На правах рукописи

Краснопевцев Антон Андреевич

ЗАЩИТА ОТ НЕСАНКЦИОНИРОВАННОГО КОПИРОВАНИЯ ПРИЛОЖЕНИЙ, КОМПИЛИРУЕМЫХ В ПРОМЕЖУТОЧНОЕ ПРЕДСТАВЛЕНИЕ

Специальность: 05.13.19 – методы и системы защиты информации, информационная безопасность

АВТОРЕФЕРАТ
диссертации на соискание ученой степени
кандидата технических наук

Автор: _____

Москва – 2011

Работа выполнена в Национальном исследовательском ядерном университете «МИФИ» (НИЯУ МИФИ)

Научный руководитель: кандидат технических наук, доцент
Петрова Тамара Васильевна

Официальные оппоненты: Доктор технических наук
Тарасов Александр Алексеевич

Кандидат технических наук
Смирнов Павел Владимирович

Ведущая организация: Московский Государственный Технический Университет имени Н.Э. Баумана

Защита состоится «24» июня 2011 г. в 15 часов 00 минут на заседании диссертационного совета ДМ 212.130.08 при Национальном исследовательском ядерном университете «МИФИ»: 115409, г. Москва, Каширское ш., д.31. Тел. для справок: +7 (495) 323-95-26, 324-73-34.

С диссертацией можно ознакомиться в библиотеке Национального исследовательского ядерного университета «МИФИ».

Отзывы в двух экземплярах, заверенные печатью, просьба направлять по адресу: 115409, г. Москва, Каширское ш., д.31, диссертационные советы НИЯУ МИФИ, тел.: +7 (495) 323-95-26.

Автореферат разослан «___» _____ 2011 г.

Ученый секретарь
диссертационного совета Горбатов В.С.

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность темы. Большинство различных процессов, связанных с жизнью общества, в той или иной степени управляются с привлечением различного программного обеспечения. Однако, совместно с ростом количества создаваемого программного обеспечения увеличивается и число незаконных действий, именуемых «компьютерным пиратством», или иными словами – несанкционированным копированием приложений. Несанкционированное копирование влечет за собой нарушение авторского права (а в некоторых случаях и патентного, а также смежных прав) разработчика программного обеспечения.

Проблема несанкционированного копирования приложений и нарушения авторского права приобретает все большую актуальность в связи с ростом популярности приложений, компилируемых в промежуточное представление. Под приложениями, компилируемыми в промежуточное представление, в работе понимается вид программного обеспечения, которое компилируется не в машинный код, а в набор команд виртуальной машины (промежуточное представление). Примерами систем, с помощью которых создаются такие приложения, являются .NET, Java и др. Основной отличительной особенностью приложений, созданных с использованием приведенных средств разработки, является то, что полученные приложения скомпилированы не в машинный код, а в промежуточное представление. По данным аналитической компании «TIOBE» более 40% разрабатываемого программного обеспечения создается с использованием именно таких средств разработки приложений. Статистика использования различных языков программирования приведена на рис. 1.

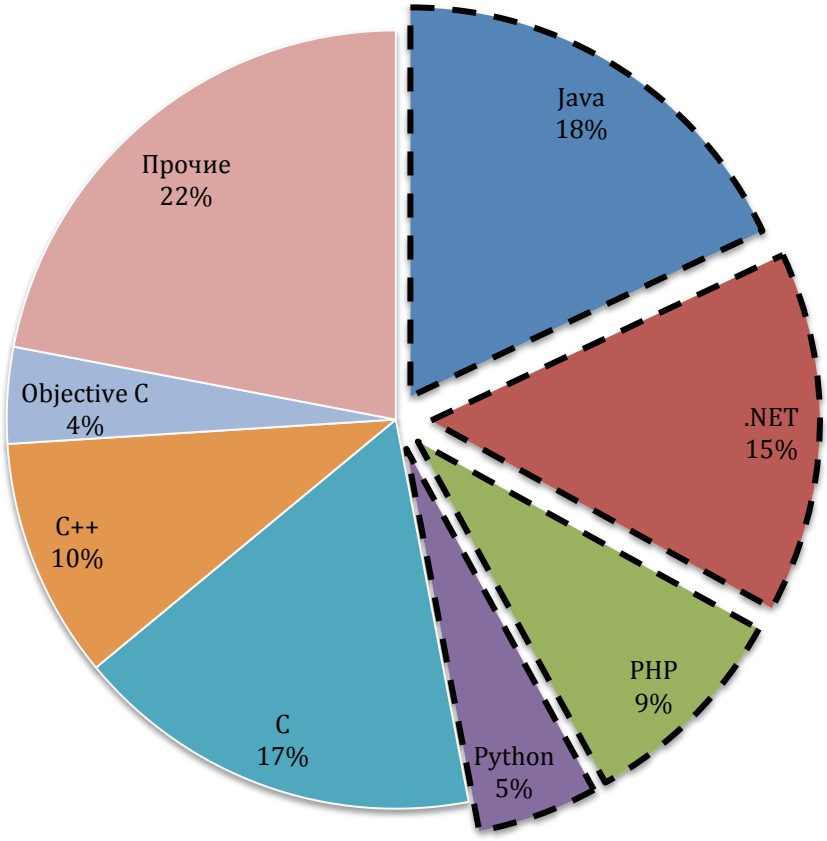

Рисунок 1- Статистика использования различных языков программирования

Пунктирной линией на рис. 1 выделены блоки, интерпретирующие языки программирования, дающие возможность создания приложений, компилируемых в промежуточное представление.

