Московский государственный университет
имени М.В. Ломоносова
Факультет вычислительной математики и кибернетики

На правах рукописи

Гайворонская Светлана Александровна

ИССЛЕДОВАНИЕ МЕТОДОВ ОБНАРУЖЕНИЯ ШЕЛЛКОДОВ В ВЫСОКОСКОРОСТНЫХ КАНАЛАХ ПЕРЕДАЧИ ДАННЫХ

Специальность 05.13.11 — математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей

АВТОРЕФЕРАТ
диссертации на соискание ученой степени
кандидата физико-математических наук

МОСКВА
2014

Работа выполнена на кафедре автоматизации систем вычислительных комплексов факультета вычислительной математики и кибернетики Московского государственного университета имени М.В. Ломоносова.

Научные руководители:	доктор физико-математических наук, профессор, член-корр. РАН Смелянский Руслан Леонидович;
Официальные оппоненты:	доктор физико-математических наук, зам. Генерального директора ФГУП Главного научно-исследовательского вычислительного центра ФНС России, Баранов Александр Павлович; кандидат физико-математических наук, доцент, доцент кафедры информационной безопасности МГИУ Бутакова Наталья Георгиевна.
Ведущая организация:	Федеральное государственное бюджетное учреждение науки Научно-исследовательский институт системных исследований Российской Академии Наук (НИИСИ РАН).

Защита состоится «19» сентября 2014 г. в 11:00 на заседании диссертационного совета Д 501.001.44 при Московском государственном университете имени М.В. Ломоносова по адресу: 119991, ГСП-1, Москва, Ленинские горы, МГУ, 2-ой учебный корпус, факультет вычислительной математики и кибернетики, аудитория 685.

С диссертацией можно ознакомиться в библиотеке факультета ВМиК МГУ имени М.В. Ломоносова, с текстом автореферата — на официальном сайте ВМиК МГУ имени М.В. Ломоносова: `http://www.cs.msu.su` в разделе «Наука» — «Работа диссертационных советов» — «Д 501.001.44».

Автореферат разослан «___» _____ 2014 г.

Ученый секретарь
диссертационного совета Д 501.001.44
к.т.н, в.н.с. В.А. Костенко

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность работы. С начала 2000-х годов и по сегодняшний день одним из ключевых инструментов киберпреступности являются ботнеты. *Ботнетом* называется сеть зараженных узлов, на которых запущен автономный процесс, выполняющий команды злоумышленника. Среди крупных ботнетов можно отметить Torpig, подробно исследованный командой ученых Университета Калифорнии в Санта-Барбаре, Zeus, по которому в 2010 году было завершено расследование ФБР и арестовано более двадцати человек, а так же ботнет Conficker, который привлекал внимание исследователей с 2008 года, и долгое время оставался одним из самых распространенных ботов на пользовательских компьютерах. Ботнеты используются для самой разнообразной криминальной деятельности, наиболее распространенными видами которой являются фишинг, организация DDoS-атак, рассылка спама, кликфрод и другие.

Несмотря на то, что набирает обороты использование веб-уязвимостей для распространения ботнетов посредством drive-by-download, заражения легитимных сайтов, значимость удаленно эксплуатируемых уязвимостей в распространенном программном обеспечении не снижается, и вряд ли будет снижаться в ближайшее время. Большая установочная база уязвимой версии программного обеспечения гарантирует возможность быстрого захвата значительного числа узлов. В качестве примера можно привести недавно исправленные уязвимости протокола RDP компании Майкрософт [1], уязвимости Java [2]. За последние три года, согласно статистике CVE [3], было опубликовано

[1] Microsoft Security TechCenter. Microsoft security bulletin ms12- 020 - critical: Vulnerabilities in remote desktop could allow remote code execution (2671387). Online: http://technet.microsoft.com/en-us/security/bulletin/ms12-020.

[2] Multi-State Information Sharing and Analysis Center. Vulnerability in oracle java runtime environment could allow remote code execution. Online: https://msisac.cisecurity.org/advisories/2013/2013-041.cfm.

[3] CVE details. Security vulnerabilities published in 2013. // Online: http://www.cvedetails.com/vulnerability-list.php.

около 15000 удаленно эксплуатируемых уязвимостей.

В настоящей работе рассматривается проблема обнаружения и фильтрации *шеллкодов* (shellcode) - вредоносного исполнимого кода, эксплуатирующего уязвимости работы с памятью. Типичными примерами таких уязвимостей являются переполнение стека, переполнение кучи, а так же некоторых других служебных структур.

