РОССИЙСКАЯ ФЕДЕРАЦИЯ

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

(19) **RU** (11) **2 768 196**(13) **C9**

(51) МПК
G06F 21/56 (2013.01)
G06F 21/62 (2013.01)
G06F 21/78 (2013.01)

(12) СКОРРЕКТИРОВАННОЕ ОПИСАНИЕ ИЗОБРЕТЕНИЯ К ПАТЕНТУ

Примечание: библиография отражает состояние при переиздании

(52) СПК
G06F 21/56 *(2022.02);* ***G06F 21/566*** *(2022.02);* ***G06F 21/602*** *(2022.02);* ***G06F 21/6218*** *(2022.02);* ***G06F 21/80*** *(2022.02);* ***G06F 3/0622*** *(2022.02);* ***G06F 3/0623*** *(2022.02);* ***G06F 3/0658*** *(2022.02);* ***G06F 3/0673*** *(2022.02);* ***H04L 63/0428*** *(2022.02)*

(21)(22) Заявка: **2020103936, 31.07.2018**

(24) Дата начала отсчета срока действия патента:
31.07.2018

Приоритет(ы):
(30) Конвенционный приоритет:
04.08.2017 US 62/541,505;
25.07.2018 US 16/045,115

(43) Дата публикации заявки: **06.09.2021** Бюл. № **25**

(45) Опубликовано: **23.03.2022**

(15) Информация о коррекции:
Версия коррекции №1 (W1 C2)

(48) Коррекция опубликована:
13.05.2022 Бюл. № 14

(85) Дата начала рассмотрения заявки PCT на национальной фазе: **04.03.2020**

(86) Заявка PCT:
EP 2018/070692 (31.07.2018)

(87) Публикация заявки PCT:
WO 2019/025423 (07.02.2019)

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

(72) Автор(ы):
ЛУКАКС Сандор (RO),
ТУРИКУ Дан-Кристьян (RO)

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

(56) Список документов, цитированных в отчете о поиске: **WO 2013141838 A1, 26.09.2013. US 20080140724 A, 12.06.2008. US 20090022324 A1, 22.01.2009. WO 2016153602 A1, 29.09.2016. RU 2333608 C2, 10.09.2008.**

(54) ЗАЩИЩЁННОЕ ЗАПОМИНАЮЩЕЕ УСТРОЙСТВО

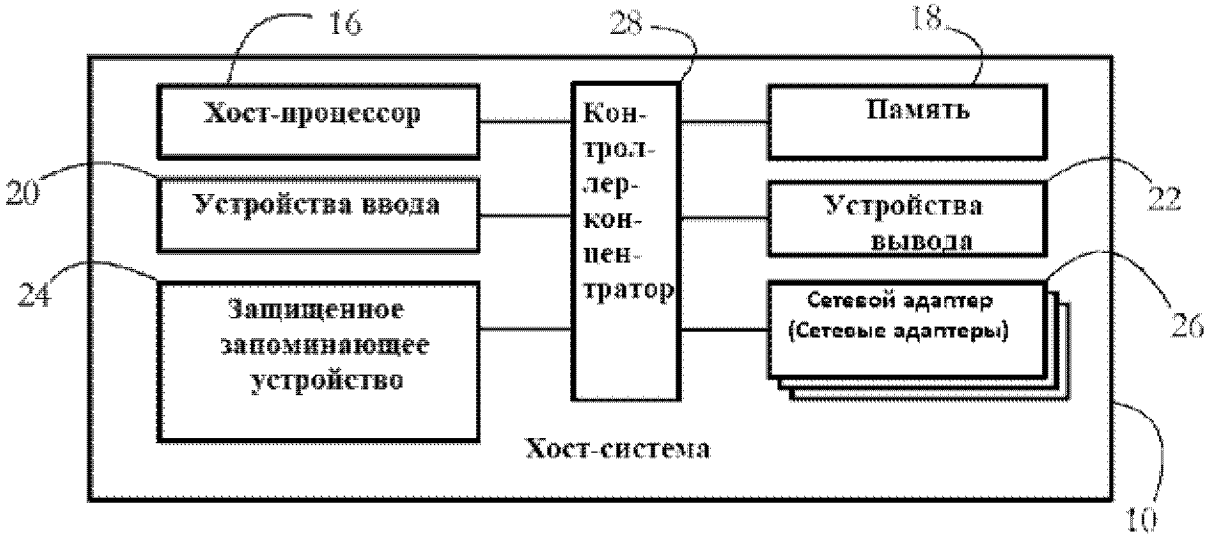
(57) Реферат:

Группа изобретений относится к области компьютерной безопасности. Техническим результатом является повышение надежности защиты данных. Система содержит первый аппаратный процессор и защищенное запоминающее устройство (ЗУ), защищенное ЗУ содержит второй аппаратный процессор и энергонезависимый элемент ЗУ, при этом первый аппаратный процессор сконфигурирован так, что: при обнаружении программным обеспечением, выполняемым на первом аппаратном процессоре, запроса на сохранение в элементе ЗУ пакета данных этот процессор шифрует указанный пакет данных, передает

RU 2768196 C9

RU 2768196 C9

исходный запрос доступа к ЗУ в интерфейс ЗУ, причем данный исходный запрос доступа к ЗУ содержит зашифрованный пакет данных, генерирует имитационный запрос доступа к ЗУ в соответствии с протоколом передачи ЗУ, который содержит по меньшей мере часть криптографического ключа, и передает этот имитационный запрос доступа к ЗУ в интерфейс ЗУ; а второй аппаратный процессор сконфигурирован так, что: после приема сообщения через интерфейс ЗУ процессор определяет, содержит ли сообщение имитационный запрос доступа к ЗУ, если сообщение содержит такой имитационный запрос доступа к ЗУ, то процессор определяет криптографический ключ в соответствии с имитационным запросом доступа к ЗУ, после приема исходного запроса доступа к ЗУ процессор использует криптографический ключ и дешифрует пакет данных, после чего определяет, содержит ли дешифрованный пакет данных вредоносное ПО. 3 н. и 18 з.п. ф-лы, 8 ил.



ФИГ. 1

RU 2768196 C9

RU 2768196 C9

RUSSIAN FEDERATION

FEDERAL SERVICE
FOR INTELLECTUAL PROPERTY

(19) **RU** (11) **2 768 196**(13) **C9**

(51) Int. Cl.
G06F 21/56 (2013.01)
G06F 21/62 (2013.01)
G06F 21/78 (2013.01)

(12) ABSTRACT OF INVENTION

Note: Bibliography reflects the latest situation

(52) CPC
G06F 21/56 *(2022.02);* ***G06F 21/566*** *(2022.02);* ***G06F 21/602*** *(2022.02);* ***G06F 21/6218*** *(2022.02);* ***G06F 21/80*** *(2022.02);* ***G06F 3/0622*** *(2022.02);* ***G06F 3/0623*** *(2022.02);* ***G06F 3/0658*** *(2022.02);* ***G06F 3/0673*** *(2022.02);* ***H04L 63/0428*** *(2022.02)*

(21)(22) Application: **2020103936, 31.07.2018**

(24) Effective date for property rights:
31.07.2018

Priority:
(30) Convention priority:
04.08.2017 US 62/541,505;
25.07.2018 US 16/045,115

(43) Application published: **06.09.2021** Bull. № **25**

(45) Date of publication: **23.03.2022**

(15) Correction information:
Corrected version no1 (W1 C2)

(48) Corrigendum issued on:
13.05.2022 Bull. № 14

(85) Commencement of national phase: **04.03.2020**

(86) PCT application:
EP 2018/070692 (31.07.2018)

(87) PCT publication:
WO 2019/025423 (07.02.2019)

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

(72) Inventor(s):
LUKAKS Sandor (RO),
TURIKU Dan-Cristian (RO)

(73) Proprietor(s):
BITDEFENDER IPR MANAGEMENT LTD (CY)

RU 2768196 C9

RU 2768196 C9

(54) PROTECTED STORAGE DEVICE

(57) Abstract:

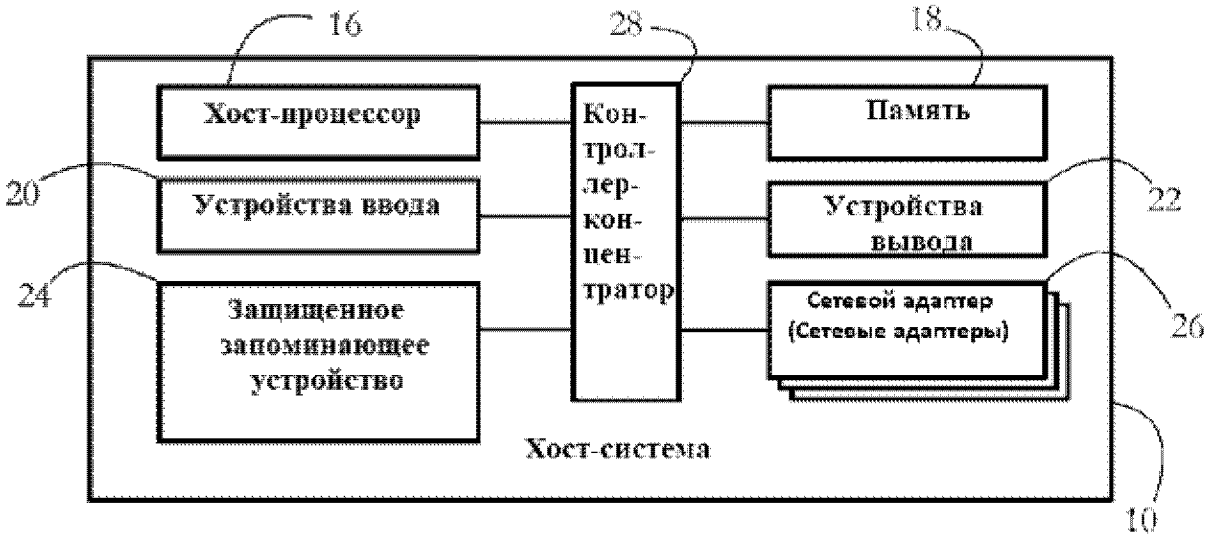
FIELD: computer security.

SUBSTANCE: system contains the first hardware processor and a protected storage device (hereinafter – SD). The protected SD contains the second hardware processor and a non-volatile SD element. The first hardware processor is configured so that: when the software running on the first hardware processor detects a request to save a data packet in the SD element, this processor encrypts the specified data packet, transmits an initial SD access request to a SD interface, wherein this initial SD access request contains an encrypted data packet, generates a simulated SD access request in accordance with a SD transfer protocol, which contains at least part of a cryptographic key, and transmits this simulated SD access request to the SD interface. The second hardware processor is configured so that: after

receiving a message through the SD interface, the processor determines whether the message contains a simulated SD access request, if the message contains such a simulated SD access request, the processor determines a cryptographic key in accordance with the simulated SD access request, after receiving the initial SD access request, the processor uses the cryptographic key and decrypts the data packet, after which it determines whether the decrypted data packet contains malware.

EFFECT: increase in the reliability of data protection.

21 cl, 8 dwg



ФИГ. 1

RU 2768196 C9

RU 2768196 C9

Родственные заявки

[0001] Настоящая заявка испрашивает приоритет по дате подачи предварительной заявки на патент США №62541505, поданной 08.04.2017, озаглавленной «Защищенное запоминающее устройство», все содержание которой включено в настоящий документ посредством ссылки.

Уровень техники

[0001] Настоящее изобретение относится к компьютерной безопасности, а именно к защите компьютера от вредоносного программного обеспечения (ПО) (от вредоносных программ).

[0002] Воздействие вредоносного ПО сказывается на множестве компьютерных систем по всему миру. Во многих своих формах, таких как компьютерные вирусы, черви, руткиты и шпионские программы, вредоносные программы для миллионов пользователей компьютеров создают серьезную опасность потери данных, утраты конфиденциальной информации, хищения идентификационных сведений и снижения производительности. Программа-вымогатель - это особо опасный тип вредоносных программ, шифрующих набор файлов, найденных на носителе данных, например, на жестком диске компьютера, и затем требующих от владельца файлов заплатить за восстановление соответствующих данных.

