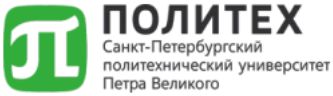

На правах рукописи

Кочетков Евгений Викторович

МЕТОД КОНТРОЛЯ ВЫПОЛНЕНИЯ ПОЛИТИКИ БЕЗОПАСНОСТИ В ОПЕРАЦИОННЫХ СИСТЕМАХ СЕМЕЙСТВА WINDOWS

Специальность 05.13.19 —
Методы и системы защиты информации,
информационная безопасность

АВТОРЕФЕРАТ
диссертации на соискание ученой степени
кандидата технических наук

Орел
2021

Работа выполнена в федеральном государственном казенном военном образовательном учреждении высшего образования «Академия Федеральной службы охраны Российской Федерации».

Научный руководитель:
доктор технических наук, доцент Козачок Александр Васильевич.

Официальные оппоненты:

доктор физико-математических наук Белеванцев Андрей Андреевич, ФГБУН Институт системного программирования им. В.П. Иванникова Российской Академии наук, ведущий научный сотрудник. .

кандидат технических наук, доцент Красов Андрей Владимирович, ФГБОУ ВО «Санкт-Петербургский государственный университет телекоммуникаций им. проф. М.А. Бонч-Бруевича», заведующий кафедрой Защищенных систем связи.

Ведущая организация:
ПАО «Институт электронных управляющих машин им. И.С. Брука»

Защита состоится 05 октября 2021 г. в __ часов
на заседании диссертационного совета У.05.13.19
федерального государственного автономного образовательного учреждения высшего образования «Санкт-Петербургский политехнический университет Петра Великого» (195251, г. Санкт-Петербург, ул. Политехническая, 29, корпус __, аудитория 175).

С диссертацией можно ознакомиться в библиотеке и на сайте www.spbstu.ru федерального государственного автономного образовательного учреждения высшего образования «Санкт-Петербургский политехнический университет Петра Великого».

Автореферат разослан __ июля 2021 года.

Ученый секретарь
диссертационного совета
У.05.13.19,
канд. физ.-мат. наук  Шенец Николай Николаевич

Общая характеристика работы

Актуальность темы. В настоящее время самым распространенным способом удаленного несанкционированного доступа в информационную систему является внедрение и исполнение вредоносного программного обеспечения (ВПО). Основной технической мерой защиты от ВПО являются средства антивирусной защиты. Эксперименты по оценке их эффективности показывают близкую к нулю вероятность пропуска цели. Однако даже при достаточно высокой вероятности обнаружения существует непустое множество вредоносных программ, оставшихся необнаруженными. Достижение принципиальной невозможности выполнения ВПО вероятностными методами обнаружения, такими как сигнатурный или эвристический, невозможно.

Таким образом, успешность реализации несанкционированного доступа к информации во многом зависит от способности политики безопасности, реализованной в операционной системе, противостоять нарушению целостности выполняемых процессов, исполняемых и конфигурационных файлов. В связи с этим более актуальными становятся вопросы формального доказательства выполнения требований политики безопасности в операционных системах.

Степень разработанности темы исследования

Результаты исследований таких ученых, как П.Д. Зегжда, Д.П. Зегжда, А.В. Хорошилов, Н.А. Гайдамакин, Дж. Кэрри, Р. Рид и др. в разработке формальных моделей обеспечения информационной безопасности и П.Н. Девянина — в области формализации и верификации политик безопасности управления доступом, доказывают принципиальную возможность достижения гарантированного уровня обеспечения целостности на основе применения формальных методов.

Диссертационная работа выполнена в соответствии с одним из основных научных направлений Академии Федеральной службы охраны Российской Федерации.

Объектом исследования является подсистема разграничения доступа и обеспечения целостности операционных систем семейства Windows.

Предметом исследования являются модели, методы и алгоритмы контроля выполнения политики безопасности операционных систем.

Целью исследования является защита исполняемых и конфигурационных файлов от угрозы нарушения целостности на основе контроля выполнения многоуровневой модели политики безопасности исполняемых и конфигурационных файлов операционной системы семейства Windows.

Для достижения поставленной цели в работе решаются следующие **задачи**:

1. Анализ эффективности существующих средств антивирусной защиты от ВПО и технологий защиты от реализации уязвимостей программ-

ного обеспечения на уровне ядра в целях обеспечения целостности исполняемых и конфигурационных файлов операционной системы.

2. Моделирование многоуровневой политики безопасности операционных систем семейства Windows, включающей механизмы дискреционного разграничения доступа, мандатных управления доступом и контроля целостности на основе уровней и категорий объектов и субъектов доступа.

3. Разработка формальной модели выполнения процесса в операционной системе, основанной на отображении всего множества системных вызовов в множество базовых операторов доступа и классификация всего множества объектов доступа на виды ресурсов по уровням целостности, что позволит существенно снизить пространство возможных состояний процесса при его выполнении.

