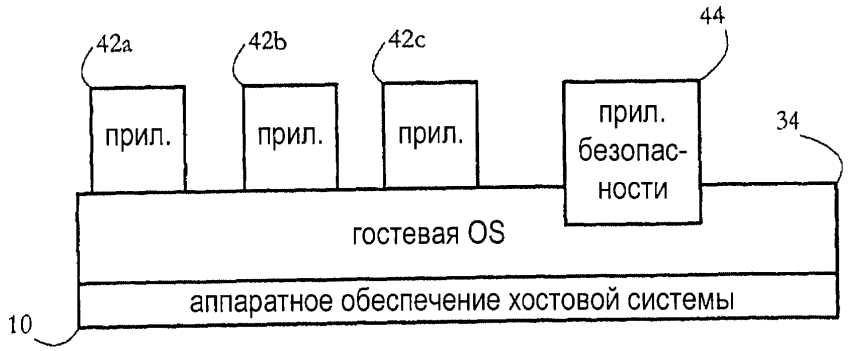
в ответ на оценку первой сущности, когда первая сущность удовлетворяет критерию оценки, обновление первого показателя; и

в ответ, определение, является ли первая исполняемая сущность вредоносной согласно обновленному первому показателю.

1

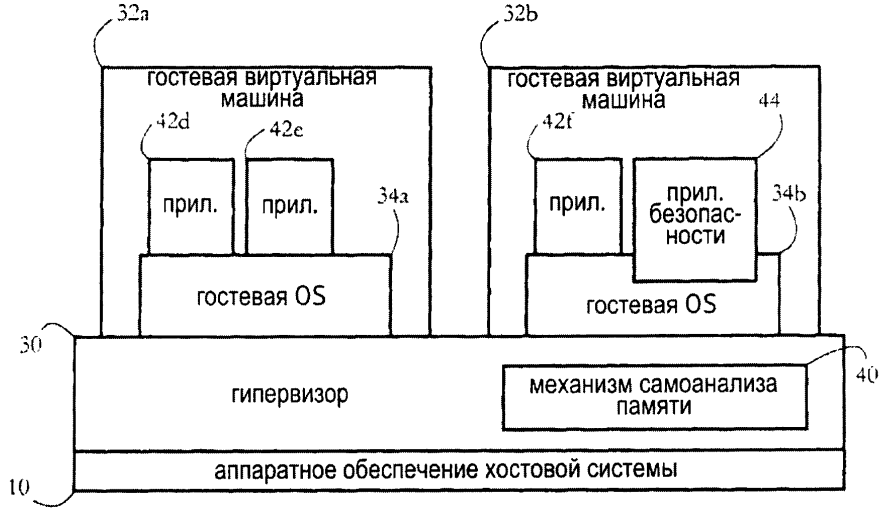


Фиг.1