[0003] Для обнаружения и/или нейтрализации вредоносного ПО используют антивирусные разработки. Однако вредоносные программы применяют различные стратегии, чтобы избежать обнаружения. Одной из таких стратегий является обфускация (умышленное запутывание), например, шифрование вредоносного кода или использование слегка различающихся версий кода на каждом зараженном компьютере (такой подход обычно называют полиморфизмом). Другим примером способа предотвращения обнаружения является такое распределение вредоносных действий между несколькими программами-агентами, когда каждый агент выполняет некоторый отдельный набор действий, который нельзя считать указывающим на вредоносное ПО, если его исполнение происходит автономно.

[0004] Существует серьезная заинтересованность в разработке надежных систем и способов защиты цифровых данных от хищения и несанкционированного изменения, в том числе с помощью вредоносного ПО.

Раскрытие сущности изобретения

[0005] В соответствии с одним аспектом компьютерная система содержит первый аппаратный процессор и защищенное запоминающее устройство (ЗУ), причем защищенное ЗУ подключено к первому аппаратному процессору через интерфейс ЗУ, сконфигурированный для приема запросов доступа к ЗУ, отформатированных в соответствии с протоколом передачи ЗУ. Защищенное ЗУ содержит второй аппаратный процессор и энергонезависимый элемент ЗУ. Первый аппаратный процессор сконфигурирован так, что при обнаружении программным обеспечением, выполняемым на первом аппаратном процессоре, запроса на сохранение в элементе ЗУ некоторого пакета данных этот процессор шифрует этот пакет данных. Первый аппаратный процессор дополнительно сконфигурирован так, что после шифрования пакета данных процессор передает исходный запрос доступа к ЗУ в интерфейс ЗУ, причем данный исходный запрос доступа к ЗУ содержит зашифрованный пакет данных. Первый аппаратный процессор дополнительно сконфигурирован так, что он генерирует имитационный запрос доступа к ЗУ в соответствии с протоколом передачи ЗУ, причем данный имитационный запрос доступа к ЗУ содержит, по меньшей мере, часть криптографического ключа, и передает этот имитационный запрос доступа к ЗУ в

интерфейс ЗУ. Второй аппаратный процессор сконфигурирован так, что после приема сообщения через интерфейс ЗУ процессор определяет, содержит ли данное сообщение имитационный запрос доступа к ЗУ. Затем, если сообщение содержит такой имитационный запрос доступа к ЗУ, то второй аппаратный процессор определяет криптографический ключ в соответствии с имитационным запросом доступа к ЗУ. Второй аппаратный процессор дополнительно сконфигурирован так, что после приема исходного запроса доступа к ЗУ процессор использует криптографический ключ и дешифрует пакет данных, после чего определяет, содержит ли дешифрованный пакет данных вредоносное ПО.

[0006] В соответствии с другим аспектом защищенное ЗУ содержит первый аппаратный процессор и энергонезависимый элемент ЗУ, причем защищенное ЗУ сконфигурировано для подсоединения ко второму аппаратному процессору через интерфейс ЗУ, сконфигурированный для приема запросов доступа к ЗУ, отформатированных в соответствии с протоколом передачи ЗУ. Второй аппаратный процессор сконфигурирован так, что при обнаружении программным обеспечением, выполняемым на втором аппаратном процессоре, запроса на сохранение в элементе ЗУ некоторого пакета данных этот процессор шифрует этот пакет данных. Второй аппаратный процессор дополнительно сконфигурирован так, что после шифрования пакета данных процессор передает исходный запрос доступа к ЗУ в интерфейс ЗУ, причем этот исходный запрос доступа к ЗУ содержит зашифрованный пакет данных. Второй аппаратный процессор дополнительно сконфигурирован так, что процессор генерирует имитационный запрос доступа к ЗУ в соответствии с протоколом передачи ЗУ, причем имитационный запрос доступа к ЗУ содержит, по меньшей мере, часть криптографического ключа, и передает этот имитационный запрос доступа к ЗУ в интерфейс ЗУ. Первый аппаратный процессор сконфигурирован так, что после приема сообщения через интерфейс ЗУ процессор определяет, содержит ли это сообщение имитационный запрос доступа к ЗУ. Затем, если сообщение содержит такой имитационный запрос доступа к ЗУ, то первый аппаратный процессор определяет криптографический ключ в соответствии с имитационным запросом доступа к ЗУ. Первый аппаратный процессор дополнительно сконфигурирован так, что после приема исходного запроса доступа к ЗУ процессор использует криптографический ключ и дешифрует пакет данных, после чего определяет, содержит ли дешифрованный пакет данных вредоносное ПО.

[0007] В соответствии с другим аспектом способ обеспечения компьютерной безопасности содержит подсоединение защищенного ЗУ к первому аппаратному процессору через интерфейс ЗУ, сконфигурированный для приема запросов доступа к ЗУ, отформатированных в соответствии с протоколом передачи ЗУ, причем защищенное ЗУ содержит второй аппаратный процессор и энергонезависимый элемент ЗУ. Способ дополнительно содержит этап, на котором при обнаружении программным обеспечением, выполняемым на первом аппаратном процессоре, запроса происходит сохранение пакета данных в элементе ЗУ при использовании первого аппаратного процессора для шифрования пакета данных. Способ дополнительно содержит этап, на котором после шифрования пакета данных первый аппаратный процессор передает исходный запрос доступа к ЗУ в интерфейс ЗУ, причем исходный запрос доступа к ЗУ содержит зашифрованный пакет данных. Способ дополнительно содержит использование первого аппаратного процессора для генерации имитационного запроса доступа к ЗУ в соответствии с протоколом передачи ЗУ, причем имитационный запрос доступа к ЗУ содержит, по меньшей мере, часть криптографического ключа, и

использование первого аппаратного процессора для передачи этого имитационного запроса доступа к ЗУ в интерфейс ЗУ. Способ дополнительно содержит этап, на котором после приема сообщения через интерфейс ЗУ второй аппаратный процессор определяет, содержит ли это сообщение имитационный запрос доступа к ЗУ. Затем, если сообщение содержит имитационный запрос доступа к ЗУ, то способ дополнительно содержит этап использования второго аппаратного процессора с целью применения криптографического ключа для дешифрования пакета данных и этап использования второго аппаратного процессора с целью определения, содержит ли дешифрованный пакет данных вредоносное ПО.

Краткое описание чертежей

[0008] Вышеизложенные аспекты и преимущества настоящего изобретения станут более понятными из нижеприведенного подробного описания со ссылками на чертежи, на которых:

[0009] На Фиг.1 проиллюстрирована представленная в качестве примера конфигурация аппаратных средств хост-системы, защищенной от угроз для компьютерной безопасности в соответствии с некоторыми вариантами осуществления настоящего изобретения.

[0010] На Фиг.2 показана представленная в качестве примера конфигурация аппаратного обеспечения защищенного ЗУ в соответствии с некоторыми вариантами осуществления настоящего изобретения.

[0011] На Фиг.3-А показаны представленные в качестве примера программные компоненты, работающие в защищенной хост-системе в соответствии с некоторыми вариантами осуществления настоящего изобретения.

[0012] На Фиг.3-В показан альтернативный набор программных компонентов в соответствии с некоторыми вариантами осуществления настоящего изобретения.

[0013] На Фиг.4 показаны представленные в качестве примера компоненты ПО, работающие в защищенном ЗУ в соответствии с некоторыми вариантами осуществления настоящего изобретения.

[0014] На Фиг.5 показан представленный в качестве примера набор этапов, выполняемых модулем компьютерной безопасности (МКБ) (CSM) с фиг.3-А-В в соответствии с некоторыми вариантами осуществления настоящего изобретения.

[0015] На Фиг.6 показана представленная в качестве примера последовательность этапов, выполняемых программным обеспечением на защищенном ЗУ, в соответствии с некоторыми вариантами осуществления настоящего изобретения.

[0016] На Фиг.7 показана представленная в качестве примера последовательность этапов, выполняемых программным обеспечением на защищенном ЗУ для поддержания семантического отображения ЗУ в соответствии с некоторыми вариантами осуществления настоящего изобретения.

Осуществление изобретения

[0017] В последующем описании подразумевается, что все перечисленные соединения между структурами могут быть прямыми рабочими соединениями или косвенными рабочими соединениями через промежуточные структуры. Набор элементов включает в себя один или несколько элементов. Любое упоминание какого-либо элемента понимается как ссылка на по меньшей мере один элемент. Термин «несколько элементов» означает «по меньшей мере два элемента». Если не требуется иное, то выполнение любых описанных этапов способа не обязательно должно происходить в показанном конкретном порядке. Первый элемент (например, данные), полученный из второго элемента, включает в себя первый элемент, равный второму элементу, а

также первый элемент, созданный посредством обработки второго элемента и - опционально - других данных. Принятие некоторого определения или решения в соответствии с некоторым параметром включает в себя принятие определения или решения в соответствии с указанным параметром и - опционально - в соответствии с другими данными. Если не указано иное, то индикатором некоторого количества / некоторых данных может быть само количество / сами данные или индикатор, отличный от самого количества / самих данных. Компьютерная программа - это последовательность команд процессора, выполняющая некоторую задачу. Компьютерные программы, описанные в некоторых вариантах осуществления настоящего изобретения, могут быть автономными программными объектами или подобъектами (например, подпрограммы, библиотеки) других компьютерных программ. Машиночитаемые носители данных включают в себя энергонезависимые носители, такие как магнитные, оптические и полупроводниковые носители данных (например, жесткие диски, оптические диски, флеш-память, динамическое ОЗУ (DRAM)), а также каналы связи, такие как проводящие кабели и оптоволоконные линии. В соответствии с некоторыми вариантами осуществления настоящее изобретение предоставляет, помимо прочего, компьютерные системы, содержащие аппаратные средства (например, один или несколько процессоров), запрограммированные для выполнения способов, описанных в настоящем документе, а также инструкции по кодированию машиночитаемого носителя для выполнения способов, описанных в настоящем документе.

[0018] Приведенное далее описание иллюстрирует варианты осуществления изобретения в качестве примера и не накладывает какого-либо ограничения.

[0019] На Фиг.1 показана представленная в качестве примера конфигурация аппаратных средств хост-системы 10, защищенной от угроз для компьютерной безопасности в соответствии с некоторыми вариантами осуществления настоящего изобретения. Представленные в качестве примера хост-системы 10 включают в себя компьютеры (например, персональный компьютер, корпоративный сервер и т.д.), мобильные вычислительные устройства (например, ноутбуки, планшетные ПК), телекоммуникационные устройства (например, смартфоны), цифровые устройства (телевизоры, игровые приставки и т.д.), носимые вычислительные устройства (например, смарт-часы) или любое другое электронное устройство, имеющее процессор и память и требующее защиты компьютерной безопасности. Для простоты проиллюстрированная хост-система представляет собой компьютерную систему; аппаратная конфигурация других хост-систем, таких как мобильные телефоны, смарт-часы и т.д., может несколько отличаться от показанной.

[0020] Хост-система 10 содержит набор физических устройств, включая аппаратный процессор 16 и элемент 18 памяти. В некоторых вариантах осуществления процессор 16 содержит физическое устройство (например, микропроцессор, многоядерную интегральную схему, сформированную на полупроводниковой подложке и т.д.), сконфигурированное для выполнения вычислительных и/или логических операций с набором сигналов и/или данных. В некоторых вариантах осуществления такие операции поступают в процессор 16 в форме последовательности команд процессора (например, машинный код или иной тип кодирования). Элемент 18 памяти может содержать энергозависимый машиночитаемый носитель (например, динамическое ОЗУ (DRAM), статическое ОЗУ (SRAM)), хранящий команды и/или данные, к которым обращается или которые генерирует процессор 16.

[0021] Устройства 20 ввода могут включать в себя устройства, позволяющие

пользователю вводить данные и/или команды в хост-систему 10, а также содержать соответствующие аппаратные интерфейсы и/или адаптеры, делающие возможным такой ввод. Представленные в качестве примера устройства ввода включают в себя, помимо прочего, кнопку, клавиатуру, мышь, джойстик, сенсорный экран, микрофон, камеру, игровой контроллер, систему обнаружения жестов и датчик обнаружения движения. Устройства 22 вывода могут включать в себя устройства отображения, такие как мониторы и динамики и прочее, а также аппаратные интерфейсы / адаптеры, такие как графические карты, позволяющие хост-системе 10 передавать данные пользователю. В некоторых вариантах осуществления устройства 20 ввода и устройства 22 вывода могут совместно использовать общую аппаратную часть, как в случае устройств с сенсорным экраном. Сетевой адаптер (сетевые адаптеры) 26 позволяет (позволяют) хост-системе 10 подключаться к коммуникационной сети, такой как локальная вычислительная сеть (LAN) и/или к другим устройствам / компьютерным системам.

