computing@computingonline.net
www.computingonline.net

ISSN 1727-6209
International Journal of Computing

ИНТЕЛЛЕКТУАЛЬНЫЙ АНАЛИЗ ДАННЫХ ДЛЯ ВЫЯВЛЕНИЯ ВРЕДОНОСНЫХ ПРОГРАММ

Дмитрий Комашинский, Игорь Котенко

Лаборатория проблем компьютерной безопасности СПИИРАН
14 линия, 39, Санкт-Петербург, 199178, Россия
komashinskiy@comsec.spb.ru, ivkote@comsec.spb.ru,
http://comsec.spb.ru

Резюме: *в статье проводится обзор наиболее значимых работ в области создания систем обнаружения и идентификации вредоносных программ на основе методов интеллектуального анализа данных. Для формализации этого процесса используются элементы методологии SADT, обобщающие основные процедурные аспекты существующих работ, посвященных данной предметной области. Выделяются основные группы сущностей, используемых для формирования типовых методик обнаружения вредоносных программ на основе данной группы методов*

Ключевые слова: *интеллектуальная обработка данных, вредоносные программы, обнаружение.*

INTELLIGENT DATA ANALYSIS FOR MALWARE DETECTION

Dmitry V. Komashinskiy, Igor V. Kotenko

Laboratory of computer security problems SPIIRAS
39, 14th Liniya, St. Petersburg, 199178, Russia
komashinskiy@comsec.spb.ru, ivkote@comsec.spb.ru,
http://comsec.spb.ru/en

Abstract: *The paper considers a state-of-the-art survey of systems for malware detection and identification based on intelligent data analysis. The SADT methodology was adopted to formalize this process in order to generalize common procedural aspects described in the analyzed papers within the area. The set of basic abstract items specifying the essence of each concrete approach to detect malware is emphasized.*

Keywords: *Intelligent data analysis, malware, detection.*

1. ВВЕДЕНИЕ

В настоящее время одной из актуальных задач в области выявления вредоносных программных объектов (далее используется аббревиатура ВП от «вредоносная программа», англ. malware) является применение методов интеллектуального анализа данных (ИАД, англ. Intelligent Data Analysis, IDA).

Концепция применения ИАД для обнаружения вредоносных программ была сформулирована Кефартом [3] и др. в 90-х годах прошлого века и получила практическое продолжение в работе Столфо, Шульца и др. [20] в 2001 году. К настоящему времени очевидно, что теоретические изыскания в данной предметной области успешно стимулируют разработку и интеграцию практических методик противодействия ВП [4, 16, 25]. Более того, наблюдается тенденция усложнения задач в данной предметной области в соответствии с текущими вызовами, определяемыми эволюцией ВП. К примеру, Масуд и др. [26] рассматривают проблему выявления ВП как задачу анализа непрерывного и изменчивого во времени потока данных, что дает понимание современных тенденций в данной предметной области.

Несмотря на наличие ряда ценных результатов, перед исследователями возникают новые, все более сложные задачи, направленные

на формирование систем обнаружения и идентификации вредоносных программ, оптимальных с точки зрении учета требований к характеристикам точности, производительности и ресурсопотребления.

Авторам представляется, что в свете наличия большого количества работ в данной области достаточно актуальной задачей является проведение их анализа, который позволит исследователям принимать более эффективные решения.

Структура статьи представляется следующим образом.

В первом разделе предлагается формальное представление процессов использования методов ИАД для построения систем обнаружения и идентификации вредоносных программ.

Во втором разделе раскрываются основы существующих подходов к выявлению вредоносных программ на примере файловых объектов формата Portable Executable (PE32), являющихся основным средством распространения вредоносных программ для операционных систем Ms Windows.

В третьем разделе рассматриваются отдельные статические и динамические подходы к выявлению вредоносных программ на основе ИАД, а также методы их комбинирования.

Итоги обзора подводятся в заключении.

2. ФОРМАЛЬНОЕ ОПИСАНИЕ ПРОЦЕССОВ ИСПОЛЬЗОВАНИЯ МЕТОДОВ ИАД

Процесс обучения системы обнаружения ВП (рис. 1) может быть представлен как последовательность действий (цепочка подпроцессов)

$$M_{Learn} = [P_{Ext}, P_{Sel}, P_{Learn}, P_{Val}], \qquad (1)$$

обеспечивающая получение из обучающего набора данных D_{Train} целевой модели M_{Dec}, построенной на основе пространства атрибутов $A_{Dec} = \{a_1, a_2, \dots, a_n\}$.

