РОССИЙСКАЯ ФЕДЕРАЦИЯ

(19) **RU** (11) **2 651 196**(13) **C1**

(51) МПК
G06F 21/56 (2013.01)

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

(12) ОПИСАНИЕ ИЗОБРЕТЕНИЯ К ПАТЕНТУ

(52) СПК
G06F 21/566 (2006.01)

(21)(22) Заявка: **2017121120, 16.06.2017**

(24) Дата начала отсчета срока действия патента:
16.06.2017

Дата регистрации:
18.04.2018

Приоритет(ы):
(22) Дата подачи заявки: **16.06.2017**

(45) Опубликовано: **18.04.2018** Бюл. № **11**

Адрес для переписки:
125212, Москва, Ленинградское ш., 39а, стр. 3, АО "Лаборатория Касперского", Управление по интеллектуальной собственности, Надежда Васильевна Кащенко

(72) Автор(ы):
Монастырский Алексей Владимирович (RU),
Павлющик Михаил Александрович (CA),
Романенко Алексей Михайлович (RU),
Головкин Максим Юрьевич (RU)

(73) Патентообладатель(и):
Акционерное общество "Лаборатория Касперского" (RU)

(56) Список документов, цитированных в отчете о поиске: **US 7246156 B2, 17.07.2007. US7448084 B1, 04.11.2008. US 8135994 B2, 13.03.2012. RU 2015969 A, 11.01.2017.**

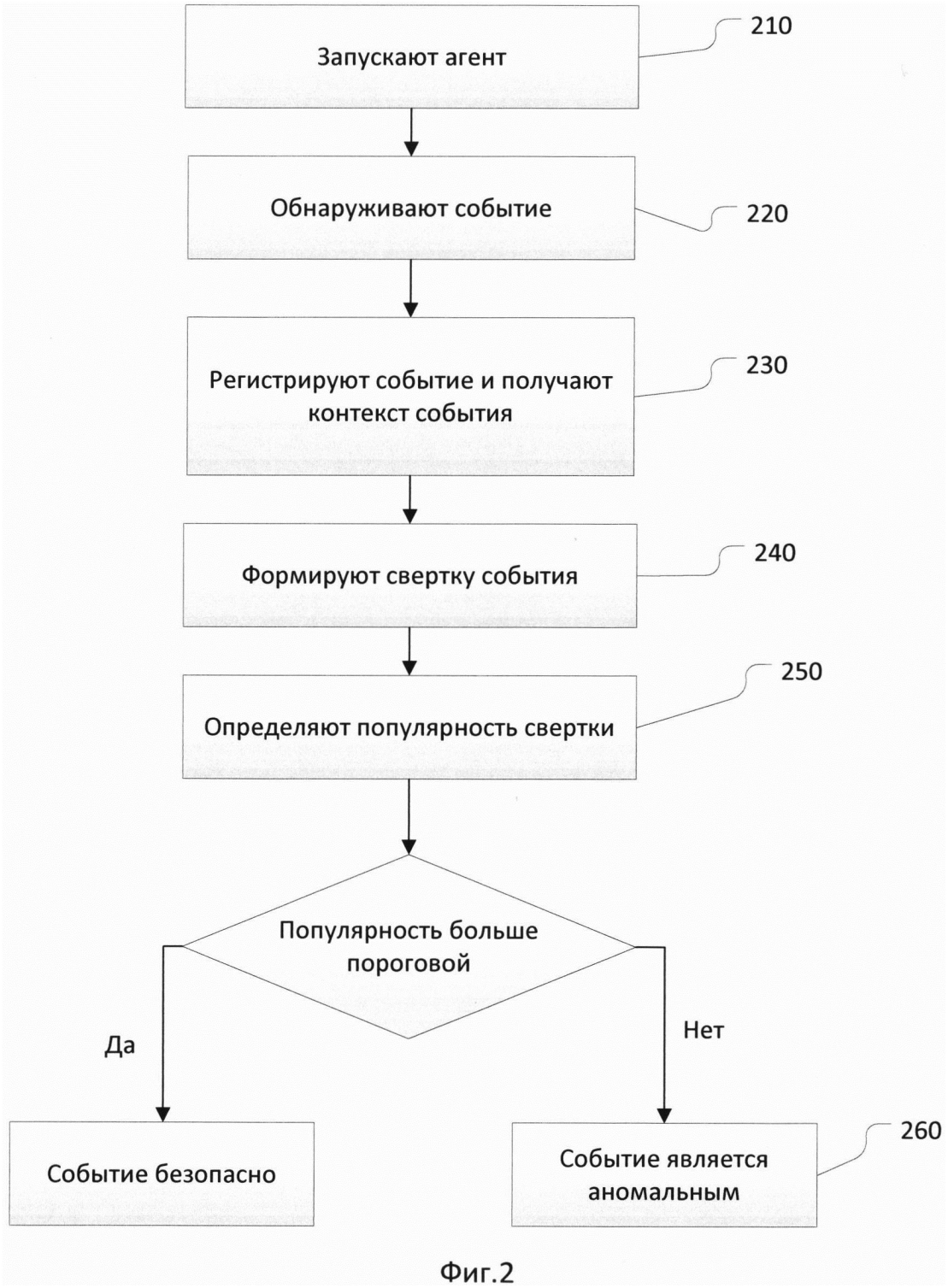
(54) Способ обнаружения аномальных событий по популярности свертки события

(57) Реферат:

Изобретение относится к способам обнаружения аномальных событий, возникающих в операционной системе. Технический результат заключается в обеспечении обнаружения аномальных событий, возникающих в операционной системе клиента в процессе исполнения программного обеспечения. Запускают агент, регистрирующий события, возникающие в операционной системе. Обнаруживают при помощи по крайней мере одного перехватчика, установленного в операционной системе, возникшее в операционной системе событие. Регистрируют событие, обнаруженное перехватчиком, и получают от компьютерного устройства контекст указанного события при помощи агента. Выделяют средством формирования свертки из полученного контекста признаки события и формируют на основании выделенных признаков свертку события. Определяют средством сравнения популярность сформированной свертки события. Признают средством сравнения обнаруженное событие аномальным, если популярность свертки указанного события ниже порогового значения. 21 з.п. ф-лы, 5 ил.

RU 2651196 C1

RU 2651196 C1



Фиг.2

RU 2651196 C1

RU 2651196 C1

RUSSIAN FEDERATION

FEDERAL SERVICE
FOR INTELLECTUAL PROPERTY

(19) **RU** (11) **2 651 196**(13) **C1**

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

(12) ABSTRACT OF INVENTION

(52) CPC
G06F 21/566 (2006.01)

(21)(22) Application: **2017121120, 16.06.2017**

(24) Effective date for property rights:
16.06.2017

Registration date:
18.04.2018

Priority:
(22) Date of filing: **16.06.2017**

(45) Date of publication: **18.04.2018** Bull. № **11**

Mail address:
125212, Moskva, Leningradskoe sh., 39a, str. 3, AO "Laboratoriya Kasperskogo", Upravlenie po intellektualnoj sobstvennosti, Nadezhda Vasilevna Kashchenko

(72) Inventor(s):
Monastyrskij Aleksej Vladimirovich (RU),
Pavlyushchik Mikhail Aleksandrovich (CA),
Romanenko Aleksej Mikhajlovich (RU),
Golovkin Maksim Yurevich (RU)

(73) Proprietor(s):
Aktsionernoe obshchestvo "Laboratoriya Kasperskogo" (RU)

RU 2651196 C1

(54) METHOD OF THE ANOMALOUS EVENTS DETECTING BY THE EVENT DIGEST POPULARITY

(57) Abstract:

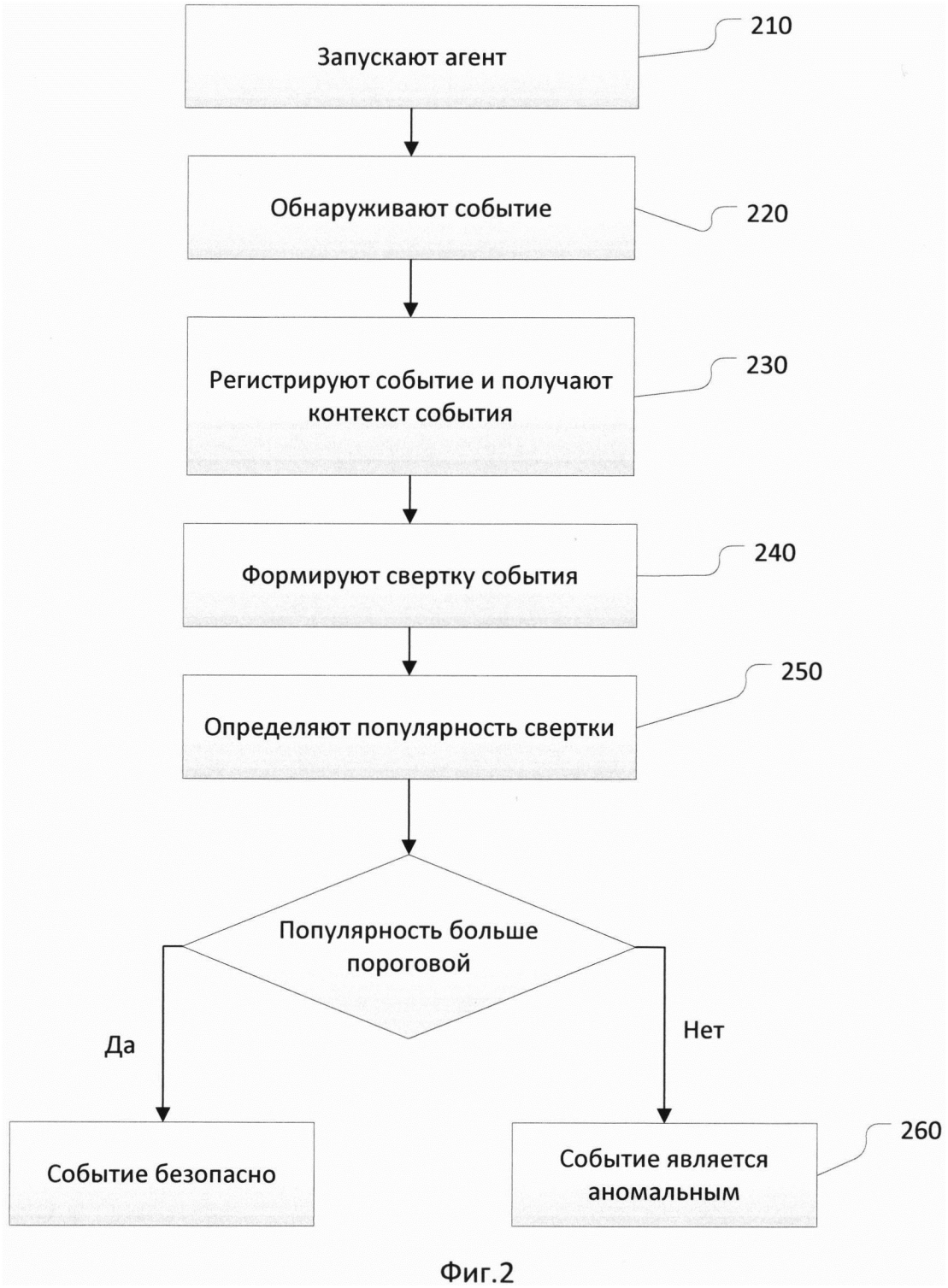
FIELD: information technologies.

SUBSTANCE: invention relates to occurring in the operating system abnormal events detecting methods. Starting the agent recording events that occur in the operating system. Using at least one installed in the operating system interceptor detecting an event that has occurred in the operating system. Recording the detected by the interceptor event and using the agent receiving the specified event context from the computer device. From the obtained context selecting the event signs by means of the digest formation and based on the selected signs forming the event digest. Formed event digest popularity is determined by means of comparison. By the comparison means recognizing the detected as abnormal one, if the specified event digest popularity is below the threshold value.

EFFECT: technical result is provision of the abnormal events detection occurring in the client's operating system during the software execution.

22 cl, 5 dwg

RU 2651196 C1



Фиг.2

RU 2651196 C1

Область техники

Настоящее изобретение относится к способам защиты компьютерных устройств от эксплуатации уязвимостей, содержащихся в программном обеспечении этих устройств.

Уровень техники

Один из самых распространенных способов проникновения вредоносного программного обеспечения на компьютерные устройства заключается в эксплуатации уязвимостей [1]([1]Era of exploits: number of attacks using software vulnerabilities on the rise https://www.kaspersky.com/rss-feeds/2017_era-of-exploits-number-of-attacks--using-software-vulnerabilities-on-the-rise), содержащихся в программном обеспечении, установленном на данном устройстве.

Для борьбы с эксплуатацией уязвимости используются пассивные методы в виде устранения самих уязвимостей[2] ([2]Vulnerability And Patch Management http://resources.infosecinstitute.com/category/certifications-training/cissp/domains/security-operations/vulnerability-and-patch-management/#gref)и активные в виде обнаружения самого факта эксплуатация уязвимостей. Пассивные методы используются для уже известных уязвимостей, активные как для известных, так и неизвестных.

Например, в публикации US 9251373 описывается технология, предотвращающая атаку, направленную на переполнение буфера, заключающаяся в дублировании фреймов стека и их сравнении во время исполнения процесса. А в публикации WO 2015113052 предлагается осуществлять мониторинг памяти процесса с целью обнаружения особенностей, характерных для известных способов эксплуатации уязвимостей: попытки выполнения вне области кода, недопустимые указатели базы, недопустимые адресы возврата, попытки применения техник возвратно-ориентированного программирования и т.д.

Существующие технологии обнаружения действительно способны обнаружить факт эксплуатации уязвимости с применением известных техник и механизмов, но, к сожалению, данные способы не способны противостоять новым техникам эксплуатации уязвимостей, которые используют новые принципы и механизмы эксплуатации. Например, для того, чтобы сделать невозможным исполнение шелл кодов[3] ([3]Execute Disable Bit Functionality Blocks Malware Code Execution: http://cache-www. intel.com/cd/00/00/14/93/149307_149307.pdf.) (англ. shellcode), были разработаны технологии, запрещающие исполнение на стеке, но на смену им пришли техники возвратно-ориентированного программирования[4] ([4]Return-Oriented Programming: Exploits Without Code Injection http://cseweb.ucsd.edu/~hovav/talks/blackhat08.htm), перед которыми данные технологии защиты оказались бессильны, и для защиты от этих атак разработали новые решения[5] ([5]US 20160196428 System and Method for Detecting Stack Pivot Programming Exploit). Поэтому возникла потребность обнаруживать отклонение функционирования компьютерной системы от нормального, которое могло бы свидетельствовать о том, что система была атакована посредством эксплуатации уязвимости в программном обеспечении. Решение данной задачи позволило бы абстрагироваться от самих техник эксплуатации уязвимостей, которые изменяются и совершенствуются, а ориентироваться на внешние проявления атаки, которые при смене техник остаются неизменными.

Раскрытие изобретения

Настоящее изобретение предназначено для обнаружения аномальных событий, возникающих в операционной системе.

Технический результат настоящего изобретения заключается в обеспечении обнаружения аномальных событий, возникающих в операционной системе клиента в

процессе исполнения программного обеспечения. Результат достигается за счет оценки популярности возникающих событий на основании популярности сверток указанных событий, сформированных на основании контекста этих событий, где событие, популярность свертки которого ниже порогового значения, признается аномальным.

Объектом настоящего изобретения является способ обнаружения аномальных событий в операционной системе компьютерного устройства, в котором: запускают агент, регистрирующий события, возникающие в операционной системе; обнаруживают при помощи по крайней мере одного перехватчика, установленного в операционной системе, возникшее в операционной системе событие; регистрируют агентом событие, обнаруженное перехватчиком, и получают агентом от компьютерного устройства контекст указанного события; выделяют средством формирования свертки из полученного контекста признаки события, и формируют на основание выделенных признаков свертку события; определяют средством сравнения популярность сформированной свертки события; признают средством сравнения обнаруженное событие аномальным, если популярность свертки указанного события ниже порогового значения.