[0022] Защищенное запоминающее устройство (ЗУ) 24 включает в себя машиночитаемый носитель, обеспечивающий энергонезависимое хранение, чтение и запись команд программного обеспечения и/или других данных. Представленные в качестве примера запоминающие устройства 24 включают в себя магнитные, оптические устройства и устройства флеш-памяти, а также съемные носители, такие как компакт-диски и/или DVD-диски и дисководы. В некоторых вариантах осуществления защищенное ЗУ 24 наделено другими аппаратными и программными компонентами, специально сконфигурированными для повышения безопасности хранимых данных, как подробно показано ниже.

[0023] В некоторых вариантах осуществления контроллер-концентратор 28 включает в себя несколько системных, периферийных и/или микросхемных шин и/или все другие схемы, обеспечивающие связь между процессором 16 и устройствами 18, 20, 22, 24 и 26. Представленные в качестве примера компоненты контроллера-концентратора 28 включают в себя контроллер памяти, контроллер ввода-вывода (I/O) и контроллер прерываний. В зависимости от изготовителя аппаратного обеспечения и от устройства некоторые или все такие контроллеры могут быть выполнены в виде единой интегральной схемы и/или могут быть частью процессора 16. В некоторых вариантах осуществления некоторые другие устройства, такие как графические адаптеры, являющиеся частью устройств 22 вывода, также могут быть частью процессора 16.

[0024] На Фиг.2 показана представленная в качестве примера конфигурация аппаратного обеспечения защищенного ЗУ 24 в соответствии с некоторыми вариантами осуществления настоящего изобретения. Некоторые варианты осуществления снижают опасность, создаваемую вредоносным ПО, путем сопряжения обычного ЗУ - первичного ЗУ 30 (например, магнитный или твердотельный накопитель) - с выделенным процессором 116 системы обеспечения безопасности, отделенным от первичного процессора хост-системы 10. Этот вспомогательный процессор может быть выполнен как одно целое с элементом ЗУ и/или дополнительным аппаратным обеспечением на общей печатной плате, имеющей конструктивные параметры обычного жесткого диска. В другом представленном в качестве примера варианте осуществления защищенное ЗУ 24 может быть выполнено как внешнее ЗУ (например, флеш-накопитель, внешний жесткий диск), подключаемое к хост-системе через обычный интерфейс, такой как универсальная последовательная шина (USB) или интерфейс/разъем Thunderbolt®.

[0025] Первичное ЗУ 30 может действовать как фактическое ЗУ для хост-системы 10. В качестве такового, первичное ЗУ 30 может хранить код и/или данные, принадлежащие операционной системе и/или другим программным приложениям,

выполняемым на хост-процессоре 16, а также данные пользователя, такие как документы, изображения, звуковые файлы и т.д.

[0026] Процессор 116 системы обеспечения безопасности содержит физическую электронную схему (например, интегральную схему, сформированную на полупроводниковой подложке), сконфигурированную для выполнения математических и/или логических операций с набором сигналов и/или данных. В некоторых вариантах процессор 116 представляет собой универсальный микропроцессор типа, используемого в качестве центрального процессора (ЦП) в персональных компьютерах и/или мобильных телефонах. Примерами таких процессоров являются обычные процессоры таких производителей, как Intel®, AMD® и ARM®. В альтернативном варианте процессор 116 содержит заказную электронную схему, такую как специализированная интегральная схема (ASIC) или программируемая в условиях эксплуатации матрица логических элементов (FPGA). Другие варианты осуществления процессора 116 включают в себя графический процессор (GPU) или комбинацию указанных выше компонентов. Использование таких специализированных процессоров может быть выгодно тем, что их архитектура может быть специально разработана и оптимизирована для параллельных вычислений и определенных специализированных задач. Параллельная специализированная архитектура может быть полезной для таких приложений, как шифрование/дешифрование, что дополнительно описано ниже.

[0027] Дополнительные аппаратные компоненты защищенного ЗУ 24 могут включать в себя память 118 системы обеспечения безопасности, содержащую энергозависимые машиночитаемые носители (например, DRAM, SRAM) для хранения команд и/или данных, к которым обращается или которые генерирует процессор 116. Устройство 24 может дополнительно содержать контроллер 34 системы обеспечения безопасности, в общем представляющий шины и/или все другие схемы, соединяющие аппаратные компоненты устройства 24. Защищенное ЗУ 24 может быть дополнительно подключено к контроллеру-концентратору 28 хост-системы 10 через интерфейс 36 ЗУ (последовательный интерфейс ATA (Serial AT Attachment) - SATA или и/или последовательный интерфейс периферии PCI Express).

[0028] В некоторых вариантах осуществления защищенное ЗУ 24 может дополнительно включать в себя вторичное ЗУ 32, содержащее энергонезависимый машиночитаемый носитель. В некоторых вариантах осуществления первичное ЗУ 30 и вторичное ЗУ 32 реализованы в виде отдельных логических разделов одного физического ЗУ - магнитного, оптического или твердотельного накопителя. Вторичное ЗУ 32 может быть невидимым для ПО, выполняемого на процессоре 16, и может быть доступно только для ПО, выполняемого на вспомогательном процессоре 116 системы обеспечения безопасности. Такая изоляция может быть достигнута с использованием аппаратной логики (схемы контроллера 34 системы обеспечения безопасности), выполненной так, чтобы предоставить хост-процессору 16 лишь некоторый ограниченный диапазон адресов ЗУ.

[0029] Вторичное ЗУ 32 может быть использовано для хранения кода системы обеспечения безопасности, а также, кроме прочего, таких данных как сигнатуры, указывающие на вредоносное ПО. Вторичное ЗУ 32 может дополнительно хранить код, который должен быть выполнен процессором 116 при запуске (загрузке). Некоторые варианты осуществления защищенного ЗУ 24 используют вторичное ЗУ 32 для хранения кодирования семантического отображения файловой системы, как более подробно показано ниже. Другие варианты осуществления ЗУ 32 могут хранить частичные копии (например, резервные копии) данных, хранимых в первичном ЗУ 30.

[0030] В некоторых вариантах осуществления первичное ЗУ 30 и вторичное ЗУ 32 являются адресуемыми через набор индикаторов местоположения (адресов). ЗУ 30, 32 в зависимости от реализации может быть разделено на отдельные адресуемые компоненты, например, секторы, блоки и/или кластеры.

[0031] На фиг.3-А показаны представленные в качестве примера программные компоненты, выполняемые на хост-процессоре 16 в соответствии с некоторыми вариантами осуществления настоящего изобретения. Операционная система (ОС) 40 содержит набор компьютерных программ, обеспечивающих интерфейс между аппаратным обеспечением хост-системы 10 и другим ПО, таким как приложения 41a-b. ОС 40 может включать в себя любую широкодоступную операционную систему, такую как Windows®, MacOS®, Linux®, iOS®, Android® и другие. Приложения 41a-b, в общем, представляют собой любые компьютерные программы, в т.ч. текстовые процессоры, приложения для работы с электронными таблицами, приложения для обработки изображений, игры, браузеры и приложения для электронной коммуникации.

[0032] Некоторые варианты осуществления настоящего изобретения дополнительно включают в себя модуль 44 компьютерной безопасности (МКБ), содержащий программное обеспечение, защищающее хост-систему 10 от вредоносных программ. Модуль МКБ 44 может включать в себя, например, компьютерные программы для обнаружения вредоносного ПО и/или компьютерные программы для деактивации такого ПО. Такие компоненты могут использовать любой способ, известный в области компьютерной безопасности. В некоторых вариантах осуществления модуль МКБ 44 дополнительно содержит компонент 42 промежуточного блока ЗУ, работающий в качестве интерфейса между ОС 40 и защищенным ЗУ 24. Представленный в качестве примера промежуточный блок 42 ЗУ может работать как драйвер ЗУ, позволяющий считывать данные из первичного ЗУ 30 и записывать данные в него. Промежуточный блок 42 ЗУ может быть дополнительно сконфигурирован для обмена сообщениями и/или данными с ПО, выполняемым на процессоре 116 системы обеспечения безопасности, как более подробно показано ниже.

[0033] На фиг.3-В показан альтернативный вариант осуществления, работающий в аппаратной среде виртуализации, например, в приложении облачных вычислений. Виртуальная машина (ВМ) - это термин, используемый в данной области техники для описания эмуляции реальной физической машины / компьютерной системы; виртуальная машина может работать с операционной системой и другими приложениями. Гипервизор включает в себя программное обеспечение, предназначенное для создания или включения множества виртуализированных устройств, таких как виртуальный процессор и виртуальное устройство управления памятью (MMU), и для предоставления таких виртуализированных устройств программному обеспечению вместо реальных физических устройств хост-системы 10. Такие операции обычно известны как раскрытие виртуальной машины. Гипервизоры могут разрешать совместное использование аппаратных ресурсов хост-системы 10 несколькими виртуальными машинами и могут дополнительно управлять таким мультиплексированием так, что каждая виртуальная машина работает независимо и не знает о других виртуальных машинах, работающих параллельно на соответствующем хосте. Примерами популярных гипервизоров являются, помимо прочего, VMware vSphere™ от VMware Inc. и гипервизор Xen с открытым исходным кодом. В настоящем описании подразумевается, что ПО выполняется на виртуальной машине, когда оно работает на виртуальном процессоре соответствующей виртуальной машины.

[0034] В представленной в качестве примера конфигурации, показанной на фиг.3-В,

гипервизор 46 предоставляет гостевую виртуальную машину (ВМ) (VM) 48a. Гостевая ВМ 48a работает с операционной системой и набором пользовательских приложений, в то время как модуль МКБ 44 действует на выделенной виртуальной машине 48b системы обеспечения безопасности, отделенной от гостевой ВМ 48a. В альтернативном варианте осуществления, показанном на фиг.3-B, модуль МКБ 44 и/или промежуточный блок 42 ЗУ могут содержать набор библиотек, загружаемых/вызываемых гипервизором 46. Как таковой, модуль МКБ 44 и/или промежуточный блок 42 ЗУ могут выполняться ниже гостевой ВМ 48a на уровне привилегий процессора гипервизора 46 (например, кольцо-1 (ring-1) или VMXroot на платформах Intel®). Конфигурации, такие как показанные на фиг.3-B, могут быть предпочтительнее конфигурации, проиллюстрированной на фиг.3-A, из-за повышенной безопасности. Виртуальные среды гостевой ВМ 48a и ВМ 48b системы обеспечения безопасности могут быть относительно хорошо изолированы друг от друга, поэтому вредоносное ПО, выполняемое на гостевой ВМ 48a, не сможет заразить или иным образом помешать программному обеспечению, выполняемому на ВМ 48b системы обеспечения безопасности.

[0035] На Фиг.4 показан представленный в качестве примера набор программных компонентов, выполняемый на защищенном ЗУ 24 (т.е. на процессоре 116) в соответствии с некоторыми вариантами осуществления настоящего изобретения. Показанное ПО включает в себя агента 50 системы обеспечения безопасности ЗУ и криптографическое устройство 52. Агент 50 может быть выполнен с возможностью поддерживать семантическое отображение файловой системы первичного ЗУ 30 и применять набор эвристических правил (например, правила принятия решения), чтобы определить, содержит ли запрос от ПО, выполняемого на хост-процессоре 16, на доступ к первичному ЗУ 30 признаки угрозы для компьютерной безопасности. Устройство 52 может быть выполнено так, что оно будет выполнять операции шифрования и/или дешифрования для пакетов данных, поступающих в первичное ЗУ 30 и/или выходящих из этого ЗУ. Работа компонентов 50 и 52 будет подробно описана далее.