Рост популярности средств разработки приложений, компилируемых в промежу-

точное представление, связан с тем, что такие средства разработки содержат большое количество различных шаблонов кода. В результате, разработчик может использовать уже готовые протестированные реализации алгоритмов. Одним из главных достоинств приложений, компилируемых в промежуточное представление, является наличие собственного менеджера памяти. В результате, при выполнении приложений не возникает ошибок, связанных с переполнением буфера, обращений к некорректному указателю и т.д. В связи с вышеприведенными преимуществами рассматриваемых средств разработки приложений, время, затрачиваемое на разработку приложения, сокращается. Как следствие, снижаются издержки, требуемые для создания, тестирования и реализации приложения.

Однако, существенным недостатком приложений, компилируемых в промежуточное представление, является то, что внутренняя архитектура таких приложений хорошо документирована и полностью описывает поведение приложения в любой момент его выполнения, что реализует собой уязвимость данного класса приложений. В результате, процесс анализа такого приложения и компрометация его системы защиты от несанкционированного копирования является достаточно простой задачей.

Исследования в области защиты приложений от несанкционированного копирования проводились как российскими, так и зарубежными учеными, среди которых следует отметить:

– Christian Collberg, который предложил математическое описание и оценку реализации различных атак на программные методы защиты, реализующие запутывающие преобразования над кодом приложения.

– Чернов А. В., также проводивший исследование по программным методам защиты приложений, предложивший классификацию различных способов программного запутывания исполняемого кода приложения;

– Boaz Barak, автор статьи, посвященной анализу программных методов защиты приложений, с обоснованием возможности автоматической компрометации для любой системы защиты, построенной на программных методах;

– Алексей Солодовников, который является автором и разработчиком программной системы защиты приложений от несанкционированного копирования ASPack.

Разработкой средств защиты от несанкционированного копирования занимались такие фирмы, как:

– компания «Актив», которая с 1994 года является лидером российского рынка защиты программного обеспечения (продукты Guardant Code, Time, Sign и пр.);

– компания «Aladdin», основанная в 1993 году, в сферу деятельности которой кроме создания систем защиты приложений входит разработка аппаратной составляющей для систем PKI (HASP);

– компания «SafeNet», являющаяся глобальной международной компанией с достаточно широкой сферой интересов, среди которых не малую долю занимает разработка программно-аппаратных систем защиты приложений от несанкционированного копирования (Sentinel);

– компания «RocKey», которая на данный момент является новатором рынка защиты приложений, компилируемых в промежуточное представление, предложившая концепцию защиты приложений посредством выгрузки приложения во внешний аппаратный модуль (RocKey).

Анализ продуктов, представленных компаниями, выявил наличие у них суще-

ственных недостатков. Среди основных недостатков следует отметить низкую сложность компрометации для программных методов защиты (в общем случае полиномиальной сложности) и сильное снижение скорости выполнения защищенного приложения для программно-аппаратных способов защиты (до 10-15 раз).

Автором работы предлагается новая методика защиты от несанкционированного копирования приложений, компилируемых в промежуточное представление, лишенная выявленных, в ходе проведенного анализа, недостатков.

Объект исследования. Приложения, компилируемые в промежуточное представление.

Предмет исследования. Уязвимости приложений, компилируемых в промежуточное представление, и методы защиты.

Цель диссертационной работы. Повышение сложности несанкционированного копирования приложений, компилируемых в промежуточное представление.

Научная задача заключается в анализе уязвимостей приложений, компилируемых в промежуточное представление, и синтезе системы защиты от несанкционированного копирования таких приложений.

В рамках решения научной задачи необходимо:

– провести анализ существующих подходов и методов защиты от несанкционированного копирования приложений, компилируемых в промежуточное представление;

– построить модель нарушителя, провести анализ возможных способов атаки на систему защиты и используемого нарушителем инструментария;

– разработать методику защиты приложений от несанкционированного копирования;

– оценить сложность компрометации предложенной системы защиты;

– разработать архитектуру систем защиты приложений с использованием предложенной в работе методики;

– разработать программно-аппаратный комплекс, реализующий систему защиты .NET приложений от копирования.

Методы исследований. Теория автоматов, методы оценки сложности вычислений, теория графов.

Научная новизна работы состоит в следующем:

– предложена и исследована математическая модель процесса защиты приложения от несанкционированного копирования, использующая разделение графа потока управления приложения. Построенная модель позволяет проводить оценку сложности компрометации различных систем защиты приложений от несанкционированного копирования;

– построена математическая модель функционирования внешнего аппаратного модуля системы защиты от несанкционированного копирования, описанная в терминах теории конечных автоматов, которая позволяет повысить сложность компрометации систем защиты приложений от несанкционированного копирования.

– предложена методика защиты приложений от несанкционированного копирования с заданным уровнем сложности компрометации, позволяющая соотносить скорость выполнения защищенного приложения и уровень сложности его компрометации.

Практическая значимость результатов заключается в следующем:

– сформулированы рекомендации по применению практических и теоретических результатов работы для использования при защите приложений от несанкционированного копирования;

– предложен метод оценки сложности компрометации систем защиты приложений, использующих внешний аппаратный модуль;

– разработан программно-аппаратный комплекс, реализующий защиту от копирования .NET приложений с использованием методики, предложенной в диссертационной работе.

Результаты работы представляют практическую ценность для реализации систем защиты от несанкционированного копирования приложений.