В настоящем изобретении контекст события есть совокупность состояний операционной системы на момент возникновения события непосредственно повлиявших на его возникновение, при этом события возникают во время выполнения процессов в операционной системе. Контекст события в частном случае включает, по меньшей мере:

- стек вызовов, на момент возникновения события;
- дамп участка памяти процесса, содержащего код, который выполнялся в момент возникновения события.

Контекст дополнительно может включать информацию о переходах из по меньшей мере Last Branch Record, Branch Trace Store, а также список модулей, загруженных в процесс до возникновения события.

В частном случае из стека вызовов в качестве признаков для формирования свертки получают по меньшей мере список процедур и функций, выполняемых в данный момент времени, список модулей, содержащих указанные процедуры и функции, а также типы данных и значения всех параметров, передаваемых в модули, где модуль - это программный объект, содержащий код, который расширяет функциональность запущенного процесса. В частном случае модулем является модуль ядра операционной системы или динамическая библиотека. А из дампа в качестве признаков для формирования свертки получают по меньшей мере такие признаки как наличие/отсутствие косвенных вызовов, позиционно независимого кода, самомодифицирующегося кода. Признаки события, выделяемые из контекста, характеризуют событие и необходимы для формирования свертки.

В частном случае в качестве перехватчиков по меньшей мере выступают средства трассировки событий, такие как Event tracing for Windows.

В частном случае при получении контекста дополнительно производят преобразования полученного контекста для представления в вид, позволяющий выделить признаки. Способами преобразования могут быть:

- квантование;
- сортировка;
- слияние (склеивание); группировка;
- настройка набора данных;
- табличная подстановка значений;
- вычисляемые значения;

- кодирование данных;

- нормализация (масштабирование).

Для нормализации могут получать информацию об отладочных символах для модулей из стека вызовов и дампа, а информация об отладочных символах содержится в файлах формата .pdb. В частном случае контекст преобразуют с помощью дизассемблирования и эмуляции, а выделенные признаки могут содержать по крайней мере такие признаки как наличие/отсутствие косвенных вызовов, позиционно независимого кода, самомодифицирующегося кода.

В частном случае популярность свертки определяют в рамках подсети, в которой расположено компьютерное устройство, на котором обнаружено событие.

В другом частном случае определяют глобальную популярность свертки события.

Популярность может определяться следующим образом:

- запросом к базе данных, содержащей информацию о популярности сверток событий,

- в случае отсутствия информации о популярности в базе данных производится сбор информации из сети о количестве обнаруженных событий и общем количестве клиентов, с которых производится сбор на текущий момент времени, после чего вычисляется популярность.

В частном случае запуск агента, обнаружение события, регистрацию события агентом и получение контекста выполняют на стороне клиента, а выделение признаков и формирование свертки, определение популярности свертки и признание обнаруженного события аномальным выполняют на стороне сервера, при этом для выделения признаков дополнительно получают сервером от агента на клиенте контекст события.

В другом частном случае запуск агента, обнаружение события, регистрацию события агентом, получение контекста и выделение признаков с формированием свертки выполняют на стороне клиента, а определение популярности свертки и признание обнаруженного события аномальным выполняют на стороне сервера, при этом дополнительно перед определением популярности получают сервером от агента на клиенте свертку события.

В частном случае свертка представляет собой либо хеш события, либо вектор события в евклидовом пространстве.

Краткое описание чертежей

Дополнительные цели, признаки и преимущества настоящего изобретения будут очевидными из прочтения последующего описания осуществления изобретения со ссылкой на прилагаемые чертежи, на которых:

Фиг. 1 изображает систему мониторинга исполнения программного обеспечения, установленного на клиенте;

Фиг. 2 изображает способ обнаружение аномальных событий на основании оценки популярности сверток событий;

Фиг. 3 изображает способ формирования набора сверток безопасных событий;

Фиг. 4 изображает способ обнаружения аномального события на основании набора сверток безопасных событий;

Фиг. 5 изображает пример компьютерной системы общего назначения, с помощью которой может быть реализовано настоящее изобретение.

Осуществление изобретения

Объекты и признаки настоящего изобретения, способы для достижения этих объектов и признаков станут очевидными посредством отсылки к примерным вариантам осуществления. Однако настоящее изобретение не ограничивается примерными вариантами осуществления, раскрытыми ниже, оно может воплощаться в различных

видах. Приведенное описание предназначено для помощи специалисту в области техники для исчерпывающего понимания изобретения, которое определяется только в объеме приложенной формулы.

Модуль (син. программный модуль[6] ([6]ГОСТ 19781-90. Обеспечение систем обработки информации программное. Термины и определения)) - программный объект, содержащий код, который расширяет функциональность запущенного процесса, например: модуль ядра операционной системы; динамическая библиотека.

Свертка в данном изобретении понимается максимально широко, не только как некоторый хеш (англ. intelligent hash), где признаки (англ. features) события сворачиваются в строку, но и вектор, где признаки события сворачиваются в координаты и др. В общем случае это любой объект, в который может быть свернуты признаки события (далее по тексту также «признаки») для осуществления математических и логических операций над ними. Сворачивание признаков есть произвольное преобразование признаков в строковое представление, векторное представление или в их совокупность.

Событие (англ. event) - идентифицированное появление определенного состояния операционной системы, сервиса или сети. Информация о событии может содержаться в сообщении[7] ([7]Например типа Windows Message https://msdn.microsoft.com/ru-ru/library/windows/desktop/ff381405(v=vs.85).aspx) программного обеспечения, например, операционной системы (либо его части), которое указывает, что произошло. События в системе бывают многих типов[8] ([8]Event Types https://msdn.microsoft.com/ru-ru/library/windows/desktop/aa363662(v=vs.85).aspx)и видов. Примеры видов:

- запуск процессов;
- загрузка модулей;
- файловые операции;
- реестровые операций;
- и др.

Контекст события - совокупность состояний операционной системы на момент возникновения события непосредственно повлиявших на его возникновение. Примеры содержимого контекста будут указаны ниже.

Аномальное событие - идентифицированное появление определенного состояния операционной системы, сервиса или сети, указывающего на возникновение неизвестной ранее ситуации. В частном случае аномальным событием является событие безопасности[9] ([9]ГОСТ Р ИСО/МЭК ТО 18044: 2007) - идентифицированное появление определенного состояния системы, сервиса или сети, указывающего на возможное нарушение политики информационной безопасности или отказ защитных мер, или возникновение неизвестного ранее состояния, которая может иметь отношение к безопасности.