[0036] В некоторых вариантах осуществления модуль МКБ 44 взаимодействует с ПО, выполняемым в защищенном ЗУ 24, (например, с агентом 50 системы обеспечения безопасности ЗУ), например, посредством обмена уведомлениями/сигналами через канал связи, управляемый промежуточным блоком 42 ЗУ. Представленные в качестве примера уведомления, поступающие от процессора 16 на защищенное ЗУ 24 (в настоящем документе называемые нисходящими уведомлениями), включают в себя индикатор операции, которую должен выполнить процессор 116, и другие данные, такие как имя файла, ключ шифрования, набор флагов и т.д. Модуль МКБ 44 может также отправлять нисходящие уведомления в ответ на определенные, относящиеся к безопасности события, как подробно описано ниже.

[0037] Представленный в качестве примера канал связи может использовать обычные средства транспортировки данных между процессором и ЗУ большой емкости. Например, модуль МКБ 44 и агент 50 системы обеспечения безопасности ЗУ могут обмениваться сообщениями в соответствии с протоколом передачи ЗУ, таким как протокол последовательного интерфейса AT (Serial ATA) или протокол интерфейса малых компьютерных систем (SCSI). Протокол передачи ЗУ задает формат связи через интерфейс ЗУ, размер и содержимое кадра/пакета, количество, размер, порядок и/или формат заголовков и/или полезной нагрузки, значение различных управляющих битов и полей данных, кодирование команд и т.д.

[0038] В некоторых вариантах осуществления нисходящее уведомление замаскировано под имитационный запрос доступа. Имитационные запросы доступа в данном документе

относятся к таким запросам доступа к ЗУ, которые должным образом отформатированы в соответствии с протоколом связи (например, правильно сформированные команды SATA), но которые следует не выполнять как таковые, а интерпретировать как уведомления. Термин "имитационный" здесь используют для того, чтобы отличать такие запросы от "исходных" или "подлинных" запросов доступа к ЗУ, представляющих собой операции доступа к ЗУ и предназначенных для выполнения как таковые.

[0039] В некоторых вариантах осуществления имитационные запросы являются недопустимыми запросами доступа к ЗУ в том смысле, что выполнение такого имитационного запроса вызывает исключение/ошибку. В таких случаях обработчик ошибок может перехватить соответствующую ошибку и определить запрос, вызывающий соответствующую ошибку, как нисходящее уведомление. В других вариантах осуществления имитационные запросы не являются ошибочными сами по себе, но могут содержать специфические, необычные или противоречивые комбинации значений параметров (флагов). В других вариантах осуществления имитационные запросы могут содержать действительные регулярные запросы; такие имитационные запросы могут быть идентифицированы как таковые, например, в соответствии с содержимым полезной нагрузки или других полей данных, определенным протоколом связи ЗУ.

[0040] Пример имитационного запроса доступа содержит запрос на доступ к адресу вне допустимого диапазона, т.е. к адресу за пределами обычного адресуемого диапазона первичного ЗУ 30. Примеры включают в себя "чтение блока B" или "запись полезной нагрузки P по адресу A", в котором адрес блока B и адрес A находятся за пределами нормального адресуемого диапазона первичного ЗУ 30. Каждый такой адрес (специфическое значение A и/или B) может соответствовать специфической команде или специфическому событию, передаваемому программному обеспечению, выполняемому внутри защищенного ЗУ 24. В другом примере нисходящие уведомления могут быть замаскированы под запросы доступа к ЗУ по адресу, который, хотя и находится в пределах нормального адресуемого диапазона, но обычно недоступен во время нормальной работы. Например, запрос на запись в ячейку ЗУ, в которой в данный момент хранится главная загрузочная запись (MBR) или критический компонент ОС (например, NTOSKRNL.EXE в Windows®), может быть перехвачен защищенным ЗУ 24 и интерпретирован как нисходящее уведомление от модуля МКБ 44.

[0041] В еще одном примере имитационные запросы доступа могут быть идентифицированы в соответствии с конкретным содержимым полезной нагрузки (например, конкретным битовым шаблоном или сигнатурой). В таких вариантах осуществления программный компонент, выполняемый на процессоре 116 системы обеспечения безопасности, может анализировать содержимое полезной нагрузки для обнаружения имитационных запросов доступа. Различные полезные нагрузки P могут соответствовать различным уведомлениям/командам для защищенного ЗУ 24.

[0042] В еще одном примере имитационные запросы доступа могут быть идентифицированы как таковые в соответствии с содержимым другого поля данных, заданного протоколом связи ЗУ, например, в соответствии с содержимым полей "Команда" и/или "Вспомогательные", указанных в протоколе последовательного интерфейса SATA (Serial ATA). В одном таком примере команда для обновления сигнатур вредоносных программ может быть закодирована в поле "Команда" имитационного запроса доступа, выданного модулем МКБ 44. Текущие сигнатуры могут быть переданы как полезная нагрузка другого имитационного запроса доступа.

[0043] Некоторые другие примеры нисходящих уведомлений включают в себя команду на сканирование данных, хранящихся в определенной ячейке ЗУ: соответствующее

местоположение может быть введено в полезную нагрузку имитационного запроса доступа. В еще одном примере имитационные запросы на запись, содержащие разные адреса A, могут указывать на разные способы или параметры защиты от вредоносного ПО. Каждый адрес или диапазон адресов A может показывать отдельный набор эвристических правил или отдельную группу сигнатур, указывающих на вредоносное ПО.

[0044] В свою очередь, промежуточный блок 42 ЗУ может получать уведомления/сигналы от устройства 24 (в данном документе называемые восходящими уведомлениями) через подтверждение и/или другую форму ответа на нисходящее уведомление, например, через аппаратные запросы прерывания (IRQ) или асинхронные уведомления SATA, отправляемые устройством 24 и обрабатываемые промежуточным блоком 42 ЗУ и/или другим обработчиком событий ОС 40. Представленные в качестве примера восходящие уведомления включают в себя, среди прочего, предупреждение об угрозе безопасности, например, индикатор вероятной атаки вымогателей и индикатор попытки обновления встроенного ПО ЗУ 24.

[0045] Еще одно представленное в качестве примера восходящее уведомление содержит запрос на криптографический ключ. В одном таком примере агент 50 системы обеспечения безопасности ЗУ может обнаружить попытку записи зашифрованного пакета данных. Такая попытка может указывать на то, что автоматическое шифрование данных на диске активировано (например, технология Bitlocker® от Microsoft®). В ответ некоторые варианты осуществления могут запросить криптографический ключ от модуля МКБ 44. Более подробная информация об этом способе приведена ниже.

[0046] Некоторые варианты осуществления реализуют версию итеративного обнаружения вредоносных программ с использованием последовательности нисходящих-восходящих уведомлений. В одном таком примере нисходящее уведомление дает команду агенту 50 системы обеспечения безопасности ЗУ отсканировать конкретный файл на предмет вредоносных программ. Агент 50 может затем передать результат сканирования в модуль МКБ 44, который может выбрать другой файл или папку в соответствии с результатом первого сканирования и передать новую цель сканирования агенту 50 системы обеспечения безопасности через новое нисходящее уведомление и т.д. Такие итеративные схемы могут позволить довольно сложным процедурам обнаружения вредоносного ПО устанавливать сложное ПО на защищенном ЗУ 24.

[0047] На Фиг.5 показана представленная в качестве примера последовательность этапов, выполняемых модулем МКБ 44 в соответствии с некоторыми вариантами осуществления настоящего изобретения. Модуль МКБ 44 может объединять относящиеся к безопасности данные из нескольких источников в хост-системе 10 и может получать уведомления о возникновении определенных событий при выполнении ПО. Одним приведенным в качестве примера источником является функция системного вызова ОС 40, которая модифицируется путем подключения к сигналу на модуль МКБ 44 каждый раз, когда происходит соответствующий системный вызов. Другие представленные в качестве примера источники уведомлений и информации о безопасности содержат минифильтры ОС. В некоторых вариантах осуществления модуль МКБ 44 может дополнительно принимать восходящие уведомления от защищенного ЗУ 24. В ответ на обнаружение наступления некоторого события модуль МКБ 44 на этапе 206 может применить набор эвристических правил, чтобы определить, подвергается ли хост-система 10 атаке в настоящее время. Если анализ показывает возможность наличия вредоносных программ, то модуль МКБ 44 может принять

соответствующие меры борьбы (этапы 212-214), например, блокирование или иным образом предотвращение выполнения вредоносного процесса и оповещение пользователя или администратора хост-системы 10. Некоторые обнаруженные события требуют выдачи нисходящего уведомления на защищенное ЗУ 24. В одном таком примере модуль МКБ 44 может дать указание агенту 50 системы обеспечения безопасности провести расследование попытки доступа к ЗУ. Нисходящие уведомления проиллюстрированы как этапы 208-210 на фиг.5.

[0048] В некоторых вариантах осуществления агент 50 системы обеспечения безопасности ЗУ воссоздает семантику файловой системы, используемой ОС 40, из метаданных, хранящихся в ЗУ 30, способом, который не зависит от файловой системы ОС 40. Иными словами, агент 50 поддерживает базу семантических знаний о файловой системе, равную альтернативе или тени файловой системы ОС 40. В обычных компьютерных системах на аппаратном уровне данные хранятся в виде блоков, в которых отсутствует семантическая информация. Например, неясно, какому файлу / какой папке принадлежит тот или иной фрагмент данных. Еще одним осложнением является фрагментация, при которой данные одного файла не собраны вместе, а рассредоточены в различных местах по всему носителю данных. Задачу осуществления служебных действий по сопоставлению связанных объектов с местами хранения и по преобразованию информации аппаратного уровня в значимые данные обычно выполняет ОС.ОС управляет такими задачами, поддерживая специализированную структуру данных, известную как файловая система, закодированную как метаданные и хранящуюся в определенном разделе носителя. Представленные в качестве примера файловые системы включают в себя, помимо прочего, FAT (таблица размещения данных), FAT32 и NTFS (файловая система новой технологии).

[0049] В некоторых вариантах осуществления семантическое отображение файловой системы содержит кодирование отображения между разделом первичного ЗУ 30 и элементом файловой системы ОС 40. Представленные в качестве примера элементы файловой системы включают в себя каталог и файл. Представленный в качестве примера элемент семантического отображения связывает диапазон адресов [A1 A2] с файлом F (где F может быть представлен в виде пути, например, C:\user\docs\Letter.txt или /home/user/docs/Letter.txt). Такое сопоставление фактически указывает на то, что данные, хранящиеся в соответствующем диапазоне адресов, являются частью файла F. Другой представленный в качестве примера элемент семантического отображения файловой системы показывает, что данный диапазон адресов [A3 A4] хранит метаданные файловой системы. Еще один представленный в качестве примера элемент семантического отображения файловой системы связывает отдельный адресуемый блок (например, блок или сектор ЗУ в отличие от диапазона адресов) с элементом файловой системы. Данные семантического отображения могут быть закодированы с использованием любого способа, известного в данной области техники, например, в виде растрового изображения, связанного списка и т.д.

[0050] На Фиг.6 показана представленная в качестве примера последовательность этапов, выполняемых агентом 50 системы обеспечения безопасности ЗУ в соответствии с некоторыми вариантами осуществления настоящего изобретения. Агент 50 принимает запросы доступа к ЗУ от ОС 40 через интерфейс 36 ЗУ. Типичный запрос содержит индикатор операции (чтение, запись), индикатор адреса и полезную нагрузку. Представленные в качестве примера запросы доступа к ЗУ имеют семантику "чтение блока из N байтов по адресу A" и "запись полезной нагрузки P по адресу A". Запросы доступа могут дополнительно содержать набор значений параметров (например, флаги,

атрибуты) соответствующего запроса. Фактический формат и кодировка запросов доступа к ЗУ могут различаться в зависимости от реализации аппаратного и программного обеспечения.

[0051] Когда запрос доступа поступает на устройство 24, агент 50 может определить в соответствии с параметрами запроса, указывает соответствующий запрос на действительный или на имитационный доступ к ЗУ (т.е. уведомление от модуля МКБ 44). В одном приведенном в качестве примера варианте осуществления запрос на доступ к адресу вне допустимого диапазона может быть индикатором такого уведомления. Если соответствующий запрос доступа содержит нисходящее уведомление, то этап 236 может выбрать и выполнить конкретное действие в соответствии с параметрами соответствующего запроса (см. некоторые примеры ниже).