4. Разработка формального логического языка задания функциональных ограничений к выполнению процесса в операционной системе с использованием операторов темпоральной логики, что позволит ограничить использование системных вызовов в зависимости от порядка их следования.

5. Разработка метода контролируемого выполнения политики безопасности операционной системы семейства Windows.

6. Разработка научно-технического предложения по практической реализации предложенного метода контроля выполнения политики безопасности операционных систем семейства Windows.

Научная новизна исследования состоит в следующем:

1. Построена формальная модель многоуровневой политики безопасности управления доступом, включающая в себя дискреционное разграничение доступа, мандатные управление доступом и контроль целостности на основе уровней и категорий, а также механизм обеспечения замкнутой программной среды для операционных систем семейства Windows.

2. Разработан метод контроля выполнения политики безопасности операционных систем семейства Windows, учитывающий требования дискреционного и мандатного управления доступом и обладающий инвариантностью к способу реализации системных вызовов за счет описания функциональных ограничений на разработанном формальном языке.

3. Разработаны научно-технические предложения по реализации метода контроля выполнения политики безопасности операционных систем семейства Windows, базирующиеся на применении полносистемного эмулятора для контроля системных вызовов и учета последовательности их совершения.

Теоретическую значимость работы состоит в развитии методов контроля реализации политики безопасности для обеспечения целостности конфигурационных и исполняемых файлов в операционных системах семействах Windows.

Практическая значимость результатов работы заключается в возможности применения предложенного метода контроля выполнения политики безопасности в операционных системах семейства Windows для защиты от ВПО, эксплуатирующего уязвимости в программном обеспечении прикладных приложений и самой ОС для обхода штатной системы разграничения доступа и дополнительных средств защиты информации, с сохранением возможности работы с приложениями из недоверенных источников без нарушения требований замкнутой программной среды за счет обеспечения целостности исполняемых и конфигурационных файлов операционной системы.

Методы исследования. Результаты диссертационной работы получены на основе использования методов системного анализа, математической логики, теории множеств, теории эффективности целенаправленных процессов, методов формальной верификации.

Положения, выносимые на защиту:

1. Многоуровневая модель политики безопасности управления доступом операционных систем семейства Windows, отличающаяся учетом мандатных категорий и уровней доступа и целостности.

2. Метод контроля выполнения политики безопасности в операционных системах семейства Windows, учитывающий требования дискреционного и мандатного управления доступом и инвариантный к способу реализации системных вызовов.

3. Научно-технические предложения по практической реализации метода контроля выполнения политики безопасности в операционных системах семейства Windows, отличающиеся применением полносистемной эмуляции для контроля системных вызовов.

Соответствие специальности научных работников

Полученные научные результаты соответствуют следующим пунктам раздела 2 «Области исследования» паспорта специальности научных работников 05.13.19 «Методы и системы защиты информации, информационная безопасность»: технологии идентификации и аутентификации пользователей и субъектов информационных процессов. Системы разграничения доступа. (п. 11); принципы и решения (технические, математические, организационные и др.) по созданию новых и совершенствованию существующих средств защиты информации и обеспечения информационной безопасности (п. 13).

Достоверность и обоснованность результатов, представленных в диссертации, подтверждается всесторонним анализом предшествующих научных работ в данной области, полученными экспериментальными данными и апробацией результатов в научных публикациях и докладах на научных и научно-практических конференциях. Представленный метод контроля выполнения политики безопасности в операционных системах семей-

ства Windows на основе обработки системных вызовов с использованием полносистемной эмуляции в известных работах не рассматривался.

Формальная постановка задачи

Разработать метод контроля выполнения политики безопасности M в операционных системах семейства Windows, обеспечивающий доказательство следующей теоремы:

Теорема 1. $\neg M(a,A,SecPolicy) \rightarrow \neg\Box a \in A$, где:

$a \in \{S \times O \times actions\}$;

$SecPolicy = DAC \wedge MAC \wedge FuncSpec$;

a – доступ субъекта к объекту;

S – множество субъектов доступа;

O – множество объектов доступа;

$actions$ – множество возможных действий субъекта по отношению к объекту;

$SecPolicy$ – спецификация политики безопасности;

A – множество совершенных доступов субъекта к объекту;

M – предикат проверки относительно заданной модели политики безопасности $SecPolicyc$;

$\Box$ – темпоральный оператор «всегда в будущем»;

DAC – дискреционное разграничение доступа;

MAC – мандатное разграничение доступа;

$FuncSpec$ – функциональное ограничения на работу прикладного программного обеспечения.

