10. Методика определения актуальных угроз безопасности персональных данных при их обработке в информационных системах персональных данных: утв. зам. дир. ФСТЭК России 14 февр. 2008 г. – М.: Аксимед, 2008. – 8 с.
11. Защита информации. Обеспечение безопасности сетей электросвязи. Общие положения: ГОСТ Р 52448-2005. – Введ. 2007-01-01. – М.: Стандартинформ, 2007. – 15 с.
12. Временное положение по организации разработки, изготовления и эксплуатации программных и технических средств защиты информации от несанкционированного доступа в автоматизированных системах и средствах вычислительной техники: Руководящий документ / Гос. техн. комиссия Рос. Федерации. – М.: ГТК РФ, 1992. – 29 с.
13. ГОСТ Р ИСО/МЭК ТО 15446-2008. Информационная технология. Методы и средства обеспечения безопасности. Руководство по разработке профилей защиты и заданий по безопасности. введ. 2009-10-01. – М.: Стандартинформ, 2010. – 107 с.
14. ГОСТ 34.003-90. Автоматизированные системы. Термины и определения. – Взамен ГОСТ 24.003-84, ГОСТ 22487-77; введ. 1992-01-01. – М., 1990. – 68 с.
15. Руководство по разработке профилей защиты и заданий по безопасности: Руководящий документ / Гос. техн. комиссия России. – М.: ГТК РФ, 2003. – 110 с.
16. ГОСТ Р 51275-2006. Защита информации. Объект информатизации. Факторы, воздействующие на информацию. Общие положения. – Взамен ГОСТ Р 51275-99; введ. 2008-02-01. – М.: Стандартинформ, 2007. – 11 с.

Статью рекомендовал к опубликованию д.т.н. профессор В.А. Терсков.

Стефаров Артем Павлович – Сибирский государственный аэрокосмический университет им. ак. М.Ф. Решетнева (СибГАУ); e-mail: Chameleo@mail.ru; 660014, г. Красноярск, пр. им. газеты «Красноярский рабочий», 31; тел.:+79135347358; кафедра САиИО; аспирант.

Жуков Вадим Геннадьевич – e-mail: vadimzhukov@mail.ru; тел.:+79029171966; кафедра безопасности информационных технологий; к.т.н.; доцент.

Stefarov Artem Pavlovich – Siberian state airspace university named after academician M.F. Reshetnev (SSAU); e-mail: Chameleo@mail.ru; 31, Krasnoyarsky Rabochy av., Krasnoyarsk, 660014, Russia; phone: +79135347358; the department of system analysis; postgraduate student.

Vadim Zhukov Genad'evich – e-mail: vadimzhukov@mail.ru; phone: +79029171966; the department of information technologies` security; cand. of eng. sc.; associate professor.

УДК 004.056

М.О. Шудрак, И.А. Лубкин, В.В. Золотарев

СТАТИЧЕСКИЙ АНАЛИЗ БИНАРНОГО КОДА В СФЕРЕ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

Рассматривается методика декомпиляции бинарного кода и возможность ее применения в сфере информационной безопасности. Основная цель работы заключается в разработке эффективного алгоритма анализа бинарного кода. Для достижения поставленной цели необходимо решить ряд задач: разработать эффективный механизм анализа низкоуровневых команд, их алгоритмического представления и провести апробацию полученной методики. Результатом работы стала эффективная методика декомпиляции и алгоритмического представления линейных участков бинарного кода, апробированная на решении таких задач как: защита программного обеспечения от несанкционированного анализа и анализе обфусцированного кода вредоносных объектов.

Декомпиляция; анализ кода; защита программного обеспечения; полиморфный код.

M.O. Shudrak, I.A. Lubkin, V.V. Zolotarev

STATIC BINARY CODE ANALYSIS IN INFORMATION SECURITY SPHERE

The article describes about technique of binary code decompilation and its application possibility in information security sphere. The main object of this work is to develop an efficient algorithm for the binary code analyze. To achieve this goal it is necessary to solve a number of tasks: to develop an effective mechanism for the low-level command analyze with algorithmic representations and conduct testing of the resulting methods. The result was an effective method of reverse engineering and algorithmic representations of binary code linear plots, tested with such tasks as: software protection against unauthorized analysis and analysis of obfuscated code threats.