[0052] В последовательности этапов 228-230 агент 50 системы обеспечения безопасности ЗУ может дополнительно декодировать семантику соответствующего запроса доступа к ЗУ в соответствии с семантическим отображением. Например, агент 50 может определить, является ли то, что подлежит записать, метаданными или фактическим файлом, создается ли новый файл, какой конкретный файл в настоящее время записывается или читается и т.д. Следующий этап 232 может применить набор эвристических правил доступа, чтобы определить, имеет ли запрошенный доступ признаки угрозы для компьютерной безопасности. Если "нет", то агент 50 может разрешить соответствующий доступ для продолжения. Если применение эвристических правил показывает, что запрос доступа требует уведомления модуля 44 компьютерной безопасности, то агент 50 может выполнить восходящее уведомление, что равносильно предупреждению от системы обеспечения безопасности.

[0053] На Фиг.7 показана представленная в качестве примера последовательность этапов, выполняемых агентом 50 системы обеспечения безопасности ЗУ для поддержания семантического отображения файловой системы. Создание отображения может быть начато, например, при загрузке. В некоторых вариантах осуществления агент 50 может определять местоположение в первичном ЗУ 30, где хранятся метаданные файловой системы, используемые ОС 40. Несколько способов для достижения этого известны в области компьютерной экспертизы. В некоторых вариантах осуществления используют программный компонент, выполняемый на хосте (например, модуль МКБ 44), для определения местоположения данных файловой системы. Модуль МКБ 44 может затем передать соответствующее местоположение агенту 50 системы обеспечения безопасности, используя нисходящие уведомления.

[0054] В последовательности этапов 254-256 агент 50 анализирует метаданные файловой системы, хранящиеся в первичном ЗУ 30, и собирает семантическое отображение в соответствии с метаданными. Некоторые варианты осуществления сохраняют полученные данные семантического отображения (т.е. теневую файловую систему) в памяти 118 системы обеспечения безопасности и/или вторичном ЗУ 32. Затем во время выполнения ПО на процессоре 16 агент 50 может контролировать запросы доступа к ЗУ и определять, имеют ли такие запросы признаки изменения метаданных (например, создание или удаление файла/каталога). Если "да", то агент 50 может обновлять семантическое отображение соответствующим образом (этапы 260-262-264 на фиг.7).

[0055] Проиллюстрированная система уведомлений и эвристических правил может быть использована для различных приложений, некоторые примеры которых приведены ниже.

Фильтрация команд для защиты аппаратного обеспечения

[0056] Тщательно проработанное вредоносное ПО может использовать определенные функции набора команд ATA (например, команду DOWNLOAD MICROCODE (ЗАГРУЗИТЬ МИКРОКОД)) для скрытного обновления микропрограмм ЗУ, чтобы ввести вредоносное ПО на уровне аппаратного обеспечения в само устройство. Одной из таких приведенных в качестве примера вредоносных программ является программа "черный ход", которую ПО, выполняемое на хосте, может использовать для изменения поведения соответствующего ЗУ и/или управления им.

[0057] Чтобы предотвратить такие сложные атаки, в некоторых вариантах осуществления агент 50 системы обеспечения безопасности ЗУ выполнен так, что он фильтрует запросы доступа к ЗУ, поступающие через интерфейс 36 ЗУ. Правила фильтрации могут быть базовыми, например, разрешать выполнение только наиболее распространенных запросов доступа (например, команды для чтения, записи, идентификации устройства и т.д.) и блокировать все другие команды/запросы. Другие варианты осуществления могут реализовывать более сложные эвристические правила фильтрации, например, правила фильтрации, адаптированные к текущему контексту. В другом примере агент 50 системы обеспечения безопасности ЗУ может обусловить выполнение некоторых команд ATA/ запросов доступа после явного подтверждения от администратора хост-системы 10.

[0058] В некоторых вариантах осуществления определенные команды/запросы не блокируются, а вместо этого ПО, выполняемое на процессоре 116 системы обеспечения безопасности (например, агент 50 системы обеспечения безопасности), эмулирует и дополнительно анализирует их. В ответ на прием такой команды ПО системы обеспечения безопасности может сгенерировать такой ответ, чтобы ПО, выполняемое на хост-процессоре 16, решило, что соответствующая команда / соответствующий запрос доступа была/был успешно выполнен. Агент 50 системы обеспечения безопасности может дополнительно регистрировать такие команды, чтобы помочь в исследовании вредоносных программ.

Перехват и интерпретация событий на аппаратном уровне

[0059] Поддержание семантического отображения (теневая файловая система) позволяет агенту 50 системы обеспечения безопасности ЗУ в ответ на прием запроса доступа обнаруживать относящиеся к безопасности события, происходящие во время выполнения ПО на процессоре 16. Например, в ответ на запрос записи агент 50 может определить, являются записываемое метаданными или фактическим содержимым файла, направлена соответствующая запись в пустой раздел хранения или перезаписывает существующую информацию, является соответствующая запись подлинной записью или нисходящим уведомлением от процессора 16 и т.д.

[0060] События файловой системы, такие как создание файла, удаление файла и перезапись файла, выполняются в соответствии со специфической для события последовательностью операций. Представленный в качестве примера шаблон может включать в себя чтение метаданных с последующей записью метаданных и затем записью полезной нагрузки. Агент 50 может использовать некоторый набор эвристических правил, позволяющий кодировать такие шаблоны, чтобы идентифицировать тип каждого события файловой системы. Кроме того, агент 50 может идентифицировать цель каждой операции чтения/записи, например, в какой файл идет запись, чтение какого файла происходит.

[0061] В отличие от традиционных систем/способов обеспечения компьютерной безопасности такой перехват событий осуществляют практически без знания ПО, выполняемого на процессоре 16. Таким образом, вредоносное ПО не может

воспрепятствовать или иным образом помешать обнаружению событий. ПО системы обеспечения безопасности, такое как агент 50, может также уведомлять модуль МКБ 44 о появлении определенных событий, которые можно считать относящимися к безопасности, используя восходящие уведомления.

Проверка на вредоносное ПО при доступе

[0062] Агент 50 системы обеспечения безопасности может обнаружить попытку открыть какой-то файл и/или попытку выполнить некоторый исполняемый файл. Другие операции, которые могут быть обнаружены таким образом, включают в себя добавление файла и назначение ЗУ для последующих записей. Каждое такое событие может быть использовано как пусковой сигнал (триггер) для запуска сканирования соответствующего файла или сканирования ресурса (основной исполняемый файл, библиотека и т.д.), принадлежащего процессу, заказавшего соответствующий доступ к ЗУ. Сканирование может происходить в то время, пока процесс, выдавший запрос доступа к ЗУ, приостановлен, или может происходить автономно, пока соответствующий процесс позволяет продолжать выполнение.

[0063] Сканирование может быть выполнено в соответствии с любым способом, известным в области компьютерной безопасности, например, путем сопоставления содержимого соответствующего файла с библиотекой сигнатур или кодов, указывающих на вредоносное ПО. Для этой цели библиотека сигнатур, указывающих на вредоносное ПО, может храниться во вторичном ЗУ 32. Периодически или по запросу может происходить обновление этой библиотеки. Некоторые варианты осуществления обновляют библиотеку сигнатур и/или другое ПО, выполняемое на защищенном ЗУ 24, посредством набора нисходящих уведомлений (например, имитационных запросов доступа к ЗУ).

[0064] В варианте сканирования при доступе агент 50 может использовать набор эвристических правил для выявления последовательности загрузки хост-системы 10 и/или ОС 40. Иными словами, анализируя последовательность запросов доступа к ЗУ, агент 50 может определить, что хост-система 10 в данный момент находится в процессе загрузки (т.е. идет инициализация аппаратного и/или программного обеспечения). Приведенная в качестве примера последовательность загрузки обычно начинается с последовательности запросов на чтение участка, хранящего структуру данных, известную как главная загрузочная запись (MBR) или таблица разделов GUID (GPT) (таблица разделов глобального уникального идентификатора).

Идентификация устройства

Read DMA Ext	A:	00000000	C:	0001
Read DMA Ext	A:	00000000	C:	0001
Read DMA Ext	A:	00000001	C:	0001
Read DMA Ext	A:	E8E088AF	C:	0001
Read DMA Ext	A:	00000002	C:	0020
Read DMA Ext	A:	E8E0888F	C:	0020
Read DMA Ext	A:	00000002	C:	0020

…

Приведенная в качестве примера последовательность запросов доступа к ЗУ, показывающая, что ОС 40 начала загрузку, проиллюстрирована ниже:

Идентификация устройства

Read FPDMA Queued	A:	00000000	C:	0001	T:	04
Read FPDMA Queued	A:	00000001	C:	0001	T:	05
Read FPDMA Queued	A:	00000002	C:	0020	T:	06

Read FPDMA Queued	A:	00000000	C:	0001	T:	07

…

[0065] Другой типичной особенностью последовательности загрузки запросов на доступ являются очень длинные последовательности запросов на чтение (например, 2-3000 последовательных запросов на чтение), прерываемые короткими последовательностями из 2-5 запросов на запись. Такие наборы могут зависеть от ОС. Характеристический анализ может объединять некоторое количество последовательных запросов на чтение/запись с анализом информации об адресе и/или анализом других параметров для выявления значения различных этапов процесса загрузки.

[0066] Некоторые варианты осуществления могут комбинировать сопоставление состава запроса доступа с информацией семантического отображения, чтобы обнаружить инициализацию ОС 40. В одном таком примере агент 50 системы обеспечения безопасности может вести итеративную процедуру для сбора информации о типе и/или расположении ПО, выполняемого на хост-процессоре 16, непосредственно из событий доступа к ЗУ. При обнаружении доступа к главной загрузочной записи агент 50 может определить информацию о разделе. Затем серия запросов на чтение поступает на заголовки томов. На основании таких запросов агент 50 может определять и проверять информацию о соответствующих томах. Затем агент 50 может автоматически идентифицировать место хранения набора важных файлов ОС (например, ресурсов, которые ОС 40 загружает при инициализации, таких как NTOSKRNL.EXE в Windows®), отслеживая последовательность операций чтения и/или записи, сопровождающую загрузку. Соответствующая последовательность загрузки может также указывать тип операционной системы (например, сборка, версия и т.д. ОС 40).

[0067] Некоторые варианты осуществления могут затем сканировать каждый файл, открываемый в течение некоторого интервала времени (например, несколько секунд) за обнаруженной фазой загрузки/инициализации. Сканирование может включать в себя проверку целостности, т.е. выявление того, было ли содержимое файла повреждено, например, вредоносным ПО. Проверка целостности может содержать сравнение хэша текущего содержимого соответствующего файла с эталонным хэшем, хранящимся во вторичном ЗУ 32.

[0068] В еще одном приведенном в качестве примера приложении, относящемся к последовательностям загрузки, некоторые варианты осуществления могут использовать защищенное ЗУ 24 в качестве агента, выполняемого вне ОС 40 и сконфигурированного для проверки целостности и/или достоверности ОС 40. Например, агент 50 может автоматически обнаруживать запрос на перезагрузку хост-системы 10. Затем, при обнаружении того, что перезагрузка действительно выполняется (например, из операций обнаружения и инициализации устройства или в ответ на нисходящее уведомление модулем МКБ 44), агент 50 может приостановить нормальную последовательность загрузки и предоставить альтернативную ОС, или агент безопасности может быть сконфигурирован для сканирования структур данных ОС 40 и/или загрузочной области первичного ЗУ 30. После завершения сканирования и принятия решения считать систему безопасной агент 50 может возобновить загрузку ОС 40.

Защита хранимых объектов

[0069] Загрузочная область первичного ЗУ 30 обычно хранит ресурсы, считываемые до загрузки ОС. Поддерживая семантическое отображение, агент 50 может определить, нацелен ли запрос на запись на соответствующую область загрузки, и в ответ может заблокировать соответствующую запись и/или уведомить модуль МКБ 44. Аналогичная стратегия может быть использована для защиты ценных объектов ОС 40 или других

приложений, таких как определенные файлы, библиотеки и т.д.