В настоящее время существует несколько десятков систем обнаружения шеллкодов, использующих как статический, так и динамический анализ программ. Тем не менее, анализ существующих систем показал, что методы, обладающие невысокой вычислительной сложностью, характеризуются большим процентом ложных срабатываний. Верно и обратное: методы, обладающие невысоким процентом ложных срабатываний, характеризуются высокой вычислительной сложностью. Кроме того, ни одно из существующих на настоящий момент решений не в состоянии обнаруживать все существующие классы шеллкодов. Это делает существующие системы слабо применимыми к реальным каналам передачи данных. Таким образом, возникает задача создания системы обнаружения шеллкодов, обеспечивающая полное покрытие существующих классов шеллкодов и характеризующаяся приемлемой вычислительной сложностью и числом ложных срабатываний.

Цель работы. Цель данной диссертационной работы - исследование методов обнаружения шеллкодов при их передаче по высокоскоростным каналам, а так же разработка метода обнаружения шеллкодов, распознающего максимально полно известные классы шеллкодов и имеющего наименьшее число ложных срабатываний.

Методы исследования. При получении основных результатов работы диссертации использовались методы теории вероятностей, теории графов, статического и динамического анализа программ.

Научная новизна. В настоящей работе разработана модель распознава-

ния объектов, позволяющая обнаруживать в объектах набор специфических признаков и относить объекты к некоторым из заданных классов. Данная модель может быть применена как к задаче обнаружения шеллкода, так и к задаче обнаружения любого другого вредоносного программного обеспечения. В рамках разработанной модели предложен метод решения задачи распознавания шеллкодов, позволяющий обеспечить полноту покрытия обнаруживаемых классов шеллкодов, а так же снизить вычислительную сложность обнаружения по сравнению с линейной комбинацией алгоритмов и оптимизировать значение ложных срабатываний.

В диссертации описан набор специфических признаков, присущих известным шеллкодам, а так же впервые предложена классификация шеллкодов.

Практическая ценность. Экспериментально реализованная система обнаружения шеллкодов может быть использована в рамках IDS/IPS систем для выявления и фильтрации шеллкодов различных классов на высокоскоростных каналах передачи данных, а так же может быть использована на конечном устройстве для сканирования любых документов на предмет содержания шеллкодов в них. Разработанная система существенно снижает вычислительную нагрузку как на IDS/IPS систему в первом случае, так и на конечное устройство.

Апробация работы. Результаты, представленные в работе, докладывались на научном семинаре лаборатории вычислительных комплексов кафедры АСВК факультета ВМиК МГУ им М. В. Ломоносова под руководством член-корр. РАН Р. Л. Смелянского; на семинаре кафедры АСВК под руководством член-корр. РАН Л. Н. Королева; на научном семинаре группы «Network and system security» исследовательского подразделения компании Майкрософт, а так же на следующих конференциях:

- Конференция «РусКрипто», Солнечногорск, Россия, 27-29 марта 2011г.

- Конференция «РусКрипто», Солнечногорск, Россия, 28-31 марта 2012г.
- Международная конференция «DEFCON-20», Лас-Вегас, США, 26-30 июля 2012г.
- Международная конференция «BlackHat-EU», Амстердам, Нидерланды, 12-15 марта 2013г.
- Международная конференция «Ломоносов», Москва, Россия, 8-13 апреля 2013г.
- Летний коллоквиум молодых ученых в области программной инженерии «SYRCoSE», Казань, Россия, 30-31 мая 2013 г.
- Международная конференция «NOPCON», Стамбул, Турция, 6 июня 2013г.

Работа была выполнена при поддержке фонда «Сколково».

Публикации. По теме диссертации имеется 5 публикаций (включая 2 в изданиях из перечня ВАК), список которых приводится в конце автореферата.

Структура и объем диссертации. Диссертация состоит из введения, пяти глав, списка литературы и приложения. Объем работы - 129 страниц, с приложением - 133 страницы. Список литературы содержит 112 наименований.

ОСНОВНОЕ СОДЕРЖАНИЕ РАБОТЫ

Введение содержит краткий обзор предметной области, обоснование актуальности диссертационной работы. Сформулирована цель исследований и приведен краткий обзор содержания диссертации.

В **первой главе** предлагается формальная модель процесса распознавания шеллкода (объекта), свойства этого процесса, его основные понятия и их свойства. В рамках построенной модели ставится задача распознавания объектов - задача отнесения объектов в некоторому фиксированному множеству классов объектов. Так же в данной главе приводится доказательство существования решения задачи распознавания шеллкодов в рамках приведенной модели и приводится алгоритм, строящий такое решение.

В **разделе 1.1** приводится формальная модель процесса распознавая шеллкода.