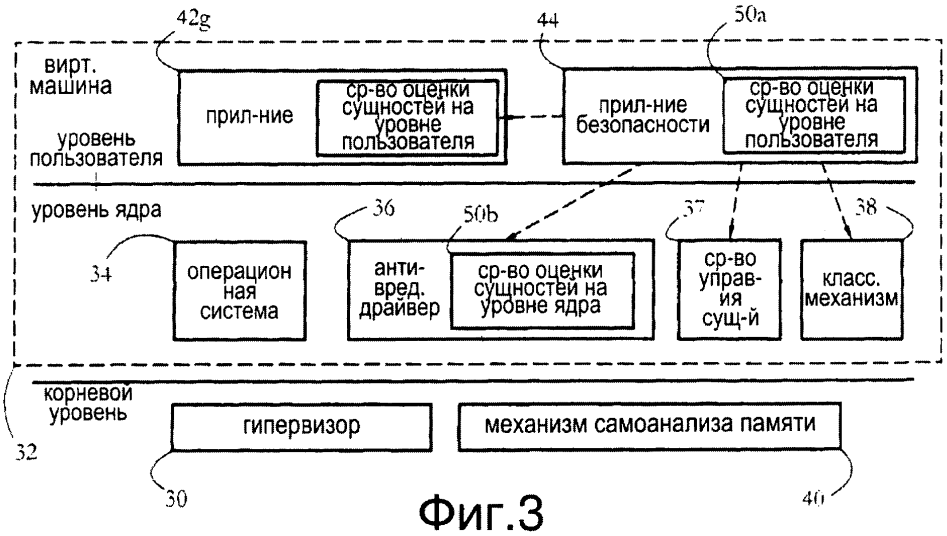


Фиг.2-А

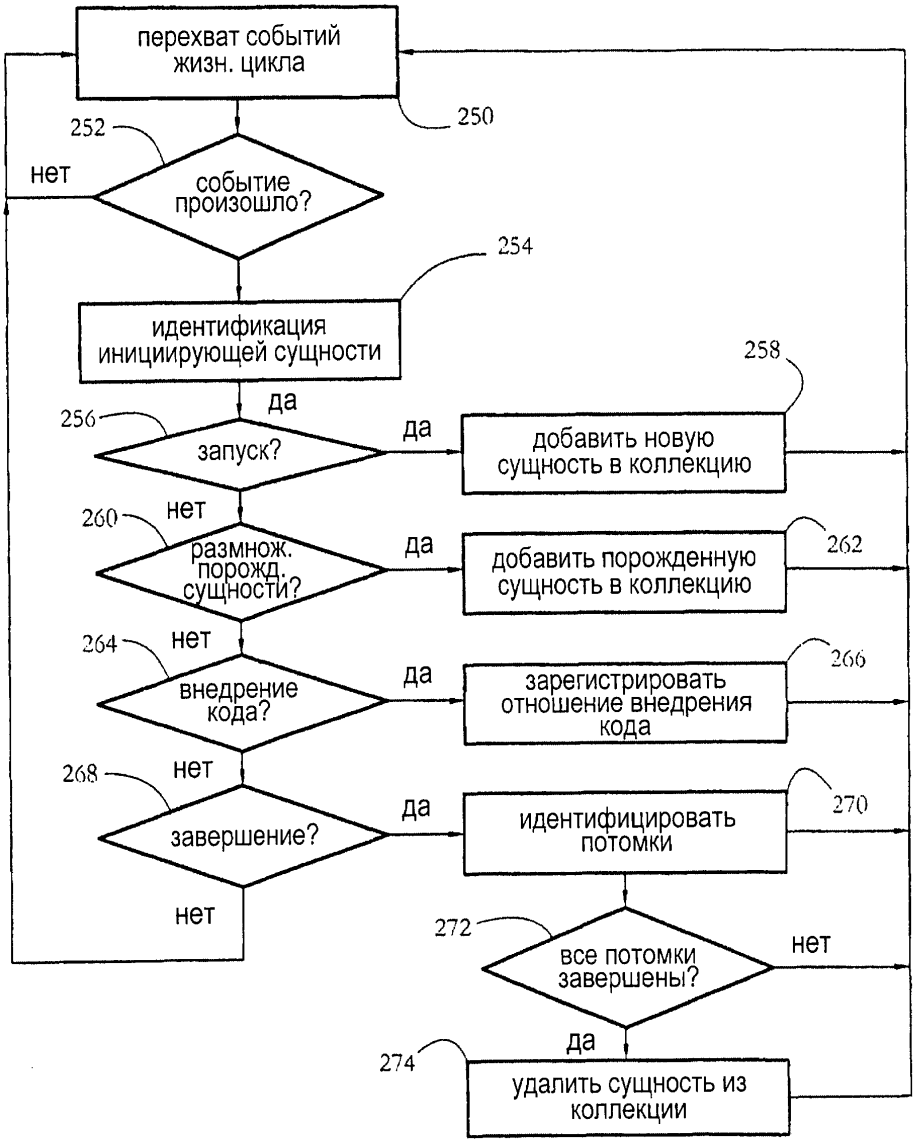
2


32a
32b
гостевая виртуальная машина
гостевая виртуальная машина
42d
42e
42f
44
прил.
прил.
прил.
прил. безопас-ности
34a
34b
гостевая OS
гостевая OS
30
гипервизор
механизм самоанализа памяти
40
аппаратное обеспечение хостовой системы
10