На Фиг. 1 изображена система мониторинга исполнения программного обеспечения, осуществляющая способы, которые являются объектами настоящего изобретения. На стороне клиента 100 установлен агент 110. Также в операционной системе клиента 100 установлены перехватчики 120, связанные с агентом 110. В частном случае в качестве перехватчиков может использоваться Event Tracing for Windows (сокр. ETW)[10] ([10]https://msdn.microsoft.com/en-us/library/ms751538(v=vs.110).aspx)120a. Назначение указанных перехватчиков 120 - обнаруживать события, возникающие в операционной системе и сообщать об этом агенту 110, например путем отправки нотификации непосредственно агенту 110, и/или записью в журнал событий (на Фиг. 1 не указан, хранится в локальной

базе данных 160) к которому имеет доступ агент 110. Агент 110 может получать нотификации как о всех возможных событиях в системе, которые могут быть обнаружены перехватчиками 120, так и только об интересующих событиях одного вида (например, только о запуске процессов). Агент 110 содержит средство сбора 130, которое используется после того как агент зарегистрирует интересующее событие, обнаруженное перехватчиками 120. Средство сбора 130 получает контекст зарегистрированного события. В частном случае в контекст события включают стек вызовов, который предшествовал возникновению события и дамп участка памяти, содержащего код, который выполнялся в момент возникновения события. Из стека вызовов получают по меньшей мере список процедур и функций, выполняемых в данный момент времени, список модулей, содержащих указанные процедуры и функции, а также типы и значения всех параметров, передаваемых в модули, например при вызове экспортируемых модулем функций Дополнительно контекст может включать информацию о переходах из по меньшей мере Last Branch Record (сокр. LBR)[11] ([11]Intel® Microarchitecture Codename Nehalem Performance Monitoring Unit Programming Guide C. 43), Branch Trace Store (сокр. BTS)[12] ([12]Там же C. 45) - буферов памяти и регистров. LBR и BTS содержат информацию о выполнении программы: адреса переходов, ветви исполнения и др. (иначе, сохраняют данные трассировки). В частном случае перехватчики 120 при обнаружении события сохраняют его контекст, который потом будет получен средством сбора 130 агента 110 (например, передан средству сбора 130 или запрошен средством сбора 130) и средству сбора 130 в таких случаях нет необходимости самостоятельно получать весь требуемый контекст или некоторую его часть. Клиент 110 и сервер 200 содержат средство формирования свертки 140. Контекст полученный средством сбора 130 передаются средству формирования свертки 140. В одном случае они могут передаваться средству формирования свертки 140 содержащемуся на клиенте 100, в другом пересылаются на сервер 200. Средство формирования свертки 140 извлекает из полученного контекста признаки (примеры будут указаны ниже) и формирует свертку. Также средство формирования свертки 140 трансформирует (преобразует) контекст события (пример ниже). Как указывалось, свертка в данном изобретении понимается максимально широко, не только как некоторый хеш (где признаки сворачиваются в строку), но и вектор (где признаки сворачиваются в координаты) и др. - иными словами, любой объект, в который может быть свернуты признаки события для осуществления математических и логических операций над ними. Для формирования сверток событий могут использоваться любые алгоритмы, известные из уровня техники, в которых, например, из полученных признаков формируются хеши для файлов (один из таких способов описан в публикации RU 2580036) или векторы для файлов (один из таких способов описан в публикации RU 2614557) или HTML страниц. Средство сравнения 150 может располагаться как на сервере 200, так и на клиенте 110. Данное средство используется для определения популярности сформированной свертки события и сравнения сформированной свертки с другими свертками, например свертками, содержащимися в наборе безопасных сверток, которые хранятся в локальной базе данных 160 или удаленной базе данных 170, взаимодействующей в том числе с сервером 200 Также локальная база данных 160 хранит журнал событий и контекст событий, обнаруженных ранее, в частности событий загрузки модулей в процессы.

Система, описанная выше, используется для мониторинга исполнения программного обеспечения, установленного на клиентах 100. В результате такого мониторинга обнаруживаются аномальные события в операционной системе, которые могут оказаться результатом эксплуатации уязвимости программного обеспечения, установленного на

клиенте 100. Следует отметить, что понятие клиент 100 в данном изобретении применяется в парадигме связи клиент-сервер[13] ([13]ГОСТ 34.321-96. Информационные технологии. Система стандартов по базам данных. Эталонная модель управления данными), т.е. это элемент вычислительной, а не сетевой архитектуры. Под клиентом 100 в данном изобретении понимается любое компьютерное устройство в сети, запрашивающее посредством агента 110 услуги, обеспечиваемые интерфейсом сервера 200, а под сервером 200 понимается любое компьютерное устройство, с которым взаимодействует упомянутый агент 110 клиента 100, получающее собранные данные от агента 110 и высылающее ему уведомления и команды (тем самым предоставляя услуги клиенту 100). Далее опишем способы, которые осуществляются описанной системой при мониторинге исполнения программного обеспечения, установленного на клиентах 100.

На Фиг. 2 изображен способ обнаружения аномальных событий в операционной системе по результатам оценки популярности свертки этого события. На этапе 210 запускают в операционной системе на стороне клиента 100 агент 110, регистрирующий события, возникающие в операционной системе клиента 100, во время выполнения процессов Далее, на этапе 220, перехватчики 120, установленные в операционной системе клиента 100, и связанными с агентом 110, обнаруживают возникшее в операционной системе событие, об этом сообщается агенту 110, который на этапе 230 регистрирует возникшее событие, и получает при помощи средства сбора 130 контекст указанного события. Контекст по меньшей мере может включать:

- стек вызовов, на момент возникновения события, где из стека вызовов получают по меньшей мере список процедур и функций, выполняемых в данный момент времени, список модулей, содержащих указанные процедуры и функции, а также типы данных и значения всех параметров, передаваемых в модули;
- дамп участка памяти (адресного пространства) процесса, содержащего код, который выполнялся в момент возникновения события;
- информацию о переходах из по меньшей мере LBR, BTS;
- список модулей, загружавшихся в процесс до возникновения события, информация об этом доступна, например, в журнале событий, хранящемся в локальной базе данных 160.

Получив контекст, из него на этапе 240 средством формирования свертки 140 выделяют признаки для формирования на основании этих признаков свертку события. Прежде чем выделить признаки в частном случае необходимо полученный контекст трансформировать (преобразовать). Трансформация контекста - комплекс методов и алгоритмов, направленных на оптимизацию представления и форматов данных с точки зрения решаемых задач и целей анализа. Трансформация контекста не ставит целью изменить информационное содержание данных, которые включает контекст. Цель трансформации представить контекст в таком виде, чтобы они могли быть использована наиболее эффективно (пример трансформации будет дан ниже). Основными способами трансформирования (преобразования) данных являются:

- квантование;
- сортировка;
- слияние (склеивание);
- группировка;
- настройка набора данных;
- табличная подстановка значений;
- вычисляемые значения;

- кодирование данных;
- нормализация (масштабирование).

Например, стек (тоже справедливо и для дампа, ниже для него будет дан пример) вызовов трансформируется следующим образом: получают отладочные символы для модулей, участвующих в стеке вызовов; нормализуют стек вызовов применением отладочных символов.

До применения отладочных символов к результатам дизассемблирования дампа:

```
.text:00428339 loc_428339:                             ; CODE XREF: _wmain+159↑j
.text:00428339                 mov     ecx, [esp+4Ch+var_3C]
.text:0042833D                 push    edi
.text:0042833E                 call    sub_404D95
.text:00428343                 or      [esp+4Ch+var_4], 0FFFFFFFFh
.text:00428348                 lea     ecx, [esp+4Ch+var_38]
.text:0042834C                 call    sub_401274
.text:00428351
.text:00428351 loc_428351:                             ; CODE XREF: _wmain+51↑j
```

После применения отладочных символов:

```
.text:00428339 loc_428339:                             ; CODE XREF: _wmain+159↑j
.text:00428339                 mov     ecx, [esp+4Ch+_pluginsList] ; this
.text:0042833D                 push    edi             ; storage
.text:0042833E                 call    PluginsList::LoadFromStorage(Storage const &)
.text:00428343                 or      [esp+4Ch+var_4], 0FFFFFFFFh
.text:00428348                 lea     ecx, [esp+4Ch+var_38] ; this
.text:0042834C                 call    boost::detail::shared_count::~shared_count(void)
.text:00428351
.text:00428351 loc_428351:                             ; CODE XREF: _wmain+51↑j
```

С помощью дизассемблирования и эмуляции получает набор признаков для полученного дампа (наличие/отсутствие косвенных вызовов (англ. indirect calls), позиционно независимого кода (англ. position independent code), самомодифицирующегося кода (англ. self-modifying code) и т.п.). После трансформации выделяют признаки и формируют свертку. Как указывалось, в качестве признаков используются: имена загруженных модулей и порядок их загрузки (берется из журнала событий), имена процедур и функций, выполняемых в данный момент, значения параметров, передаваемые в модули перед вызовом экспортируемых этими модулями процедур и функций (берется из стека вызовов), информация о переходах (берется из дампа, LBR, BTS), наличие/отсутствие indirect calls, position independent code, self-modifying code (дамп). Свертка может формироваться любым способом, известным из уровня техники. Трансформация контекста, как и формирование свертки могут осуществляться и на стороне клиента 100, и на стороне сервера 200 (это справедливо для всех способов, осуществляемых системой мониторинга исполнения программного обеспечения), для осуществления указанных операций на стороне сервера 200 контекст (для трансформации) и признаки (для формирования свертки) предварительно пересылаются серверу 200 клиентом 100. На этапе 250 определяют популярность свертки события на данный момент времени. Популярность определяют запросом к базе данных (локальной базе данных 160 или удаленной базе данных 170). Под популярностью понимается, вычисленное некоторым способом:

- общее количество обнаружений событий на текущий момент времени, популярность свертки которого определяется; либо
- число клиентов 100 на которых данное событие, популярность свертки которого определяется, было обнаружено на текущий момент времени, независимо от числа обнаружений на клиенте.

