(19) **RU** (11) **2 142 674** (13) **C1**

(51) МПК[6] **H 04 L 9/32**

РОССИЙСКОЕ АГЕНТСТВО
ПО ПАТЕНТАМ И ТОВАРНЫМ ЗНАКАМ

(12) ОПИСАНИЕ ИЗОБРЕТЕНИЯ К ПАТЕНТУ РОССИЙСКОЙ ФЕДЕРАЦИИ

(21), (22) Заявка: **98105209/09, 19.07.1996**

(24) Дата начала действия патента: **19.07.1996**

(30) Приоритет: **25.08.95**

(30) Приоритет: **25.08.1995 US 08/519,307**

(46) Дата публикации: **10.12.1999**

(56) Ссылки: **US 5557346 A, 17.09.96. US 4924515 A, 08.05.90. US 5412718 A, 02.05.95. EP 0356065 A2, 28.02.90. US 5052040 A, 24.09.91. EP 0586022 A1, 11.01.90. WO 95-23468 A2, 31.08.95. US 5421006 A, 30.05.95.**

(85) Дата перевода заявки PCT на национальную фазу: **25.03.98**

(86) Заявка PCT:
US 96/11925 (19.07.96)

(87) Публикация PCT:
WO 97/07657 (06.03.97)

(98) Адрес для переписки:
129010, Москва, ул.Большая Спасская, 25, стр.3, "Городисский и Партнеры", Емельянову Е.И.

(71) Заявитель:
Интел Корпорейшн (US)

(72) Изобретатель: **Дэвид В.Оксмит (US), Роберт К.Науэрхейз (US)**

(73) Патентообладатель:
Интел Корпорейшн (US)

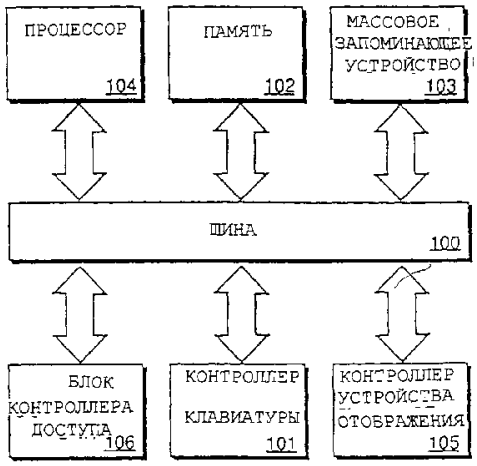
RU 2142674 C1

(54) УПРАВЛЕНИЕ ДОСТУПОМ С ИСПОЛЬЗОВАНИЕМ ПАРАМЕТРИЗИРОВАННОЙ ХЭШ-ФУНКЦИИ

(57) Реферат:

Изобретение относится к управлению доступом в вычислительной системе. Блок хранения получает блок данных, включающий в себя зашифрованный исполняемый модуль и сигнатурный компонент. Блок разделения, связанный с блоком хранения, отделяет сигнатурный компонент от зашифрованного исполняемого модуля. Блок дешифрирования, соединенный с блоком разделения, дешифрирует зашифрованный исполняемый модуль, используя сигнатурный компонент в качестве ключа. Это приводит к получению дешифрированной исполняемой программы. Блок идентификации, связанный с блоком дешифрирования, находит идентификационную метку в дешифрированной исполняемой программе и идентифицирует составной ключ, присвоенный идентификационной метке. Блок формирования сигнатуры, связанный с блоком идентификации, выполняет ключевой криптографический хэш-алгоритм для дешифрированной исполняемой программы, используя составной ключ в качестве ключа. Технический результат изобретения заключается в повышении надежности системы за счет обнаружения скрытых ошибок в интерфейсах между составляющими подсистемы. 5 с. и 7 з.п. ф-лы, 7 ил.

RU 2142674 C1



ФИГ.1

RU 2142674 C1

RU 2142674 C1

(19) **RU** (11) **2 142 674** (13) **C1**

(51) Int. Cl.[6] **H 04 L 9/32**

RUSSIAN AGENCY
FOR PATENTS AND TRADEMARKS

(12) ABSTRACT OF INVENTION

(21), (22) Application: **98105209/09, 19.07.1996**

(24) Effective date for property rights: **19.07.1996**

(30) Priority: **25.08.95**

(30) Priority: **25.08.1995 US 08/519,307**

(46) Date of publication: **10.12.1999**

(85) Commencement of national phase: **25.03.98**

(86) PCT application:
US 96/11925 (19.07.96)

(87) PCT publication:
WO 97/07657 (06.03.97)

(98) Mail address:
129010, Moskva, ul.Bol'shaja Spasskaja, 25, str.3, "Gorodisskij i Partnery", Emel'janovu E.I.

(71) Applicant:
Intel Korporejshn (US)

(72) Inventor: **Dehvid V.Oksmit (US), Robert K.Nauehrkhejz (US)**

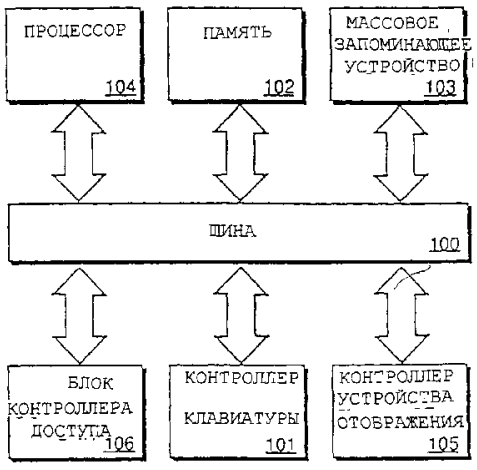
(73) Proprietor:
Intel Korporejshn (US)

(54) ACCESS CONTROL USING PARAMETER-CONTROLLED HASH FUNCTION

(57) Abstract:

FIELD: computer engineering. SUBSTANCE: storage unit receives data unit, which includes encrypted execution unit and signature component. Separation unit, which is connected to storage unit, separates signature component from encrypted execution unit. Decoding unit, which is connected to separation unit, deciphers encrypted execution unit using signature component as key in order to produce deciphered executable application. Identification unit, which is connected to decoding unit, searches for identification label in deciphered executable application and identifies complex key assigned to identification label. Signature generation unit, which is connected to identification unit, executes key cryptographic algorithm for deciphered executable application using complex key as key. EFFECT: increased reliability due to detection of hidden interface errors between system components.

12 cl, 7 dwg



ФИГ.1

RU 2142674 C1

RU 2142674 C1

Настоящее изобретение относится к управлению доступом в компьютерной системе. Более конкретно, настоящее изобретение относится к устройству и способу идентификации происхождения исполняемого модуля и использованию этой идентификации для определения уровня прав доступа, предоставляемых исполняемому модулю.

ПРЕДШЕСТВУЮЩИЙ УРОВЕНЬ ТЕХНИКИ

Нарушение защиты в компьютерных системах можно подразделить на преднамеренные и случайные. К видам преднамеренного доступа относятся несанкционированное считывание данных, несанкционированное изменение данных и несанкционированное разрушение данных. Большинство операционных систем предоставляют процессам средства для порождения других процессов. В такой среде возможно создание ситуации, при которой ресурсы операционной системы и файлы пользователя используются некорректно. Вормы и вирусы - два обычных способа некорректного использования. Защита компьютерной системы зависит от способности идентифицировать источник программ, которые должны быть выполнены, и проверять, что эти программы не были изменены таким образом, что могут представлять угрозу защите системы.

В дополнение к проверке подлинности источника программы также необходимо убедиться, что файлы, сегменты памяти, ЦПУ и другие ресурсы компьютерной системы могут быть использованы только теми процессами, которые получили соответствующее санкционирование от операционной системы. Существует несколько причин для обеспечения такой защиты. Наиболее очевидное - это потребность предотвратить вредное намеренное нарушение ограничения доступа. Самое важное - необходимость гарантировать, что каждый программный компонент, действующий в системе, использует системные ресурсы способами, согласующимися с установленными правилами для использования этих ресурсов. Защита может улучшить надежность, обнаруживая скрытые ошибки в интерфейсах между составляющими подсистемами. Раннее обнаружение ошибок в интерфейсе может предотвратить вывод из строя исправных подсистем другими подсистемами, которые неправильно функционируют.