Фиг.2-B

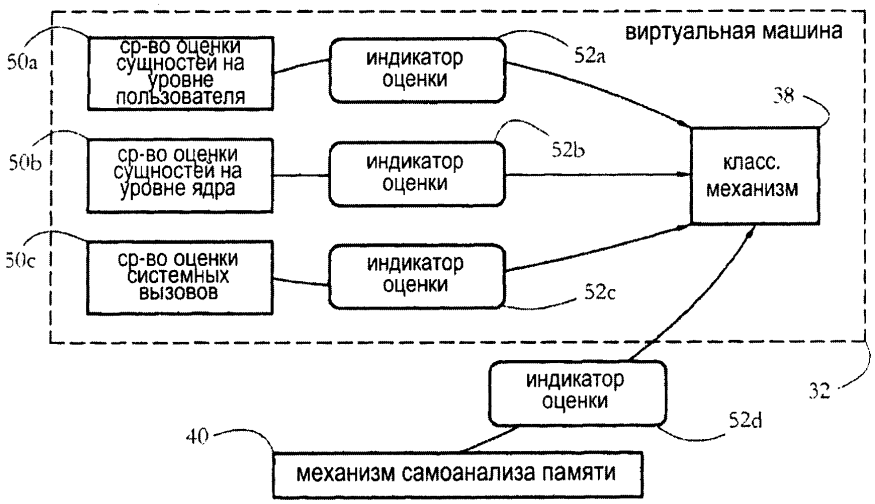

42g
44
50a
вирт. машина
уровень пользователя
прил-ние
ср-во оценки сущностей на уровне пользователя
прил-ние безопасности
ср-во оценки сущностей на уровне пользователя
уровень ядра
36
50b
37
38
34
операцион ная система
анти-вред. драйвер
ср-во оценки сущностей на уровне ядра
ср-во управ-ия сущ-й
класс. механизм
корневой уровень
32
гипервизор
механизм самоанализа памяти
30
40