В качестве *исследуемого объекта* рассматривается набор битовых исполнимых строк $\{\mathfrak{s}_i\}_{i=1}^{n} = \{\mathfrak{s}_1, \ldots, \mathfrak{s}_n\}$. Множеству рассматриваемых объектов сопоставляется набор *признаков* $\{\mathfrak{f}_j\}_{j=1}^{k}$, каждым из которых обладает хотя бы один из объектов $\mathfrak{s}_i$ рассматриваемого множества. На множестве объектов $\{\mathfrak{s}_i\}_{i=1}^{n}$ вводится множество вычислимых функций $\{\mathfrak{F}_j(\mathfrak{s}_j)\}_{j=1}^{k}$, ставящих в соответствие объекту $\mathfrak{s}_i$ значение признака $\mathfrak{f}_j$, где признаки могут принимать значения из множества $\{0, 1\}$. Значение признака $\mathfrak{f}_j$ равно 0 в том случае, если объект $\mathfrak{s}_i$ не обладает признаком $\mathfrak{f}_j$ и 1 в противном случае.

Производится разбиение множества исследуемых объектов на *классы* - подмножества $\{K_1, ..., K_l\}$, где каждый класс определяется некоторой структурой признаков из $\{\mathfrak{f}_k\}$. Взаимосвязь между признаками, определяющими некоторый класс K_j, и самим классом K_j, задается формулой $P_j(S) = [\mathfrak{F}_{j_{1_1}}(S) \vee \cdots \vee \mathfrak{F}_{j_{1_m}}(S)] \wedge \cdots \wedge [\mathfrak{F}_{j_{k_1}}(S) \vee \cdots \vee \mathfrak{F}_{j_{k_n}}(S)]$, представляемой в виде конъюнктивной нормальной формы (КНФ). На множестве классов задается отношение частичного порядка $\prec$, определяющее, в какой последовательности необходимо проверить признаки для отнесения некоторого объекта $\mathfrak{s}_i$ к одному из классов объектов.

Так как множество $\{\mathfrak{F}(\mathfrak{s})\}$ представляет из себя множество вычислимых функций, то для каждой функции $\mathfrak{F}_j \in \{\mathfrak{F}(\mathfrak{s})\}$ существует алгоритм $\mathfrak{A}_k$, реа-

лизующий ее. На множестве алгоритмов $\{\mathfrak{A}\}$, так же сохраняется отношение частичного порядка $\prec$.

Каждому объекту $\mathfrak{s}_j$ сопоставлен *информационный вектор* $\tilde{\alpha}(\mathfrak{s}_j) = (\alpha_1, \ldots, \alpha_l)$, кодирующий информацию о принадлежности объекта $\mathfrak{s}_j$ к классам $\{K_1, \ldots, K_l\}$.

Вводится понятие *классификатора* μ_i- сущности, содержащей структуру некоторого подмножества $\{\mathfrak{A}_k\}'$ алгоритмов $\mathfrak{A}_k$, представляющую из себя граф, в котором узлы являются алгоритмами из подмножества $\{\mathfrak{A}_k\}'$, а дуги соответствуют путям передачи входного объекта между алгоритмами. Все алгоритмы, входящие в классификатор μ_i структурированы с учетом отношения частичного порядка. Классификатор μ_i способен распознавать объекты класса K_j в том случае, если пересечение множества признаков, вычисляемых алгоритмами, которые содержатся в классификаторе μ_i, и множества признаков, определяющих класс K_j, не пусто. Каждый классификатор корректно распознает обнаруживаемые им классы объектов с некоторой вероятностью.

В **разделе 1.2** в рамках построенной модели рассматривается задача распознавания объектов - задача отнесения объектов к некоторым классам из заданного списка классов. Пусть задано фиксированное множество классов $\{K_l\}$ объектов. Пусть так же задано множество классификаторов $\{\mu_m\}$, способных распознавать объекты всех классов из множества $\{K_l\}$, при этом каждый классификатор распознает некоторое непустое подмножество классов $\{K_l\}$. Задача состоит в построении распознающего алгоритма $\mathfrak{A}'$, представляющего из себя такую суперпозицию классификаторов, для которой выполнено следующее условие:

- Для любого объекта $\mathfrak{s}_i$, принадлежащего любому классу K_j из множества заданных классов, структура классификаторов, построенная алгоритмом $\mathfrak{A}'$, распознает объект $\mathfrak{s}_i$ как объект класса K_j.

В качестве доказательства решения этой задачи в данном разделе указан алгоритм построения распознающего алгоритма $\mathfrak{A}'$. Распознающий алгоритм в представленном решении представляет из себя линейную структуру классификаторов, отсортированных в соответствии с отношением частичного порядка классификаторов (рисунок 1.1).

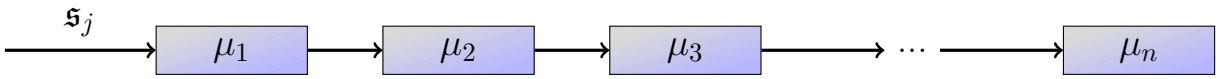

Рис. 1.1. Линейная структура классификаторов.