Процесс в типовом случае работает в рамках области защиты. Эта область определяет ресурсы, к которым процесс может иметь доступ. Каждая область определяет набор объектов и типы операций, которые могут быть вызваны для каждого объекта. Способность выполнять операцию над объектом определяется как право доступа. Область - это набор прав доступа, каждое из которых в типовом случае представляет собой упорядоченную пару вида: "имя объекта, набор прав". Например, если область D имеет право доступа "файл F, {чтение, запись}", то процесс, исполняющийся в области D, может осуществлять как считывание, так и запись в файл F. Однако выполнение других операций над этим объектом не будет разрешено. Области могут быть разделены или они могут совместно использовать право доступа. Связь между процессом и областью также может быть статической или динамической. Таким образом, важно ограничить области защиты, доступные каждому процессу.

Таким образом существует необходимость в устройстве и способе предоставления защищенной от подделки сигнатуры исполняемого модуля, которая может быть использована, чтобы идентифицировать происхождение исполняемого модуля, определить, было ли сделано какое-либо изменение исполняемого модуля, уровень прав доступа и разрешение использования исполняемого модуля операционной системой.

СУЩНОСТЬ ИЗОБРЕТЕНИЯ

Описаны способ и устройство для управления доступом в компьютерной системе. Первый вариант осуществления контроллера доступа включает блок хранения. Блок хранения хранит блок данных, включающий в себя сигнатурный компонент и зашифрованный исполняемый модуль. Блок разделения связан с блоком хранения. Блок разделения получает блок данных и отделяет сигнатурный компонент от зашифрованного исполняемого модуля. Блок дешифрования связан с блоком разделения. Блок дешифрования получает зашифрованный исполняемый модуль и расшифровывает зашифрованный исполняемый модуль, преобразуя его в исполняемую программу. Это достигается выполнением алгоритма дешифрования, который использует сигнатурный компонент в качестве ключа для того, чтобы расшифровать зашифрованный исполняемый модуль. Блок идентификации связан с блоком дешифрования. Блок идентификации получает исполняемую программу, которую нужно использовать, и идентифицирует ключ, указанной идентификационной метки в исполняемой программе для вычисления криптографически зашифрованного по ключу хэшированного значения исполняемой программы. Блок генерации сигнатуры связан с блоком дешифрирования. Блок генерации сигнатуры получает исполняемую программу и вычисляет криптографически зашифрованное по ключу хэшированное значение для исполняемой программы с использованием заполненного ключа, идентифицированного блоком идентификации. Блок проверки связан с блоком хэширования. Блок проверки сравнивает ключевое хэшированное значение с сигнатурным компонентом, чтобы проверить источник блока данных и не была ли сделана какая-либо модификация блока данных. Блок назначения прав связан с блоком хэширования. Блок назначения прав получает ключ, используемый для вычисления ключевого хэшированного значения исполняемой программы, и присваивает права исполняемой программе в соответствии с правами, связанными с ключом.

Второй вариант осуществления настоящего изобретения раскрывает способ управления доступом в компьютерной системе. Сначала принимают блок данных, включающий в себя сигнатурный компонент и зашифрованный исполняемый модуль. После того, как блок данных получен, сигнатурный компонент отделяют от зашифрованного исполняемого модуля. Затем исполняемый

RU 2142674 C1

RU 2142674 C1

модуль расшифровывают посредством выполнения алгоритма дешифрования, который использует сигнатурный компонент в качестве ключа. Идентифицируют составной ключ, соответствующий идентифицированной метке в исполняемой программе. Составной ключ используют, чтобы вычислить ключевое хэшированное значение исполняемой программы. После того, как ключевое хэшированное значение вычислено, ключевое хэшированное значение сравнивают с сигнатурным компонентом, чтобы проверить источник блока данных. Исполняемой программе присваивают права в соответствии с правами, предварительно назначенными для ключа.

КРАТКОЕ ОПИСАНИЕ ЧЕРТЕЖЕЙ

Настоящее изобретение поясняется в нижеследующем подробном описании, иллюстрируемым чертежами. Описание и чертежи не предназначаются для ограничения изобретения конкретным вариантом осуществления, а приведены для пояснения, облегчения понимания сущности изобретения.

Фиг. 1 иллюстрирует первый вариант осуществления контроллера доступа, используемого в компьютерной системе.

Фиг. 2 иллюстрирует структурную схему первого варианта осуществления блока кодирования, соответствующего настоящему изобретению.

Фиг. 3 иллюстрирует процедуру кодирования блока информации с использованием блока кодирования, соответствующего настоящему изобретению.

Фиг. 4 иллюстрирует структурную схему второго варианта осуществления контроллера доступа, соответствующего настоящему изобретению.

Фиг. 5 иллюстрирует структурную схему третьего варианта осуществления системы обработки видеосигнала, соответствующего настоящему изобретению.

Фиг. 6 - блок-схема, иллюстрирующая способ кодирования.

Фиг. 7 - блок-схема, иллюстрирующая способ управления доступом в компьютерной системе.

ПОДРОБНОЕ ОПИСАНИЕ ВАРИАНТОВ ОСУЩЕСТВЛЕНИЯ ИЗОБРЕТЕНИЯ

Предложен новый блок контроллера доступа. Ниже изложены различные конкретные детали примеров осуществления изобретения. Однако специалистам в данной области техники должно быть понятно, что настоящее изобретение может быть осуществлено без использования этих конкретных деталей. Кроме того, общеизвестные методы, процедуры, компоненты и схемы не описываются подробно, чтобы не затенять сущность настоящего изобретения.

Некоторые части нижеследующего подробного описания представлены в терминах алгоритмов и символических представлениях операций над битами данных в памяти компьютерной системы. Эти алгоритмические описания и представления являются средствами, используемыми специалистами в данной области обработки данных для того, чтобы наиболее эффективно передать сущность их работы другим специалистам в данной области техники. Алгоритм представляет собой последовательность операций, ведущую к желаемому результату. Операции - это действия, осуществляемые над материальными объектами с помощью материальных средств. Обычно, хотя не обязательно, эти материальные объекты представляют собой электрические и магнитные сигналы, которые можно хранить, перемещать, объединять, сравнивать и подвергать другим манипуляциям. Как оказалось, иногда удобно, преимущественно по причинам общепринятого словоупотребления, ссылаться на эти сигналы как на биты, значения, элементы, символы, знаки, выражения, числа или подобное. Нужно помнить, однако, что все эти и подобные термины должны быть связаны с соответствующими физическими величинами и просто являются удобными обозначениями, относящимися к этим величинам. Если специально не оговорено другое, должно быть принято во внимание, что во всем настоящем изобретении такие термины, как "обработка", "вычисление", "расчет", "определение", "отображение" или подобные, относятся к действиям и процессам компьютерной системы или подобного электронного вычислительного устройства, которое манипулирует данными или преобразует данные, представляемые в виде физических (электронных) величин в регистрах компьютерной системы и запоминающих устройствах, в другие данные, также представленные в виде физических величин в запоминающих устройствах компьютерной системы или регистрах или других устройствах, таких, как устройство хранения информации или устройства передачи или отображения.

Фиг. 1 иллюстрирует в виде структурной схемы вычислительную систему первого варианта осуществления настоящего изобретения. Вычислительная система включает шину 100, интерфейс клавиатуры 101, внешнюю память 102, массовое запоминающее устройство 103, процессор 104, контроллер устройства отображения 105. Шина 100 соединена с контроллером устройства отображения, интерфейсом клавиатуры 101, микропроцессором 104, памятью 102 и массовым запоминающим устройством 103. Контроллер устройства отображения может быть соединен с устройством отображения. Интерфейс клавиатуры 101 может быть соединен с клавиатурой.