Фиг.3

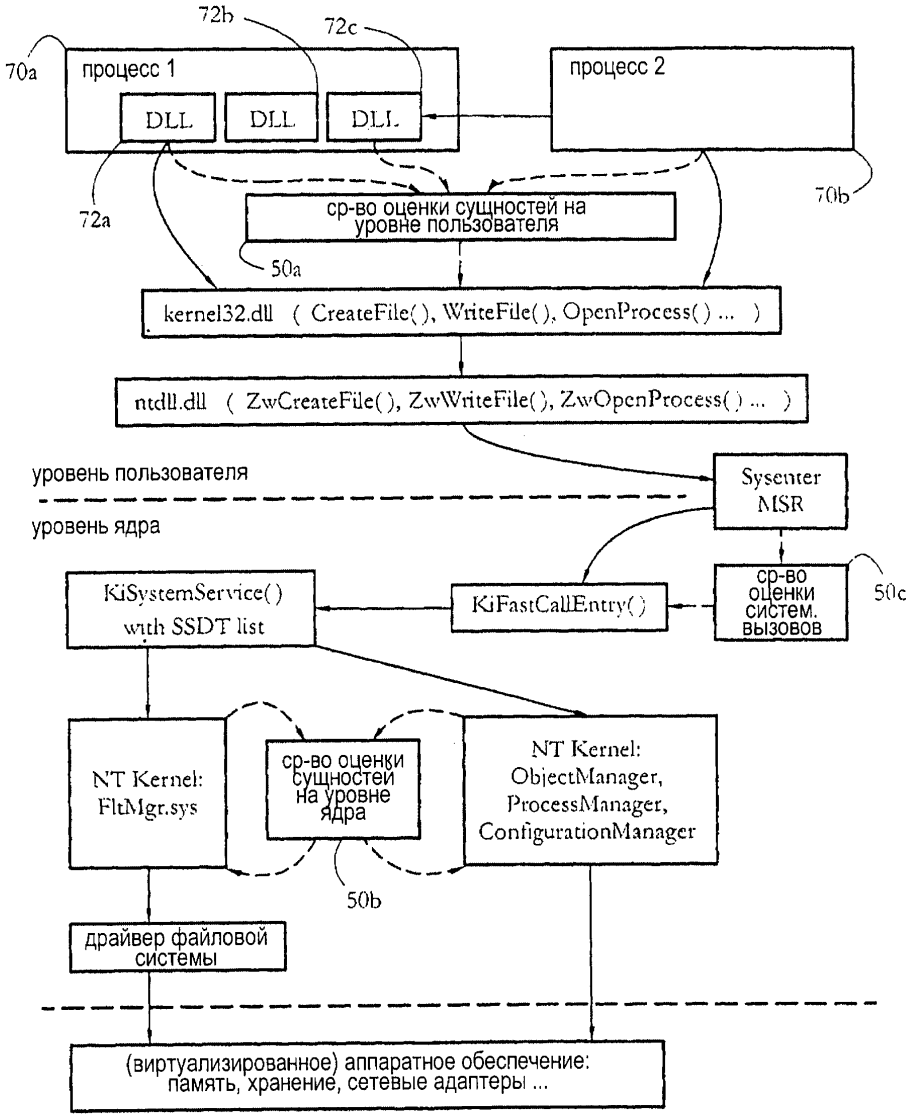

перехват событий жизн. цикла
250
252
нет
событие произошло?
254
идентификация инициирующей сущности
да
256
запуск?
да
258
добавить новую сущность в коллекцию
нет
260
размнож. порожд. сущности?
да
262
добавить порожденную сущность в коллекцию
нет
264
внедрение кода?
да
266
зарегистрировать отношение внедрения кода
нет
268
завершение?
да
270
идентифицировать потомки
нет
272
все потомки завершены?
нет
да
274
удалить сущность из коллекции

