УДК 004.492.3

В.М. РУВИНСКАЯ, А.В. МОЛДАВСКАЯ, М.И. ГРИГОРЕНКО
Одесский национальный политехнический университет

ИНФОРМАЦИОННАЯ ТЕХНОЛОГИЯ ДЕТЕКТИРОВАНИЯ ВРЕДОНОСНЫХ ПРОГРАММ НА ОСНОВЕ СЦЕНАРИЕВ

Статья посвящена описанию информационной технологии детектирования вредоносных программ на основе сценариев. Показана востребованность такой технологии в области антивирусного анализа. Описана подсистема автоматического перехвата функций, вызываемых вредоносной программой в операционной системе, функциональные требования к ней и основной метод её работы.

Ключевые слова: вредоносные программы, поведенческий анализ, сценарии, перехват функций.

V.M. RUVINSKAYA, A.V. MOLDAVSKAYA, M.I. GRIGORENKO
Odessa National Polytechnic University

INFORMATIONAL TECHNOLOGY OF DETECTING MALICIOUS SOFTWARE BASED ON SCRIPTS

Abstract – This paper describes an informational technology for detecting malicious software based on scripts. We use scripts to represent the relations between separate malware actions and to make them easier for understanding. The demand for such technology in the field of malware analysis is demonstrated. The described technology includes creating malware behaviour scripts with machine learning, making datasets from observing program behaviour to use them for machine learning and in the detection process, saving data and loading it from outer resources, and , finally, making inference. A subsystem for automated malware-invoken Windows API functions hooking is described, together with its main working method, conceptual class structure and functional requirements. We propose to use splicing for hooking the API and describe a method for doing automated splicing using generated hooking functions. In the end, we give a brief explanation of an object-oriented design for data collection and data storing.

Keywords: malicious software, malware, behaviour analysis, scripts, hooking, splicing

Введение

Вредоносное программное обеспечение с каждым годом распространяется во всё больших объёмах. По исследованиям антивирусных компаний ежегодно количество новых угроз, с которыми сталкиваются аналитики, возрастает почти вдвое [1]. Создатели вредоносных программ учитывают существующие методы анализа и создают новые технологии внедрения и сокрытия угроз, что затрудняет обнаружение. Исходя из этого, актуальной является разработка системы анализа, позволяющей ускорить обработку антивирусными компаниями новых штаммов вредоносных программ. Наиболее перспективным при этом является направление поведенческого анализа, так как вредоносное поведение большинства новых угроз остаётся типичным, несмотря на изменения и усложнения, вносимые авторами в исходный код вредоносных программ [2].

На сегодняшний день существует ряд средств, частично автоматизирующих процесс работы аналитиков: в основном это среды эмуляции, генерирующие отчёты о поведении. Существует ряд проблем, связанных с использованием этих сред, основные из которых следующие: адаптированность под одну либо ограниченное количество операционных систем, даже принадлежащих одному семейству; наличие у ряда вредоносных программ защиты от отладки; генерируемый отчёт требуется проанализировать вручную [3].

Основные исследования в этом направлении ведутся в области классификации вредоносного поведения по векторам признаков, есть разработки, использующие ассоциативные правила [4–6]. Однако при использовании правил и векторов не учитывается порядок вызова действий, вектора не позволяют представлять многовариантное поведение одного и того же класса вредоносных программ, отображать и анализировать причинно-следственные связи между действиями, объяснять полученный результат.

Работа посвящена обнаружению вредоносных программ с помощью системы автоматизированного распознавания вредоносного поведения. Аппарат распознавания предлагается построить на основе сценариев, что было ранее предложено в [7]. На данный момент область поведенческого анализа вредоносных программ изучена значительно слабее, нежели область статического анализа, то есть анализа кода [3]. При использовании представления в виде сценариев, в отличие от ранее разработанных решений, анализируются не только отдельные потенциально вредоносные действия или их наборы, а и последовательности таких действий с иерархической структурой и правилами взаимосвязи, целостно описывающие потенциально вредоносные образцы поведения. Особенность предлагаемого подхода состоит в том, что сценарии строятся на основе экземпляров существующих вредоносных программ, то есть предлагается использовать методы машинного обучения для конструирования сценариев.