Шина 100 может быть отдельной шиной или комбинацией нескольких шин. Например, шина 100 может включать шину ISA (промышленной стандартной архитектуры), шину EISA (расширенной промышленной стандартной архитектуры), системную шину, X-шину, параллельную системную шину PS/2, шину PCI (межсоединений периферийных компонентов), шину PCMCIA (платы памяти персональных компьютеров Международной Ассоциации) или другие шины. Шина 100 может также включать комбинацию любых этих шин. Шина 100 обеспечивает коммуникационные связи между компонентами компьютерной системы. Интерфейс клавиатуры 101 может быть контроллером клавиатуры или другим интерфейсом клавиатуры. Интерфейс клавиатуры 101 может быть

RU 2142674 C1

RU 2142674 C1

специализированным устройством или частью другого устройства - такого, как контроллер шины или другой контроллер. Интерфейс клавиатуры 101 обеспечивает соединение клавиатуры с компьютерной системой и передает сигналы от клавиатуры к компьютерной системе. Внешняя память 102 может включать динамическое оперативное запоминающее устройство (ДОЗУ), статистическое запоминающее устройство (СОЗУ) или другие запоминающие устройства. Внешняя память 102 хранит информацию и данные, полученные из массового запоминающего устройства 103 и процессора 104 для использования процессором 104. Массовое запоминающее устройство 103 может быть накопителем на жестком диске, накопителем на гибком диске, накопителем CDROM, флэш (сверхбыстродействующее) - ПЗУ или другим массовым запоминающим устройством. Массовое запоминающее устройство 103 обеспечивает информацией и данными внешнюю память 102.

Процессор 104 обрабатывает информацию и данные из внешней памяти 102 и сохраняет информацию и данные во внешней памяти 102. Процессор 104 также принимает сигналы от контроллера клавиатуры 101 и передает информацию и данные контроллеру устройства отображения 105 для отображения на дисплее. Процессор 104 также передает видеоизображения контроллеру устройства отображения для отображения на дисплее. Процессор 104 может быть микропроцессором с полным набором команд (CISC-процессором), микропроцессором с сокращенным набором команд (RISC-процессором), микропроцессором со сверхдлинным словом команды (VLIW-процессором) или другим процессорным устройством. Контроллер устройства отображения 105 обеспечивает соединение устройства отображения с вычислительной системой и действует как интерфейс между устройством отображения и вычислительной системой. Контроллер устройства отображения может быть монохромным адаптером (MDA), цветным графическим адаптером (CGA), цветным графическим адаптером (MCGA), графическим адаптером (EGA), видеоадаптером (VGA), видеоадаптером (XGA) или другим контроллером устройства отображения. Устройство отображения может быть телевизионным приемником, компьютерным монитором, плоской индикаторной панелью или другим устройствам отображения. Устройство отображения принимает информацию и данные от процессора 104 через контроллер устройства отображения 105 и отображает информацию и данные для пользователя вычислительной системы.

Вычислительная система также включает блок контроллера доступа 106. Блок контроллера доступа 106 соединен с шиной 100. Набор ключей, которые связаны с правами доступа в компьютерной системе, хранится в блоке контроллера доступа 106. Каждый ключ определяет область, в которой программа функционирует. Ключи также определяют один или более составных ключей, которые используются в качестве параметров криптографической хэш-функции для генерации сигнатуры программы. Сигнатура программы в дальнейшем используется в качестве ключа шифрования для шифрования исполняемой программы.

Блок контроллера доступа 106 получает процесс, который должен быть выполнен процессором 104 из массового запоминающего устройства 103 или другого устройства ввода-вывода, подсоединенного к шине 100. Процесс включает в себя зашифрованный исполняемый модуль и сигнатурный компонент. Прежде чем вычислительная система исполнит программу, блок контроля доступа 106 проверяет, правильно ли построена сигнатура программы, из известного составного ключа. Проверяя сигнатурный компонент процесса, блок контроля доступа 106 идентифицирует происхождения процесса, проверяет, не был ли процесс модифицирован таким образом, что стал представлять угрозу компьютерной системе, и определяет уровень доступа, который операционная система должна предоставить процессу. Затем блок контроллера доступа 106 позволяет исполняемой программе выполняться с правами, присвоенными ключам, используемым при получении составного ключа.

Фиг. 2 иллюстрирует структурную схему первого варианта осуществления блока кодирования файла настоящего изобретения. Блок кодирования файла 210 включает генератор сигнатур 221 и блок шифрования 230. Генератор сигнатур 221 выполняет операции по созданию сигнатур исполняемой программы, которая будет выполнена процессором 104. Блок шифрования 230 шифрует файл, содержащий исполняемую программу, используя сигнатуру в качестве ключа. Генератор сигнатур 221 выполняет криптографическую ключевую хэш-функцию на открытом тексте исполняемой программы, генерируя зашифрованный текст. Генератор сигнатур 221 использует ключи, которые являются составными ключами тех ключей, которые хранятся в блоке управления доступом 106. Каждый из составных ключей, используемых в криптографической хэш-функции, связан с набором прав доступа. Эти права присваиваются исполняемой программе перед ее выполнением.

Генератор сигнатур 221 включает блок вычисления 222 и блок шифрования 223. Генератор сигнатур 221 может использовать блок вычисления и блок шифрования 223 для выполнения любого числа криптографических ключевых хэш-функций или алгоритмов шифрования на открытом тексте исполняемой программы. Ключи могут быть индивидуальными симметричными ключами или общими асимметричными ключами. Различие состоит в степени защиты, требуемой копией ключа операционной системы. Генератор сигнатур 221 может использовать такие стандартные алгоритмы как Lucifer, Madryga, NewDES, FEAL, REDOC, LOKI, Khufu, Khafre или IDEA, с тем, чтобы генерировать криптографическое ключевое хэшированное значение для исполняемой программы. В первом варианте осуществления настоящего изобретения блок вычисления 222 и блок шифрования 223 используют формирование Цепочки Шифрованных Блоков (ФЦШБ) Стандарта

RU 2142674 C1

RU 2142674 C1

Шифрования Данных (СШД) для того, чтобы генерировать криптографическое ключевое хэшированное значение для исполняемой программы.

Фиг. 3 иллюстрирует операции, выполняемые блоком вычисления 222 и блоком шифрования 223, когда он использует ФЦШБ, чтобы генерировать ключевое хэшированное значение для исполняемой программы. Формирование цепочки использует механизм обратной связи. Результаты шифрования предыдущих блоков поступают обратно на шифрование текущего блока. Другими словами, предыдущий блок используется для изменения шифрования следующего блока. Каждый блок зашифрованного текста зависит и от блока открытого текста, который его породил, и от предыдущих блоков открытого текста. В ФЦШБ открытый текст обрабатывается согласно операции "исключающее ИЛИ" совместно с предыдущим блоком зашифрованного текста прежде, чем он будет зашифрован.

В этом примере блок кодирования 210 получает файл, содержащий исполняемую программу, с размером 24 байта. Генератор сигнатур 221 разбивает файл в 24 байта на три раздела по 8 байтов. Первые 8 байтов открытого текста представляются как P1 в блоке 301. P1 подвергается обработке по процедуре "исключающее ИЛИ" с инициирующим вектором (IV), хранящимся в блоке вычисления 222. Это приводит к получению первого результата. Инициирующий вектор является функцией первого составного ключа, связанного с набором прав доступа, которые будут присвоены исполняемой программе. После этого P1 обрабатывается по процедуре "исключающее ИЛИ" с IV; блок шифрования 223 выполняет ключевой алгоритм шифрования, используя второй составной ключ для первого результата, формируя первый зашифрованный результат C1. Ключевой алгоритм шифрования может быть одним из ряда различных ключевых алгоритмов шифрования, включая любой из ключевых алгоритмов шифрования, перечисленных ранее. Блок вычисления 222 обрабатывает по процедуре "исключающее ИЛИ" первый зашифрованный результат со вторым 8-байтовым разделом, представляемым как P2, чтобы получить второй 8-байтовый результат. Блок шифрования 223 выполняет ключевой алгоритм шифрования, используя второй составной ключ для второго результата и формируя второй зашифрованный результат C2. Блок вычисления 222 обрабатывается по процедуре "исключающее ИЛИ" второй зашифрованный результат с третьим 8-байтовым разделом, чтобы сформировать третий 8-байтовый результат. Блок шифрования 223 выполняет ключевой алгоритм шифрования, используя второй составной ключ для третьего результата. Это приводит к получению третьего зашифрованного результата C3, который используется в качестве сигнатуры исполняемой программы.