Апробация работы. Основные результаты работы опубликованы в статьях [1; 3—10] в рецензируемых изданиях, входящих в перечень рекомендованных ВАК при Минобрнауки РФ. Публикации [1; 3; 11] проиндексированы в Scopus и WoS. В статье [4] осуществляется обоснование возможности применения верификации программ для обнаружения в них вредоносного кода. Разработка формальной модели выполнения процесса в операционной системы рассмотрена в [1]. Формальный логический язык задания функциональных требований к выполнению процессов рассмотрен в [7]. Комплекс алгоритмов, лежащий в основе метода контроля выполнения политики безопасности операционных систем семейства Windows, и подход к практической реализации разработанного решения представлены в статьях [8; 10; 2]. Работы [5; 6; 9] посвящены исследованию особенностей функционирования ВПО и методам их обнаружения, вопросам применения и эффективности современных средств антивирусной защиты. В работе [3] разработана формальная многоуровневая модель политики безопасности управления доступом операционных систем семейства Windows.

По теме диссертации опубликовано 10 научных работ, в том числе 8 в изданиях из перечня ВАК при Минобрнауки РФ, 2 — в изданиях, индексируемых в базах Scopus и WoS. Получено 5 свидетельств ФИПС о регистрации программ для ЭВМ.

Результаты работы обсуждались на следующих конференциях:

1. Межрегиональная научно-практическая конференция «Информационная безопасность и защита персональных данных. Проблемы и пути их решения», БГТУ, Брянск, Россия, 2014 [12].

2. Всероссийская научно-техническая конференция «Безопасные информационные технологии», МГТУ им. Н.Э. Баумана, Москва, 16 ноября, 2016 [13].

3. Международная конференция «Новые горизонты», БГТУ, Брянск, Россия, 2016, [14].

4. International conference on information technology and nanotechnology — 2017 : ITNT 2017, Самара, Россия, 2017, [15].

5. 26-я научно-техническая конференция методы и технические средства обеспечения безопасности информации, Санкт-Петербургский политехнический университет Петра Великого, 2017 [16].

6. Открытая конференция ИСП РАН им. В.П. Иванникова, 2017.

7. Ежегодная межвузовская научно-техническая конференция студентов, аспирантов и молодых специалистов имени Е.В. Арменского, Национальный исследовательский университет «Высшая школа экономики», 2021.

8. Международная научно-техническая конференция «Безопасные информационные технологии», МГТУ имени Н.Э. Баумана, 2021.

Личный вклад. Все представленные в диссертации результаты получены лично автором.

Содержание работы

Во **введении** обосновывается актуальность исследований, проводимых в рамках данной диссертационной работы, ставится цель, определяются задачи, формулируются научная новизна и практическая значимость представляемой работы.

Первая глава посвящена описанию многоуровневой модели политики безопасности управления доступом операционных систем семейства Windows.

В разделе 1.1 рассмотрены особенности существующей политики безопасности управления доступом и мандатного контроля целостности (Mandatory integrity control) операционных систем семейства Windows. Основными ограничениями которого являются:

– по умолчанию большинство файлов и каталогов имеют средний уровень целостности (Middle integrity), ввиду этого, граница доверия между объектами становится размытой;

– назначение низкого уровень целостности процессу (Low integrity) производится явно, однако для большинства браузеров и других приложений этого не делается;

– назначение уровня целостности файлов не учитывает источник их получения, например, съемные носители, сетевые диски, а также файлы, полученные из сети.

В разделе 1.2 представлен подход к верификации моделей методом Model checking с использованием языка темпоральной логики действий Лэмпорта (TLA+). Под верификацией модели методом Model checking понимается автоматическое доказательство соответствия поведения системы заданной для нее спецификации.

В разделе 1.3 представлено описание формальной модели многоуровневой политики безопасности управления доступом операционных систем семейства Windows, включающей в себя:

– определение переменных модели;
– установление структур субъектов и объектов модели;
– реализация дискреционного и многоуровневого разграничения доступа, замкнутой программной среды;
– установление начальных значения переменных модели;
– определение предикатов действий (операторов);
– инварианты корректности и безопасности.

Состояние модели определяется набором значений следующих переменных: множество доступов A, множество объектов доступа O, множество субъектов доступа S, множество пользователей U, множество групп пользователей G.

В модели определены следующие структуры: *Objects*, *Subjects*, *Users*, *Groups*. Пример определения структуры *Users* представлен следующим выражением:

```
Users ≜ [
    uid:       UserIDs,          isAdmin:  BOOLEAN
    cl:        ConfLevs,         ci:       IntLevs,
    confCats:  SUBSET ConfCats,  intCats:  SUBSET IntCats,
    groups:    SUBSET GroupIDs,  wlExec:   SUBSET ObjectIDs
]
```

где *uid* – идентификатор пользователя; *UserIDs* – множество возможных идентификаторов пользователей; *isAdmin* – бинарное значение, определяющее, является ли пользователь администратором; *cl* – максимальный уровень конфиденциальности, доступный пользователю; *il* – максимальный уровень целостности, доступный пользователю; *confCats* – категории конфиденциальности, доступные пользователю; *intCats* – категории целост-

ности, доступные пользователю; *groups* – группы, в которые входит пользователь; *wlExec* – множество идентификаторов исполняемых объектов, разрешенных для запуска новых субъектов данным пользователем.