Целью данной работы является снижение объёма ручного труда при обработке аналитиками новых штаммов вредоносных программ с помощью системы на основе сценариев, позволяющей осуществить как автоматизированное накопление и обновление сценариев, так и проведение логического вывода для распознавания вредоносного поведения. В связи с этим поставлены следующие **задачи**:

- описать информационную технологию детектирования вредоносных программ на основе сценариев;
- провести сравнительный анализ существующих средств получения информации о программном поведении;
- определить способы получения информации о поведении программ в операционной системе и описать подсистему сбора данных о поведении, которые используются для проведения машинного обучения.

1 Информационная технология поведенческого детектирования вредоносных программ на основе сценариев

Технология детектирования вредоносных программ на основе сценариев состоит из двух этапов:

1. Автоматизированное формирование сценариев, позволяющее:

1.1. Извлекать из примеров вредоносных программ события, происходящие при их работе, либо получать такого рода информацию из сторонних источников и хранить ее в БД.

1.2. Формировать сценарии:

- вводить сценарии вручную;
- получать выборку примеров, исходя из поведения обучающих образцов в системе, и на её основе получать сценарии автоматически (машинное обучение);
- корректировать введенные вручную сценарии с помощью обучения на основе примеров;
- хранить полученные сценарии на основе выбранных структур данных и, соответственно, записывать их и считывать;
- просматривать и редактировать сценарии.

В результате будут получены сценарии во внутреннем, определённом при создании технологии представлении, т.е. в виде выбранной структуры данных. Такой структурой данных могут быть, например, конечные автоматы либо байесовские сети [8].

2. Распознавание вредоносных программ, включающее:

- Получение данных о поведении распознаваемой программы.
- Сопоставление данных о поведении со сценариями базы и принятие решения на основе сопоставления.

В зависимости от знаний, заложенных в сценарии, система распознавания может выдавать двоичный результат (программа является или не является вредоносной), один из нескольких вариантов в качестве результата (например, выдавать название категории вредоносных программ: вирус, червь, рекламная программа и т.п.), вероятностный результат (программа вредоносна с определённой степенью достоверности) и т.п.

Технология детектирования вредоносных программ показана на рис.1. Отметим, что на схеме отображены также этапы, требующие вмешательства пользователя.

На первом этапе пользователем является эксперт-вирусный аналитик, задачей которого является формирование сценариев. На этапе 1.1 при работе подсистема сбора информации о вредоносных программах пользователь вводит в БД событий информацию о вредоносных программах, а также подает вредоносные программы на вход анализатору, который анализирует их поведение и заносит результат в БД последовательность событий, происходящих при работе программы. Также предусмотрена возможность получать события из других источников, к примеру, это могут быть данные из автономных программ-анализаторов событий в системе, открытая статистика, собираемая антивирусными сообществами автоматически или вручную и т.д. На этапе 1.2 предусмотрено интерактивное формирование сценариев для доступа эксперта к БЗ сценариев с возможностью просматривать, редактировать, сохранять и загружать сценарии.

На втором этапе информационной технологии с системой работает либо вирусный аналитик, либо конечный пользователь, их задачей является детектирование новых вредоносных программ. На входе – исследуемая программа и БЗ сценариев.

2 Подсистема сбора данных о вредоносных программах и их поведении (VA)

2.1 Общие сведения

В данной работе предлагается программная подсистема сбора и хранения данных о вредоносных программах, реализующая этап 2.1 описанной выше технологии поведенческого детектирования вредоносных программ, состоящая их двух частей:

- анализатор поведения Virus Analysis (VA), в качестве результата анализа подсистема выдаёт подробную информацию о вызываемых во вредоносной программе API-функциях и событиях, показывает подробный отчет, который содержит имя функции, ее параметры и их значения, время вызова, возвращаемое значение;
- управляющая программа поддержки работы с БД, содержащей описания вредоносных программ;
- набор программ, конвертирующих информацию о вредоносных программ из различных сторонних источников в БД.