Блок генерации сигнатуры 221 генерирует сигнатуру исполняемой программы, которая является функцией всех символов в файле. Таким образом, если исполняемая программа изменяется, можно будет обнаружить модификацию, повторно вычисляя криптографическое ключевое хэш-значение и сравнивая повторно вычисленное значение с первоначальной сигнатурой.

Блок шифрования 230 выполняет операции по шифрованию исполняемой программы посредством выполнения алгоритма шифрования с использованием сигнатуры, созданной в результате работы ключевого криптографического хэш-алгоритма в качестве ключа. Это приводит к получению зашифрованного исполняемого модуля. Шифрование исполняемой программы обеспечивает дополнительный уровень защиты, чтобы предотвратить несанкционированное считывание исполняемой программы третьим лицом. Блок шифрования 230 может использовать все разнообразие алгоритмов шифрования. Зашифрованный исполняемый модуль и сигнатура посылаются в виде файла вычислительной системе для исполнения.

Фиг. 4 иллюстрирует структурную схему второго варианта осуществления контроллера доступа настоящего изобретения. Блок управления доступом 400 включает блок хранения 410, блок разделения 420, блок дешифрования 430, блок идентификации 440, блок генерации сигнатуры 450, блок проверки 460 и блок назначения прав 470.

Блок хранения 410 получает блок данных, включающий зашифрованный исполняемый модуль и сигнатурный компонент. Блок хранения 410 может включать ДОЗУ, СОЗУ или другие виды оперативной памяти. Блок хранения 410 использует признак, чтобы указать компьютерной системе, является ли хранящийся файл исполняемым модулем или исполняемой программой. Признак указывает компьютерной системе на то, что блок хранения 410 используется в качестве временной памяти, когда хранящийся файл - исполняемый модуль. Признак указывает компьютерной системе на то, что блок хранения 410 используется в качестве места исполнения, когда файл - исполняемая программа.

Блок разделения 420 соединен с блоком хранения 410. Блок разделения 420 получает блок данных из блока хранения 410 и отделяет зашифрованный исполняемый модуль от сигнатурного, компонента. Это позволяет блоку управления доступом 400 обрабатывать зашифрованный исполняемый модуль и сигнатурный компонент независимо.

Блок дешифрования 430 соединен с блоком хранения 420 и блоком хранения 410. Блок дешифрования 430 получает зашифрованный исполняемый модуль в форме зашифрованного текста и сигнатурного компонента из блока разделения 420. Блок дешифрования 430 расшифровывает зашифрованный исполняемый модуль, используя сигнатурный компонент в качестве ключа дешифрования. Блок дешифрования 430 преобразует зашифрованный исполняемый модуль в дешифрованную исполняемую программу.

Блок идентификации 440 соединен с блоком дешифрования 430 и блоком хранения 410. Блок идентификации 440 получает исполняемую программу от блока дешифрования 430. Блок идентификации 440

RU 2142674 C1

RU 2142674 C1

считывает идентификационную метку в исполняемой программе и идентифицирует соответствующий составной ключ, который присвоен идентификационной метке. Этот составной ключ - обычно тот же самый ключ, который используется блоком генерации сигнатуры 221, чтобы генерировать ключевое хэшированное значение для исполняемой программы. В первом варианте осуществления настоящего изобретения, идентифицирующий процессор 440 содержит поисковую таблицу, согласовывающую различные идентификационные метки с различными составными ключами. Составной ключ связан со специфическими правами доступа, которые предоставляются исполняемой программе.

Блок генерации сигнатуры 450 соединен с блоком идентификации 440 и блоком хранения 410. Блок генерации 450 получает идентификацию составного ключа, присвоенного идентификационной метке исполняемой программы. Блок генерации сигнатуры 450 выполняет операции по вычислению криптографического ключевого хэш-значения дешифрованной исполняемой программы, полученной блоком идентификации 440, используя идентификацию составного ключа, полученную блоком идентификации 440. Блок генерации сигнатуры 450 хранит множество ключей, которым соответствуют специфические права доступа в компьютерной системе. Эти ключи используются для формирования множество составных ключей для кодирования и декодирования исполняемых программ и дешифрованных исполняемых программ.

Блок проверки 460 связан с блоком генерации сигнатуры 450 и блоком хранения 410. Блок проверки 460 получает сигнатурный компонент исполняемого модуля из блока хранения 410 и ключевое хэшированное значение дешифрованной исполняемой программы из блока генерации сигнатуры 450. Блок проверки 460 сравнивает ключевое хэшированное значение дешифрованной исполняемой программы с сигнатурным компонентом исполняемого модуля. Если они совпадают, блок проверки 460 разрешает выполнение дешифрированной исполняемой программы компьютерной системой. Если они не совпадают, блок проверки 460 принимает решение, что исполняемый модуль был изменен, и не разрешает его выполнения компьютерной системой.

В первом варианте осуществления настоящего изобретения блок генерации сигнатуры 450 не получает идентификацию составного ключа, используемого для вычисления ключевой хэш-функции дешифрированной исполняемой программы. Вместо этого блок генерации сигнатуры 450 вычисляет несколько ключевых хэшированных значений дешифрованной исполняемой программы, используя составные ключи, полученные перестановками ключей, хранящихся в блоке генерации сигнатуры 450. Эти ключевые хэшированные значения принимаются блоком проверки 460, который определяет, соответствует ли любой из ключевых хэшированных значений первоначальному сигнатурному компоненту. Точно так же, если имеется соответствие между сигнатурным компонентом исполняемого модуля и любым из вычисленных ключевых хэшированных значений дешифрованной исполняемой программы, блок проверки 460 позволяет осуществлять выполнение дешифрированной исполняемой программы компьютерной системой. Если нет никаких совпадений, блок проверки 460 принимает решение, что исполняемый модуль был изменен и не должен выполняться компьютерной системой.

Блок назначения прав 470 соединен с блоком проверки 460 и блоком хранения 410. Блок назначения прав 470 получает идентификацию составного ключа, используемого для вычисления соответствующего ключевого хэшированного значения для сигнатурного компонента исполняемого модуля. Когда блок назначения прав 470 получает сигнал от блока проверки 460, показывающий, что дешифрированная исполняемая программа должна быть выполнена компьютерной системой, блок назначения прав 470 выполняет операции, необходимые для присвоения прав, доступных программе, идентифицируя права, связанными со специфическими составными ключами, используемыми для вычисления соответствующего ключевого хэшированного значения. В первом варианте осуществления настоящего изобретения блок назначения прав 470 может содержать поисковую таблицу, согласовывающие различные составные ключи с различными уровнями прав доступа. После того, как блок назначения прав 470 присвоит соответствующие права дешифрированной исполняемой программе, блок назначения прав 470 изменяет признак в блоке хранения 410, чтобы указать компьютерной системе, что блок хранения 410 используется в качестве исполняемого модуля. Компьютерная система воспринимает указания, что блок хранения 410 содержит исполняемую программу, и переходит к ее выполнению.

Фиг. 5 иллюстрирует в форме структурной схемы типовую компьютерную систему для четвертого варианта осуществления настоящего изобретения. Компьютерная система включает шину 500, микропроцессор 510, память 520, устройство хранения данных 530, контроллер клавиатуры 540 и контроллер устройства отображения 550.