В настоящей модели замкнутая программная среда реализуется на основе следующих правил:

– создание исполняемых файлов может осуществлять только администратор системы;

– модификация исполняемых файлов запрещена;

– добавление идентификаторов исполняемых объектов в список разрешенных осуществляет только администратор;

– создание субъекта пользователем возможно только при вхождении исполняемого файла во множество разрешенных для запуска объектов данного пользователя.

В соответствии с правилами реализации многоуровневого разграничения доступа для любого состояния системы должен выполняться инвариант *MACSafety*:

$$
\begin{array}{l}
MACSafety \triangleq \\
\quad \wedge \forall o \in O: \\
\qquad \vee \wedge o.type \in Containers \\
\qquad\quad \wedge \forall ch \in SelectAllChilds(o): \\
\qquad\qquad \wedge \vee \text{"ccnr"} \in o.ext\_attr \\
\qquad\qquad\quad \vee \wedge ch.cl \leq o.cl \\
\qquad\qquad\qquad \wedge \vee o.confCats = \{\} \\
\qquad\qquad\qquad\quad \vee ch.confCats \subseteq o.confCats \\
\qquad\qquad \wedge \vee \text{"icnr"} \in o.ext\_attr \\
\qquad\qquad\quad \vee \wedge ch.cl \geq o.cl \\
\qquad\qquad\qquad \wedge \vee o.intCats = \{\} \\
\qquad\qquad\qquad\quad \vee ch.intCats \subseteq o.intCats \\
\qquad \vee o.type \notin Containers \\
\quad \wedge \forall s \in S: \\
\qquad \wedge s.cl \leq SelectUser(s.uid).cl \\
\qquad \wedge s.confCats \subseteq SelectUser(s.uid).confCats \\
\qquad \wedge s.il \leq SelectUser(s.uid).il \\
\qquad \wedge s.intCats \subseteq SelectUser(s.uid).intCats
\end{array}
$$

В инварианте *MACSafety* происходит проверка того, что для каждого объекта, если он является контейнером, должен содержаться флаг *ccnr* или *icnr* в его расширенных атрибутах, или уровни конфиденциальности и целостности вложенных объектов должны быть ниже уровней самого объекта соответственно. При отсутствии флагов *ccnr* или *icnr* категории вложенных объектов контейнера должны быть подмножествами категорий самого контейнера. Выполнение этих правил касается также и субъектов, созданных пользователями.

Модель является верифицированной, если для каждого из возможных состояний системы выполняются все инварианты — глобальные свойства безопасности. На каждом шаге верификации выполнение всех инвариантов подтверждает, что все переменные находятся в допустимых доменах значений, а данное состояние не является запрещенным.

Инвариант представляет собой логическое выражение, включающее в себя темпоральные операторы. Инварианты модели разделены на группы (раздел 1.4):

– по корректности: *TypeInv* (соответствие переменных заданному типу), *NoCyclesInContainers* (отсутствие образования циклов в иерархии объектов), *UserACLCardinality* (для каждого пользователя существует не более одного ACL для одного объекта), *GroupACLCardinality* (для каждой группы существует не более одного ACL для одного объекта).

– жизнеспособности: *OneAdminExists* (доступен режим администрирования), *MACSafety* (отсутствие нарушения мандатных правил разграничения доступа для иерархии и вложенности категорий объектов).

– безопасности: *MACAccessSafety* (отсутствие перехода в запрещенное состояние в связи с нарушением мандатных правил разграничения доступа), *DACAccessSafety* (отсутствие перехода в запрещенное состояние в связи с нарушением дискреционных правил разграничения доступа).

Вторая глава посвящена описанию метода контроля выполнения политики безопасности операционных систем семейства Windows. В разделе 2.1 описано применение метода формальной верификации «Model checking» для решения задачи контроля выполнения политики безопасности операционной системы. Политика безопасности операционной системы при этом описывается в виде спецификации. На основе спецификаций и модели выполнения субъектов операционной системы проверяется насколько согласуется реальное выполнение субъектов с заданной моделью политики безопасности. Так как происходит именно математическая верификация, решение о согласованности является корректным.