Как уже упоминалось, системы генерации отчётов о поведении программы являются на сегодняшний день основным инструментом, частично автоматизирующим процесс анализа вредоносных программ. Большинство из существующих программных продуктов не подходят для полноценного динамического анализа вредоносного ПО, так как практически все анализаторы используют отладчики для получения необходимых данных, а вредоносные программы обычно защищены от средств отладки. Предлагаемая подсистема применяет другую технологию, описанную ниже. Кроме того, предоставляется графический пользовательский интерфейсом для хранения и предоставления результатов анализа в удобной форме. Подсистема работает с локальной базой данных, что наиболее важно для интеграции в систему поведенческого анализа. Предусмотрена возможность вручную добавлять и редактировать данные о вредоносном ПО.

Рис.1. Детализация информационной технологии детектирования вредоносных программ

В таблице 1 рассмотрены некоторые существующие приложения, являющиеся анализаторами вызываемых в программе API-функций и использующие различные технологии их перехвата для сбора информации о поведении программ [3, 9–11]; производится сравнение приложений между собой и с предлагаемой подсистемой VA.

Таблица 1

Сравнение средств перехвата по их функционалу

Функции	Средства перехвата			
	ZeroWine	Malpimp	CWSandbox	VA
Анализ файлов	+	+	+	+
Получить данные из источника	-	-	+	+
Просмотр данных	+	+	+	+
Добавление данных	-	-	-	+
Редактирование данных	-	-	-	+
Ручное добавление событий	-	-	-	+

2.2 Утилита VA

Наиболее удобным и точным методом получения данных об вызываемых в программе функциях является перехват этих функций. Существует множество различных способов перехвата:

1. Перехват через таблицу импорта процесса (IAT).
2. Перехват через таблицы виртуальных функций.
3. Перехват, основанный на создании функции-трамплина в теле целевой функции до кода обработчика (т.н. сплайсинг).

В разработанной подсистеме был использован метод перехвата, основанный на создании трамплина [12]. Алгоритм работы данного метода при ручном анализе следующий:

1. Создается обработчик, который выполняет нужные действия с перехваченной функцией. Например: выводит имя функции, ее параметры и время вызова).
2. В теле целевой функции с помощью дизассемблера или отладчика ищется подходящее место для записи команды перехода (JMP) на обработчик. При этом некоторые операции целевой функции перезаписываются, поэтому их нужно воспроизвести в теле обработчика по завершению его работы.
3. В теле обработчика, в конце, записывается команда перехода обратно в тело целевой функции на следующие, после перезаписанных в п. 2, команды.
4. Целевая функция продолжает свое нормальное выполнение.

Для выполнения данных действий необходим доступ к адресному пространству целевого процесса. Наиболее удобным в данном случае является использование динамически подключаемых библиотек DLL.

Данный метод является универсальным, однако очень трудоемким, если возникает необходимость перехвата большого количества API функций, т.к. нужно вручную для каждой функции найти подходящее место и перезаписать команды, а затем воспроизвести их. Предлагается следующий принцип автоматизации: благодаря используемым стандартам компиляторов программ, в большинстве API функциях присутствует одинаковая часть – пролог. К примеру, в ОС семейства Windows пролог выглядит следующим образом:

```
MOV EDI, EDI
PUSH EBP
MOV EBP, ESP
```

Общий размер данных операций составляет 5 байт, что соответствует размеру команды перехода JMP. Таким образом, можно значительно сократить необходимую трудоемкость для создания обработчиков перехваченных API-функций, т.е. обеспечить их автоматическую генерацию, что и было реализовано в VA.

2.3 Описание функциональных требований

К подсистеме VA были поставлены следующие функциональные требования:

1. Система должна производить анализ вредоносных файлов, выбранных пользователем.
2. Система должна хранить данные, полученные в результате анализа.
3. Система должна предоставлять удобное средство для просмотра хранимых данных.
4. Система должна предоставлять возможность ручного добавления и редактирования данных о вредоносном ПО.
5. Необходимо предусмотреть возможность работы с другими типами источников.

На рис.2 приведена диаграмма концептуальных классов для подсистемы. Можно заметить, что система условно разбивается на три абстрактные логические части: контроль и управление данными (контроллер, менеджер БД), графический интерфейс и, собственно, сами данные (сущности источники, событие, вредоносное ПО). Система использует множество различных запросов к БД, поэтому необходимо реализовать удобный инструментарий для работы с базой данных. Для этого хорошо подойдет паттерн проектирования TableDataGateway [13], в соответствии с которым реализация запросов выносится в отдельные классы для каждой таблицы базы данных, и SQL-коды запросов хранятся только в этих классах.