Внедрение результатов исследований. Метод защиты переменных приложения был использован при разработке системы защиты приложений от несанкционированного копирования, предлагаемой компанией «Актив».

Разработанная методика защиты была использована для построения системы защиты Flash-приложений от несанкционированного копирования. Разработанная система защиты применяется для противодействия несанкционированному копированию учебных курсов, выпускаемых компанией «Интерактивное Детство».

Теоретические и прикладные результаты, полученные в ходе выполнения диссертационной работы, использованы в учебном курсе «Языки программирования» кафедры «Криптология и дискретная математика» НИЯУ МИФИ для создания лабораторной работы.

Предложенная методика защиты приложений от несанкционированного копирования была реализована для защиты .NET приложения, используемого в Центре вирусных исследований и аналитики «Eset», для исследования графа потока управления вредоносного программного обеспечения.

Предложенный способ защиты приложений, посредством разделения конечного автомата, а также принцип работы внешнего аппаратного модуля, были реализованы для защиты программного комплекса, используемого в компании ООО «Связьмонтаж-комплектация», для создания смет.

Публикации и апробация работы. Результаты диссертации изложены в 8 публикациях, 4 из которых опубликованы в рецензируемых журналах ВАК РФ. Результаты работы докладывались на конференциях и семинарах различного уровня:

– Инфофорум-2009 – 29-30.01.2009г, г. Москва, получены диплом и медаль в номинации «Лучшая научная работа года»;

– 2-я Международная телекоммуникационная конференция студентов и молодых ученых «Молодежь и наука», 12.10. 2009, НИЯУ МИФИ, представленная на конференции работа получила грант по программе У.М.Н.И.К.;

– НТТМ-2009, 24-27.06.2009, Москва, ВВЦ, получена медаль за «Успехи в научно-техническом творчестве»;

– Научная сессия МИФИ 2011, 01-05.02.2011, НИЯУ МИФИ.

Основные положения, выносимые на защиту:

– математическая модель процесса защиты от несанкционированного копирования приложений, компилируемых в промежуточное представление;

– математическая модель функционирования внешнего аппаратного модуля системы защиты от несанкционированного копирования;

– методика защиты приложений от несанкционированного копирования;

– архитектура и программно-аппаратный комплекс, реализующий защиту приложений, с использованием предложенной в работе методики.

Структура работы. Работа состоит из введения, четырех глав, заключения, списка литературы, включающего 105 наименований, и одного приложения. Текст диссертации изложен на 150 страницах, включая 15 рисунков и 2 таблицы.

СОДЕРЖАНИЕ РАБОТЫ

Во **введении** обосновывается актуальность темы диссертации, выделяются и формулируются цель и задачи исследования, описывается структурно-логическая схема диссертационной работы.

В **первой главе** представлены результаты анализа особенностей архитектуры приложений, компилируемых в промежуточное представление. Рассмотрены проблемы, с которыми сталкиваются разработчики систем защиты от несанкционированного копирования для приложений, компилируемых в промежуточное представление. Основным недостатком таких приложений является простота анализа их кода. Большинство существующих систем защиты приложений от несанкционированного копирования направлены на усложнение процесса анализа кода приложения, однако, для приложений, компилируемых в промежуточное представление, такой подход не дает необходимого результата.

Способы защиты приложений от несанкционированного копирования делятся на два основных класса:

– программные способы защиты приложений;
– программно-аппаратные способы защиты приложений.

Примерами программных способов защиты приложений являются следующие:

– осуществляющие запутывающие преобразования графа потока управления защищаемого приложения (Dotfuscator);
– осуществляющие выгрузку элементов приложения в библиотеки, скомпилированные в машинный код (.NET Reactor);
– реализующие шифрование кода приложения (Code Veil).

Основным недостатком программных способов защиты приложений является низкая сложность их компрометации. Более того, для большого количества программных методов защиты существуют решения, осуществляющие несанкционированное копирование защищенного приложения в автоматическом режиме. Менее существенным недостатком является сильное увеличение объема исполняемого файла после защиты, а также незначительное снижение скорости выполнения защищенного приложения.

Примерами программно-аппаратных способов защиты приложений от несанкционированного копирования являются следующие:

– программно-аппаратные комплексы, основанные на идентификации внешнего аппаратного модуля (старые версии Sentinel, Guardant, Aladdin);
– программно-аппаратные комплексы, осуществляющие шифрование кода защищаемого приложения (Aladdin, Guardant);
– программно-аппаратные комплексы, осуществляющие выгрузку кода приложения во внешний аппаратный модуль (RocKey 7 .NET, Java Card).

Программно-аппаратные способы защиты приложений обладают более высокой сложностью компрометации относительно программных методов защиты приложе-

ний. Такая сложность компрометации связана с тем, что для анализа защищенного приложения необходимо исследовать внешний аппаратный модуль посредством методов «черного ящика» или посредством физического анализа. Основным недостатком программно-аппаратных способов защиты приложений является значительное (в 10-15 раз) снижение скорости выполнения защищенного приложения. Кроме того, существуют программно-аппаратные методы защиты приложений, для которых возможно построить метод, реализующий автоматическую компрометацию.

Общим недостатком для обоих приведенных классов систем защиты приложений является отсутствие обоснования стойкости к различного рода атакам и математического обоснования сложности компрометации.

Результаты анализа систем защиты от несанкционированного копирования приложений, компилируемых в промежуточное представление, приведены в табл. 1.