Фиг.4

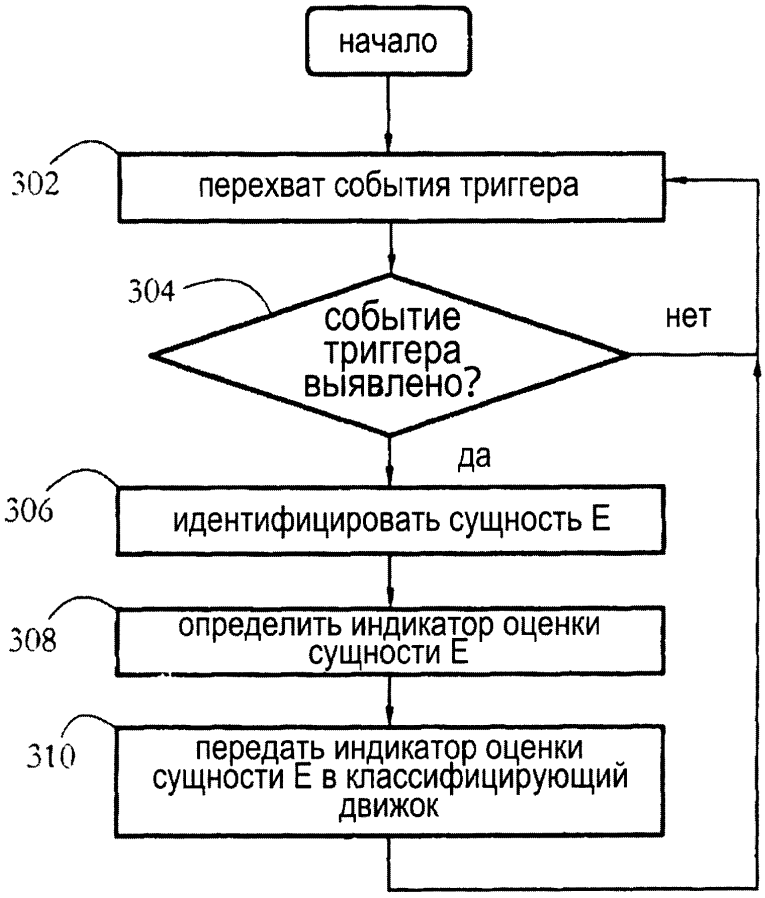
4


виртуальная машина
50a
ср-во оценки сущностей на уровне пользователя
индикатор оценки
52a
38
50b
ср-во оценки сущностей на уровне ядра
индикатор оценки
52b
класс. механизм
50c
ср-во оценки системных вызовов
индикатор оценки
52c
индикатор оценки
32
52d
40
механизм самоанализа памяти