Микропроцессор 510 может представлять собой микропроцессор с полным набором команд (CISC-процессор), микропроцессор с сокращенным набором команд (RISC-процессор) или другое процессорное устройство. Микропроцессор выполняет команды или код, хранящиеся в памяти 520, и выполняет операции над данными, хранящимися в памяти 520. Кроме того, компьютерная система 500 включает устройство хранения данных 530 (такое, как накопитель на жестком, гибком или оптическом диске), которое соединено с шиной 515. Контроллер устройства отображения 550 также соединен с шиной 515. Контроллер устройства отображения 550 обеспечивает соединение устройства отображения с компьютерной системой. Контроллер клавиатуры 540 обеспечивает соединение клавиатуры с компьютерной системой и передает сигналы от клавиатуры к компьютерной системе.

RU 2142674 C1

Память, 520 соединена с микропроцессором 510 через шину 500. Память 520 может быть динамическим запоминающим устройством (ДОЗУ), статическим оперативным запоминающим устройством (СОЗУ) или другим запоминающим устройством. Память 520 может хранить команды или код, исполняемые процессором 510 и являющиеся частью прикладных программ, программ операционной системы или других компьютерных программ. Память 520 включает модуль хранения 521, модуль разделения 522, модуль дешифрования 523, модуль идентификации 524, модуль генерации сигнатуры 525, модуль проверки 526 и модуль назначения прав 527. Модуль хранения 521 включает первое множество исполняемых команд процессора, которое выполняются процессором 510 так, как показана фиг. 7. Модуль хранения выполняет функции, подобные тем, что выполняет блок хранения 410 на фиг. 4. Модуль разделения 522 включает второе множество исполняемых команд процессора, которые выполняются процессором 510 так, как показано на фиг. 7. Модуль разделения 522 выполняет функции, подобные тем, что выполняет блок разделения 420 на фиг. 4. Модуль дешифрования 523 включает третье множество исполняемых команд процессора, которое выполняются процессором 510 так, как показано на фиг. 7. Модуль дешифрования 523 выполняет функции, подобные тем, что выполняет блок дешифрования 430 на фиг.4. Модуль идентификации 524 включает четвертое множество исполняемых команд процессора, которые выполняются процессором 510 так, как показано на фиг.7. Модуль идентификации 524 функционирует аналогично блоку идентификации 440 на фиг.4. Модуль генерации сигнатуры 525 включает пятое множество исполняемых команд процессора, которое выполняются процессором 510 так, как показано на. фиг.7. Модуль генерации сигнатуры 525 выполняет функции, подобные тем, что выполняет блок генерации сигнатуры 450 на фиг.4. Модуль проверки 526 включает шестое множество исполняемых команд процессора, которые выполняются процессором 510 так, как показано на фиг.7. Модуль проверки 526 выполняет функции, подобные тем, что выполняет блок проверки 460 на фиг. 4. Модуль назначения прав 527 включает седьмое множество исполняемых команд процессора, которые выполняются процессором 510 так, как показано на фиг.7. Модуль назначения прав 527 функционирует аналогично блоку назначения прав 470 на фиг. 4.

Фиг. 6 - блок-схема, иллюстрирующая способ кодирования исполняемой программы, которая должна быть выполнена компьютерной системой. Сначала получают исполняемую программу, как показано в блоке 601. Затем получают составной ключ, определяющий связанные права, которые будут присвоены исполняемой программе, как показано в блоке 602. Выполняют ключевой криптографический хэш-алгоритм для исполняемой программы. Используемый составной ключ может быть или индивидуальными симметричными ключами или общими асимметричными ключами. Это приводит к получению зашифрованного ключевого хэшированного значения которое, служит в качестве сигнатуры для исполняемой программы. Это показано в блоке 603.

Затем зашифровывают исполняемую программу, используя зашифрованное ключевое хэшированное значение в качестве ключа. В результате этого получают исполняемый модуль. Это показано в блоке 604. После того, как исполняемая программа зашифрована в исполняемый модуль, посылают исполняемый модуль и сигнатурный компонент в компьютерную систему на обработку и выполнение. Это показано в блоке 605.

Фиг. 7 - блок-схема, иллюстрирующая способ управления доступом в компьютерной системе. Сначала получают блок данных, включающий в себя сигнатурный компонент и исполняемый модуль, как показано в блоке 701. Отделяют сигнатурный компонент от исполняемого модуля, как показано в блоке 702. Расшифровывают исполняемый модуль, используя сигнатурный компонент в качестве ключа. Это приводит к получению дешифрованной исполняемой программы, что показано в блоке 703.

Затем определяют идентификационную метку в дешифрированной исполняемой программе, как показано в блоке 704. Идентифицируют составной ключ, связанный с идентификационной меткой, как показано в блоке 705. Вычисляют ключевое криптографическое хэш-значение для дешифрированной исполняемой программы, используя составной ключ, связанный с идентификационной меткой, как показано в блоке 706. Затем проверяют источник блока данных и был ли идентифицирован блок, сравнивая сигнатурный компонент в блоке данных с вычисленным ключевым криптографическим хэш-значением для дешифрированной исполняемой программы. Это показано в блоке 707. Если сигнатурный компонент в блоке данных не соответствует вычисленному ключевому криптографическому хэш-значению, то дешифрированную исполняемую программу не выполняют, что показано в блоке 708. Если сигнатурный компонент в блоке данных соответствует вычисленному ключевому криптографическому хэш-значению, то присваивают соответствующие права дешифрированной исполняемой программе, как определено составным ключом. Это показано в блоке 709. Наконец, выполняют дешифрированную исполняемую программу, как показано в блоке 710.

В предшествующем описании изобретения приведены ссылки на конкретные варианты его осуществления. Очевидно, однако, что могут быть сделаны различные модификации и изменения изобретения без изменения сущности и объема изобретения, определяемого формулой изобретения, которая прилагается.

Соответственно описания и чертежи носят скорее иллюстративный, чем ограничительный характер.

Специалистам в данной области техники при рассмотрении предшествующего описания должны быть очевидны различные изменения и модификации настоящего изобретения, поэтому конкретные варианты

RU 2142674 C1

RU 2142674 C1

осуществления, показанные в описании, никоим образом не предназначены для ограничения. Следовательно, ссылки на подробности специфических вариантов осуществления не предназначены для ограничения объема формулы изобретения, которая содержит только существенные признаки изобретения.

Формула изобретения:

1. Способ формирования закодированного исполняемого модуля, включающий выполнение криптографической ключевой хэш-функции для исполняемой программы для генерирования сигнатурного компонента с использованием первого ключа, имеющего связанный набор прав доступа, присваиваемых исполняемой программе, и выполнение алгоритма шифрования для исполняемой программы с использованием сигнатурного компонента в качестве второго ключа.

2. Способ по п.1, отличающийся тем, что при выполнении криптографической ключевой хэш-функции осуществляют выполнение алгоритма формирования цепочки шифрованных блоков стандарта шифрования данных.

3. Способ управления доступом, включающий отделение сигнатурного компонента от исполняемого модуля в блоке данных, дешифрирование исполняемого модуля в исполняемую программу с использованием сигнатурного компонента, вычисление криптографического ключевого хэш-значения исполняемой программы с использованием ключа и проверку источника блока данных путем сравнения сигнатурного компонента с криптографическим ключевым хэш-значением.

4. Способ по п.3, отличающийся тем, что включает нахождение идентификационной метки в исполняемой программе и поиск ключа, соответствующего идентификационной метке, для выполнения криптографический ключевой хэш-функции для исполняемой программы.

5. Способ по п. 3, отличающийся тем, что включает назначение прав исполняемой программе в соответствии с правами, связанными с ключом.

6. Блок для кодирования исполняемой программы, содержащий блок генерации сигнатуры для выполнения криптографической ключевой хэш-функции для исполняемой программы для генерирования сигнатурного компонента с использованием первого ключа, имеющего связанный набор прав доступа, присваиваемых исполняемой программе, и первый блок шифрования, связанный с блоком генерации сигнатуры, для шифрования исполняемой программы с использованием сигнатурного компонента в качестве второго ключа.