Работа с зашифрованными данными

[0070] В некоторых версиях ОС 40 есть возможность хранить данные в зашифрованном виде в первичном ЗУ 30. Одним из таких примеров является функция Bitlocker® в Microsoft® Windows®. Если хранимые данные зашифрованы, то модуль, выполняемый вне ОС 40, (в т.ч. агент 50 системы обеспечения безопасности ЗУ) не может получить доступ к соответствующим данным или к системным метаданным, которые позволяют создавать семантическое отображение файловой системы.

[0071] Однако агент 50 может сотрудничать с модулем МКБ 44 для получения ключа шифрования или информации, способствующей получению соответствующего ключа. Такая информация может содержать, например, пароль, секретный код, одноразовый номер и т.д., и поэтому ее называют материалом ключа шифрования. Модуль МКБ 44 может предоставлять пользовательский интерфейс для запроса пароля пользователя или иного секретного кода, используемого в отношении соответствующего ключа, и сообщать пароль / секретный код агенту 50. В другом варианте осуществления модуль МКБ 44 может взаимодействовать напрямую с агентом шифрования ОС (например, с модулями Bitlocker®) для получения материала ключа. В ответ на получение материала ключа модуль МКБ 44 может сообщить сам материал ключа или место хранения упомянутого материала ключа агенту 50 через нисходящее уведомление (например, через имитационный запрос доступа к ЗУ).

[0072] Получив ключ шифрования, агент 50 может использовать криптографическое устройство 52 для дешифрования хранимых данных, чтобы создать и/или поддержать семантическое отображение файловой системы. В некоторых вариантах осуществления агент 50 может дополнительно использовать ключ шифрования для выполнения онлайн-сканирования/анализа трафика данных, поступающего в первичное ЗУ 30 и/или выходящего из него, практически без знания ОС 40 или другого ПО, выполняемого на хост-процессоре 16. В одном примере в ответ на перехват запроса на запись агент 50 системы обеспечения безопасности может дешифровать и проанализировать соответствующую полезную нагрузку перед записью исходной (зашифрованной) полезной нагрузки по намеченному адресу. В некоторых вариантах осуществления в ответ на дешифрование соответствующей полезной нагрузки агент 50 может сохранить незашифрованную версию полезной нагрузки во вторичном ЗУ 32.

Общее обнаружение шифрования

[0073] Некоторые варианты осуществления агента 50 системы обеспечения безопасности ЗУ могут автоматически определять, зашифрованы ли данные, хранящиеся в разделе (например, в блоке, секторе и т.д.) ЗУ 30. Для этого можно использовать критерии сложности информации, как, например, энтропия, или иные способы, известные в данной области техники. Чтобы не допустить принятия неправильного решения, некоторые варианты осуществления могут использовать метаданные, доступные для соответствующего файла, чтобы определить, может ли соответствующий файл иметь высокую энтропию без обязательного шифрования. Примерами являются данные, сжатые в соответствии с такими форматами, как MP3, JPG, MPG, ZIP и т.д. Чтобы определить, попадает ли соответствующий файл в одну из этих категорий, некоторые варианты осуществления могут найти раздел заголовка соответствующего файла в соответствии с метаданными и выполнить поиск информации о типе файла в соответствующем заголовке.

[0074] В некоторых вариантах осуществления каждая запись семантического отображения файловой системы может быть дополнена набором флагов, указывающих,

например, зашифрован ли соответствующий элемент файловой системы (файл, папка и т.д.), является ли соответствующий элемент файловой системы сжатым или нет и т.д. Агент 50 может поддерживать эти флаги вместе с остальными данными семантического отображения.

Обнаружение усовершенствованных программ-вымогателей

[0075] Описанные здесь системы и способы позволяют автоматически обнаруживать попытку шифрования хранимых данных. Обнаружение происходит независимо от ПО, выполняемого на процессоре 16, и виртуально незаметно для этого ПО. Такое автоматическое обнаружение шифрования предназначено для выявления программ-вымогателей и иного типа вредоносного ПО, действия которых приводят к несанкционированному или непреднамеренному шифрованию пользовательских данных.

[0076] Одно приведенное в качестве примера эвристическое правило обнаружения содержит обнаружение попытки перезаписать незашифрованный контент зашифрованным контентом. Другой набор эвристических правил обнаружения использует статистику для сравнения текущего потока запросов доступа к ЗУ с "нормальным" шаблоном, соответствующим конкретному пользователю и/или запущенному приложению. Для обнаружения некоторые варианты осуществления задают набор пользовательских профилей и/или профилей приложений, указывающих, например, какие приложения/процессы обычно запускает конкретный пользователь, какие места хранения обычно доступны соответствующему пользователю, что такое типичная схема запросов к ЗУ, связанных с каждым процессом/приложением и т.д. Если текущая последовательность запросов доступа к ЗУ выходит за границы шаблона "нормальности", например, если агент 50 обнаруживает необычный всплеск активности по созданию файла, причем содержимое соответствующих файлов зашифровано, то агент 50 может решить, что идет атака программы-вымогателя.

[0077] Если агент 50 обнаруживает подозрительную активность шифрования, то агент 50 может приостановить соответствующую активность (например, заблокировать набор подозрительных записей) и/или передать сигнал на модуль МКБ 44, используя механизм восходящего уведомления. В свою очередь, модуль МКБ 44 может использовать это уведомление как предупреждение о возможной атаке или может применить дополнительную эвристику, например, для корреляции событий, информация о которых поступает от агента 50, с другими указывающими на вредоносное ПО событиями, происходящими в хост-системе 10 или на других компьютерах, подключенных к хост-системе 10 по сети связи.

Теневое копирование объектов для приложений, таких как управление версиями ПО и резервное копирование

[0078] В некоторых вариантах осуществления агент 50 системы обеспечения безопасности ЗУ автоматически обнаруживает попытку удаления или перезаписи файла и в ответ сохраняет копию удаленных/перезаписанных данных в отдельном месте - либо в первичном ЗУ 30, либо во вторичном ЗУ 32. Таким образом, теневая копия удаленного или перезаписанного файла сохраняется вместе со вновь записанными данными. Некоторые варианты осуществления сохраняют более двух последовательных версий одного и того же файла, причем по меньшей мере одна из соответствующих версий является нешифрованной. Этот механизм может позволить безопасное восстановление данных, потенциально восстанавливая соответствующий файл в любой из сохраненных версий.

[0079] Возможные применения такого теневого копирования активов включают в себя, среди прочего, обеспечение безопасности, резервное копирование и управление

версиями ПО. В варианте осуществления компьютерной безопасности агент 50 может обнаружить, что например, имела место многократная перезапись какого-то конкретного файла в течение относительно небольшого интервала времени. Такая файловая активность может указывать на вредоносное ПО. Некоторые варианты осуществления могут сравнивать последовательные версии одного и того же файла, чтобы определить, например, зашифрована ли более новая версия, по сравнению со старой нешифрованной версией. Как упомянуто ранее, такие изменения в шифровании могут указывать на атаку программы-вымогателя. В более общем смысле, сохранение резервной копии файла может помешать вредоносному ПО изменить важные ресурсы (объекты) ОС, предотвращая таким образом, например, злонамеренное перехватывание функций ОС. Если такая модификация обнаружена, то агент 50 может уведомить модуль МКБ 44. В свою очередь, модуль МКБ 44 может дать команду агенту 50 откатить изменения в конкретном файле.

Оптимизация

[008] Некоторые варианты осуществления могут быть дополнительно оптимизированы, чтобы снизить вычислительные затраты на расшифровку семантики файловой системы с уровня защищенного ЗУ 24. В гибридном варианте осуществления модуль МКБ 44 доставляет индикаторы семантики агенту 50 системы обеспечения безопасности, в то время как агент 50 может перехватывать и противодействовать злонамеренным событиям, осуществляемым на ЗУ, таким как удаление файла, перезапись файла, несанкционированное шифрование данных и т.д. В таких вариантах осуществления модуль МКБ 44 может содержать облегченный фильтр доступа к ЗУ, имеющий функциональность, аналогичную функциональности минифильтра файловой системы в Windows®. Фильтр доступа к ЗУ может определять, среди прочего, например, попытку записи в файл, имя и/или местоположение на диске соответствующих данных / соответствующего файла и идентичность процесса, выполняющего операцию записи. Затем фильтр доступа к ЗУ может передавать такую семантическую информацию агенту 50 системы обеспечения безопасности ЗУ через нисходящее уведомление. Агент 50 может подтвердить получение, используя пакет подтверждения или восходящее уведомление. В некоторых вариантах осуществления нисходящие уведомления поступают в очередь уведомлений.

[0081] На уровне защищенного ЗУ 24 агент 50 может идентифицировать попытку записи и попытаться сопоставить соответствующую попытку с данными из очереди нисходящих уведомлений. Успешное сопоставление позволяет агенту 50 выявить семантику соответствующей попытки записи, благодаря чему агент 50 может применить более изощренную эвристику для определения, является ли соответствующая попытка записи подозрительной, должна ли она быть предотвращена и т.д. Неудача сопоставления попытки записи, обнаруженной на аппаратном уровне (т.е. из конкретных изменений метаданных), с попыткой записи, переданной через нисходящее уведомление, может указывать на то, что вредоносное ПО способно избежать обнаружения минифильтром доступа к ЗУ ОС 40. В таких случаях агент 50 системы обеспечения безопасности может заблокировать соответствующую попытку записи и/или уведомить модуль МКБ 44 через восходящее уведомление.

[0082] Настоящее изобретение относится к системам и способам защиты хост-системы от угроз для компьютерной безопасности, среди прочего, таких как вредоносное ПО. Описанные системы и способы особенно подходят для защиты хост-систем (например, компьютеров, устройств мобильной связи и т.д.) от сложных вредоносных программ, способных разрушить традиционные средства защиты. Представленные в качестве

примера приложения включают в себя защиту от программ-вымогателей и от хищения фирменных, личных и/или конфиденциальных данных.

[0083] Некоторые варианты осуществления настоящего изобретения основаны на наблюдении, что вредоносное ПО может успешно проникать в трафик данных между процессором хост-системы и энергонезависимой памятью (например, магнитный, оптический или твердотельный накопитель). ПО системы обеспечения безопасности, действующее на соответствующем процессоре, может быть не в состоянии заблокировать или предотвратить все такие вмешательства, что ведет к существенной опасности хищения или потери данных. Чтобы решить эту проблему, представленные в качестве примера варианты осуществления настоящего изобретения помещают части защитного ПО на отдельный процессор, сконфигурированный для перехвата, анализа и/или выборочного блокирования трафика данных между главным процессором хост-системы и ЗУ. Вспомогательный процессор системы обеспечения безопасности может быть интегрирован с ЗУ и/или с дополнительным аппаратным обеспечением, например, на общей печатной плате, чтобы сформировать улучшенное защищенное ЗУ. Указанное защищенное ЗУ может иметь обычный конструктивный размер жесткого диска или другого энергонезависимого ЗУ и может быть подключено к остальной части аппаратного обеспечения хост-системы через обычный интерфейс/разъем ЗУ, такой как последовательный интерфейс SATA или интерфейс/разъем межсоединения периферийных компонентов (PCI) Express. В альтернативном варианте осуществления защищенное ЗУ (т.е. ЗУ + вспомогательный процессор) может быть выполнено как внешний накопитель, подключенный к хост-системе, например, посредством универсальной последовательной шины (USB) или другого обычного интерфейса/разъема.

[0084] В обычных системах защиты от вредоносных программ предотвращение, обнаружение и меры противодействия реализуются в программном обеспечении, работающем на том же физическом процессоре, на котором действует также вредоносный код. Дополнительно в обычных системах как вредоносные, так и разрешенные программы могут обращаться к одному и тому же физическому устройству хранения (например, жесткий диск). Такое построение потенциально позволяет тщательно проработанному вредоносному коду разрушить ПО системы обеспечения безопасности. В противоположность этому в некоторых вариантах осуществления настоящего изобретения доступом к физической памяти управляет вспомогательный процессор, отличный от основного процессора, на котором работают пользовательские приложения (и, возможно, вредоносный код). Поэтому ПО системы обеспечения безопасности, выполняемое на вспомогательном процессоре, недоступно для вредоносных программ.