Фиг.5

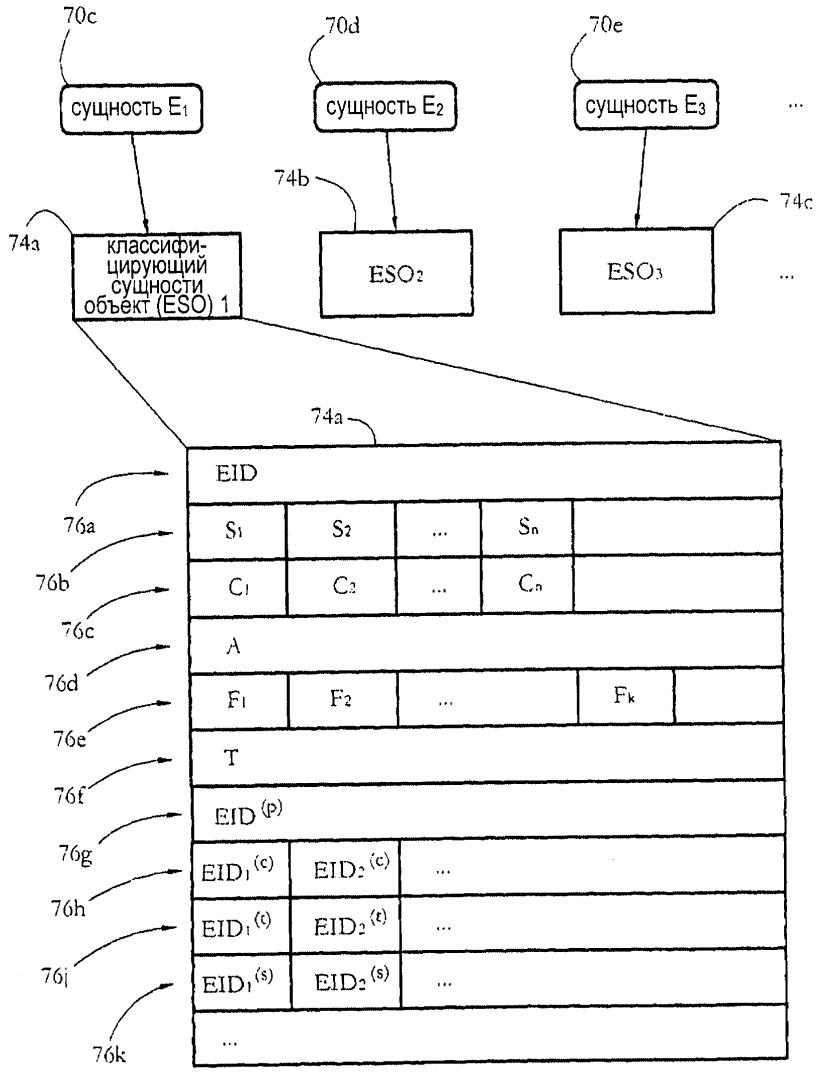

72b
72c
70a
процесс 1
DLL
DLL
DLL
процесс 2
70b
72a
ср-во оценки сущностей на уровне пользователя
50a
kernel32.dll (CreateFile( ), WriteFile( ), OpenProcess( ) ...)
ntdll.dll (ZwCreateFile( ), ZwWriteFile( ), ZwOpenProcess( ) ...)
уровень пользователя
уровень ядра
Sysenter MSR
KiSystemService( ) with SSDT list
KiFastCallEntry( )
ср-во оценки систем. вызовов
50c
NT Kernel: FltMgr.sys
ср-во оценки сущностей на уровне ядра
NT Kernel: ObjectManager, ProcessManager, ConfigurationManager
50b
драйвер файловой системы
(виртуализированное) аппаратное обеспечение: память, хранение, сетевые адаптеры ...

Фиг.6


начало
302
перехват события триггера
304
событие триггера выявлено?
нет
да
306
идентифицировать сущность E
308
определить индикатор оценки сущности E
310
передать индикатор оценки сущности E в классифицирующий движок