7. Блок для кодирования исполняемой программ по п.6, отличающийся тем, что блок генерации сигнатуры дополнительно включает блок вычисления и второй блок шифрования, который выполняет алгоритм формирования цепочки зашифрованных блоков стандарта шифрования данных.

8. Блок управления доступом, включающий блок разделения для отделения сигнатурного компонента, полученного в результате выполнения ключевой хэш-функции для исполняемой программ, от зашифрованного исполняемого модуля в блоке данных, блок дешифрирования, связанный с блоком разделения, для дешифрирования зашифрованного исполняемого модуля с преобразованием его в дешифрированную исполняемую программу с сигнатурным компонентом, блок генерации сигнатуры, соединенный с блоком дешифрирования, для вычисления криптографического ключевого хэш-значения для дешифрированной исполняемой программы с использованием ключа, и блок проверки, соединенный с блоком генерации сигнатуры, для сравнения криптографического ключевого хэш-значения с сигнатурным компонентом.

9. Блок управления доступом по п.8, отличающийся тем, что блок генерации сигнатуры дополнительно содержит компонент для хранения ключа, предназначенный для хранения ключа, используемого блоком генерации сигнатуры.

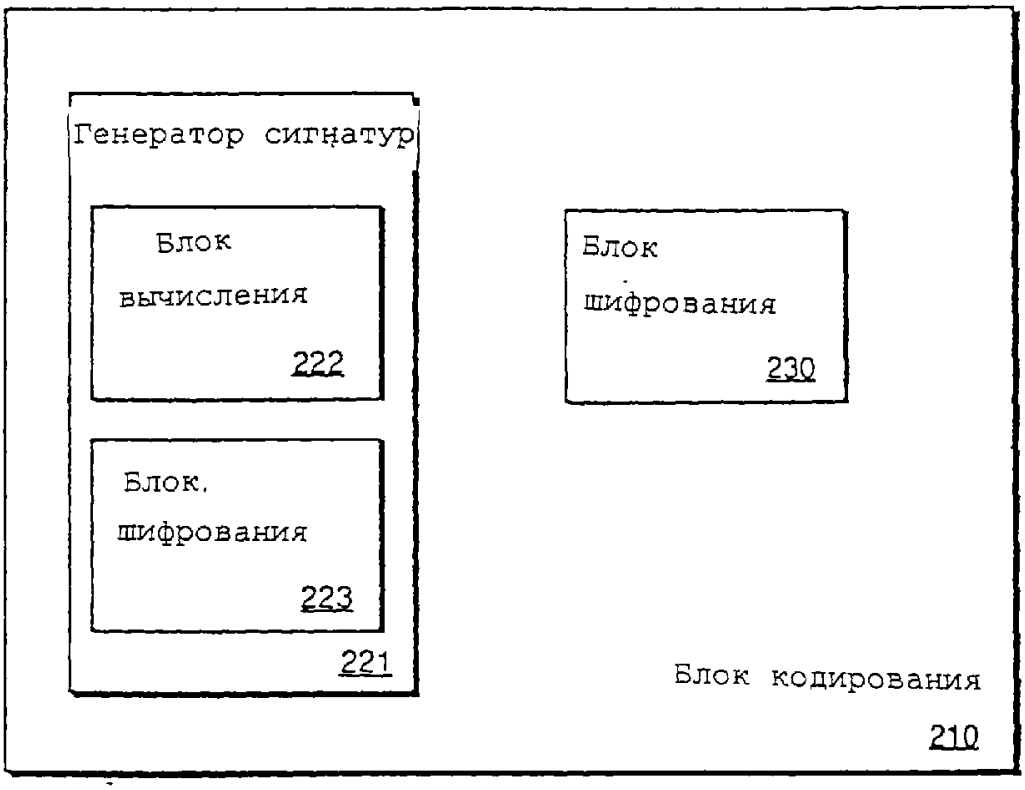
10. Блок управления доступом по п.8, отличающийся тем, что дополнительно содержит блок идентификации, соединенный с блоком дешифрирования, предназначенный для идентификации ключа, указывающего на идентификационную метку в исполняемой программе, для вычисления криптографического хэш-значения для дешифрированной исполняемой программы.

11. Блок управления доступом по п.8, отличающийся тем, что содержит блок назначения прав, связанный с блоком генерации сигнатуры, для присвоения права дешифрированной исполняемой программе в соответствии с правами, связанными с ключом.

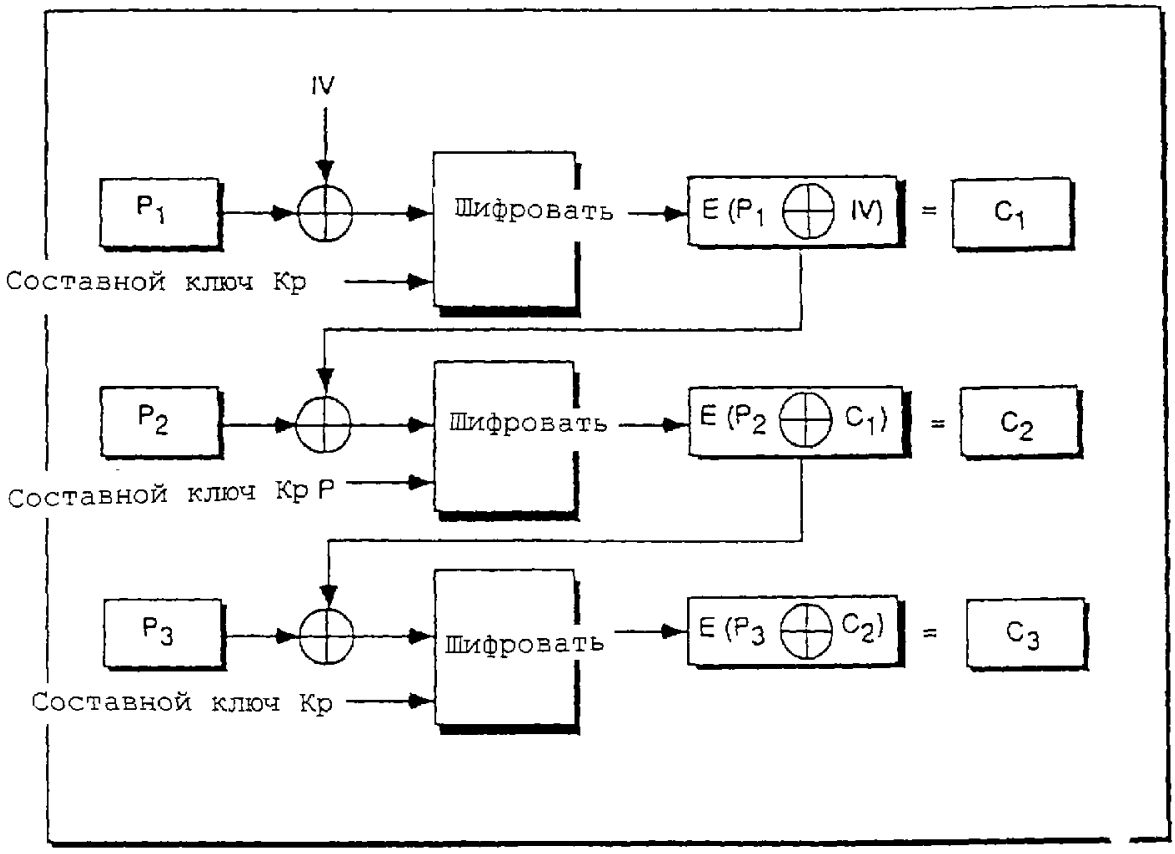
12. Компьютерная система, содержащая шину, память, соединенную с шиной, блок разделения, предназначенный для отделения сигнатурного компонента от зашифрованного исполняемого модуля в блоке данных, блок дешифрирования, связанный с блоком разделения, для дешифрирования зашифрованного исполняемого модуля с преобразованием его в исполняемую программу с сигнатурным компонентом, блок генерации сигнатуры, связанный с блоком дешифрирования, для вычисления криптографического ключевого хэш-значения для исполняемой программы с использованием ключа, и блок проверки, связанный с блоком генерации сигнатуры, предназначенный для сравнения криптографического ключевого хэш-значения с сигнатурным компонентом.