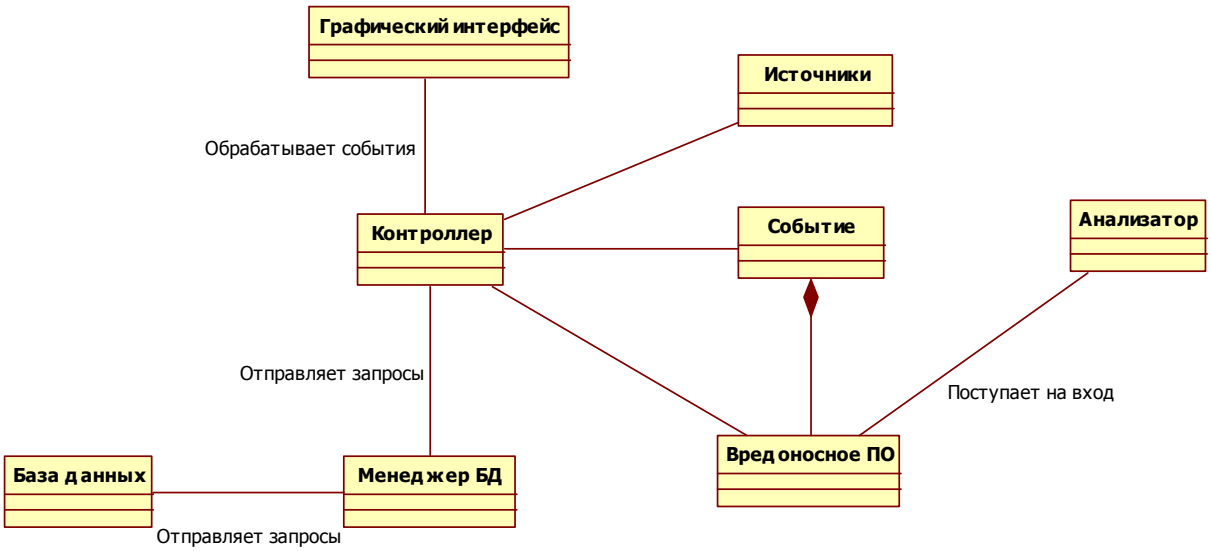

Рис.2. UML-диаграмма концептуальных классов подсистема сбора данных о вредоносных ПО

Выводы

В данной работе предложена и описана *информационная технология поведенческого детектирования вредоносных программ*, основанная на использовании сценариев. Особенностью предлагаемого подхода является то, что сценарии строятся на основе экземпляров существующих вредоносных программ и их поведении, то есть предлагается использовать методы машинного обучения для конструирования сценариев, а для их ведения и хранения применяется БЗ.

Для сбора данных о программном поведении разработана *подсистема VA*. Перехват событий, происходящих при работе вредоносных программ, производится с помощью создания функции-трамплина. Предложен принцип автоматизации данного метода для операционных систем Windows. Для хранения полученных результатов была спроектирована база данных и система для удобного ведения этой базы, которая также обладает возможностью извлечения данных из сторонних источников.

Литература

1. Malware AV-TEST – The independent IT-security institute [Електронний ресурс]. – Режим доступу : http://www.av-test.org/en/statistics/malware/.

2. Лисенко С. М. Дослідження методу опорних векторів як засобу ідентифікації шкідливого програмного забезпечення / С. М. Лисенко, М. П. Южека // Вісник Хмельницького національного університету. Технічні науки. – 2013. – № 6. – С. 194–201.

3. Egele M. Survey on Automated Dynamic Malware-Analysis Techniques and Tools / Egele M., Scholte T., Kirda E., Kruegel C. // Journal in ACM Computing Surveys. – 2012. – № 44. – P. 1–49.

4. Learning and Classification of Malware Behavior / Konrad Rieck, Thorsten Holz, Carsten Willems, Patrick Düssel, Pavel Laskov // Lecture Notes in Computer Science. – 2008. – № 5137. – P. 108–125.

5. Маслова Н.А. О применении интеллектуального анализа данных для защиты информации корпоративных систем / Н.А. Маслова // Искусственный интеллект. – 2009. – № 4. – С. 66–74.