Также популярность может быть глобальная (в рамках всех доступных подсетей) или локальная (популярность только в рамках некоторой подсети), например, в рамках той подсети, в которой исследуемое событие обнаружено. Таким образом, базы данных 160 и 170, к которым обращаются для определения популярности свертки события, хранят некоторое число, которое в частном случае вычисляется сервером 200. Популярность может вычисляться любым способом известным из уровня техники. На этапе 260 признают обнаруженное на клиенте 100 событие аномальным, если популярность свертки обнаруженного события на текущий момент времени ниже порогового значения. При этом сценариев может быть несколько, событие признается аномальным, если:

- локальная популярность свертки события ниже порогового значения;
- глобальная популярность свертки события ниже порогового значения;
- локальная и глобальная популярность свертки события ниже порогового значения.

Пороговые значения для глобальной популярности свертки события и популярности в подсети (локальной) задаются независимо. События, признанные аномальными, далее будут дополнительно исследованы и при необходимости заблокированы на клиентах 100, агентом 110.

Система мониторинга исполнения программного обеспечения используется также для формирования сверток безопасных событий (следовательно, данную систему можно использовать для получения свертки от любого события) и набора сверток безопасных событий. Способ формирования набора сверток безопасных событий изображен на Фиг. 3. Под безопасными событиями понимаются события, возникновения которых не является последствием эксплуатации уязвимости или выполнения вредоносного программного обеспечения. На этапе 300 запускают в операционной системе на стороне по меньшей мере одного заведомо безопасного клиента 100 (безопасный клиент, это клиент, который не содержит вредоносного программного обеспечения и не может быть атакован, в процессе осуществления способа, посредством эксплуатации уязвимости) агент 110, регистрирующий события по меньшей мере одного вида, возникающие в операционной системе клиента 100, где видами событий могут являться:

- запуск процессов;
- загрузка модулей;
- файловые операции;
- реестровые операций; и др.

На этапе 310 обнаруживают перехватчиками 120, установленными в операционной системе клиента 100, и связанными с агентом 110, возникшее в операционной системе событие. Далее (этап 320) регистрируют агентом 110 возникшее событие и получают средством сбора 130 агента 110 контекст указанного события, возможный состав контекста приводился выше. Из контекста на этапе 330 средством формирования свертки 140 выделяют признаки и формируют на основе выделенных признаков свертку события, затем добавляют свертку события в набор сверток безопасных событий (этап 340). В частном случае этапы с 330 по 340 выполняются на сервере 200 (для чего контекст, полученный с клиентов 100 на этапе 320, пересылается серверу в «сыром» (англ. raw) или уже трансформированном виде), а набор сверток безопасных событий сохраняется в удаленной базе данных 170 и впоследствии может быть загружен на любой клиент 100, или клиент 100 может организовать запрос к базе 170, не загружая весь набор в локальную базу данных 160.

Наборы сверток безопасных событий используются для обнаружения аномальных событий на клиентах 100. Сам набор может храниться как локально в базе данных 160,

так и удаленно в удаленной базе данных 170, к которой может обратиться клиент 100. Способ обнаружения аномальных событий на основании набора сверток безопасных событий изображен на Фиг. 4. Также, как и во всех других способах, на этапе 400 запускают в операционной системе на стороне клиента 100 агент 110, регистрирующий события, возникающие в операционной системе клиента 100, на этапе 410 обнаруживают перехватчиками 120, установленными в операционной системе, и связанными с агентом 110, возникшее в операционной системе событие. Далее (этап 420) регистрируют возникшее событие агентом 110, и получают контекст указанного события, возможный состав контекста события приводился выше. На этапе 430 выделяют из контекста признаки и формируют на основе выделенных признаков свертку события. Полученную свертку события на этапе 440 сравнивают с множеством заранее сформированных сверток безопасных событий из набора, сформированного способом, описанным выше. На заключительном этапе 450 признают событие аномальным, если при сравнении сформированная свертка обнаруженного события не совпадает ни с одной сверткой события из множества сверток событий из указанного набора.

В настоящем изобретении под агентом 110, перехватчиками 120, средством сбора 130, средством формирования свертки 140, средством сравнения 150 в настоящем изобретении понимается реальные устройства, системы, компоненты, группа компонентов, реализованных с использованием аппаратных средств, таких как интегральные микросхемы (англ. application-specific integrated circuit, ASIC) или программируемой вентильной матрицы (англ. field-programmable gate array, FPGA) или, например, в виде комбинации программных и аппаратных средств, таких как микропроцессорная система и набор программных инструкций, а также на нейроморфных чипах (англ. neurosynaptic chips) Функциональность агента 110, перехватчиков 120, средства сбора 130, средства формирования свертки 140, средства сравнения 150 может быть реализована исключительно аппаратными средствами, а также в виде комбинации, где часть функциональности реализована программными средствами, а часть аппаратными. В некоторых вариантах реализации часть агента 110, перехватчиков 120, средства сбора 130, средства формирования свертки 140, средства сравнения 150 могут быть исполнены на процессоре компьютера общего назначения (например, который изображен на Фиг. 5), тоже относится к клиенту 100 и серверу 200.

Как указывалось, данное изобретение обеспечивает также формирование сверток для любых событий, возникающих в операционной системе.

Фиг. 5 представляет пример компьютерной системы общего назначения, персональный компьютер или сервер 20, содержащий центральный процессор 21, системную память 22 и системную шину 23, которая содержит разные системные компоненты, в том числе память, связанную с центральным процессором 21. Системная шина 23 реализована, как любая известная из уровня техники шинная структура, содержащая в свою очередь память шины или контроллер памяти шины, периферийную шину и локальную шину, которая способна взаимодействовать с любой другой шинной архитектурой. Системная память содержит постоянное запоминающее устройство (ПЗУ) 24, память с произвольным доступом (ОЗУ) 25. Основная система ввода/вывода (BIOS) 26, содержит основные процедуры, которые обеспечивают передачу информации между элементами персонального компьютера 20, например, в момент загрузки операционной системы с использованием ПЗУ 24.

Персональный компьютер 20 в свою очередь содержит жесткий диск 27 для чтения и записи данных, привод магнитных дисков 28 для чтения и записи на сменные магнитные диски 29 и оптический привод 30 для чтения и записи на сменные оптические

диски 31, такие как CD-ROM, DVD-ROM и иные оптические носители информации. Жесткий диск 27, привод магнитных дисков 28, оптический привод 30 соединены с системной шиной 23 через интерфейс жесткого диска 32, интерфейс магнитных дисков 33 и интерфейс оптического привода 34 соответственно. Приводы и соответствующие компьютерные носители информации представляют собой энергонезависимые средства хранения компьютерных инструкций, структур данных, программных модулей и прочих данных персонального компьютера 20.

Настоящее описание раскрывает реализацию системы, которая использует жесткий диск 27, сменный магнитный диск 29 и сменный оптический диск 31, но следует понимать, что возможно применение иных типов компьютерных носителей информации 56, которые способны хранить данные в доступной для чтения компьютером форме (твердотельные накопители, флеш карты памяти, цифровые диски, память с произвольным доступом (ОЗУ) и т.п.), которые подключены к системной шине 23 через контроллер 55.