[0085] Специалисту в данной области техники будет понятно, что вышеуказанные варианты осуществления могут быть изменены многими способами, не выходя за пределы объема изобретения. Соответственно, объем изобретения определяет приведенная далее формула изобретения и ее юридические эквиваленты.

(57) Формула изобретения

1. Компьютерная система, содержащая первый аппаратный процессор и защищенное запоминающее устройство (ЗУ), в которой защищенное ЗУ подключено к первому аппаратному процессору через интерфейс ЗУ, сконфигурированный для приема запросов доступа к ЗУ, отформатированных в соответствии с протоколом передачи ЗУ, причем защищенное ЗУ содержит второй аппаратный процессор и энергонезависимый элемент

ЗУ, при этом:

первый аппаратный процессор сконфигурирован так, что:

при обнаружении программным обеспечением, выполняемым на первом аппаратном процессоре, запроса на сохранение в элементе ЗУ пакета данных этот процессор шифрует указанный пакет данных,

после шифрования пакета данных процессор передает исходный запрос доступа к ЗУ в интерфейс ЗУ, причем данный исходный запрос доступа к ЗУ содержит зашифрованный пакет данных, генерирует имитационный запрос доступа к ЗУ в соответствии с протоколом передачи ЗУ, причем этот имитационный запрос доступа к ЗУ содержит по меньшей мере часть криптографического ключа, и передает этот имитационный запрос доступа к ЗУ в интерфейс ЗУ;

а второй аппаратный процессор сконфигурирован так, что:

после приема сообщения через интерфейс ЗУ процессор определяет, содержит ли сообщение имитационный запрос доступа к ЗУ, затем, если сообщение содержит такой имитационный запрос доступа к ЗУ, то процессор определяет криптографический ключ в соответствии с имитационным запросом доступа к ЗУ,

после приема исходного запроса доступа к ЗУ процессор использует криптографический ключ и дешифрует пакет данных, после чего определяет, содержит ли дешифрованный пакет данных вредоносное программное обеспечение (ПО).

2. Компьютерная система по п.1, в которой второй аппаратный процессор дополнительно сконфигурирован так, что после определения того, содержит ли дешифрованный пакет данных вредоносное ПО, если дешифрованный пакет данных содержит вредоносное ПО, этот процессор передает уведомление, отформатированное в соответствии с протоколом передачи ЗУ, в интерфейс ЗУ, причем указанное уведомление сконфигурировано так, чтобы первый аппаратный процессор интерпретировал сообщение уведомления как предупреждение об угрозе безопасности.

3. Компьютерная система по п.2, в которой уведомление сконфигурировано так, чтобы вызвать аппаратное прерывание в первом аппаратном процессоре.

4. Компьютерная система по п.1, в которой сообщение содержит адрес, указывающий местоположение в элементе ЗУ, причем второй аппаратный процессор сконфигурирован так, чтобы определить, содержит ли сообщение имитационный запрос доступа к ЗУ в соответствии с этим адресом.

5. Компьютерная система по п.4, в которой определение, содержит ли сообщение имитационный запрос доступа к ЗУ, включает в себя сравнение адреса с заранее заданным адресом, и затем определение, что сообщение содержит имитационный запрос доступа к ЗУ в соответствии с результатом сравнения.

6. Компьютерная система по п.1, в которой сообщение содержит запрос на запись полезной нагрузки в элемент ЗУ и в которой второй аппаратный процессор сконфигурирован так, чтобы определить, содержит ли сообщение имитационный запрос доступа к ЗУ в соответствии с полезной нагрузкой.

7. Компьютерная система по п.1, в которой второй аппаратный процессор дополнительно сконфигурирован так, что:

поддерживает теневую файловую систему, содержащую карту соответствия между множеством пакетов данных, хранимых на элементе ЗУ, и множеством файлов первичной файловой системы, поддерживаемой операционной системой, исполняемой на первом аппаратном процессоре;

после приема исходного запроса доступа к ЗУ идентифицирует в соответствии с теневой файловой системой файл из множества файлов так, чтобы пакет данных

составлял часть этого файла; и

после идентификации указанного файла определяет, содержит ли этот файл вредоносное ПО.

8. Компьютерная система по п.1, в которой второй аппаратный процессор дополнительно сконфигурирован так, что:

определяет в соответствии с теневой файловой системой, указывает ли другое сообщение, полученное от первого аппаратного процессора через интерфейс ЗУ, на создание нового файла; и

затем, если это другое сообщение указывает на создание нового файла, обновляет теневую файловую систему, чтобы указать на создание нового файла.

9. Компьютерная система по п.1, в которой второй аппаратный процессор дополнительно сконфигурирован так, что после дешифрования пакета данных этот процессор выполняет запись дешифрованного пакета данных в элемент ЗУ.

10. Компьютерная система по п.1, в которой интерфейс ЗУ содержит элемент, выбранный из группы, включающей последовательный интерфейс SATA (Serial ATA) и интерфейс универсальной последовательной шины (USB).

11. Защищенное ЗУ, содержащее первый аппаратный процессор и энергонезависимый элемент ЗУ, причем защищенное ЗУ сконфигурировано для подсоединения ко второму аппаратному процессору через интерфейс ЗУ, сконфигурированный для приема запросов доступа к ЗУ, отформатированных в соответствии с протоколом передачи ЗУ, в котором:

второй аппаратный процессор сконфигурирован так, что:

при обнаружении программным обеспечением, выполняемым на втором аппаратном процессоре, запроса на сохранение в элементе ЗУ пакета данных этот процессор шифрует указанный пакет данных,

после шифрования пакета данных процессор передает исходный запрос доступа к ЗУ в интерфейс ЗУ, причем указанный исходный запрос доступа к ЗУ содержит зашифрованный пакет данных, генерирует имитационный запрос доступа к ЗУ в соответствии с протоколом передачи ЗУ, причем данный имитационный запрос доступа к ЗУ содержит по меньшей мере часть криптографического ключа, и передает этот имитационный запрос доступа к ЗУ в интерфейс ЗУ; и

первый аппаратный процессор сконфигурирован так, что:

после приема сообщения через интерфейс ЗУ процессор определяет, содержит ли сообщение имитационный запрос доступа к ЗУ; затем, если сообщение содержит такой имитационный запрос доступа к ЗУ, то процессор определяет криптографический ключ в соответствии с имитационным запросом доступа к ЗУ,

после приема исходного запроса доступа к ЗУ процессор использует криптографический ключ и дешифрует пакет данных, после чего определяет, содержит ли дешифрованный пакет данных вредоносное ПО.

12. Способ обеспечения компьютерной безопасности, включающий в себя следующие этапы:

подсоединение защищенного ЗУ к первому аппаратному процессору через интерфейс ЗУ, сконфигурированный для приема запросов доступа к ЗУ, отформатированных в соответствии с протоколом передачи ЗУ, при этом защищенное ЗУ содержит второй аппаратный процессор и энергонезависимый элемент ЗУ;

при обнаружении программным обеспечением, выполняемым на первом аппаратном процессоре, запроса на сохранение пакета данных в элементе ЗУ использование первого аппаратного процессора для шифрования пакета данных;

после шифрования пакета данных передача первым аппаратным процессором

исходного запроса доступа к ЗУ в интерфейс ЗУ, причем исходный запрос доступа к ЗУ содержит зашифрованный пакет данных;

использование первого аппаратного процессора для формирования имитационного запроса доступа к ЗУ в соответствии с протоколом передачи ЗУ, причем имитационный запрос доступа к ЗУ содержит по меньшей мере часть криптографического ключа;

использование первого аппаратного процессора для передачи имитационного запроса доступа к ЗУ в интерфейс ЗУ;

после приема сообщения через интерфейс ЗУ использование второго аппаратного процессора для определения, содержит ли данное сообщение имитационный запрос доступа к ЗУ;

затем, если сообщение содержит имитационный запрос доступа к ЗУ, то использование второго аппаратного процессора для применения криптографического ключа для дешифрования пакета данных; и

после дешифрования пакета данных использование второго аппаратного процессора для определения, содержит ли этот дешифрованный пакет данных вредоносное ПО.

13. Способ по п.12, дополнительно содержащий этап, на котором после определения того, содержит ли дешифрованный пакет данных вредоносное ПО, если дешифрованный пакет данных содержит вредоносное ПО, используют второй аппаратный процессор для передачи уведомления, отформатированного в соответствии с протоколом передачи ЗУ, в интерфейс ЗУ, причем данное уведомление сконфигурировано так, чтобы первый аппаратный процессор интерпретировал сообщение уведомления как предупреждение об угрозе безопасности.

14. Способ по п.13, в котором уведомление сконфигурировано так, чтобы вызвать аппаратное прерывание в первом аппаратном процессоре.

15. Способ по п.12, в котором сообщение содержит адрес, указывающий местоположение в элементе ЗУ, и в котором второй аппаратный процессор сконфигурирован так, чтобы определить, содержит ли сообщение имитационный запрос доступа к ЗУ в соответствии с этим адресом.

16. Способ по п.15, в котором определение, содержит ли сообщение имитационный запрос доступа к ЗУ, включает в себя следующие этапы:

использование второго аппаратного процессора для сравнения адреса с заранее заданным адресом и

затем определение, что сообщение содержит имитационный запрос доступа к ЗУ в соответствии с результатом сравнения.

17. Способ по п.12, в котором сообщение содержит запрос на запись полезной нагрузки в элемент ЗУ и в котором второй аппаратный процессор сконфигурирован так, чтобы определить, содержит ли сообщение имитационный запрос доступа к ЗУ в соответствии с полезной нагрузкой.

18. Способ по п.12, далее содержащий:

использование второго аппаратного процессора для поддержания теневой файловой системы, содержащей карту соответствия между множеством пакетов данных, хранимых на элементе ЗУ, и множеством файлов первичной файловой системы, поддерживаемой операционной системой, исполняемой на первом аппаратном процессоре;

после приема исходного запроса доступа к ЗУ использование второго аппаратного процессора для идентификации в соответствии с теневой файловой системой файла из множества файлов так, чтобы пакет данных образовывал часть файла; и

после идентификации файла использование второго аппаратного процессора для определения, содержит ли данный файл вредоносное ПО.

19. Способ по п.12, далее содержащий:

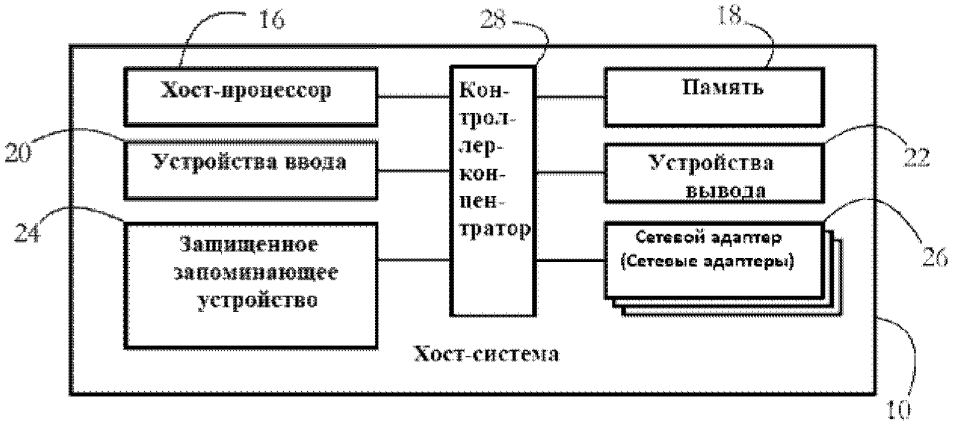
использование второго аппаратного процессора для определения в соответствии с теневой файловой системой, указывает ли другое сообщение, полученное от первого аппаратного процессора через интерфейс ЗУ, на создание нового файла; и

затем, если это другое сообщение указывает на создание нового файла, то использование второго аппаратного процессора для обновления теневой файловой системы, чтобы указать на создание нового файла.