Таблица 1 – Сравнительная характеристика методов защиты приложений

Методы защиты приложений	Характеристика способа / Способ защиты	Оценочная степень замедления приложения (разы)	Возможность автоматической компрометации	Возможность автоматической защиты	Задание уровня сложности компрометации
Программные методы	Запутывание кода приложений	**1,2-5**	**+**	**+**	**–**
	Выгрузка кода в «неуправляемые» библиотеки	**4-6**	**+**	**+**	**–**
	Шифрование кода приложения	**4-6**	**+**	**+**	**+**
Программно-аппаратные методы	Идентификация внешнего аппаратного модуля	**1,2**	**+**	**+**	**–**
	Шифрование кода приложения	**6-10**	**+**	**+**	**+**
	Выгрузка кода приложения во внешний аппаратный модуль	**10-15**	**–**	**–**	**–**
	Авторский метод	**0,5-3**	**–**	**+**	**+**

В табл. 1 знак «+» или «–» показывает наличие или отсутствие указанной характеристики. Под оценочной степенью замедления приложения понимается соотношение скорости выполнения приложения до и после его защиты.

По результатам проведенного анализа было выявлено, что все существующие

методы защиты приложений от несанкционированного копирования обладают существенными недостатками. Предлагаемая в работе методика защиты от несанкционированного копирования приложений, компилируемых в промежуточное представление, лишена выявленных недостатков.

Во **второй главе** представлены результаты исследований проблем построения систем защиты приложений, компилируемых в промежуточное представление, от несанкционированного копирования с использованием теории автоматов, теории графов и теории сложности вычислений. Построена модель нарушителя, рассмотрен основной инструментарий, который может быть использован с целью снижения сложности компрометации предложенной системы защиты приложений, а также приведены типовые атаки, которые могут применяться для снижения сложности компрометации предложенной системы защиты приложений от несанкционированного копирования.

Основной класс нарушителя – нарушитель с правами легитимного пользователя, проводящий анализ и исследование защищенного приложения с целью его дальнейшей компрометации. Нарушитель может располагать одной или несколькими лицензионными копиями защищенного программного обеспечения. Допускается возможность запуска защищенного приложения и исследования системы защиты на протяжении всего процесса выполнения защищенного приложения.

Среди инструментов, которые использует аналитик для исследования кода защищенного приложения, в рамках сформулированной модели нарушителя, рассматриваются:

- дизассемблер промежуточного кода;
- декомпилятор;
- оптимизатор кода приложения;
- отладчик уровня пользователя;
- отладчик уровня ядра операционной системы;
- приложение для получения трассы приложения;
- аппаратный отладчик.

В результате проведенного моделирования возможных действий нарушителя были сформулированы следующие требования, предъявляемые к разрабатываемой системе защиты приложений от копирования:

- анализ приложения без внешнего аппаратного модуля должен являться вычислительно сложной задачей;
- разработанный метод должен позволять реализацию системы на ключах быть реализуемым на практике;
- возможность задания необходимого уровня защищенности приложения.

В работе в виде модели, интерпретирующей приложение, предлагается использовать специфичный конечный автомат Мура, который представлен в виде семи объектов:

$A = \{X, Y, S, S_b, S_e, h, f\}$, где X – входной алфавит автомата A, Y – выходной алфавит автомата, S – множество состояний автомата, $S_b \subset S$ – множество начальных состояний автомата, $S_e \subset S$ – множество конечных или выходных состояний автомата, h – отображение $h: S \times X \to S$ или функция переходов, f – отображение $f: S \to Y$ или функция выходов. Особенность предлагаемой модели заключается в том, что работа приложения обязательно должна начинаться с некоторого состояния $s_0 \in S_b$, а заканчиваться

состоянием $s_t \in S_e$.

Построение конечного автомата, описывающего работу приложения, происходит в соответствии с описывающей приложение другой математической моделью – графом потока управления приложения.

Граф потока управления $G=\{U,V\}$ приложения задается набором множеств: U – множество вершин рассматриваемого графа, V – множество дуг рассматриваемого графа. Для графа потока управления множество U состоит из линейных блоков приложения. Под линейными блоками приложения (в дальнейшем базовыми блоками приложения) понимаются участки программного кода приложения, не содержащие ветвлений, переходов и инструкций передачи управления. Множество дуг V включает в себя пары элементов из множества U, то есть $V \subset \{U \times U\}$, таким образом, что дуга между двумя вершинами существует тогда и только тогда, если переход из одной вершины осуществим в другую. Другими словами, если инструкция (или набор инструкций) приложения, представленная на графе вершиной u_k будет вызвана сразу после обращения к инструкции, представленной вершиной u_i, то вершины u_k и u_i будут связаны между собой дугой $\{u_k,u_i\}$. Важно отметить, что рассматриваемый граф потока управления приложения является ориентированным, в связи с тем, что при исполнении кода приложения важен порядок вызова инструкций (базовых блоков).

Конечный автомат A, описывающий приложение, строится в соответствии с графом потока управления приложения следующим образом:

– множество состояний – виртуальные метки, обозначающие начало отдельного базового блока в приложении;

– множество входов составляют переменные, в соответствии со значениями которых осуществляется передача управления в приложении;

– множество выходов составляют базовые блоки приложения;

– начальными состояниями автомата помечаются такие состояния, которые напрямую достижимы из точки входа защищаемого приложения;

– конечными состояниями автомата помечаются состояния, которые на графе потока управления приложения представлены висячими вершинами;

– функции переходов и выходов конечного автомата представляют собой таблицу переходов в соответствии с графом потока управления.