Decompilation; code analyze; software security; obfuscated code.

Введение. На сегодняшний день применение декомпиляция бинарного кода в сфере информационной безопасности ограничено, в виду наличия в этом процессе ряда фундаментальных проблем.

Анализ бинарного кода является не тривиальной задачей, в силу того что в процессе компиляции исходного кода программы теряется, важная с точки зрения анализа, информация [1]. Таким образом, полученная в ходе декомпиляции информация, часто не представляет практического интереса.

Однако декомпиляция занимает важное место в сфере информационной безопасности и может быть успешно использована в следующих областях:

- защита программного обеспечения;
- анализ вредоносных объектов;
- поиск уязвимостей.

В данной статье, авторы предлагают новый подход к декомпиляции бинарного кода и его применение к областям, которые описаны выше.

Декомпиляция кода. Процесс анализа бинарного кода требует некоторого уровня его абстракции от машинного кода к человеку [2]. Таким образом, основная идея описываемой методики заключается в том, что декомпиляция бинарного кода не требует представления машинного кода на языке высокого уровня, достаточно восстановить алгоритм функционирования участка анализируемого кода (тем самым добиться соответствующего уровня абстракции).

Методика интерпретации инструкций, предложенная авторами данной статьи, рассматривает инструкции как совокупность элементарных операций, над ресурсами, где в качестве ресурса могут выступать регистры, память или регистры флагов процессора. Рассмотрим алгоритм работы декомпилятора подробно на рис. 1.

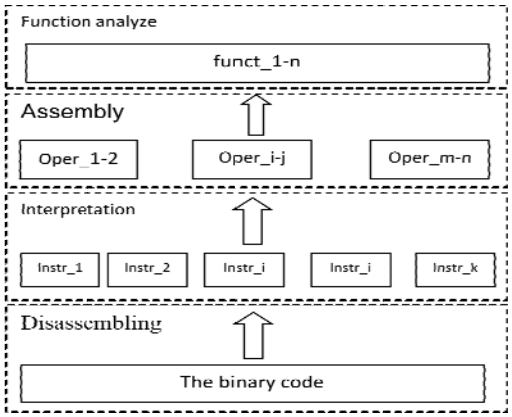

Рис. 1. Алгоритм анализа кода

Все операции в рамках методики были сгруппированы в общий базис, описывающий все множество возможных инструкций (табл. 1).

Таблица 1

Описание операций

Операция	Описание
+ , –	Сложение и вычитание
*	Умножение
. /.	Целочисленное деление
%	Взятие остатка от деления
and	Логическое «и»
or	Логическое «или»
xor	Логический «хог»
<<	Циклический сдвиг влево
>>	Циклический сдвиг вправо
[]	Доступ к отдельным битам
[<–	Запись в память по определенному адресу
<]	Чтение из памяти по определенному адресу
NOT	Логическое «НЕ»
«Transit»	Пересылка информации от одного ресурса к другому
«Const»	Обозначение констант
«Special».	Обозначение специальных инструкций

Для отображения информации об инструкции предлагается следующая схема: ресурс, соответствующий регистру, обозначается фигурными скобками, в которые заключён его номер. Ресурс оперативной памяти обозначается квадратными скобками, в которые заключён адрес ресурса. Номер такого ресурса выносится за скобки. Результат отделяется знаком «=». Операции и константы записываются «как есть». Представленная форма записи позволяет представлять инструкции в виде математических формул.

Тем самым наглядно показывая процесс преобразования информации для участка машинного кода.

За этапом интерпретации, согласно предлагаемому алгоритму декомпиляции следует этап сборки, идея такого процесса состоит в том, что ресурс–источник одной операции может выступать в роли приёмника в другой операции, таким образом, устанавливая связь между операциями на участке кода.

Развивая методику сборки, для установки связи между операциями было предложено использовать аппарат синтаксических деревьев для описания преобразований над ресурсами. В таком дереве узлы представляют собой операторы из базового набора операций, описанного выше, а листья соответствуют ресурсам или константам.