В разделе 2.2 представлена формальная модель выполнения процесса в операционной системе. В основе ее построения лежит принцип разделения всех взаимодействующих элементов операционной системы на субъекты и объекты. К субъектам относятся процессы, совершающие действия по отношению к объектам. Объектами являются ресурсы операционной системы и процессы, в отношении которых другими субъектами совершаются действия:

– «Процесс» (P);
– «Оперативная память» (M);
– «Внешняя память» (E);
– «Периферийные устройства» (D);
– «Сетевая подсистема» (N).

Определение 1. *Действие процесса в операционной системе (ДП) – операция a_o, совершаемая субъектом $p_x^{k_p}$ по отношению к объекту $o_y^{k_o}$.*

Каждое ДП описывается тройкой элементов:

$$\omega_i =< p_x^{k_p},\ a_o,\ o_y^{k_o} >, \tag{1}$$

где $p_x^{k_p}$ – субъект категории k_p с идентификатором x; $a_o \in A_o$ – некоторое действие субъекта по отношению к объекту; $o_y^{k_o}$ – некоторый объект категории k_o с идентификатором y.

Операторное представление выражения 1 имеет следующий вид:

$$\omega_i = {p_j}^{k_p} \xrightarrow{a_o} {o_y}^{k_o}.$$

Важно отметить, что прикладная интерпретация каждой операции зависит от объекта, к которому она применяется, и его категории:

- *Create* (c) – создание нового объекта;
- *Delete* (d) – удаление объекта;
- *Read* (r) – чтение информации из объекта;
- *Write* (w) – запись информации в объект.

Представленная модель применительно к задаче анализа позволяет построить модель выполнения процесса в операционной системе путем наблюдения за его выполнением или исследования бинарного исполняемого файла. Во втором случае после обработки будет получена структура, позволяющая произвести процедуру формальной верификации. Применительно к задаче синтеза она позволяет построить модель выполнения процесса в операционной системе с заранее заданными свойствами безопасности.

Модель безопасного исполнения программного кода строится на основе аксиом безопасного исполнения программного кода (таблица 1).

Таблица 1 — Базис аксиом «Безопасного исполнения программного кода»

Формула	**Интерпретация**
$p^* \xrightarrow{c,r,w,d} m^3$	Выполнение операций над памятью только в своем адресном пространстве
$p^* \xrightarrow{c,r,w,d} e^5$	Выполнение операций над файловыми объектами только в своем домашнем каталоге
$p^* \xrightarrow{r} e^2$	Чтение системных файловых объектов и конфигурации
$p^* \xrightarrow{r,w} e^3$	Чтение и запись параметров пользовательского окружения
$p_i^* \xrightarrow{d} p_j^*$	Процесс может завершать только свою работу

Представленный базис можно рассматривать как аксиоматический, так как его выполнение обеспечивает безопасность исполнения программного кода в аспекте защиты от вредоносного кода.

В разделе 2.3 представлен формальный логический язык задания функциональных ограничений на выполнение процессов в операционной системе. Для его описания, в соответствии с определением логики первого порядка, заданы следующие подмножества:

$$FormSpec = Func \cup Pred \cup Var \cup Log \cup Aux.$$

При этом множество функциональных символов *Func* включает в себя возможные ДП в операционной системе. Множество предикатных символов включает в себя базовые предикаты темпоральной логики CTL* и предикат проверки безопасности *isSecure*:

$$Pred = \{isSecure, AX, AF, AG, AU, AR, EX, EF, EG, EU, ER, EC\}.$$

Множество символов предметных переменных включает в себя следующие элементы: $Var = p,m,n,e,d,cat$, где O – объекты модели выполнения процесса в операционной системе, *cat* – категория объекта.

Множество логических символов включает в себя следующие элементы: $Log = \{\neg, \wedge, \vee, \rightarrow, \exists, \forall\}$. Множество вспомогательных символов включает в себя следующие элементы: $Aux = \{,()\}$.

В разделе 2.4 представлена функциональная схема метода контроля выполнения политики безопасности (рисунок 1). В его основе лежит последовательное выполнение ряда этапов, реализованных в виде отдельных блоков.

Блок 1 «Предварительная обработка». Производится построение модели исполняемого файла, представляющей собой структуру Крипке, в которой под состоянием понимается ДП, а связи между состояниям означают потенциальную возможность программы совершить следующее ДП из данного состояния.

Блок 2 «Формирование комплекса функциональных ограничений». На основе априорной информации о функциональном назначении программы с учетом аксиом «Безопасного исполнения программного кода» формируется комплекс функциональных ограничений, выполнение которых, с одной стороны, необходимо для безопасного использования программы, с другой стороны, минимально ограничивает ее функциональные возможности.

Блок 3 «Задание спецификации безопасного исполнения». Производится преобразование комплекса функциональных ограничений, выраженное на формальном языке *FormSpec*, в формулы темпоральной логики.