Множество состояний S – множество виртуальных меток в приложении. Каждая виртуальная метка соответствует началу базового блока приложения. Таким образом, множество состояний будет строиться в соответствии с ветвлениями графа потока управления приложения P.

Множество входов X – набор переменных приложения P, в соответствии со значениями которых, происходит тот или иной переход в графе потока управления защищаемого приложения. Иными словами, если в зависимости от значения некоторой переменной в теле программы происходит переход от одной вершины графа потока управления к другой, то данная переменная является элементом множества X.

Множество выходов Y – набор инструкций, которые приложение будет выполнять при переходе из одного состояния в другое.

Отображения, интерпретирующие функцию переходов h и функцию выходов f, задаются таблицей связей между состояниями автомата и связей между состояниями автомата и элементами множества выходов.

Постановка задачи. Пусть защищаемое приложение представлено в виде ко-

нечного автомата ***A.***

Необходимо, используя алгебру $\Sigma(A_s,R)$, построить набор преобразований $R:(A,k)\rightarrow A'$, $A' = \{ X', S', Y', S_s', S_e', h', f' \}$ таким образом, чтобы процесс восстановления автомата A, по известному набору множеств $\{X', S', Y'\}$, зависел от длины ключа K и являлся вычислительно сложной задачей.

Введенный ключ необходим для задания уровня сложности компрометации системы защиты приложений. Длина ключа определяет количество добавленных базовых блоков защищенного приложения, что задает уровень сложности компрометации системы защиты.

Для достижения поставленной задачи необходимо использовать следующий набор преобразований, принадлежащих множеству R. Зададим алгебру $\Sigma(A_s,R)$, где A_s – множество конечных автоматов Мура, интерпретирующих приложения, R – набор преобразований. Множество преобразований, входящих в алгебру Σ, осуществляемых над конечным автоматом A, состоит из следующих преобразований:

– $r_u: (A,k)\rightarrow A'$, $A'= r_u\ (A,k)=\{X', Y', S', S_s', S_e', f', h'\}$– преобразование, аргументами которого являются автомат A, интерпретирующий защищаемое приложение P, и элемент ключа $k\in K$. В зависимости от элемента ключа, преобразование осуществляет добавление вершины к графу потока управления приложения, или, с точки зрения конечного автомата, осуществляется добавление элемента к множеству выходов Y, множеству входов X и множеству состояний S. Множества начальных и конечных состояний $\{S_b,S_e\}$ автомата A остаются неизменными, чтобы не нарушить логику работы приложения P.

– $r_d:(A,k)\rightarrow A'$, $A'= r_d\ (A,k)=\{X', Y', S', S_s', S_e', f', h'\}$ – добавление «мертвого» кода к графу потока управления приложения. Рассматриваемое преобразование осуществляет добавление вершины к оригинальному графу потока управления приложения. В результате, код, который был добавлен, не несет никакой функциональной нагрузки. Выполнение внедренного кода не несет никакой модификации переменных. Однако, в связи с тем, что сложность восстановления оригинального графа потока управления, и, как следствие оригинального автомата A, основывается на количестве базовых блоков приложения, применение описываемой операции позволит увеличить сложность анализа защищенного приложения.

– $r_c:(A,k)\rightarrow A'$, $A'= r_c\ (A,k)=\{X', Y', S', S_s', S_e', f', h'\}$ – усложнение пути в графе потока управления защищаемого приложения. Суть преобразования заключается в том, что существующий элемент множества выходов $y \in Y$ конечного автомата A приложения P отображается в несколько элементов. В результате применения данного преобразования к графу потока управления защищаемого приложения происходит замена одной вершины графа на несколько. Причем логика работы защищенного приложения не изменяется. Добавленные вершины, в совокупности, интерпретируют тот же участок кода приложения, что и оригинальная вершина.

В результате осуществления представленных преобразований увеличивается число базовых блоков в защищаемом приложении, что в свою очередь приводит к усложнению процесса компрометации защищенного приложения.

Процесс защиты приложения реализуется за счет разделения конечного автомата A, интерпретирующего защищаемое приложение, на две группы объектов:

- множества X, S, Y;
- множества S_b, S_e, а также отображения f, h.

В результате такого разбиения конечного автомата, кортеж из множества входов, выходов и состояний предоставлен злоумышленнику для анализа, а остальные объекты автомата скрыты во внешнем аппаратном модуле.

Для совершения несанкционированного копирования защищаемого приложения злоумышленнику необходимо восстановить конечный автомат, описывающий оригинальное приложение.

Как показано в работе, при реализации атаки методом полного перебора для восстановления оригинального графа потока управления, сложность процесса восстановления составит $\boldsymbol{O(2^{n^2})}$, оценка нижней границы сложности компрометации системы защиты производится по теореме Кэли для деревьев и составляет $\boldsymbol{\Omega(n^{n-1})}$, где $\boldsymbol{n}$ число базовых блоков приложения или число состояний автомата $\boldsymbol{A}$, описывающего защищенное приложение.

Одним из основных способов снижения сложности компрометации системы защиты, предложенной в работе, является анализ переменных приложения. Для предотвращения анализа переменных защищенного приложения предлагается использовать аналог задачи «Псевдонимов», которая формулируется следующим образом:

В коде приложения существует набор указателей, тогда задача определения средствами статического анализа того факта, что данные указатели ссылаются на одну область памяти, относится к классу NP-полных задач.