Далее в разделе проводится исследование представленного решения. Показывается, что представленное решение обеспечивает полноту покрытия классов объектов и оптимально с точки зрения вероятности ошибок. Показано, что вычислительная сложность работы распознающего алгоритма оценивается значением $\mathfrak{C} = \sum_{i=1}^{m} \mathfrak{c}_{\mu_i} = \sum_{\forall j:\mathfrak{A}_i \in \{\mu_i\}_{i=1}^{m}} \mathfrak{c}_{\mathfrak{A}_i}$, где $\mathfrak{c}_{\mu_i}$ - вычислительная сложность классификатора μ_i, а $\mathfrak{c}_{\mathfrak{A}_i}$ - вычислительная сложность алгоритма $\mathfrak{A}_i$, входящего в состав некоторого классификатора. Таким образом, вычислительная сложность распознающего алгоритма равна суммарной сложности классификаторов, входящих в структуру классификаторов.

В **разделе 1.3** рассмотрен вопрос о существовании более оптимального по сложности выполнения решения задачи распознавания. Принимается следующее допущение. Каждый алгоритм $\mathfrak{A}_i$ может сбрасывать входящие в него объекты, то есть не передавать объект на вход другим алгоритмам, если он не обнаружил во входном объекте соответствующий признак. Такой же функцией могут обладать и сами классификаторы.

С учетом этих допущений рассматривается следующая *задача распознавания объектов*. Пусть задано фиксированное множество классов $\{K_1, \ldots, K_l\}$ объектов. Пусть так же задано множество классификаторов $\{\mu_1, \ldots, \mu_m\}$, спо-

собных распознавать объекты всех классов их множества $\{K_1, \ldots, K_l\}$, при этом каждый классификатор распознает непустое подмножество заданных классов. Кроме того, будем считать, что классификаторы могут сбрасывать объекты. Задача состоит в построении распознающего алгоритма $\mathfrak{A}''$, представляющего из себя суперпозицию классификаторов, такую, что выполнены следующие условия:

- Для любого объекта $\mathfrak{s}_i$, принадлежащего любому классу K_j из множества заданных классов, структура классификаторов, построенная алгоритмом $\mathfrak{A}'$, распознает объект $\mathfrak{s}_i$ как объект класса K_j. Другими словами, выполняется требование полноты покрытия заданных классов объектов.

- Полученная структура классификаторов не ухудшает значения вероятности ошибок второго рода: $\mathfrak{P} \leq max\mathfrak{P}_{\mu_\text{i}}$.

- Полученная структура оптимизирует суммарную вычислительную сложность входящих в нее алгоритмов: $\mathfrak{C} \leq \sum_i \mathfrak{c}_{\mathfrak{A}_i}$, где $\{\mathfrak{A}_i\}$ - множество алгоритмов, входящих в множество классификаторов $\{\mu_i\}$.

В **разделе 1.4** предлагается подход к решению задачи распознавания, допускающей сбрасывание объектов. Подход заключается в следующем. Все классификаторы организаются в направленный ориентированный *граф* G, множество вершин $\{V\}$ соответствует классификаторам непосредственно, а множество ребер $\{E\}$ соответствует передаче данных между классификаторами (рисунок 1.2). Передача потоков данных осуществлялась только между классификаторами, пересечение обнаруживаемых классов объектов которых не пусто. Анализируемый поток данных, состоящий из множества объектов $\{\mathfrak{s}_j\}$ поступает одновременно на классификаторы, обнаруживающие различные классы объектов.