6. Ye Y. An intelligent PE-malware detection system based on association mining / Ye Y., Wang D., Li T., Ye D., Jiang Q. // Journal in Computer Virology. – 2008. – № 4. – P. 323–334.

7. Рувинская В. М. Эвристические методы детектирования вредоносных программ на основе сценариев / В. М. Рувинская, Е. Л. Беркович, А. А. Лотоцкий // Искусств. интеллект . – 2008. – № 3. – С. 197–207.

8. Молдавская А.В. Применение методов машинного обучения для формирования сценариев поведения вредоносных программ / А.В. Молдавская, В.М. Рувинская // Информатика и математические методы в моделировании. – 2014. – № 4(2). – С. 149–157.

9. Malik. Malpimp – Advanced API Tracing Tool [Електронний ресурс]. – Режим доступу : http://securityxploded.com/malpimp.php.

10. Zero Wine: Malware Behavior Analysis. [Електронний ресурс]. – Режим доступу : http://zerowine.sourceforge.net/.

11. Willems, T. Holz, and F. Freiling. Toward Automated Dynamic Malware Analysis Using CWSandbox / Willems C., Holz T., Freiling F. IEEE Security and Privacy. – 2007. – № 5(2). – P. 32–39. – ISSN: 1540-7993.

12. Abramov A. API Hooking with MS Detours [Електронний ресурс]. – Режим доступу : http://www.codeproject.com/KB/DLL/funapihook.aspx. – 2008.

13. Фаулер М., Райс Д., Фоммела М. Архитектура корпоративных программных приложений. – М. : Вильямс, 2004. – 404 с. – ISBN: 0-321-12742-0.

References

1. “Malware AV-TEST – The independent IT-security institute.” (20.06.2014) URL: http://www.av-test.org/en/statistics/malware/

2. Ly`senko S.M., Iuzheka M.P. Doslidzhennia metodu oporny`h vektorov iak zasobu identy`fikacii shkidly`vogo programnogo zabezpechennia// Herald of Khmelnytsky National University, 2013, Vol. 6, pp.194–201.

3. Egele, M., Scholte, T., Kirda, E., Kruegel, C. Survey on Automated Dynamic Malware-Analysis Techniques and Tools. *Journal in ACM Computing Surveys*, 2012, Vol. 44, pp.1-49.

4. Konrad Rieck, Thorsten Holz, Carsten Willems, Patrick Düssel, Pavel Laskov. Learning and Classification of Malware Behavior, *Lecture Notes in Computer Science,* 2008, Vol. 5137, pp.108-125.

5. Maslova, N.A. O primenenii intellektualnogo analiza danny`h dl`a zashity` informacii korporativny`h sistem. *Iskusstvennyi intellekt*, 2009, No.4, pp. 66-74.

6. Ye Y. An intelligent PE-malware detection system based on association mining, *Journal in Computer Virology*, 2008, No.4, pp.323-334.

7. Ruvinskaia V.M., Berkovich E.L., Lototskii A.A. Evristicheskie metody` detektirovaniia vredonosny`h programm na osnove scenariiev, *Iskusstvennyi intellekt,* 2008, No.3, pp.197-207.

8. Moldavskaia A.V., Ruvinskaia V.M. Primeneniie metodov mashinnogo obucheniia dlia formirovaniia scenariiev, *Informatika i matematicheskiie metody` v modelirovanii*, 2014, Volume 4, No.2, pp. 149-157.

9. Malik A. “Malpimp - Advanced API Tracing Tool” (20.06.2014). URL: http://securityxploded.com/malpimp.php

10. “Zero Wine: Malware Behavior Analysis.” (20.06.2014). URL: http://zerowine.sourceforge.net/

11. Willems C., Holz T., Freiling F. Toward Automated Dynamic Malware Analysis Using CWSandbox. *IEEE Security and Privacy*, 2007, Voume 5, No.2, pp.32-39.

12. Abramov A. “API Hooking with MS Detours” (20.06.2014). URL: http://www.codeproject.com/KB/DLL/funapihook.aspx

13. Fauler M., Rays, D., Fomella, M. Arhitektura korporativny`h programmny`h prilozhenii. Moscow, Vil'iams, 404 p.

Рецензія/Peer review : 7.8.2014 р. Надрукована/Printed :1.10.2014 р.
Рецензент: д.т.н., проф. В.А. Крісілов