Блок 4 «Формальная верификация (статический этап)». Выполнение процедуры формальной верификации методом «Model checking». В качестве исходных данных выступают модель исполняемой программы (струк-

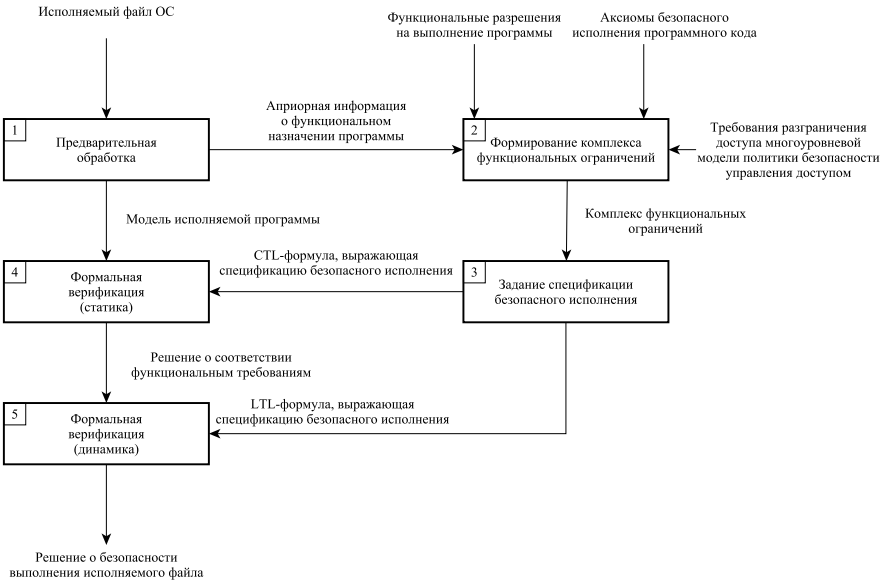

Рисунок 1 — Функциональная схема метода контроля выполнения политики безопасности

тура Крипке) и спецификация безопасного исполнения, выраженная в виде CTL^*-формулы. Результатом работы данного блока является решение о выполнении заданной моделью требуемых свойств безопасного использования на статическом этапе проверки.

Блок 5 «Формальная верификация (динамический этап)». Запуск исследуемой программы под контролем монитора исполнения. Каждый системный вызов, совершенный программой, формирует трассу исполнения. Выполнение процедуры формальной верификации методом «Model checking» выполняется после каждого шага. Исходными данными являются трасса исполнения и LTL-формула, выражающая спецификацию безопасного исполнения. Если трасса исполнения программы выполняет заданные свойства, то монитор исполнения разрешает выполнить ДП.

Третья глава посвящена описанию научно-технических предложений по практической реализации метода контроля выполнения политики безопасности операционных систем семейства Windows.

В разделе 3.1 представлен комплекс алгоритмов, составляющих основу метода контроля выполнения политики безопасности в операционных системах семейства Windows.

Алгоритм работы системы контроля выполнения политики безопасности операционных систем семейства Windows представлен на рисунке 2.

```
1  Function VerifyedExecCodeSystem(B, FR, L, CTT)
       Result: Decision
2      // Проверка ограничений
3      if checkMinCriterias(B) ≠ true then
4          return false
5      // Построение модели
6      M ← buildModel(B,CTT,L)
7      // Статический этап
8      φ ← buildSpec(AX,FR)
9      if Verify(M,φ) ≠ true then
10         return false
11     // Динамический этап
12     if SecurityMonitor(B,M,φ) ≠ true then
13         return false
14 return true
```

Рисунок 2 — Алгоритм контроля выполнения политики безопасности операционных систем семейства Windows.

Исходными данными для алгоритма *VerifyedExecCodeSystem* являются следующие параметры: B – исследуемый бинарный файл; AX – аксиомы безопасного выполнения программного кода; FR – функциональные требования, заданные на формально логическом языке; L – функция оценки; CTT («Call Translate Table») – таблица соответствия системных (API) вызовов операционной системы и действий процесса. Параметр CTT может принимать два значения: $SCTT$ для системных вызовов и $APICTT$ для вызовов с использованием API-интерфейса.

Алгоритм мониторинга исполнения *SecurityAutomataMonitor* осуществляет подготовку и запуск исследуемого исполняемого файла B с отслеживанием его выполнения в рамках конфигурации безопасного исполнения *safeConfig* (рисунок 3). Каждый вызов системной функции, производимый исполняемым файлом, изменяет состояние автомата безопасности. В случае перехода автомата в состояние, отсутствующее в модели, но не противоречащее спецификации, исполнение программы продолжается, а данная трасса добавляется во множество трасс исполнения T, полученных на этапе построения модели M. В случае перехода автомата в запрещенное состояние в рамках спецификации выполнение исполняемого файла приостанавливается. Текущая трасса исполнения сохраняется для дальнейшего анализа. Таким образом, предлагаемый подход предпо-