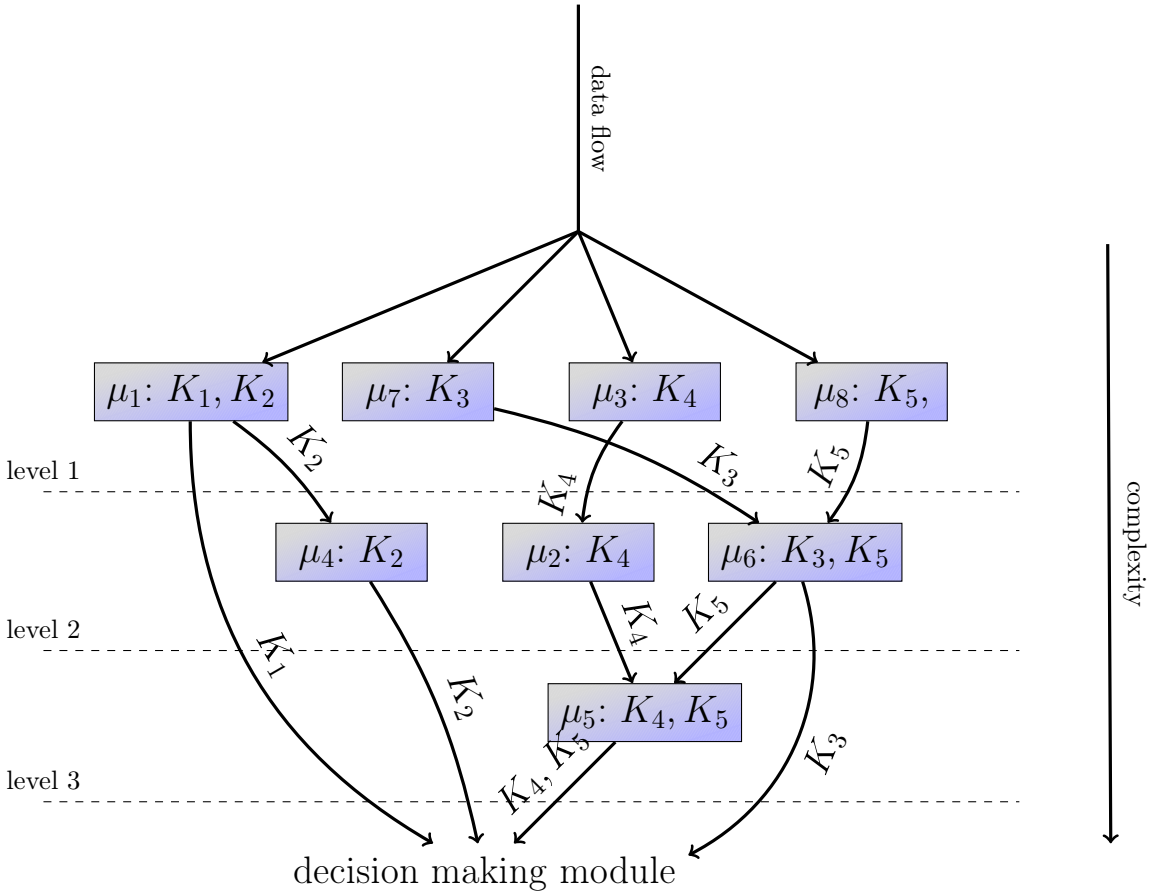

Рис. 1.2. Возможный пример топологии графа принятия решений.

Задача распознавания объектов сводится к задаче построения графа G классификаторов, обладающего следующими свойствами:

- Каждый уровень графа обеспечивает полное покрытие классов объектов, обнаруживаемых доступными для организации уровня классификаторов. Это значит, что если из множества классификаторов $\{\mu_i\}$ некоторое подмножество $\{\mu_j\}'$ составляет уровни $1 \dots j-1$, то для организации уровня j полнота покрытия классов должна обеспечиваться для классов, обнаруживаемых классификаторами из множества $\{\mu_i\} \backslash \{\mu_j\}'$;

- Ни один классификатор не встречается в графе более одного раза. Данное требование необходимо в связи с необходимостью оптимизации вычислительной сложности запуска графа G;

- Каждый уровень классификатора оптимален с точки зрения вычислительной сложности;

- Каждый уровень классификатора сохраняет отношение частичного порядка алгоритмов, входящих в классификаторы.

Далее в разделе приводится алгоритм построения графа G.

В **разделе 1.5** проводится анализ алгоритма построения графа G и приводится анализ представленного решения. Показано, что предложенное решение обеспечивает полное покрытие классов объектов. Доказано, что полученная структура классификаторов не ухудшает значения вероятности ошибок второго рода и что полученная структура оптимизирует суммарную вычислительную сложность входящих в нее алгоритмов. Таким образом показано, что построенный граф G удовлетворяет требованиям поставленной задачи.

Во **второй главе** исследуемая область ставится в соответствие построенной модели. В качестве объектов $\{\mathfrak{s}_j\}$ рассматривается набор исполнимых строк, полученных путем *дизассемблирования* байтовых строк. В качестве множества признаков $\{\mathfrak{f}_j\}_{j=1}^{k}$ рассматривается множество признаков, характерных для шеллкодов, а в качестве классов $\{K_1, ..., K_l\}$ объектов рассматриваются классы шеллкодов.

В **разделе 2.1** выделяются и подробно описываются признаки шеллкодов. Все представленные в разделе признаки выделены автором путем чтения научных статей, посвященных вредоносному исполнимому коду, а так же анализа исходных кодов (в случае его доступности) и изучения принципов их работы.

В разделе так же приводится классификация признаков по нескольким критериям: универсальность (является ли признак общим для всех шеллкодов или определяет только некоторое семейство шеллкодов); способ обнаружения признака (выявляет статическим или динамическим анализом кода)

и платформа, для которой данный признак актуален (x86, ARM).

В **разделе 2.2** впервые приводится классификация шеллкодов. Всего выделено 8 классов шеллкодов. базирующихся на признаках активатора; 3 класса, базирующихся на признаках декриптора; 6 классов, базирующихся на обфускации полезной нагрузки вредоносного объекта и 2 класса, базирующихся на зоне адресов возврата.