В дальнейшем пространство атрибутов A_{Dec} и работающая в его рамках целевая модель M_{Dec} используются в процессе эксплуатации системы (рис. 2).

Выделенные элементы процесса обучения данных систем могут быть охарактеризованы следующим образом.

Подпроцесс извлечения признаков P_{Ext} обеспечивает формирование начального пространства атрибутов $A_{Raw} = \{a_1, a_2, \dots, a_r\}$, в рамках которого формируется описание $D_{Raw} = \{d_1^R, d_2^R, \dots, d_m^R\}$ каждого элемента входного набора данных $D_{Train} = \{d_1^T, d_2^T, \dots, d_m^T\}$.

Формирование A_{Raw} осуществляется за счет использования предварительно выбранной исследователем модели представления объектов M_{View}, целью которой является обеспечение единого подхода к рассмотрению элементов множества D_{Train} в рамках выделенного аспекта, обеспечивая тем самым преобразование M_{View} : $D_{Train} \rightarrow A_{Raw}$.

Рис. 1 – Модель процесса обучения систем обнаружения ВП

Задачей подпроцесса выделения значимых признаков P_{Sel} является оптимизация размерности и эффективности использования начального пространства атрибутов A_{Raw}, на основе которого построено множество описаний элементов входного набора данных D_{Raw} с получением новых пространства атрибутов A_{Dec} и описания входного набора данных D_{Dec}.

Рис. 2 – Модель процесса эксплуатации систем обнаружения ВП

На практике задача оптимизации может быть связана как с выделением A_{Dec} из A_{Raw}, так и с полным или частичным формированием множества A_{Dec} на основе атрибутов, не

входящих в начальное пространство A_{Raw}. Часто в интересах минимизации временных и ресурсных затрат задача конструирования признаков по умолчанию сводится к формированию модели представления объектов M_{View} таким образом, чтобы оптимизация начального пространства атрибутов не требовала формирования новых.

Это объясняет фокус данного процесса на использовании определенной при подготовке эксперимента процедуры $M_{Sel} : A_{Raw} \rightarrow A_{Dec}$, $A_{Dec} \subset A_{Raw}$ выделения значимых признаков A_{Dec} из числа существующих A_{Raw}. Подпроцесс обучения модели P_{Learn} является основополагающим во всем процессе обучения системы и обеспечивает формирование целевой модели M_{Dec} на основе использования оптимизированного набора данных D_{Dec} применительно к выбранному исследователем методу обучения. Подпроцесс оценивания модели P_{Val} предоставляет возможность получить количественную оценку качества предиктивной способности полученной на предыдущем шаге целевой модели M_{Dec} на основе тестового набора данных $D_{Check} = \{d_1^C, d_2^C, \dots, d_m^C\}$.

Процесс эксплуатации эвристической системы обнаружения РПВ может быть представлен как цепочка подпроцессов)

$$M_{Func} = [P_{Ext},\ P_{Func}]\ . \qquad (2)$$

Он использует полученные на этапе обучения пространство атрибутов A_{Dec} для формирования оптимизированного представления объектов множества и целевую модель M_{Dec}. Подпроцесс извлечения признаков P_{Ext} по своей сути идентичен аналогичному подпроцессу фазы обучения систем обнаружения РПВ. Его основным отличием является то, что извлекаются признаки, относящиеся только к оптимизированному пространству атрибутов, $M_{View} : D_{Load} \rightarrow A_{Dec}$. Размер множества входящих объектов D_{Load} в данном случае не имеет значения и в общем случае $|D_{Load}| = 1$. Целью подпроцесса эксплуатации целевой модели P_{Func} является формирование метки класса объектов множества D_{Load}.

В сокращенном виде, идея формального представления системы выявления ВП на основе ИАД может быть выражена в виде кортежа

$$S = \langle M_{view}, M_{fs}, M_l, D_{tr}, E \rangle, \qquad (3)$$

где M_{view} – используемые модели представления анализируемых объектов,

M_{fs} – метод(ы) выделения значимых признаков,

M_l – метод(ы) обучения,

D_{tr} – используемые для обучения данные и

E – применяемые программные средства поддержки вычислений.

Полученное представление используется для структурирования настоящей работы.