В рамках данной работы используется следующая интерпретация задачи «Псевдонимов»:

Пусть имеется программа $\boldsymbol{P}$, в которой используется некоторое конечное множество переменных $\boldsymbol{V}$. Обращения на чтение либо запись к переменной $\boldsymbol{v_i}$ образуют собой некоторое конечное множество $\boldsymbol{W_i}$. В результате, в коде приложения вместо отдельного обращения к переменной на чтение либо запись поставим некоторое число, которое единственным образом идентифицирует обращение к переменной. Например, при нахождении обращения к элементу множества $\boldsymbol{W_k}$ данное обращение будет заменяться индексом; в результате, некоторый индекс $\boldsymbol{I}$ соответствует процессу записи (либо чтения) $\boldsymbol{w_k}$ переменной $\boldsymbol{v_k}$. Таким образом, при попытке восстановления оригинальных обращений к переменным возникает задача, аналогичная рассмотренной выше задаче «Псевдонимов».

В результате применения такого подхода для защиты переменных становится вычислительно сложно осуществить атаки частичного восстановления графа потока управления защищенного приложения с использованием переменных. Задача восстановления оригинальных обращений к переменным соответствует сложности решения задачи «Псевдонимов». Таким образом, реализация атак, основанных на установлении взаимосвязей между базовыми блоками за счет отслеживания работы с переменными приложения, является NP-трудной.

Для предотвращения реализации различных атак на внешний аппаратный модуль в работе предложена следующая модель его работы. Функционирование внешнего аппаратного модуля описывается при помощи конечного автомата $\boldsymbol{B}=\{\boldsymbol{X},\boldsymbol{Y},\boldsymbol{S},\boldsymbol{h},\boldsymbol{f}\}$, в котором множество $\boldsymbol{X}$ – входной алфавит конечного автомата $\boldsymbol{B}$. Множество $\boldsymbol{Y}$ – все выходные слова автомата, то есть все возможные выходы аппаратного модуля. $\boldsymbol{S}$ – множество состояний автомата. При описании работы внешнего аппаратного модуля множество $\boldsymbol{S}$ содержит четыре элемента, то есть, рассматриваемый конечный автомат может находиться в четырех состояниях, а именно:

– штатное состояние;
– состояние **«Тревога»;**
– состояние **«Ошибка»;**
– состояние **«Отказ».**

Штатное состояние автомата регламентирует корректное функционирование внешнего аппаратного модуля, при котором происходит обработка запросов и нормальное функционирование защищенного приложения. При получении некорректных данных от защищенного приложения (неверная последовательность смены состояний, неверная последовательность обращений к переменным, длительное время перехода из одного состояния в другое) автомат, интерпретирующий внешний аппаратный модуль, переходит в состояние **«Тревога»**. После этого автомат переходит в состояние **«Ошибка»**. Корректная работа внешнего аппаратного модуля может быть восстановлена за счет подачи ему команды «Сброс». При многократном переходе в состояние **«Тревога»**, в соответствии с политикой безопасности, аппаратный модуль переходит в состояние **«Отказ»**. Дальнейшая работа приложения при переходе внешнего аппаратного модуля в режим **«Отказ»** невозможна.

Автоматный граф, интерпретирующий поведение аппаратного модуля в соответствии с автоматом ***B,*** представлен на рис. 2.

Рисунок 2 - Автоматный граф работы внешнего аппаратного модуля

Совместное применение представленных в работе методов защиты приложений от несанкционированного копирования реализует методику защиты.

Методика защиты приложений, предложенная в работе, реализуется за счет следующей последовательности шагов:

– построение графа потока управления защищаемого приложения;
– построение конечного автомата защищаемого приложения по полученному графу потока управления;
– применение запутывающих преобразований к полученному конечному авто-

мату, интерпретирующему защищаемое приложение;

– «разделение» конечного автомата на две группы объектов;

– подготовка низкоуровневого программного обеспечения для внешнего аппаратного модуля;

– выгрузка функции переходов и функции выходов, а также графа потока управления защищаемого приложения во внешний аппаратный модуль;

– защита переменных приложения посредством предложенного в работе способа защиты;

– выгрузка списка обращений к переменным во внешний аппаратный модуль.

В результате проведения патентного поиска было установлено, что предложенная в работе методика защиты приложений от копирования имеет патентную чистоту и обладает всеми признаками и характеристиками изобретения. Оформлена заявка на получение патента РФ.

В **третьей главе** диссертационной работы предложена архитектура программно-аппаратных комплексов, реализующих предложенную в работе методику защиты приложений от несанкционированного копирования.

Для реализации предлагаемого в работе метода защиты приложений предлагается следующий набор функциональных модулей:

– модуль дизассемблирования приложения;
– модуль генерации графа потока управления приложения;
– модуль создания и работы с внутренним представлением приложения;
– модуль запутывания кода приложения;
– модуль защиты переменных приложения;
– модуль генерации файлов для аппаратной составляющей системы защиты;
– модуль работы с аппаратной составляющей системы защиты.

Модуль дизассемблирования приложения. Отвечает за получение из незащищенного приложения кода этого приложения на языке ассемблера соответствующей виртуальной машины (в случае защиты .NET приложений – MSIL). Результат работы модуля используется для построения графа потока управления приложения, а также при построении конечного автомата, интерпретирующего приложение.

Модуль генерации графа потока управления приложения. Реализует собой функционал по построению графа потока управления защищаемого приложения.