В **третьей главе** приводится обзор существующих методов обнаружения шеллкодов. В терминах введенной в главе 1 модели, существующие методы обнаружения шеллкодов представляют из себя классификаторы $\{\mu_i\}_{i=1}^{m}$, в состав которых входит один или несколько алгоритмов, обнаруживающих признаки шеллкодов во входных объектах.

В **разделе 3.1** описаны показатели эффективности, относительно которых производится обзор методов обнаружения шеллкодов:

- вероятность ошибок первого рода. *Ошибкой первого рода* считается принятие методом вредоносного объекта за легитимный. Другими словами, критерий оценивает точность обнаружения методами шеллкода;
- вероятность ошибок второго рода. *Ошибкой второго рода* считает принятие методом легитимного объекта за вредоносный. Таким образом, критерий оценивает вероятность ложных срабатываний методов;
- вычислительная сложность метода;
- покрытие классов шеллкодов.

В **разделе 3.2** приводится классификация существующим методов обнаружения шеллкодов по способу выполняемого анализа и по обнаруживаемым участкам шеллкодов. По способу выполняемого анализа методы обнаружения шеллкодов разделены на три класса - методы статического анализа, методы динамического анализа и методы гибридного анализа. К группе статического

анализа относятся методы, осуществляющие анализ кода без его непосредственного выполнения. К группе динамического анализа относятся методы, анализирующие код программы в процессе ее непосредственного выполнения. Гибридные методы используют комбинацию статических и динамических методов анализа кода программы. Подробно рассмотрены как достоинства, так и недостатки каждого из этих подходов.

В **разделе 3.3** приводится подробное описание существующих методов обнаружения шеллкодов и их сравнение.

В **разделе 3.4** приводятся результаты обзора. Резюмируется сравнение существующих методов обнаружения шеллкодов по вероятности ошибок первого и второго рода, по вычислительной сложности и по покрытию классов шеллкодов. Приводятся следующие выводы:

- Методы, обладающие невысокой вычислительной сложностью, характеризуются большим процентом ложных срабатываний.
- Методы, обладающие невысоким процентом ложных срабатываний, характеризуются высокой вычислительной сложностью.
- Ни один из существующих методов обнаружения вредоносных исполнимых инструкций не обеспечивает полного покрытия классов вредоносных инструкций.

Показывается актуальность разработки классификатора, обеспечивающего полное покрытие выделенных классов вредоносного исполнимого кода, обладающего меньшей вычислительной сложностью, чем линейная комбинация входящих в него методов и минимизирующего вероятность ложных срабатываний входящих в него методов.

В **четвертой главе** приводится описание разработанной инструментальной среды обнаружения шеллкодов, названной Demorpheus. Средство обнару-

жения шеллкодов Demorpheus поддерживает два основных режима работы: работа в режиме анализа сетевого трафика и работа в режиме анализа переданных системе на вход файлов. В случае, если средство запущено в первом режиме работы, Demorpheus считывает входные данные с указанного сетевого интерфейса непосредственно. Во втором случае, происходит обработка и анализ входного файла. Результатом работы средства является выделение из переданного объема входных данных объектов, носящих вредоносных характер, и их классификация.

В **разделе 4.1** приводится описание архитектуры Demorpheus. Общая схема архитектуры приведена на рисунке 4.3.

Рис. 4.3. Инструментальная среда Demorpheus.

Модуль *"Сбор сетевой сессии"* отвечает за восстановление потока данных из пакетов, считываемых с сетевого интерфейса. Модуль запускается только в том случае, если средство запущено в режиме анализа сетевого трафика.

Модуль *"Дизассемблирование входного потока"* преобразует входной байтовый поток в набор инструкций целевого процессора. Средство поддерживает три набора команд процессоров: набор команд платформы x86 и набор команд платформы ARM с возможностью переключения в режим Thumb.

Модуль *"Восстановление служебных структур"* преобразует набор инструкций ассемблера в некоторые служебные структуры, требуемые для работы средства: граф потока управления, граф потока инструкций, набор потоков, дизассемблированных с каждого смещения.

Модуль *"Библиотека шеллкодов"* содержит набор классификаторов. Некоторая часть классификаторов в библиотеке содержит по одному алгоритму, определяющему один из признаков шеллкодов, перечисленных в главе 2. Часть классификаторов представляет из себя декомпозированных алгоритмов обнаружения шеллкодов. Часть классификаторов в библиотеке являются существующими методами обнаружения шеллкодов. Каждый классификатор имеет универсальный интерфейс для запуска гибридным классификатором.

Модуль *"Гибридный классификатор"* осуществляет выполнение оптимальной топологии классификаторов, предоставляемых библиотекой шеллкодов. Кроме того, модуль поддерживает режим работы, в котором осуществляется единовременное построение оптимальной топологии. Построение топологии осуществляется по алгоритму, описанному в главе 1.

В **разделе 4.2** подробно рассмотрен каждый из компонентов системы и приведены соответствующие алгоритмы.