Модель представления объектов устанавливает формальное представление объектов конкретного типа, которое будет обрабатывать система. Как правило, эта модель включает обобщенную модель объекта и на ее основе вводится понятие признаков, совокупность которых тем или иным образом характеризует каждый уникальный экземпляр объекта;

Методы выделения значимых признаков позволяют произвести минимизацию количества признаков, используемых для описания объектов.

Методы классификации (кластеризации) устанавливают конкретный подход к обучению, используемый в системе.

Используемые *средства поддержки вычислений* позволяют организовать практические аспекты реализации системы;

Источники данных используются при обучении системы и начальной оценке показателей точности решений.

3. ОСНОВНЫЕ ПОДХОДЫ К АНАЛИЗУ ИСПОЛНЯЕМЫХ ФАЙЛОВЫХ ОБЪЕКТОВ

На рис.3 представлена обобщенная последовательность действий, необходимая для получения относительно полного представления о внутренних структурных и функциональных особенностях бинарных исполняемых файлов формата Portable Executable (PE32), до настоящего времени являющегося основным источником угроз для персональных компьютеров.

Как показано на рис. 3, существует две основные группы подходов к анализу исполняемых файловых объектов.

Это подходы, реализующие (1) статические и (2) динамические методики их обработки и принятия решения о степени их вредоносности.

Статические подходы предоставляют более быстрые и менее затратные способы анализа объектов, не требующие их выполнения (или моделирования процесса выполнения) в интерпретирующей их среде.

В зависимости от типа файлов, интерпретирующей средой может операционная система или

какое-либо специализированное программное приложение, например Интернет-браузер.

Основным недостатком данной группы подходов является их неспособность эффективно решать проблему множественности статических представлений объекта, обладающего уникальным поведенческим паттерном (поведением).

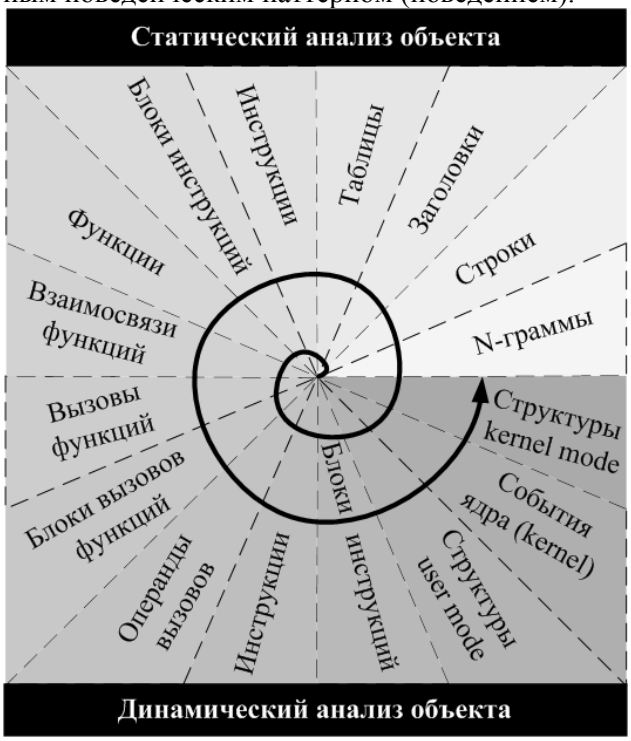

Рис. 3 – Анализ объектов формата Portable Executable

Это ограничивает применимость статических методов и, в итоге, объясняет недостаточную точность при использовании их в отрыве от других подходов. В качестве простейшего примера можно показать, что построение конечного исполняемого файла на основе одного и того же набора исходных файлов (программного проекта) с использованием разного инструментария (разных опций компиляции) позволяет сформировать множество его статически различных функциональных копий. Это, по сути, и приводит к необходимости использовать все более и более сложные методы статического анализа. Данная особенность широко используется злоумышленниками за счет использования различных инструментальных средств компиляции, обфускации, упаковки и программной защиты. При комбинировании данных средств можно с уверенностью утверждать о возможности формирования нескольких десятков тысяч структурно различных копий объекта, обладающего одними и теми же функциональными свойствами.

Группа динамических подходов объединяет более медленные и дорогие с точки зрения используемых вычислительных ресурсов способы получения достоверной информации о функциональности анализируемого объекта. Однако, данные способы позволяют нивелировать проблемы, присущие статическим подходам. Вместе с тем, динамический анализ тоже обладает рядом недостатков.