Модуль создания и работы с внутренним представлением приложения. Реализует собой функционал по построению конечного автомата, интерпретирующего защищаемое приложение, а также основной набор функций по работе с полученным автоматом (применение запутывающих преобразований, «разделение» автомата и т.д.).

Модуль запутывания кода приложения. Осуществляет запутывающие преобразования над внутренним представлением защищаемого приложения. Результаты работы модуля запутывания кода интерпретируются на сам код приложения посредством функционала внутреннего представления. С функциональной точки зрения, модуль обладает возможностью по добавлению новой вершины к оригинальному графу потока управления приложения, модификации пути в графе без изменения логики работы приложения. Кроме того, у модуля запутывания кода приложения присутствует функционал по добавлению и генерации «мертвого» и недостижимого кода.

Модуль защиты переменных приложения. Отвечает за защиту переменных

приложения предложенным в работе методом. Среди основных функций, реализуемых модулем,– поиск переменных в функции и классе приложения, поиск обращений к переменной приложения, модификация обращения к переменной, добавление кода по работе защищенного приложения с внешним аппаратным модулем. Кроме того, модуль осуществляет генерацию последовательности обращений к переменным для реализации методики.

Модуль генерации файлов для аппаратной составляющей системы защиты. Осуществляет создание всех необходимых файлов, выгружаемых во внешний аппаратный модуль. Иными словами, в связи с тем, что каждый отдельный файл внешнего аппаратного модуля является исполняемым файлом, модуль генерации файлов аппаратного модуля осуществляет компиляцию файлов.

Модуль работы с аппаратной составляющей системы защиты. Реализует работу с внешним аппаратным модулем в виде основного функционала по записи данных, а также основную техническую работу с аппаратным модулем (поиск, настройка и т.д.). Кроме того, одной из функций модуля работы с внешним аппаратным модулем является возможность записи низкоуровневого программного обеспечения, полученного от модуля генерации файлов аппаратного модуля, во внешний аппаратный модуль.

Общая архитектура программно-аппаратного комплекса представлена на рис 3.

Рисунок 3 - Архитектура комплекса защиты приложений от копирования

В **четвертой главе** приведено описание реализации в виде программного обеспечения основных модулей, предложенных в архитектуре комплекса защиты приложений. Разработка программно-аппаратного комплекса осуществлялась на языке программирования C# платформы .NET. Выбор языка обусловлен тем, что для работы

с внутренним представлением .NET приложений (дизассемблирование, модификация) наиболее полно подходят существующие в данном языке классы и пространства имен.

В ходе реализации, для предотвращения снижения производительности приложения после процесса защиты, был использован метод кэширования запросов к внешнему аппаратному модулю. При многократном обращении приложения к аппаратному модулю, например, при работе в цикле, наблюдались крайне высокие (до 10 раз) замедления защищенного приложения. Для того, чтобы избавиться от этой проблемы был использован механизм кэширования запросов, который состоит в том, что если в результате перехода состояние автомата, интерпретирующего защищенное приложение, не изменяется (приложение выполняет некоторый цикл), то адрес перехода не запрашивается у внешнего аппаратного модуля, а берется из некоторой переменной. В результате, при первом выполнении тела цикла происходит обращение к аппаратному модулю, в дальнейшем переход осуществляется в соответствии с имеющимся адресом вызываемой функции приложения.

Для снижения замедления в случае с приложением с большим количеством переходов был использован метод параллельной обработки данных. В случае, если, следующий переход является безусловным (например, вызов функции), то получение адреса такого перехода осуществляется совместно с выполнением кода защищенного приложения в отдельном потоке.

В ходе тестирования проводилась защита опытных образцов программного кода и оценивались потери в производительности по сравнению с незащищенным аналогом. Для осуществления тестирования производительности защищенного приложения было выделено несколько классов приложений, дающих наибольшее замедление, а именно:

- приложения, состоящие из цикла, с большим количеством вычислений за проход;
- приложения, состоящее из набора коротких циклов;
- приложения с большим количеством переходов.

По результатам тестирования было установлено, что среднее замедление защищенного приложения, относительно его оригинала, составляет не более чем в 3 раза.

Основной причиной возникновения замедлений в работе защищенного приложения является более низкая скорость обращения к блоку памяти внешнего аппаратного модуля по сравнению со скоростью обращения к блоку оперативной памяти.

Предложенный в работе метод кэширования обращений к внешнему аппаратному модулю позволил снизить замедление работы защищенного приложения.

Разработанный в ходе создания методики защиты приложений способ защиты переменных приложения использован при разработке программно-аппаратного комплекса, выпускаемого компанией «Актив», для противодействия несанкционированному копированию программного обеспечения, что позволило привести сложность компрометации разработанной системы защиты к классу NP полных задач.

Методика реализована в компании «Интерактивное детство» для защиты Flash-приложений от несанкционированного копирования, что позволило повысить сложность совершения несанкционированного копирования учебных курсов компании «Интерактивное детство» с высокой сложностью компрометации и защитить авторское право разработчика учебных курсов.

Результаты диссертационной работы использованы на кафедре «Криптология и дискретная математика» НИЯУ МИФИ для углубления учебного курса «Языки про-

граммирования». В результате внедрения создана лабораторная работа, позволяющая слушателям курса получить знания по внутренней архитектуре .NET сборок, а также по методам анализа приложений и существующих способах их защиты от несанкционированного копирования.