В **разделе 4.3** представлены результаты испытания прототипа. Тестирования прототипа проводились на четырех различных наборах данных:

1. Набор шеллкодов. Для апробации прототипа было использовано 1536 образцов шеллкодов, сгенерированных средством автоматического генерации вредоносного исполнимого кода Metasploit, а так же 42 образца

шеллкодов, доступных в публичном репозитории эксплойтов exploitdb. Средство тестировалось в режиме анализа трафика.

2. Набор легитимных бинарных файлов. В данном тестовом наборе было использовано 200 исполнимых PE Windows файлов и 200 исполнимых ELF-файлов unix-подобных операционных систем, содержащихся в директории `/usr/bin`. При тестировании на данном тестовом наборе, средство использовалось в режиме анализа файлов.

3. Набор случайных данных. Средство так же тестировалось на случайно сгенерированных данных. Для проведения тестов был использован тестовый набор, содержащих 300 Мб данных.

4. Набор мультимедийных данных. Для проведения тестов был использован тестовый набор, содержащих 300 Мб мультимедийных данных.

Набор данных	Л. структура FN	Г. структура FN	Л. структура FP	Г. структура FP
шеллкоды	0.2	0.204	n/a	n/a
легитимные	n/a	n/a	0.0064	0.019
случ. данные	n/a	n/a	0	0
мультимедия	n/a	n/a	0.005	0.04

Таблица 4.1. Результаты значений ошибок первого рода (FN) и второго рода (FP) для линейной и гибридной структур классификаторов.

При проведении экспериментов осуществлялось сравнение гибридного классификатора с линейной структурой классификаторов. Несмотря на то, что при построении гибридной структуры ошибки первого рода не учитывались, данный парамент важен для понимания, как выбор структуры влияет

Набор данных	Линейная структура	Гибридная структура
шеллкоды	6.9	11
легитимные программы	15	92.6
случайные данные	11	320
мультимедия	8	325

Таблица 4.2. Результаты значений пропускной способности на тестовой машине. Значение оценивается в Мб/сек.

на точность обнаружения шеллкодов. Результаты экспериментов для сравнения ошибок первого и второго рода, а так же пропускной способности на тестовой виртуальной машине, приведены в таблицах 4.1 и 4.2, соответственно. Для тех наборов данных, для которых вычисление ошибки первого или второго рода не имеет смысла, это значение принимает вид n/a.

Как видно из приведенных таблиц, очевиден компромисс между сложностью алгоритма, выраженной в данном случае пропускной способностью, и долей ложных срабатываний. Значение доли ложных срабатываний для гибридной структуры классификатора несколько выше, чем для линейной топологии, однако очевидно значительное преимущество гибридной структуры с точки зрения пропускной способности.

Сравнение результатов времени выполнения линейной и гибридной структур классификаторов для разных тестовых наборов приведены на рисунке 4.4. Сравнение относительного времени работы линейной структуры и гибридной структуры показывает, что гибридная структура в разы эффективнее.

В **пятой главе** (заключении) формулируются основные результаты работы и возможные направления исследования задачи обнаружения шеллкодов в высокоскоростных каналах передачи данных.