RU 2142674 C1

RU 2142674 C1



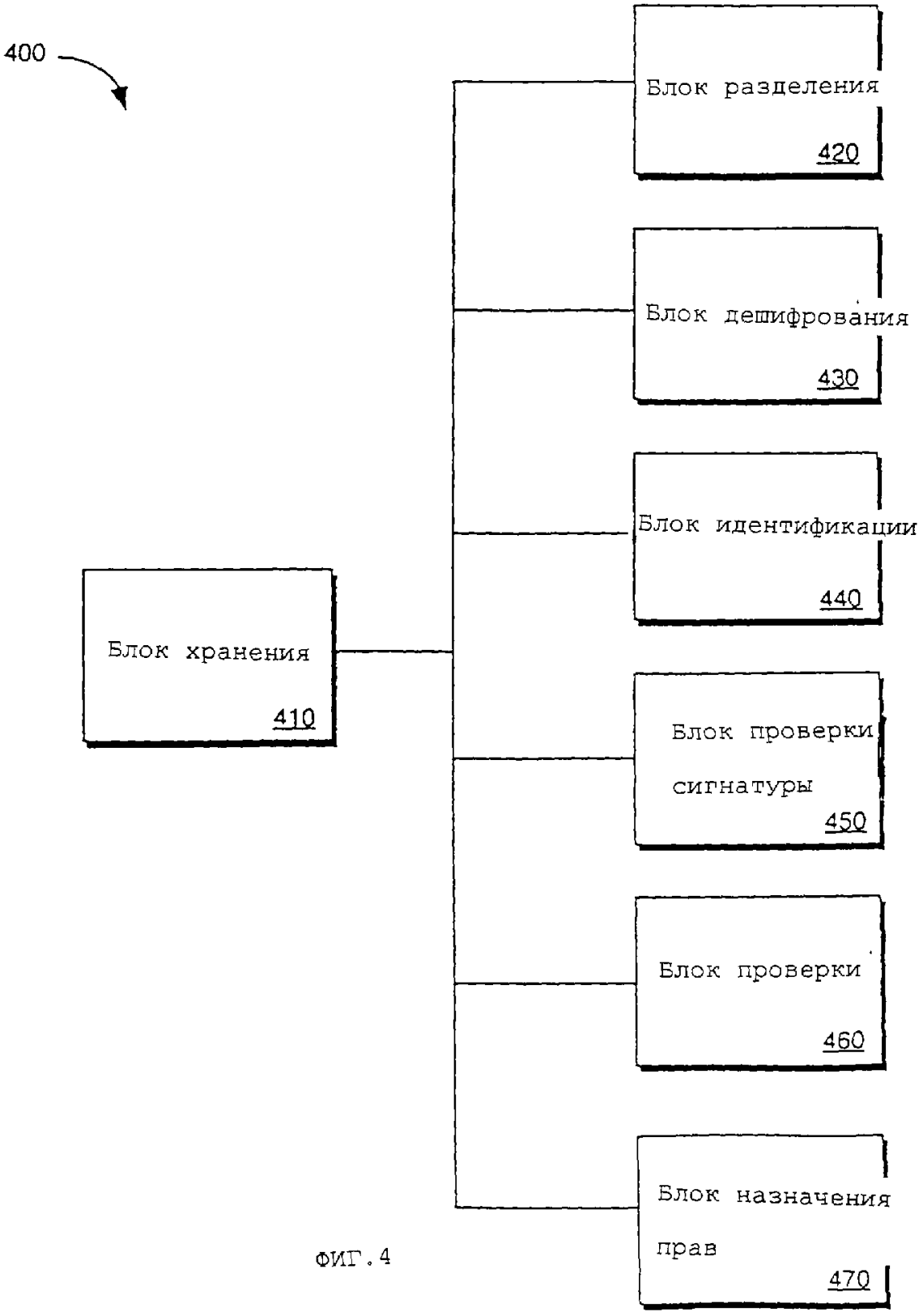
ФИГ. 2



ФИГ. 3

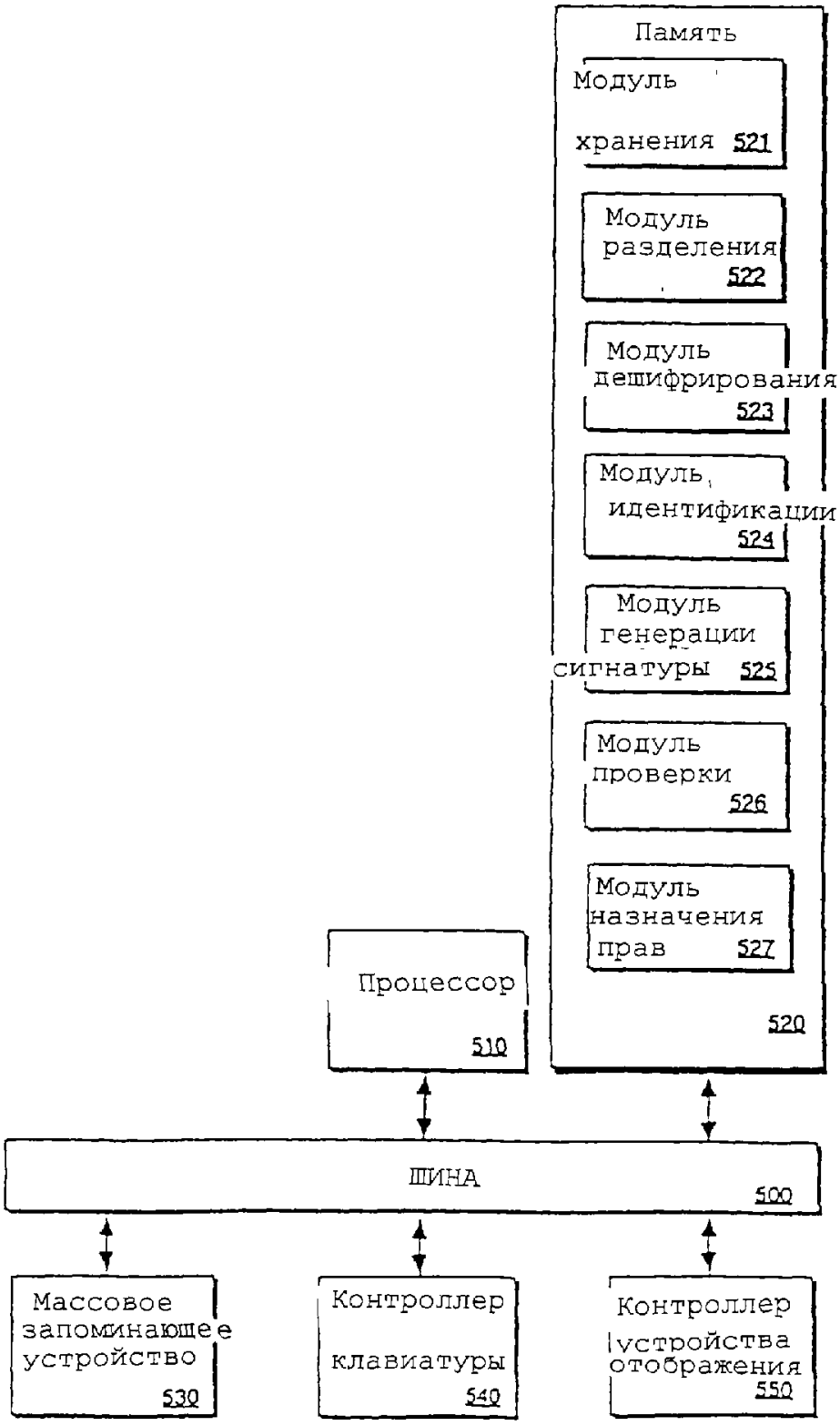
RU 2142674 C1

RU 2142674 C1



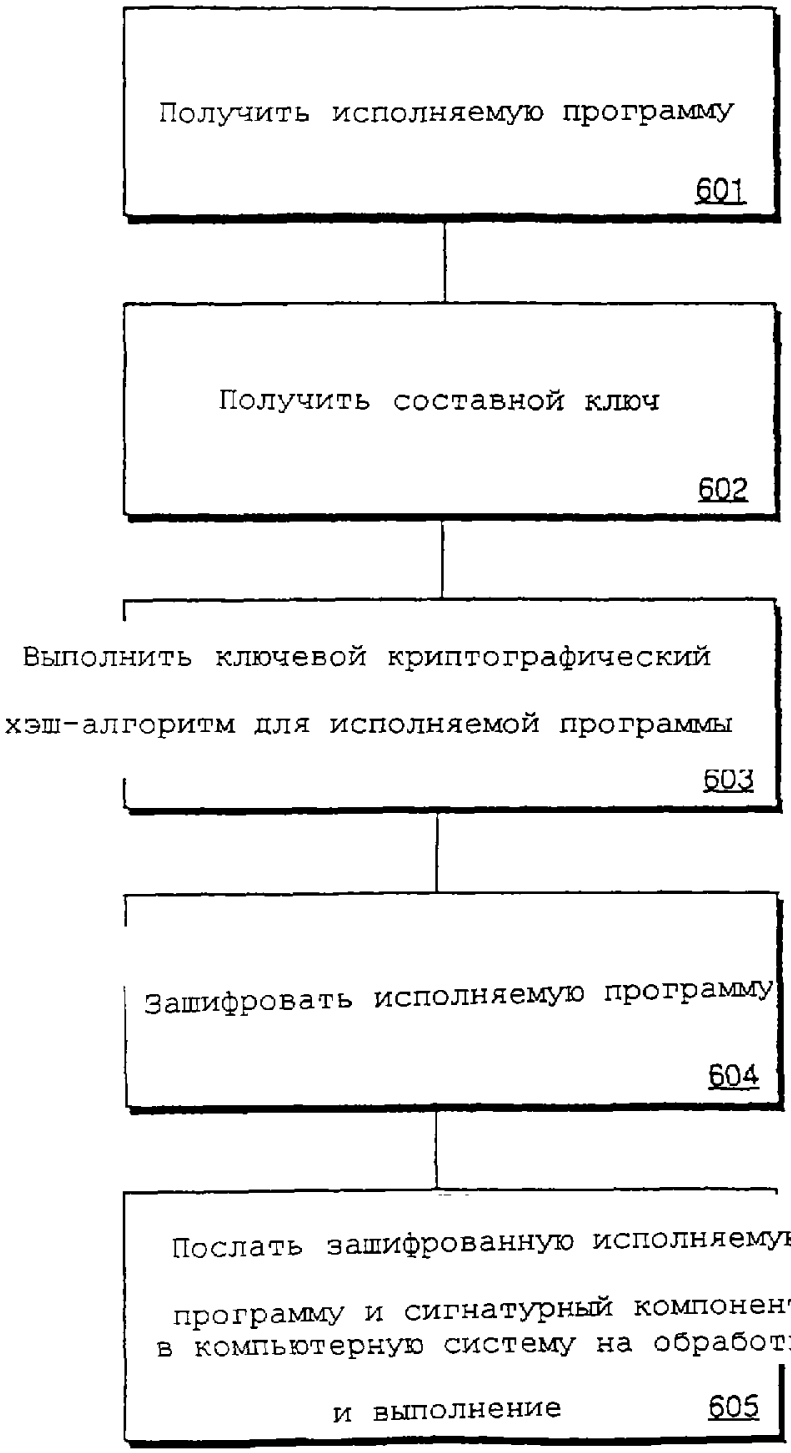