Компьютер 20 имеет файловую систему 36, где хранится записанная операционная система 35, а также дополнительные программные приложения 37, другие программные модули 38 и данные программ 39. Пользователь имеет возможность вводить команды и информацию в персональный компьютер 20 посредством устройств ввода (клавиатуры 40, манипулятора «мышь» 42). Могут использоваться другие устройства ввода (не отображены): микрофон, джойстик, игровая консоль, сканнер и т.п. Подобные устройства ввода по своему обычаю подключают к компьютерной системе 20 через последовательный порт 46, который в свою очередь подсоединен к системной шине, но могут быть подключены иным способом, например, при помощи параллельного порта, игрового порта или универсальной последовательной шины (USB). Монитор 47 или иной тип устройства отображения также подсоединен к системной шине 23 через интерфейс, такой как видеоадаптер 48. В дополнение к монитору 47, персональный компьютер может быть оснащен другими периферийными устройствами вывода (не отображены), например, колонками, принтером и т.п.

Персональный компьютер 20 способен работать в сетевом окружении, при этом используется сетевое соединение с другим или несколькими удаленными компьютерами 49. Удаленный компьютер (или компьютеры) 49 являются такими же персональными компьютерами или серверами, которые имеют большинство или все упомянутые элементы, отмеченные ранее при описании существа персонального компьютера 20, представленного на Фиг. 5. В вычислительной сети могут присутствовать также и другие устройства, например, маршрутизаторы, сетевые станции, пиринговые устройства или иные сетевые узлы.

Сетевые соединения могут образовывать локальную вычислительную сеть (LAN) 50 и глобальную вычислительную сеть (WAN). Такие сети применяются в корпоративных компьютерных сетях, внутренних сетях компаний и, как правило, имеют доступ к сети Интернет. В LAN- или WAN-сетях персональный компьютер 20 подключен к локальной сети 50 через сетевой адаптер или сетевой интерфейс 51. При использовании сетей персональный компьютер 20 может использовать модем 54 или иные средства обеспечения связи с глобальной вычислительной сетью, такой как Интернет. Модем 54, который является внутренним или внешним устройством, подключен к системной шине 23 посредством последовательного порта 46. Следует уточнить, что сетевые соединения являются лишь примерными и не обязаны отображать точную конфигурацию сети, т.е. в действительности существуют иные способы установления соединения техническими средствами связи одного компьютера с другим.

В заключение следует отметить, что приведенные в описании сведения являются примерами, которые не ограничивают объем настоящего изобретения, определенного формулой. Специалисту в данной области становится понятным, что могут существовать и другие варианты осуществления настоящего изобретения, согласующиеся с сущностью и объемом настоящего изобретения.

(57) Формула изобретения

1. Способ обнаружения аномальных событий в операционной системе компьютерного устройства, в котором:

а) запускают агент, регистрирующий события, возникающие в операционной системе;

б) обнаруживают при помощи по крайней мере одного перехватчика, установленного в операционной системе, возникшее в операционной системе событие;

в) регистрируют событие, обнаруженное перехватчиком, и получают от компьютерного устройства контекст указанного события при помощи агента;

г) выделяют средством формирования свертки из полученного контекста признаки события и формируют на основании выделенных признаков свертку события;

д) определяют средством сравнения популярность сформированной свертки события;

е) признают средством сравнения обнаруженное событие аномальным, если популярность свертки указанного события ниже порогового значения.

2. Способ по п. 1, в котором контекст события есть совокупность состояний операционной системы на момент возникновения события непосредственно повлиявших на его возникновение, при этом события возникают во время выполнения процессов в операционной системе.

3. Способ по п. 1, в котором контекст события включает по меньшей мере:

• стек вызовов на момент возникновения события;

• дамп участка памяти процесса, содержащего код, который выполнялся в момент возникновения события.

4. Способ по п. 3, в котором контекст дополнительно включает информацию о переходах из по меньшей мере Last Branch Record, Branch Trace Store.

5. Способ по п. 3, в котором контекст события дополнительно включает список модулей, загруженных в процесс до возникновения события.

6. Способ по п. 3, в котором из стека вызовов в качестве признаков для формирования свертки получают по меньшей мере список процедур и функций, выполняемых в данный момент времени, список модулей, содержащих указанные процедуры и функции, а также типы данных и значения всех параметров, передаваемых в модули.

7. Способ по п. 3, в котором из дампа в качестве признаков для формирования свертки получают по меньшей мере такие признаки, как: наличие/отсутствие косвенных вызовов, позиционно независимого кода, самомодифицирующегося кода.

8. Способ по пп. 5 и 6, в котором модуль - это программный объект, содержащий код, который расширяет функциональность запущенного процесса.

9. Способ по пп. 5 и 6, в котором модулем является модуль ядра операционной системы или динамическая библиотека.

10. Способ по п. 1 в котором качестве перехватчика по меньшей мере выступает средство трассировки событий, такое как Event tracing for Windows.

11. Способы по п. 1, в котором на шаге в) дополнительно производят преобразование полученного контекста для представления в вид, позволяющий выделить признаки события.

12. Способ по п. 11, в котором способами преобразования являются:

• квантование;
• сортировка;
• слияние (склеивание);
• группировка;
• настройка набора данных;
• табличная подстановка значений;
• вычисляемые значения;
• кодирование данных;
• нормализация (масштабирование).

13. Способ по п. 12, в котором для нормализации получают информацию об отладочных символах для модулей из стека вызовов и дампа.

14. Способ по п. 11, в котором преобразуют контекст с помощью дизассемблирования и эмуляции.

15. Способ по п. 14, в котором выделенные признаки содержат по крайней мере такие признаки как: наличие/отсутствие косвенных вызовов, позиционно независимого кода, самомодифицирующегося кода.

16. Способ по п. 1, в котором популярность свертки определяют в рамках подсети, в которой расположено компьютерное устройство, на котором обнаружено событие.

17. Способ по п. 1, в котором определяют глобальную популярность свертки события.

18. Способ по п. 1, в котором популярность определяется следующим образом:
• запросом к базе данных, содержащей информацию о популярности сверток событий,
• в случае отсутствия информации о популярности в базе данных производится сбор информации из сети о количестве обнаруженных событий и общем количестве клиентов, с которых производится сбор на текущий момент времени, после чего вычисляется популярность.