Приложение А содержит описание алгоритма восстановления графа

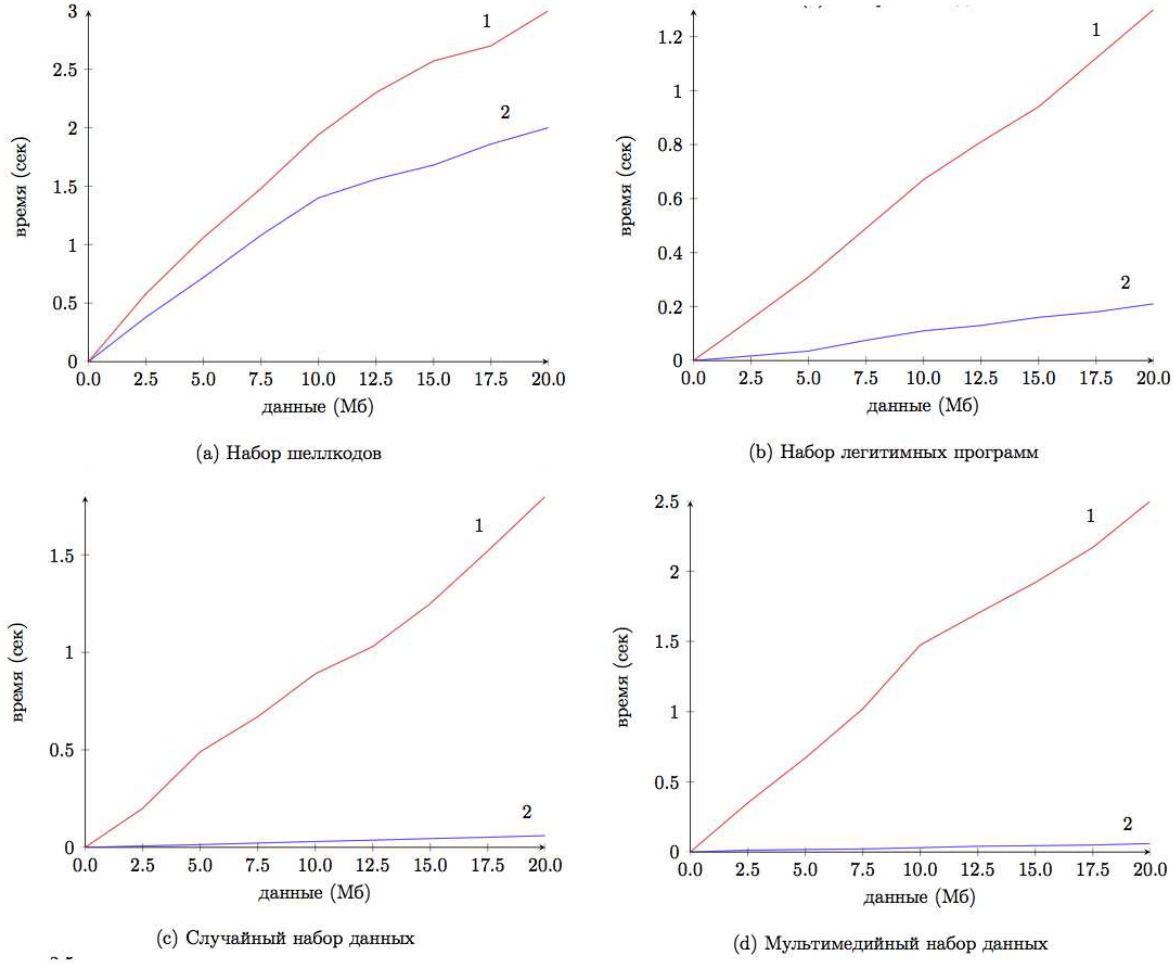

Рис. 4.4. Сравнение времени работы для линейной и гибридной топологии детекторов на вредоносном и легитимном наборе данных. Линия 1 соответсвует линейной топологии, 2 - гибридной.

потока инструкций (IFG) из набора дизассемблированных инструкций. Приводится анализ сложности представленного алгоритма.

Основные результаты работы.

1. Проведен анализ существующих шеллкодов и разработана их классификация, позволяющая полностью покрыть все известные образцы шеллкодов.

2. Предложена модель распознавания объектов, позволяющая обнаруживать в объектах набор специфичных признаков и распознавать объекты в соответствии с предложенной классификацией. В рамках представленной модели разработан алгоритм распознавания шеллкодов, позволяющий покрыть все известные классы шеллкодов; снизить вычислительную сложность обнаружения шеллкодов; минимизировать количество ложных срабатываний.

3. Разработанные алгоритмы классификации были реализованы и апробированы в рамках экспериментальной системы Demorpheus на четырех наборах данных: на наборе эксплойтов, на легитимных программах, на случайных и мультимедийных данных. Система продемонстрировала практически нулевое число ложных срабатываний, высокое значение пропускной способности по сравнению с линейной комбинацией существующих аналогов.

ПУБЛИКАЦИИ ПО ТЕМЕ ДИССЕРТАЦИИ

1. Гайворонская С. А. *Методы обнаружения вредоносного исполнимого кода в высокоскоростных каналах передачи данных.* // Системы высокой доступности. — 2011. — Т. 2, № 7. — С. 70–75.

2. Гайворонская С. А. *Гибридный метод обнаружения шеллкодов.* // Системы высокой доступности. — 2012. — Т. 2, № 8. — С. 33–44.

3. Гайворонская С. А., Гамаюнов Д. Ю. *Иерархическая топология декомпозированных алгоритмов для обнаружения вредоносного исполнимого кода* // Материалы XX Международной научной конференции студентов, аспирантов и молодых ученых Ломоносов 2013 [Электронный ресурс]. — Материалы международного молодежного научного форума

ЛОМОНОСОВ-2013. — МАКС Пресс Москва, 2013. — С. 10–13.

4. Gaivoronski S., Gamayunov D. *Hide and seek: Worms digging at the internet backbones and edges* // Proceedings of the 7th Spring/Summer Young Researchers' Colloquium on Software Engineering (SYRCoSE 2013). — National Research Technical University Kazan, Russia: Kazan, 2013. — P. 94–107.

5. Гайворонская С. А., Петров И. С. *Исследование средств автоматической генерации вредоносного исполнимого кода некоторого класса для мобильных платформ* // Программные системы и инструменты. Тематический сборник (2013). — Т. 14. — Изд-во факультета ВМиК МГУ, 2013. — С. 72–82.