Таким образом, для формирования итогового выражения необходимо осуществить рекурсивный проход по дереву вида: {левый лист, узел, правый лист}. Выход из узла дерева, формирует скобки в итоговом выражении. Такие деревья необходимо формировать для каждого ресурса являющегося выходным.

Для определения выходных ресурсов можно использовать несколько утверждений:

1) Выходным называется тот ресурс, который не является промежуточным.

2) В качестве выходных ресурсов можно рассматривать ресурсы памяти (в том числе стек программы), регистры общего назначения, так как чаще всего он является выходным ресурсом при вызове функций.

В процессе полиморфной генерации бинарного кода, встает задача учета ресурсов процессора, для обеспечения неизменности алгоритма работы участка кода, на котором производится генерация.

Для учета ресурсов процессора необходимо учитывать не только регистры и флаги процессора, но и регистры, которые используются для адресации памяти (например, mov [eax], 3) так как изменения регистров, используемых для адресации в памяти, в некоторых случаях, может привести к нарушению алгоритма работы участка кода.

Таким образом, для учета ресурсов процессора, необходимо учитывать их изменения после каждой инструкции.

Для учета ресурсов процессора используется битовая маска регистров общего назначения, таким образом, для генерации кода, возможно, использовать только те регистры, чей флаг не взведен. Также стоит отметить, что регистры, чей флаг взведен, нельзя использовать в качестве приемника информации, в качестве источника информации для других регистров их использование оправдано, так как такая пересылка информации не вносит изменений в алгоритм работы участка кода.

Апробация методики. Апробация вышеописанной методики проводилась на двух следующих задачах:

1) защита участка программного кода;
2) автоматизированный анализ полиморфного кода.

А. Защита участка программного кода. На практике, любая атака на программное обеспечение, основывается на предварительном анализе защитного механизма приложения, зачастую с использованием дизассемблирования бинарного кода. Цель такого исследования заключается в том, чтобы восстановить алгоритм защиты, выделить его слабые стороны или недокументированные возможности, для его последующей модификации и (или) автоматизации процесса преодоления. Такое преодоление гарантированно достигается за конечное время вследствие конечности программы, а время, за которое злоумышленник преодолеет систему, зависит от сложности этой системы и его квалификации. Способствует такой ситуации широкий выбор инструментов для исследования программного кода.

Технология полиморфной генерации кода позволяет проводить запутывающие преобразование в бинарном коде защищаемого объекта. В рамках данной работы производиться встраивание (вставка запутывающих инструкций между инструкциями защищаемого кода) запутывающего кода в бинарный код защищаемого объекта. Такое встраивания позволяет увеличить время, за которое злоумышленник восстановит алгоритм защиты, за счет усложнения исследуемого кода (рис. 3).

Таким образом, согласно рис. 3, для встраивания полиморфной инструкции необходимо еще несколько дополнительных этапов, таких как генерация инструкции и её компиляция.

Стоит отметить, что реализация этапа интерпретации не требует анализа всего множества инструкций архитектуры x86. Это обусловлено тем, что компиляторы используют не весь набор инструкций данной архитектуры.

Методика генерации кода (описанная ранее) была реализована в виде программного средства, которое включает в себя ряд базовых модулей, таких как: анализатор кода и генератор полиморфного кода, а также базу данных содержащую набор инструкций и соответствующий им алгоритм (рис. 5).