В первую очередь, как было показано выше, это относится к проблеме соответствия моделируемой при анализе внешней среды (окружения) и ожидаемой (реальной) среды. Например, некоторые вредоносные приложения выполняют явные деструктивные действия только при наличии условий, определенных внешней средой. Их отсутствие не позволяет выявить в процессе анализа деструктивный функционал, и, тем самым, может привести к ложному решению аналитика.

Во-вторых, цена разработки и поддержки корректных моделей оказывается иногда непомерно высокой за счет трудоемкости и сложности подобной работы. Как правило, любой программный инструментарий, поддерживающий процесс динамического анализа, имеет определенные недостатки, позволяющие вредоносному коду успешно выявлять их наличие и противостоять им. В качестве примера, можно привести понятия так называемых подходов противодействия отладочным средствам и программным эмуляторам (anti-debugging and anti-emulation tricks).

Наличие проблем в указанных группах подходов и обуславливает природу показанного на рис. 3 итеративного процесса анализа неизвестных объектов от простого к сложному, а также объясняет причину, по которой в настоящее время используются и те, и другие подходы, несмотря на все их недостатки.

4. АНАЛИЗ ИССЛЕДОВАНИЙ

На рис. 4 показано относительное место каждой используемой группы признаков в рамках базиса, определяемого степенью сложности разработки технологии, обеспечивающей их сбор, и средним временем самого процесса сбора.

Например, достаточно очевидно то, что процесс программной реализации простейшего средства сбора N-грамм значительно проще, а получаемое средство универсальнее и быстрее по сравнению с процессом разработки системы дизассемблирования, требующей учета, по крайней мере, поддерживаемых файлового формата и набора команд CPU.Обзор наиболее

Рис. 4 - Показатели процессов сбора признаков

значимых работ в области применения методов ИАД для задачи выявления вредоносных программных объектов проведем в соответствии с основными используемыми группами признаков.

N-граммы. Первой работой, в которой было представлено применение N-грамм для извлечения признаков вредоносного кода, является работа Кепхарта и др. [3], опубликованная в 1995 году и посвященная актуальной на тот момент проблеме выявления зараженных загрузочных секторов. Одной из основополагающих работ, в которых исследовалась эффективность N-грамм для построения системы обнаружения ВП формата PE32, является работа Шульца, Столфо и др. [20]. Они применили данную группу признаков с установленной длиной, равной двум байтам.

В дальнейшем эффективность N-грамм неоднократно обсуждалась другими группами исследователей. Колтер и Малуф [5] в 2004 году продолжили исследование вопроса обнаружения вредоносных PE32-файлов с помощью N-грамм с длиной в диапазоне от 1 до 10 байт, извлеченных из областей, содержащих машинный код. Авторы сообщили, что при выполнении экспериментов они выявили оптимальное для задачи соотношение длинb N-граммы и количества N-грамм в виде 4:500. В 2007 - 2008 годах Масуд, Кхан и др. [12-14] активно использовали N-граммы для построения общего подхода к обнаружению ВП за счет комбинирования групп признаков. В 2009 году Менахем и др. [15] применили данную подгруппу признаков для обоснования применимости иерархических методов комбинирования средств классификации для выявления ВП. В 2010 году в ряде работ [7, 19] N-граммы использовались для продолжения экспериментов. Сантос и др. [19] выполнили исследование применимости метода k ближайших соседей (k Nearest Neighbours, kNN) и выявления оптимальных значений N и k для задачи выявления ВП. Другая работа [7] была посвящена актуальной проблеме минимизации начального количества N-грамм за счет введения понятия позиции N-граммы в рамках значимого региона анализируемого объекта.

Одним из важнейших преимуществ N-грамм, как признаков ВП, является простота процесса их извлечения и наглядность. Вместе с тем, данная группа признаков обладает и существенными недостатками, определяемыми в первую очередь их потенциально большим количеством [7], что усложняет проведение дальнейших процедур выделения значимых признаков. Кроме того, системы, использующие только данную группу признаков, уязвимы по отношению к использованию разнообразных методов обфускации контента (например, упаковка, установка дополнительных защитных программных слоев для усложнения реверсинга и т.д.).

Строки. Строковые данные, также часто упоминаемые как интерпретируемые (читаемые) строки, можно рассматривать в качестве группы признаков, семантически близкой к рассмотренным выше байтовым

последовательностям (N-граммам). Их основным отличиями следует считать (1) переменную длину, (2) принадлежность значений символов строки к определенному диапазону значений, определяющему алфавит, и (3) наличие терминирующего символа, завершающего строку. Как правило, значение последнего зависит от контекста процедуры извлечения признаков. Так, например, для процедуры анализа файлов формата PE32 терминирующий символ исторически связан с понятием нулевого символа, определяющего завершение строки (C-строки).