лагает возможность исключения выполнения потенциально небезопасных действий.

```
1  Function SecurityMonitor(B,SA,attributes)
2      ω_cur ← ϵ
3      hProcess ← startProcess(B,Attributes)
4      while command ≠ exit and hProcess.isALive do
5          sysCall ← getSysCall()
6          lockProcess(hProcess)
7          ω_new ← TranslateSysCall(sysCall,SCTT)
8          if (ω_cur,ω_new) ∈ SA.goodR then
9              ω_cur ← ω_new
10             unlockProcess(hProcess)
11         else
12             suspendProcessAndAlert(hProcess)
13             return false
14     return true
```

Рисунок 3 — Алгоритм мониторинга исполнения

В разделе 3.2 рассмотрены научно-технические предложения по практической реализации представленного метода контроля выполнения политики безопасности.

В качестве полносистемного эмулятора предполагается использование «QEMU», ключевой особенностью которого является поддержка системы плагинов, разработанной «Институтом системного программирования им. В.П. Иванникова РАН».

Плагин контролируемого выполнения обрабатывает каждый системный вызов, полученный от контролируемого процесса. В случае выхода контролируемого процесса за рамки спецификации безопасности работа эмулятора приостанавливается, при этом выполнение процесса завершается. База спецификаций хранится в виде набора файлов в файловой системе базовой операционной системы и доступна для программы управления системы контролируемого выполнения процессов (рисунок 4). Для каждого запускаемого в эмуляторе приложения происходит загрузка соответствующей спецификации в модуль контролируемого исполнения, интегрированный в виде плагина в экземпляр «QEMU», через протокол QMP.

В **заключении** приведены основные результаты работы, которые заключаются в следующем:

1. Проанализирована эффективность существующих средств антивирусной защиты от ВПО и технологий защиты от реализации уязвимостей программного обеспечения на уровне ядра в целях обеспечения целостности исполняемых и конфигурационных файлов операционной системы.

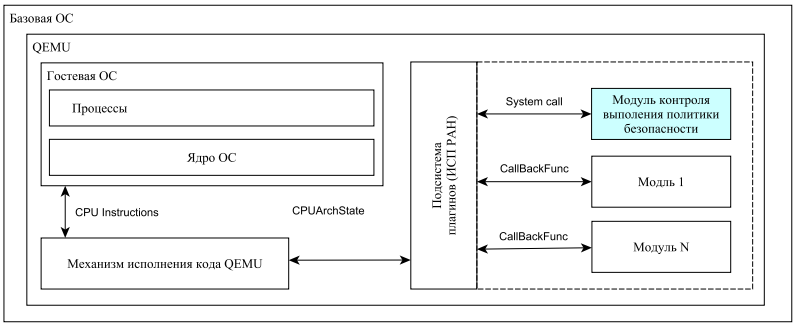

Рисунок 4 — Реализация метода контроля выполнения политики безопасности с использованием полносистемного эмулятора «QEMU»

2. Построена модель многоуровневой политики безопасности операционных систем семейства Windows, включающей механизмы дискреционного разграничения доступа, мандатных управления доступом и контроля целостности на основе уровней и категорий объектов и субъектов доступа.

3. Разработана формальная модель выполнения процесса в операционной системе, основанная на отображении всего множества системных вызов во множество базовых операторов доступа и классификация всего множества объектов доступа на виды ресурсов по уровням целостности, что позволило существенно снизить пространство возможных состояний процесса при его выполнении.

4. Разработан формальный логический язык задания функциональных ограничений к выполнению процесса в операционной системе с использованием операторов темпоральной логики, что позволило ограничить использование системных вызовов в зависимости от порядка их следования.

5. Разработан метод контроля выполнения политики безопасности операционной системы семейства Windows.

6. Разработано научно-техническое предложение по практической реализации предложенного метода контроля выполнения политики безопасности операционных систем семейства Windows.

Направлением дальнейших исследований является разработка метода виртуальной изоляции выполняемых процессов в операционной системе с использованием полносистемной эмуляции аппаратного обеспечения.

Публикации автора по теме диссертации

В изданиях, индексируемых Scopus и Web of Science

1. *Кочетков Е. В.*, *Козачок А. В.* Формальная модель функционирования процесса в операционной системе // Труды СПИИРАН. — 2017. — Т. 51, № 2. — С. 78—96.

2. *Kochetkov E. V.*, *Kozachok A. V.* Prototype of a Verified Program Code Execution System // Programming and Computer Software. — 2018. — Т. 44, № 3. — С. 190—199.