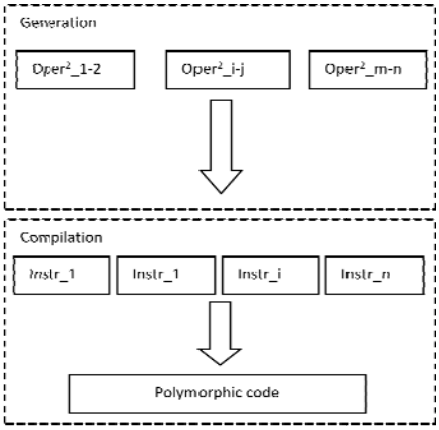

Рис. 3. Алгоритм генерации полиморфного кода

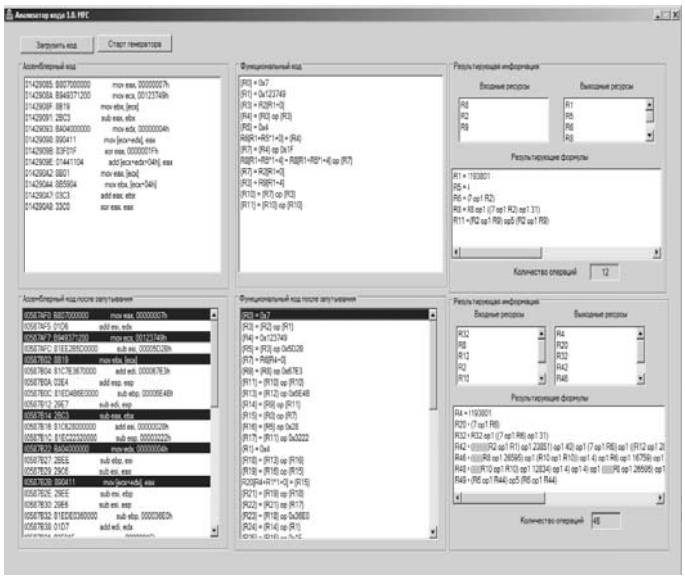

Рис. 4. Генератор полиморфного кода

Полученные результаты позволяют говорить об эффективности предложенной методики, количество операций на участке кода, количество входных и выходных ресурсов, а также результирующих формул возросло, что непременно приведет к росту ресурсов требуемых для исследования данного участка кода, что повышает защищенность описываемого участка от исследования.

В. Автоматизированный анализ полиморфного кода. Второй вариант апробации предложенной методики заключается в автоматизации анализа секций полиморфного кода вируса.

Полиморфный вирусный код, это код который модифицирует сам себя (свое бинарное представление), при этом алгоритм выполняемый участком кода остается неизменным. Такой алгоритм представляет собой сигнатуру полиморфного вируса, и может быть использоваться для обнаружения вируса.

Необходимо отметить, что для обнаружения полиморфного кода необходимо руководствовать следующим утверждением: полиморфный код это код, который не имеет выходных ресурсов и не участвует в их формировании.

Таким образом, используя вышеописанную методику декомпиляции бинарного кода, а также утверждение о полиморфном коде, была реализована программа, позволяющая детектировать полиморфные инструкции вирусного кода в автоматизированном режиме.

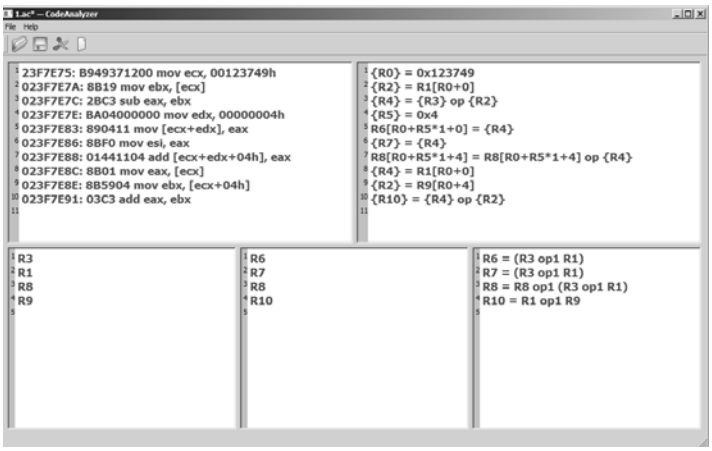

Рис. 5. Анализатор полиморфного кода вируса

Описанное средство позволяет детектировать полиморфные инструкции без ручного (с использованием средств дизассемблирования) анализа вирусного кода. Однако необходимо отметить, что на данный момент, описанное детектирование возможно лишь на линейных участках полиморфного кода (участок кода, без использования ветвления и вызова подпрограмм). Работа поддержана в рамках гранта федеральной целевой программы "Научные и научно-педагогические кадры инновационной России" на 2009–2013 гг.

Заключение. Таким образом, в данной статье были описана методика декомпиляции бинарного кода посредством применения оригинального алгоритма представления инструкции в алгоритмическом виде.