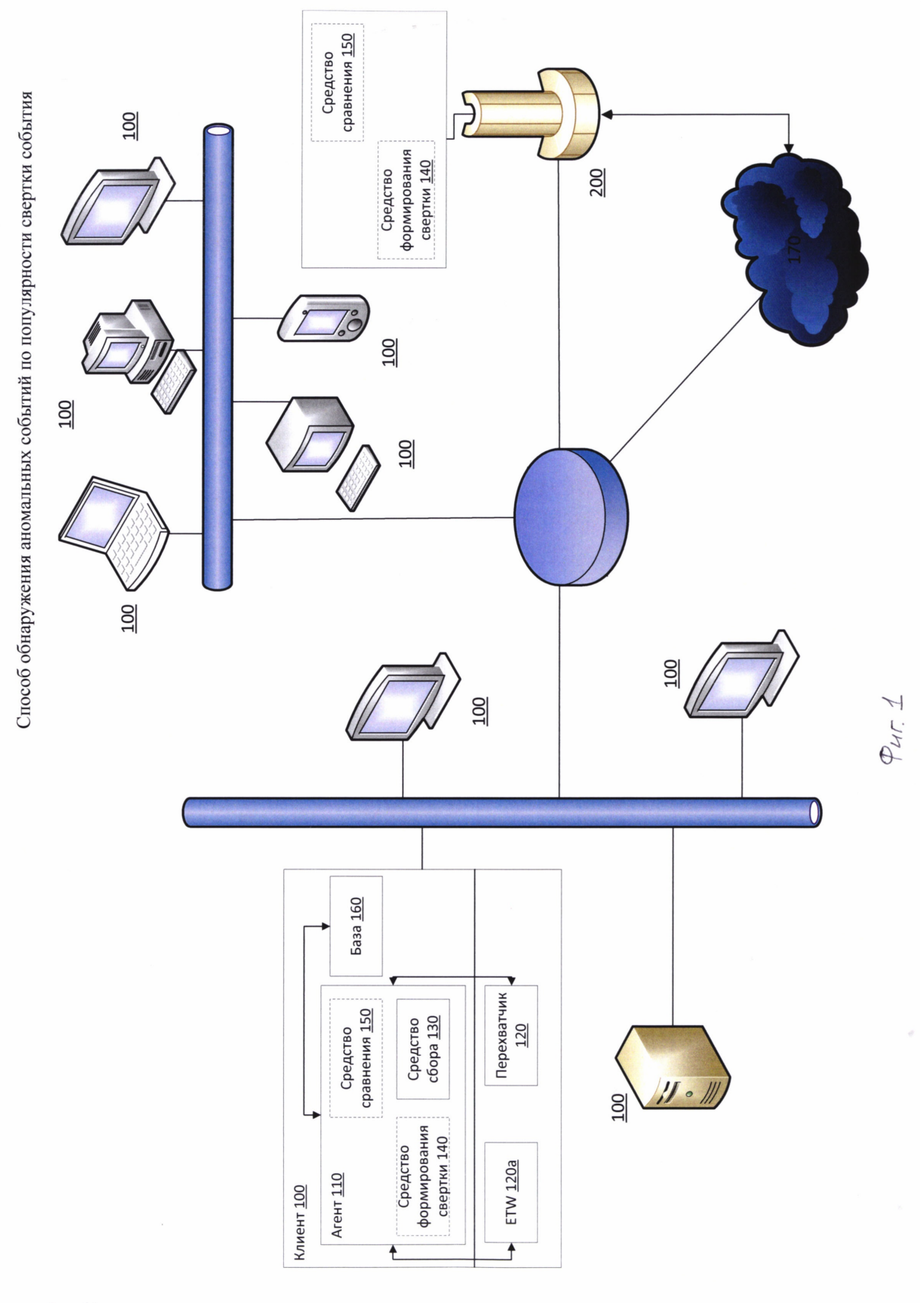
19. Способ по п. 1, в котором этапы с а) по в) выполняются на стороне клиента, а этапы с г) по е) выполняются на стороне сервера, при этом на этапе г) дополнительно получают сервером от агента на клиенте контекст события.

20. Способ по п. 1, в котором в котором этапы с а) по г) выполняются на стороне клиента, а этапы д) и е) выполняются на стороне сервера, при этом на этапе д) получают сервером от агента на клиенте сформированную свертку.

21. Способ по п. 1, в котором свертка представляет собой либо хеш события, либо вектор события в евклидовом пространстве.

22. Способ по п. 1, в котором признаки события, выделяемые из контекста, характеризуют событие и необходимы для формирования свертки.

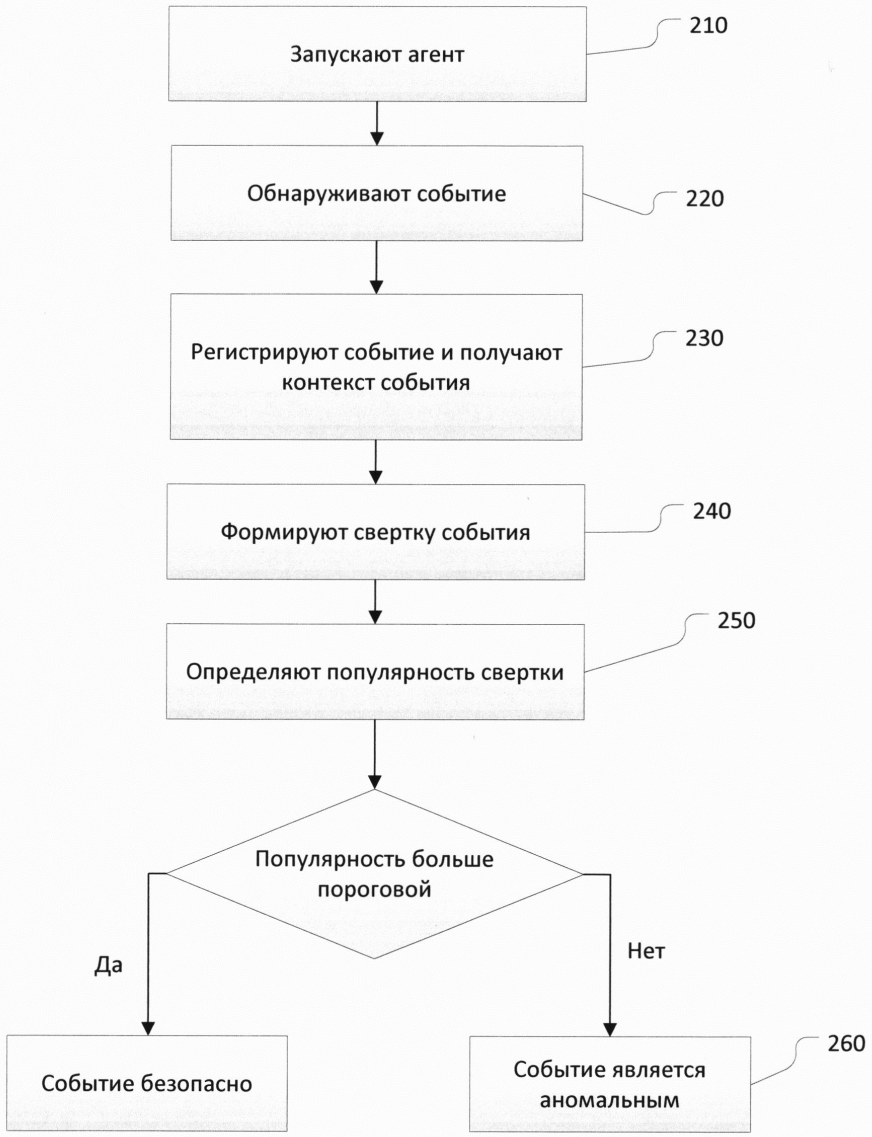
1


Способ обнаружения аномальных событий по популярности свертки события
Средство сравнения 150
Средство формирования свертки 140
200
170
100
Клиент 100
Агент 110
Средство сравнения 150
Средство формирования свертки 140
База 160
Средство сбора 130
Перехватчик 120
ETW 120a

Фиг. 1

2

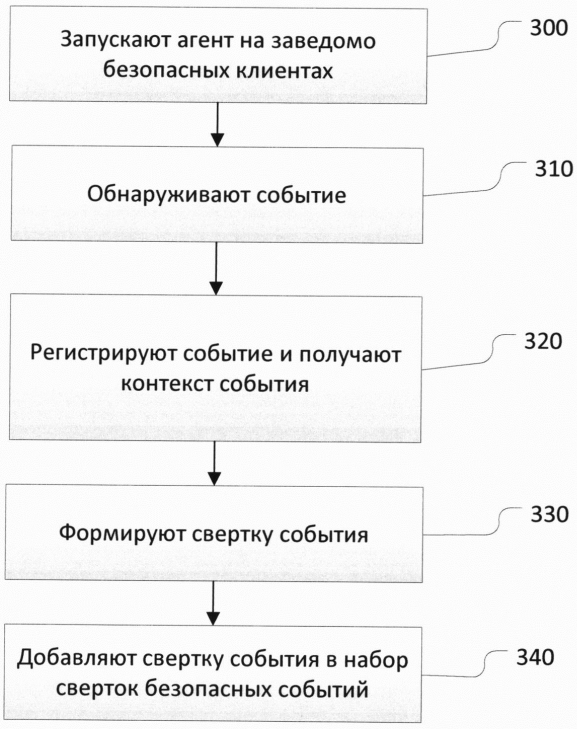
Способ обнаружения аномальных событий по популярности свертки события