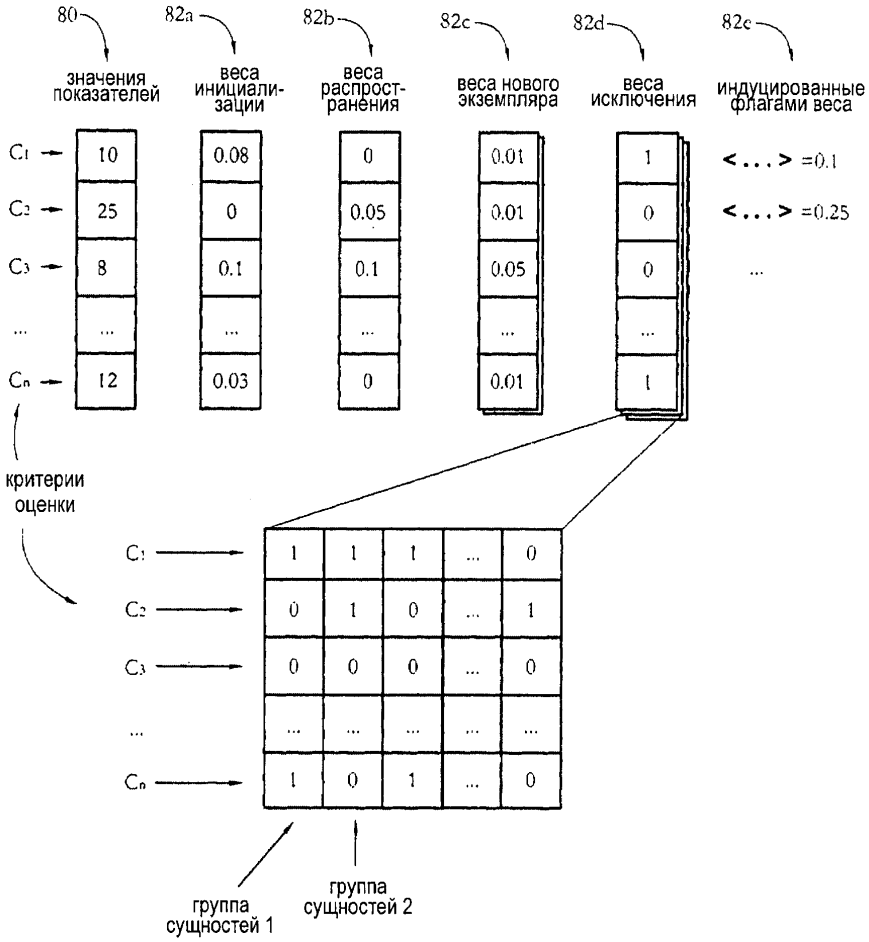
Фиг.7

7

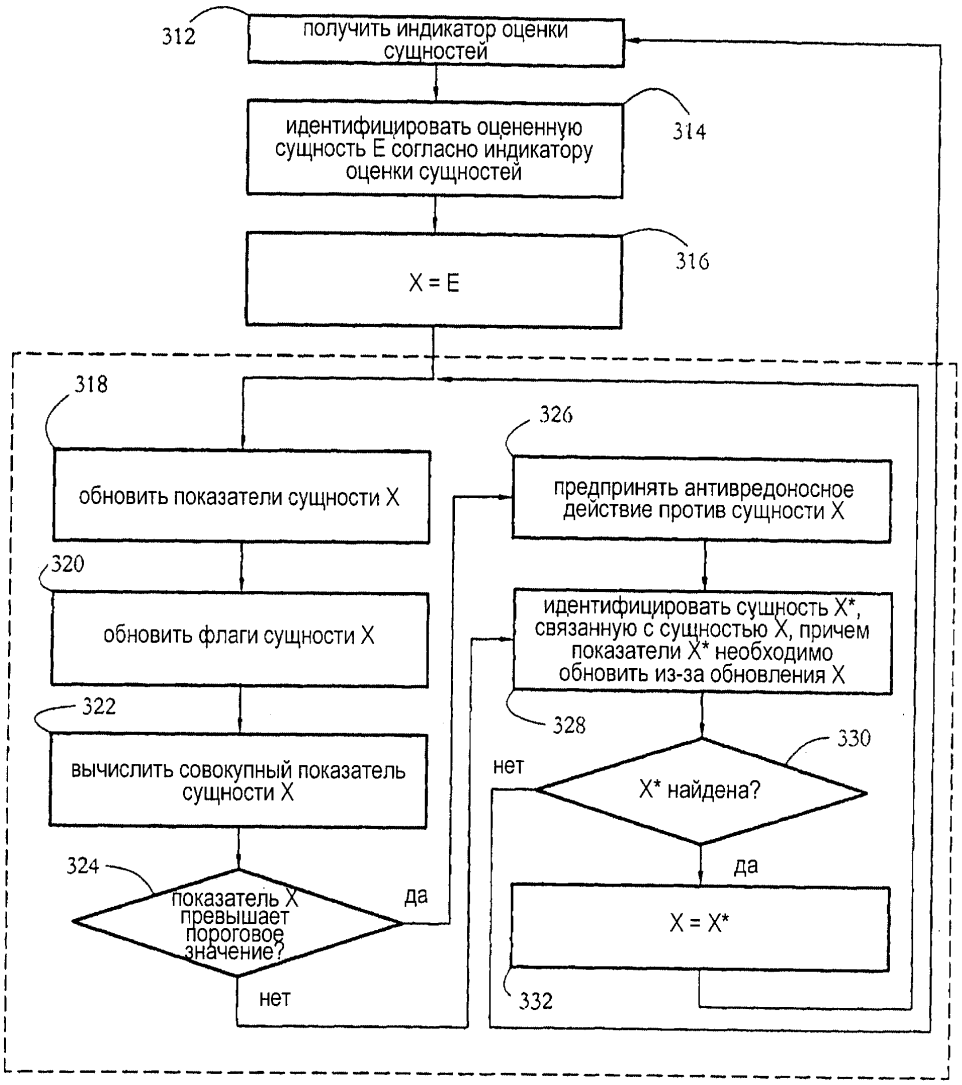

70c
70d
70e
сущность E1
сущность E2
сущность E3
...
74b
74c
74a
классифи-
цирующий
сущности
объект (ESO) 1
ESO2
ESO3
...
74a
EID
76a
S1 S2 ... Sn
76b
C1 C2 ... Cn
76c
A
76d
F1 F2 ... Fk
76e
T
76f
EID (P)
76g
EID1 (c) EID2 (c) ...
76h
EID1 (t) EID2 (t) ...
76j
EID1 (s) EID2 (s) ...
76k
...