В изданиях из перечня ВАК Минобрнауки РФ

3. *Кочетков Е. В.*, *Козачок А. В.* Многоуровневая модель политики безопасности управления доступом операционных систем семейства windows // Вопросы кибербезопасности. — 2021. — Т. 41, № 1. — С. 41—54.
4. *Кочетков Е. В.*, *Козачок А. В.* Обоснование возможности применения верификации программ для обнаружения вредоносного кода // Вопросы кибербезопасности. — 2016. — Т. 16, № 3. — С. 25—32.
5. *Кочетков Е. В.*, *Козачок А. В.* Обнаружение вредоносного программного обеспечения на основе выявления аномальной сетевой активности // Информационные системы и технологии. — 2016. — Т. 97, № 5. — С. 107—114.
6. *Кочетков Е. В.*, *Козачок А. В.*, *Татаринов А. М.* Обоснование возможности построения эвристического механизма распознавания вредоносных программ на основе статического анализа исполняемых файлов // Вестник компьютерных и информационных технологий. — 2017. — Т. 153, № 3. — С. 50—56.
7. First Order Logic For Program Code Functional Requirements Description / E. V. Kochetkov [и др.] // Вопросы кибербезопасности. — 2017. — Т. 21, № 3. — С. 2—7.
8. *Кочетков Е. В.*, *Козачок А. В.* Комплекс алгоритмов функционирования системы безопасного исполнения программного кода // Труды ИСП РАН. — 2017. — Т. 29, № 3. — С. 17—30.
9. *Кочетков Е. В.* Анализ эффективности современных подходов к защите от вредоносного кода // Информационные системы и технологии. — 2017. — Т. 102, № 4. — С. 104—115.
10. *Кочетков Е. В.*, *Козачок А. В.* Подход к реализации системы верифицированного исполнения программного кода // Труды ИСП РАН. — 2017. — Т. 29, № 6. — С. 7—24.

В сборниках трудов конференций

11. *Kochetkov E. V.*, *Kozachok A. V.*, *Bochkov M. V.* Heuristic malware detection mechanism based on executable files static analysis //. — Samara, Russia : CEUR Workshop Proceedings, 2017. — С. 132—139.
12. *Кочетков Е. В.*, *Козачок А. В.* Методика обнаружения вредоносного ПО на основе выявления подозрительной сетевой активности // Информационная безопасность и защита персональных данных. Проблемы и пути их решения. — Брянск : БГТУ, 2014. — С. 51—54.
13. *Кочетков Е. В.*, *Козачок А. В.* Подход к построению формальной модели функционирования процесса в операционной системе // Безопасные информационные технологии: Сб. тр. науч.-техн. конф. — Москва : МГТУ им. Н. Э. Баумана, 2016. — С. 158—161.

14. *Кочетков Е. В.*, *Козачок А. В.* Подход к применению верификации программ для обнаружения вредоносного кода // Новые горизонты: материалы международной конференции-конкурса. — Брянск : БГТУ, 2016. — С. 152—156.
15. *Kochetkov E. V.*, *Kozachok A. V.*, *Bochkov M. V.* Novel approach to constructing static heuristic malware detection mechanism // Information Technology and Nanotechnology — 2017. — Samara : Samara State University, 2017. — P. 882–887.
16. *Кочетков Е. В.*, *Козачок А. В.* Формальная модель функционирования процесса в операционной системе // Методы и технические средства обеспечения безопасности информации: Материалы 26-й научно-технической конференции 26 – 29 июля 2017 года. — Санкт-Петербург : Изд-во Политехн. ун-таи, 2017. — С. 10—12.

Свидетельства ФИПС о регистрации программ для ЭВМ

17. Программное средство построения модели исполняемого файла на языке Promela : Свидетельство о государственной регистрации программы для ЭВМ № 2016616577 / Е. В. Кочетков, А. В. Козачок. — Заявл. 06.2016.
18. Программное средство верификации моделей программ на языке Promela : Свидетельство о государственной регистрации программы для ЭВМ № 2016616578 / Е. В. Кочетков, А. В. Козачок. — Заявл. 06.2016.
19. Программное средство задания спецификаций программ на основе функциональных требований к ним : Свидетельство о государственной регистрации программы для ЭВМ № 2017611368 / Е. В. Кочетков, А. В. Козачок. — Заявл. 06.2016.
20. Программное средство генерации автомата безопасности для программы на основе спецификации функциональных требований : Свидетельство о государственной регистрации программы для ЭВМ № 2017611274 / Е. В. Кочетков, А. В. Козачок. — Заявл. 06.2016.
21. Система безопасного исполнения программного кода с использованием методов формальной верификации и автомата безопасности : Свидетельство о государственной регистрации программы для ЭВМ № 2017616014 / Е. В. Кочетков, А. В. Козачок. — Заявл. 05.2017.

Кочетков Евгений Викторович

Автореф. дис. на соискание ученой степени канд. тех. наук

Подписано в печать ___.___._____. Заказ № _____
Формат 60×90/16. Усл. печ. л. 1. Тираж 100 экз.
Типография _____