ФИГ. 4

RU 2142674 C1



ФИГ. 5

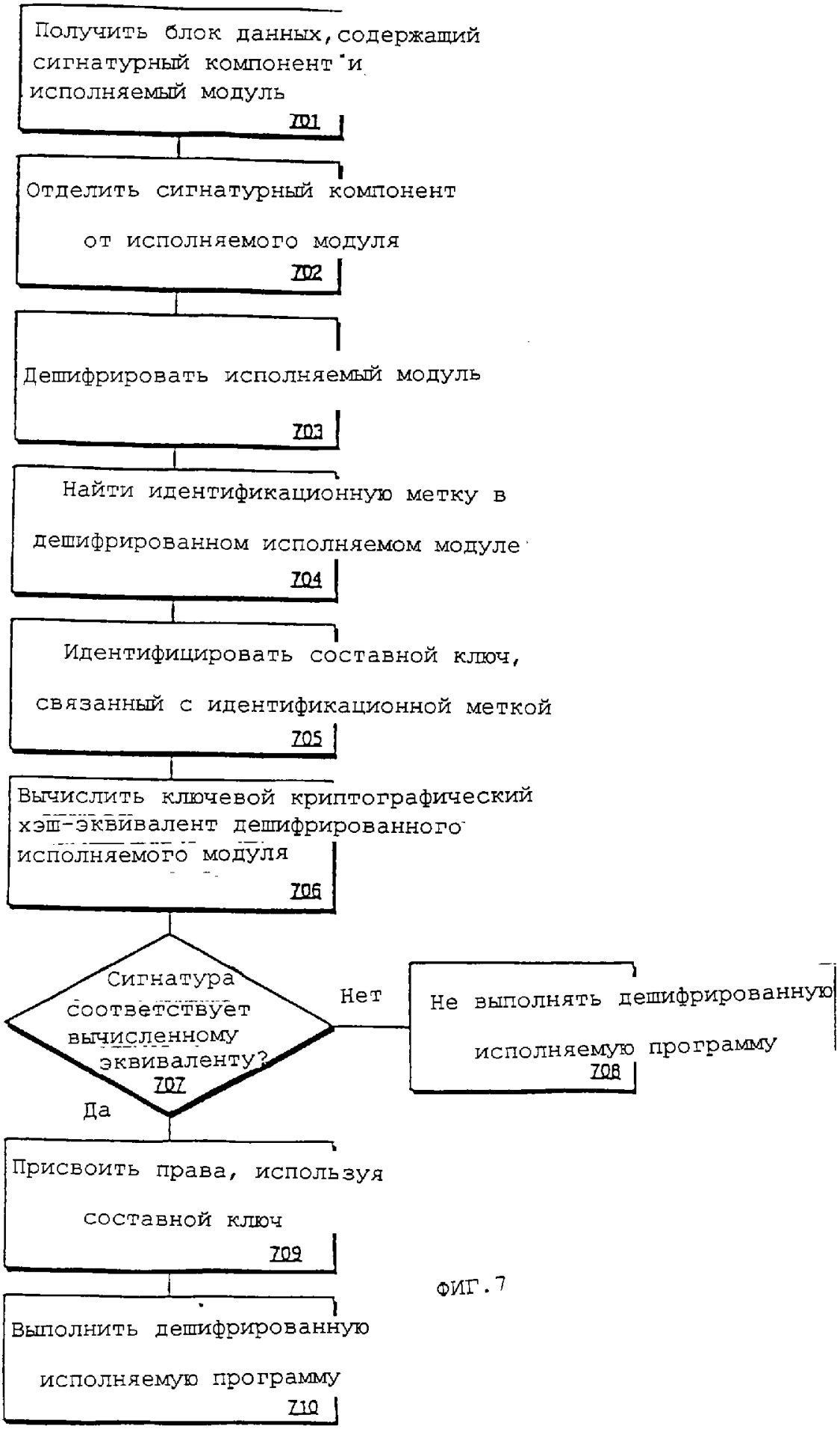
RU 2142674 C1



ФИГ.6

RU 2142674 C1

RU 2142674 C1



ФИГ.7

RU 2142674 C1

RU 2142674 C1