Фиг.8

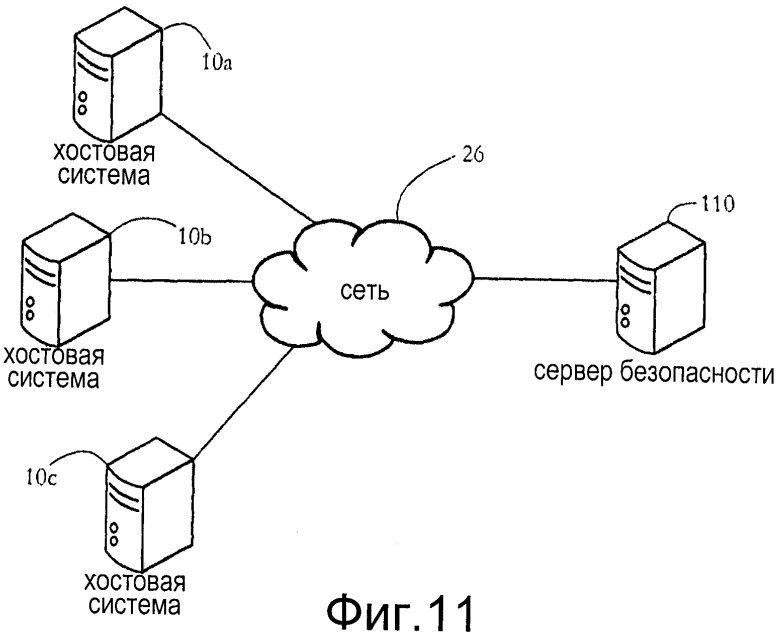
8


80
82a
82b
82c
82d
82e
значения показателей
веса инициали-зации
веса распрост-ранения
веса нового экземпляра
веса исключения
индуцированные флагами веса
C1
C2
C3
Cn
10
25
8
12
0.08
0
0.1
0.03
0
0.05
0.1
0
0.01
0.01
0.05
0.01
1
0
0
1
<...> =0.1
<...> =0.25
критерии оценки
1 1 1 ... 0
0 1 0 ... 1
0 0 0 ... 0
1 0 1 ... 0
группа сущностей 1
группа сущностей 2

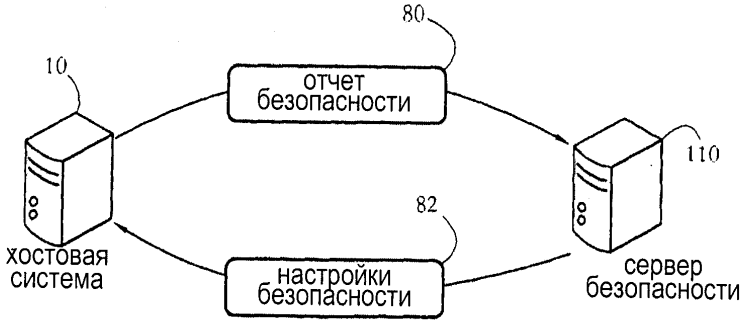
Фиг.9


312
получить индикатор оценки сущностей
идентифицировать оцененную сущность E согласно индикатору оценки сущностей
314
X = E
316
318
обновить показатели сущности X
320
обновить флаги сущности X
322
вычислить совокупный показатель сущности X
324
показатель X превышает пороговое значение?
да
нет
326
предпринять антивредоносное действие против сущности X
идентифицировать сущность X*, связанную с сущностью X, причем показатели X* необходимо обновить из-за обновления X
328
330
нет
X* найдена?
да
X = X*
332

Фиг.10


10a
хостовая система
26
10b
сеть
110
хостовая система
сервер безопасности
10c
хостовая система

Фиг.11


80
10
отчет безопасности
110
82
хостовая система
настройки безопасности
сервер безопасности

Фиг.12