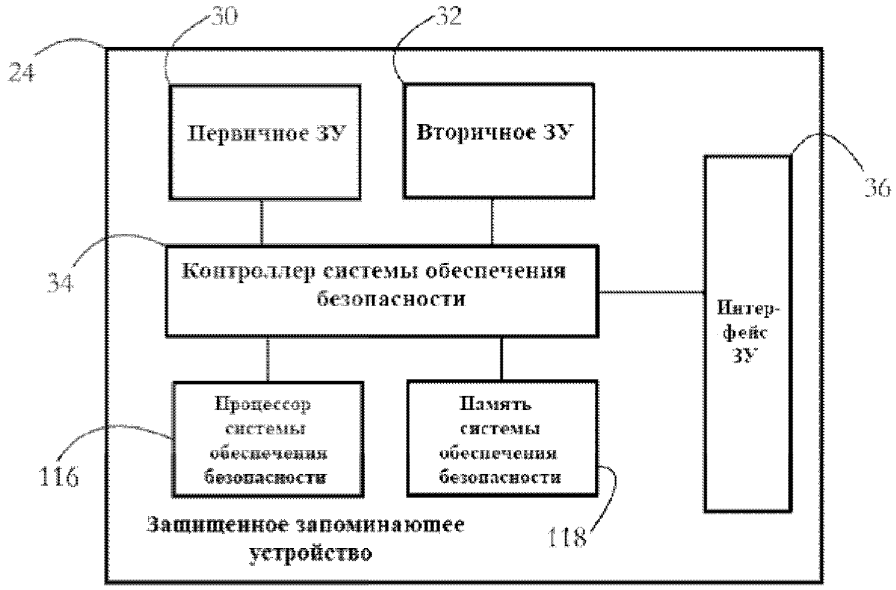
20. Способ по п.12, дополнительно содержащий этап, на котором после дешифрования пакета данных используют второй аппаратный процессор для записи дешифрованного пакета данных в элемент ЗУ.

21. Способ по п.12, в котором интерфейс ЗУ содержит элемент, выбранный из группы, включающей в себя последовательный интерфейс SATA (Serial ATA) и интерфейс универсальной последовательной шины (USB).

1

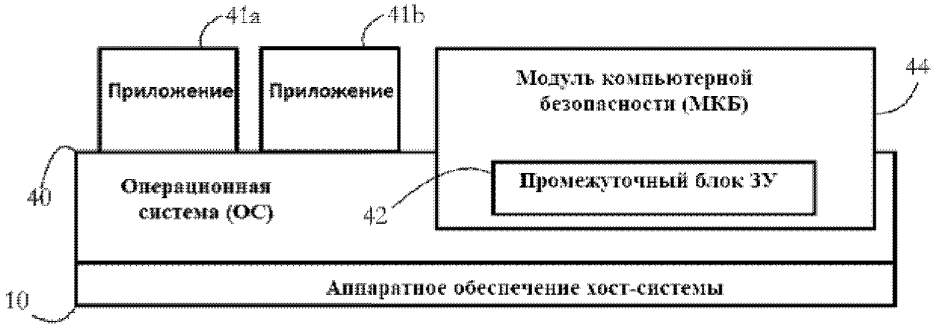

16
28
18
Хост-процессор
Контроллер-концентратор
Память
20
Устройства ввода
Устройства вывода
22
24
Защищенное запоминающее устройство
Сетевой адаптер (Сетевые адаптеры)
26
Хост-система
10

ФИГ. 1


30
32
24
Первичное ЗУ
Вторичное ЗУ
36
34
Контроллер системы обеспечения безопасности
Интерфейс ЗУ
Процессор системы обеспечения безопасности
Память системы обеспечения безопасности
116
118
Защищенное запоминающее устройство

ФИГ. 2

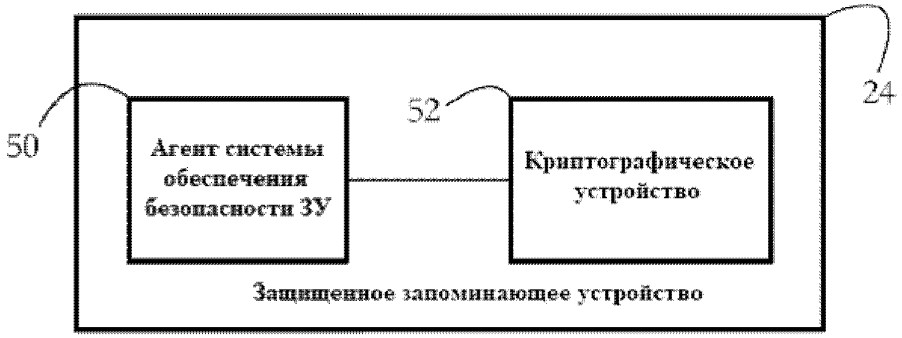
2



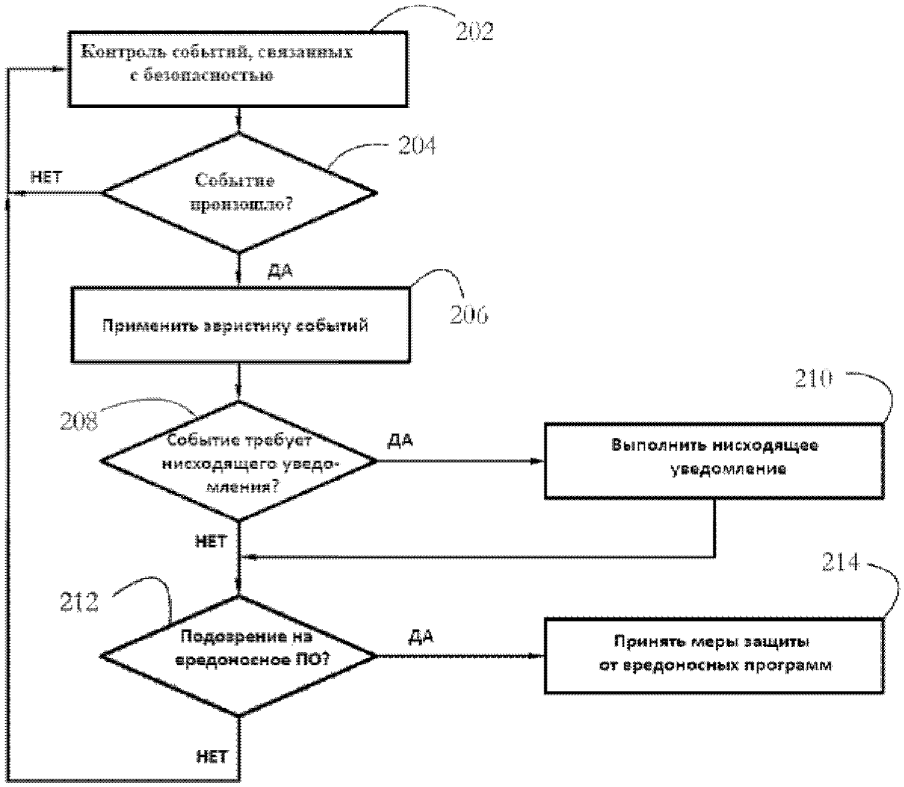
ФИГ. 3-А

ФИГ. 3-В

3

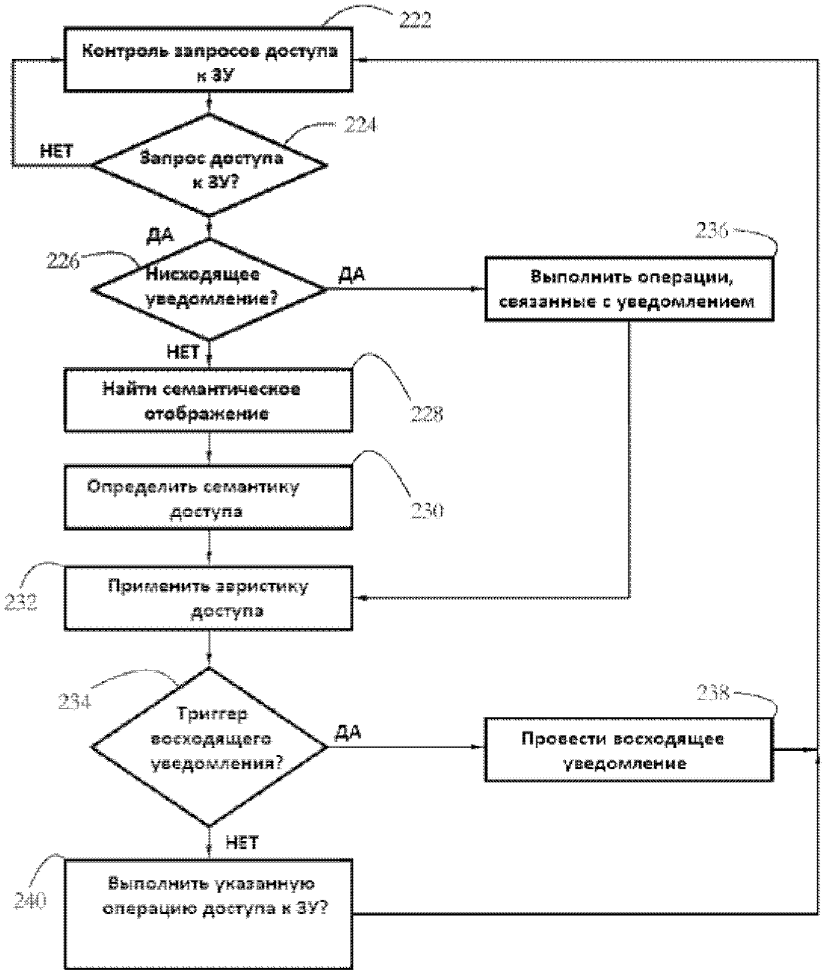

24
50
Агент системы обеспечения безопасности ЗУ
52
Криптографическое устройство
Защищенное запоминающее устройство

ФИГ. 4

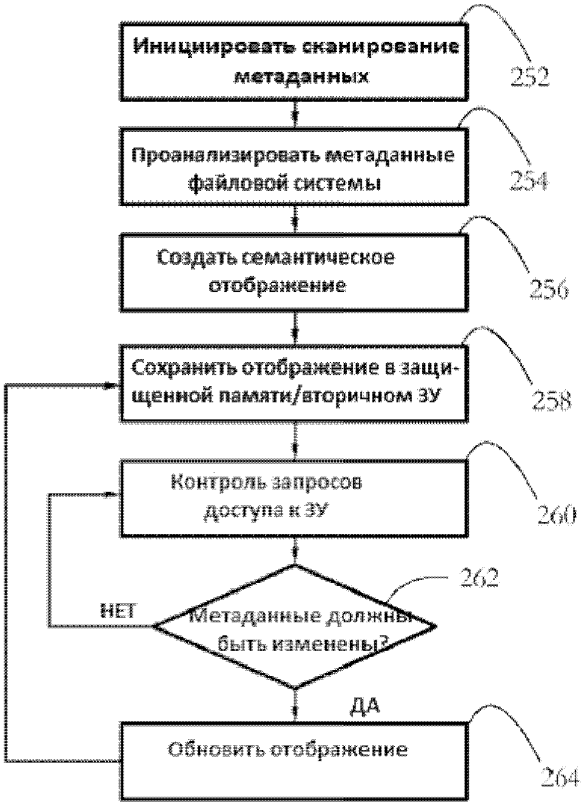

202
Контроль событий, связанных с безопасностью
204
Событие произошло?
НЕТ
ДА
206
Применить эвристику событий
208
Событие требует нисходящего уведомления?
ДА
210
Выполнить нисходящее уведомление
НЕТ
212
Подозрение на вредоносное ПО?
ДА
214
Принять меры защиты от вредоносных программ
НЕТ

ФИГ. 5

4


222
Контроль запросов доступа к ЗУ
224
НЕТ
Запрос доступа к ЗУ?
ДА
226
Нисходящее уведомление?
ДА
236
Выполнить операции, связанные с уведомлением
НЕТ
Найти семантическое отображение
228
Определить семантику доступа
230
232
Применить эвристику доступа
234
Триггер восходящего уведомления?
ДА
238
Провести восходящее уведомление
НЕТ
240
Выполнить указанную операцию доступа к ЗУ?

ФИГ. 6


Инициировать сканирование метаданных
252
Проанализировать метаданные файловой системы
254
Создать семантическое отображение
256
Сохранить отображение в защищенной памяти/вторичном ЗУ
258
Контроль запросов доступа к ЗУ
260
262
НЕТ
Метаданные должны быть изменены?
ДА
Обновить отображение
264

ФИГ. 7