Необходимо отметить, что описанная методика декомпиляции может найти широкое применение в различных аспектах информационной безопасности, особенно в разделах, касающихся защиты программного обеспечения и анализа вредоносного кода, что и было продемонстрировано в предыдущих разделах.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Бушцев Д.Н.* Метод защиты программных средств на основе запутывающих преобразований: [Электронный ресурс]: Дис. … канд. техн. наук. – Томск: РГБ, 2006 (Из фондов Российской Государственной Библиотеки).
2. *Золотарев В.В.* Метод исследования программных средств защиты информации на основе компонентной модели информационной среды // Известия ЮФУ. Технические науки. – 2008. – № 8 (85). – С. 87-94.
3. *Кукарцев А.М., Лубкин И.А.* Методика защиты программного кода от несанкционированной модификации и исследования посредством его хеширования // Вестник Сибирского государственного аэрокосмического университета. – 2008. – Вып. 1. – С. 56-60.

Статью рекомендовал к опубликованию д.ф.-м.н., профессор М.В. Носков.

Шудрак Максим Олегович – Сибирский государственный аэрокосмический университет им. М.Ф. Решетнева (СибГАУ); e-mail: mxmssh@gmail.com; 660014, г. Красноярск, пр. им. газеты «Красноярский рабочий», 31; тел.: 89233088703; кафедра безопасности информационных технологий; аспирант.

Лубкин Иван Александрович – e-mail: lubkin@rambler.ru; тел.: 83912328627; кафедра безопасности информационных технологий; аспирант.

Золотарев Вячеслав Владимирович – e-mail: amida@land.ru; тел.: 89050874847; кафедра безопасности информационных технологий; к.т.н.; доцент.

Shudrak Maxim Olegovich – Siberian Aerospace State University named after M.F. Reshetnev (SibSAU); e-mail: mxmssh@gmail.com; 31, Krasnoyarsky Rabochy av., Krasnoyarsk, 660014, Russia; phone: +79233088703; the department of information technology security; postgraduate student.

Lubkin Ivan Alexandrovich – e-mail: lubkin@rambler.ru; phone: +73912328627; the department of information technology security; postgraduate student.

Zolotarev Vyacheslav Vladimirovich – e-mail: amida@land.ru; phone: +79050874847; the department of information technology security; cand. of eng. sc.; associated professor.

УДК 004.056

А.А. Таран

ПРИЛОЖЕНИЯ АЛГОРИТМА ANTMINER+ К ЗАДАЧЕ КЛАССИФИКАЦИИ СОБЫТИЙ ПРИ АНАЛИЗЕ СЕТЕВОГО ТРАФИКА

Статья посвящена исследованию технологий классификации данных с помощью алгоритма AntMiner+ и в частности их приложениям к обнаружению вторжений при исследовании сетевого трафика. Особое внимание при этом уделено свойствам алгоритма, позволяющим автоматически получать понятные, читаемые и явно связанные с предметной областью задачи описания исследуемых множеств. Сделан вывод о применимости рассматриваемого алгоритма к задачам автоматической генерации сигнатур атак и профилей нормального поведения системы. Описаны результаты экспериментов, подтверждающие выдвинутые гипотезы о свойствах алгоритма.

Обнаружение вторжений; классификация данных; анализ сетевого трафика; AntMiner+; аномалии; сигнатуры; извлечение правил.

A.A. Taran

APPLICATION OF ANTMINER+ ALGORITHM TO EVENT CLASSIFICATION FOR NETWORK TRAFFIC ANALYSIS

The article deals with the technologies of data classification based on the algorithm AntMiner+ in the analysis of the network traffic for intrusion detection. Special attention is paid to the properties of the algorithm which allow us automatically receive understandable, human readable and strongly related with the problem under consideration and describing sets. Some conclusion about applicability of the methods to the tasks of automated generation of attack's signatures and profiles of system's normal behavior are made. The paper also explores the results of experiments which confirm the hypothesis about algorithm's properties.

Anomaly detection; data classification; network traffic analysis; AntMiner+; signatures; rule conduction.

Введение. Системы обнаружения вторжений (СОВ) наряду с криптографическими средствами, как правило, составляют основу организации противодействия угрозам несанкционированного доступа к данным в компьютерной системе, осо-