Фиг.2

3

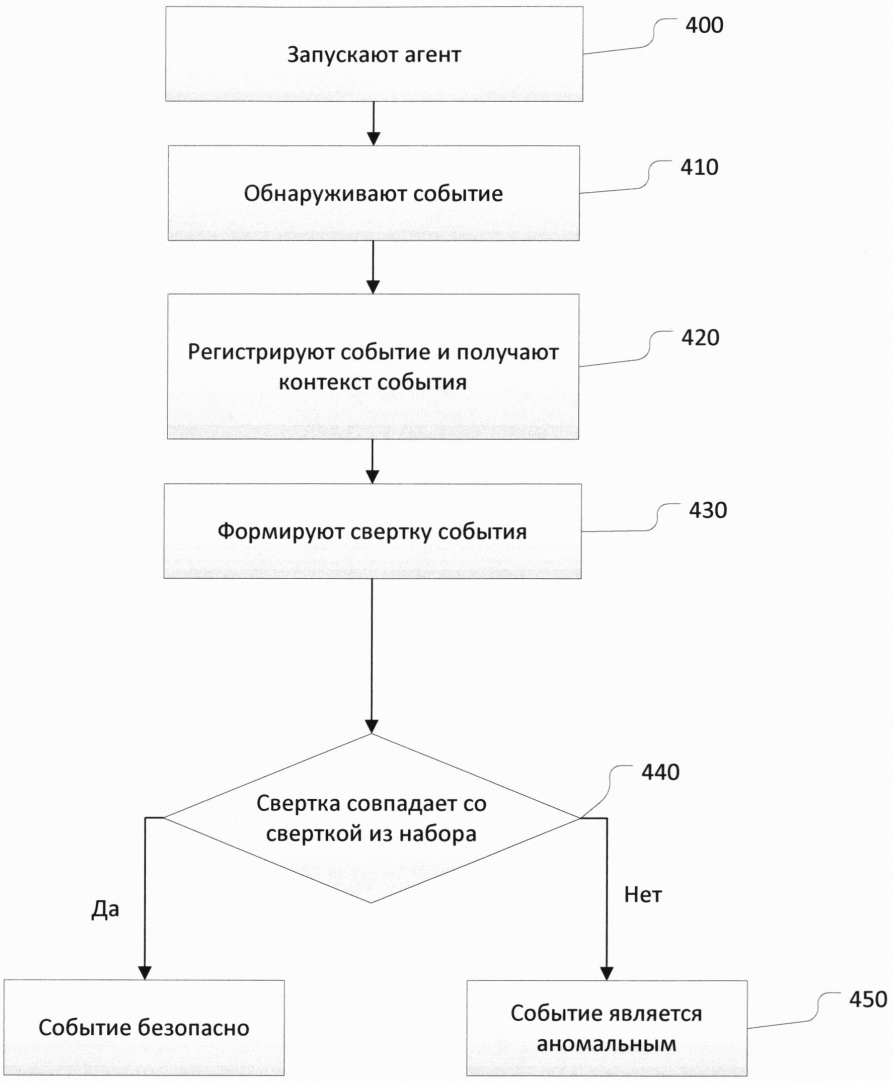
Способ обнаружения аномальных событий по популярности свертки события



Фиг.3

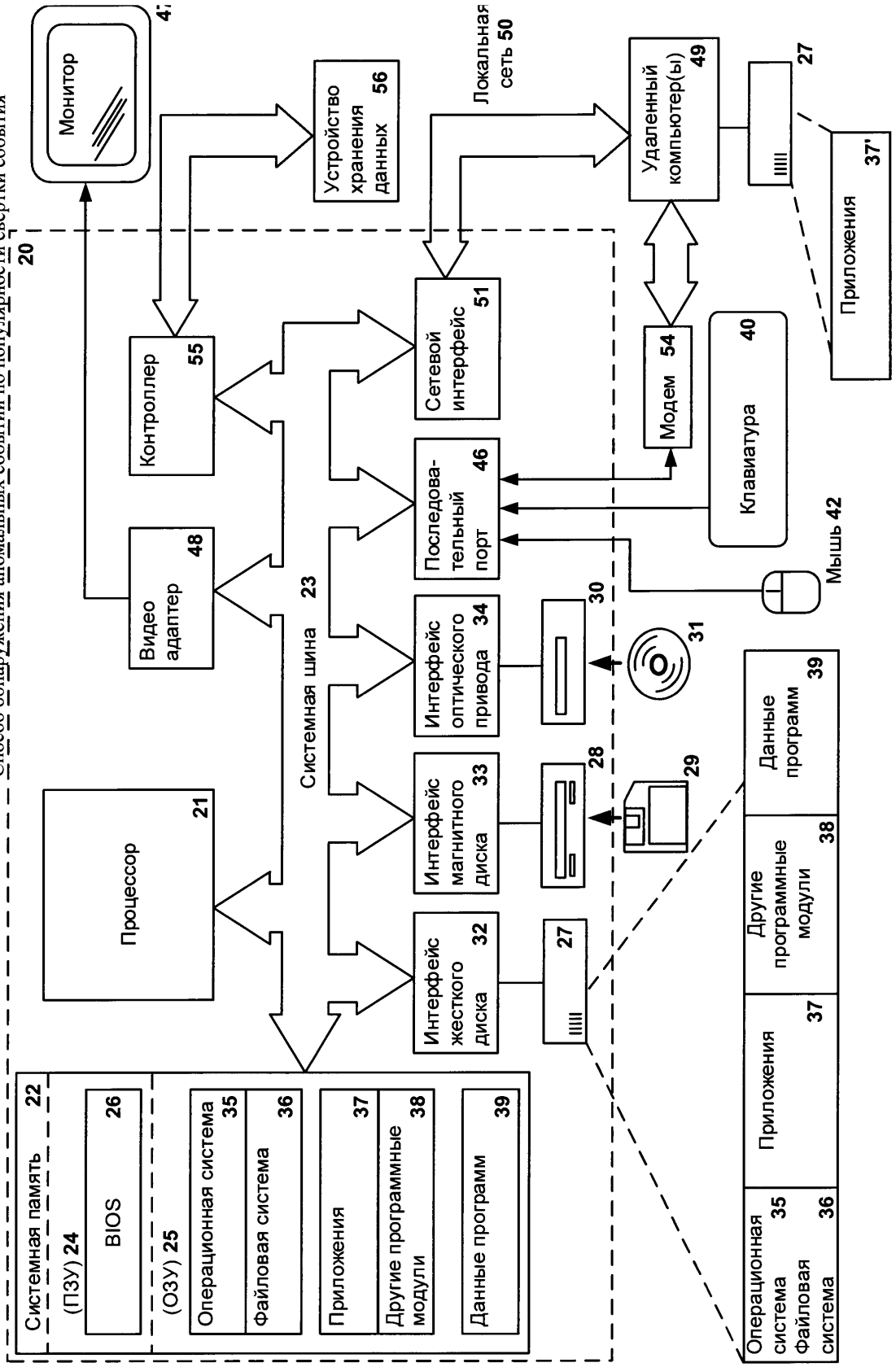
4

Способ обнаружения аномальных событий по популярности свертки события



Фиг.4

5


Способ обнаружения аномальных событий по популярности свертки события
20
Монитор
47
Устройство хранения данных 56
Локальная сеть 50
Удаленный компьютер(ы) 49
27
Приложения 37'
Контроллер 55
Сетевой интерфейс 51
Модем 54
Клавиатура 40
Видео адаптер 48
Последовательный порт 46
Мышь 42
Системная шина 23
Процессор 21
Интерфейс оптического привода 34
30
31
Интерфейс магнитного диска 33
28
29
Интерфейс жесткого диска 32
27
Данные программ 39
Другие программные модули 38
Приложения 37
Операционная система 35
Файловая система 36
Системная память 22
(ПЗУ) 24
BIOS 26
(ОЗУ) 25
Операционная система 35
Файловая система 36
Приложения 37
Другие программные модули 38
Данные программ 39

Фиг.5