Элементы предложенной методики защиты приложений от несанкционированного копирования, а именно:

– способ защиты приложения посредством разделения конечного автомата;
– принцип работы внешнего аппаратного модуля

были реализованы для защиты приложения, осуществляющего анализ графа потока управления вредоносного программного обеспечения, для Центра вирусных исследований и аналитики «Eset». Также подобная система защиты приложений была реализована для противодействия несанкционированному копированию программного обеспечения для создания смет компании ООО «Связьмонтажкомплектация». В результате, несанкционированное копирование защищенных приложений стало вычислительно трудной задачей.

В **заключении** приведены основные результаты диссертационной работы, а также представлены выводы, полученные в ходе выполнения работы.

ОСНОВНЫЕ ВЫВОДЫ И РЕЗУЛЬТАТЫ РАБОТЫ

1. Проведен анализ существующих способов защиты программного обеспечения от несанкционированного копирования для приложений, компилируемых в промежуточное представление. Обоснована необходимость создания новой методики защиты приложений от несанкционированного копирования;
2. Построена модель нарушителя, реализующего атаки на систему защиты приложений, компилируемых в промежуточное представление. Построенная модель нарушителя позволила осуществить анализ возможных атак на систему защиты, что позволило предотвратить возможность реализации выявленных атак в предложенном в работе методе защиты;
3. Предложена математическая модель системы защиты от несанкционированного копирования приложений, компилируемых в промежуточное представление, которая позволяет производить оценку сложности компрометации систем защиты приложений от несанкционированного копирования;
4. Представлена математическая модель работы внешнего аппаратного модуля, описанная в терминах теории конечных автоматов. Модель позволяет повысить сложность компрометации системы защиты приложений и предотвратить возможность реализации атак, основанных на динамическом исследовании приложения.
5. Сформулирована методика защиты приложений от несанкционированного копирования, обеспечивающая защиту с заданным уровнем сложности компрометации. Системы защиты, реализующие предложенную методику, обладают высокой сложностью компрометации. Задача несанкционированного копирования приложения относится к классу задач EXPTIME;
6. Построена архитектура для создания программных решений, реализующих систему защиты приложений от несанкционированного копирования. На основании предложенной архитектуры разработан программно-аппаратный комплекс, реализующий защиту приложений от несанкционированного копирования. Основным преиму-

ществом предложенного программно-аппаратного комплекса является возможность задания уровня сложности компрометации системы защиты, сохраняя при этом класс сложности задачи компрометации защищенного приложения;

7. Предложенный в рамках методики метод защиты переменных был использован при построении системы защиты приложений компании «Актив», что привело к увеличению сложности компрометации предлагаемой компанией системы защиты приложений. Методика защиты была использована и реализована для защиты Flash-приложений от несанкционированного копирования для компании «Интерактивное Детство»;

8. Проведенный анализ методов защиты приложений от копирования был использован при создании лабораторной работы для курса «Языки программирования» кафедры «Криптология и дискретная математика» НИЯУ МИФИ. Слушатели курса могут ознакомиться со способами анализа .NET приложений, а также со способами построения различных систем защиты приложений от несанкционированного копирования для данного класса приложений;

9. Разработана система защиты для противодействия несанкционированному копированию приложения, используемого для анализа графа потока управления вредоносного программного обеспечения в Центре вирусных исследований и аналитики «Eset». Предложенная система защиты позволила привести сложность анализа приложения к классу задач EXPTIME;

10. Разработанная система защиты приложений применяется для обеспечения безопасности приложения, используемого для составления смет, в компании ООО «Связьмонтажкомплектация». В результате реализации системы защиты приложений от несанкционированного копирования, с использованием предложенной в работе методики, копирование защищенных приложений является вычислительно трудной задачей;

Основные публикации по теме диссертации:

1. Краснопевцев А.А., Букасов В.А. Разработка средств автоматической защиты приложений, содержащих байт-код. – Материалы 10-й международной научно-практической конференции, Таганрог, 24-27 июня 2008. – с. 15-16
2. Букасов В.А, Краснопевцев А.А. Автоматизация динамической распаковки программ. – Материалы 10-й международной научно-практической конференции, Таганрог, 24-27 июня 2008.– с. 13-14
3. Краснопевцев А.А. Разработка средств автоматизации для переноса байт-кода во внешний аппаратный модуль. – Технологии Microsoft в теории и практике – М.:Вузовская книга, 2008. – с. 139-141
4. Краснопевцев А.А. Разработка автоматической защиты от несанкционированного копирования .NET приложений. – Научная сессия МИФИ-2009. XII Московская международная телекоммуникационная конференция студентов и молодых ученых «Молодежь и Наука». Тезисы докладов в 2-х частях. Ч. 2. М.:МИФИ, 2009. – с. 271-272.
5. **Краснопевцев А.А. Разработка автоматической защиты от несанкционированного копирования .NET приложений с использованием внешнего аппаратного модуля. //Безопасность информационных технологий 2009. №1. С.54-58**

6. **Краснопевцев А.А. Разработка автоматической защиты приложений и передаваемых, обрабатываемых и хранимых ими данных. //Безопасность информационных технологий 2010. № 3. С.82-85. А.А. Краснопевцев О защите приложений с использованием внешнего аппаратного модуля //Безопасность информационных технологий 2011. № 1. С.102-104.**
7. **Фомичев В.М., Варфоломеев А.А., Туманов Ю.М., Коренева А.М., Краснопевцев А.А. О реализации метода полного опробования ключей криптосистем в условиях различных математических моделей распределенных вычислений. //Безопасность информационных технологий 2011. № 1. С. 80-82.**