Уже упомянутую работу Шульца [20] можно считать первой работой, официально использовавшей строковые данные для построения систем обнаружения ВП на основе ИАД. Несмотря на это, следует отметить, что подобные признаки активно использовались в антивирусной индустрии и ранее для формирования сигнатур. Колтер и Малуф [5] на практике показали, что очень часто наиболее значимые для процесса детектирования ВП N-граммы соответствуют определенным последовательностям символом, относящимся к строкам. В настоящее время в силу определенных недостатков статические строковые признаки в меньшей степени значимы при проведении исследований, однако все равно принимаются во внимание. Например, Лу и другие [11] используют строковые признаки в качестве одной из групп признаков при построении комбинированной схемы методов для обнаружения ВП.

К достоинствам строк, как признаков, следует отнести указанные выше преимущества использования N-грамм и интерпретируемость. Основными недостатками строк являются проблемы, присущие N-граммам.

Заголовки и таблицы. Данные заголовков исполняемых файлов являются важным источником информации для систем обсуждаемого класса. Шульц и др. [20] применяют данные таблиц импорта для формирования подмножества признаков, несущих информацию об используемых библиотечных вызовах функций операционной системы. Менахем и др. [15] используют значения заголовков PE32 и информацию директорий импорта, экспорта и ресурсов для построения многослойной иерархической системы классификации, основанной на применении признаков различных групп. Пердиски и др. [17] используют данные заголовков для формирования набора производных признаков (таких, как энтропия секций). Маттик [16] использует числовые данные заголовков и некоторые производные признаки для анализа применимости статического анализа. Шахзад и др. [22] производит анализ заголовочных структур исполняемых файлов формата ELF (используемых в Unix-подобных ОС) и тем самым обосновывает применимость подходов, традиционно используемых для структурного анализа исполняемых файлов ОС Windows (PE32) для статического обнаружения вредоносных файлов на других платформах. Основным достоинством данной группы признаков является относительная простота их извлечения и интерпретации. Использование данных этого уровня, в конечном счете, позволяет получить значительную долю информации о начальном состоянии объекта непосредственно перед началом его интерпретации исполняющей средой. Например, заголовочные и табличные данные файлового формата PE32 позволяют с достаточной точностью судить о начальном состоянии адресного пространства процессов, его выполняющих. Вместе с тем, наличие широкого спектра методов противодействия статическому анализу на практике не всегда позволяет в полной мере воспользоваться преимуществами заголовочных и табличных данных.

Инструкции и их совокупности. Группы признаков, извлекаемых из анализируемых объектов на уровне анализа вложенного программного кода (для исполняемых форматов это, как правило, достигается дизассемблированием), позволяют получать низкоуровневую статическую информацию, описывающую поведенческие аспекты, дающие представление об отдельных управляющих командах и их совокупностях (последовательностях команд и их логических объединений). Йе и др. [24] используют производные частотные характеристики наличия машинных команд и их последовательностей в рамках отдельных функциональных блоков. Масуд и др. [12-14] применяют последовательности описаний машинных команд (семантический тип операции и типы аргументов). Сиддики и др. [23] предлагают подход к формированию описаний последовательностей машинных команд на основе значений их опкодов. Алазаб [1] и др. используют статические данные о вызовах функций для улучшения показателей системы выявления ВП на основе N-грамм. Кинэйбл и Костакис [4] предлагают методику, использующую статические данные о связях совокупностей инструкций (функций) для подготовки статического графа вызовов функций, используемого для оценки подобия ранее неизвестных объектов существующим описаниям вредоносных. Данные группы признаков

являются наиболее информативными по сравнению с описанными выше и позволяют достигнуть максимальной (по сравнению с другими статическими типами данных) точности принятия решения. Вместе с тем, их очевидным недостатком является достаточная сложность обработки и невозможность их полноценного использования в условиях широкого применения злоумышленниками средств программной защиты и упаковки.

Динамические признаки. Авторы данного обзора представили подход [8] к выявлению вредоносных программ на основании использования ими в процессе работы системных сервисов ядра для ОС Windows XP. Использовались числовые и категориальные производные признаки, характеризующие, например, количество вызовов и типы доступа к отдельным объектам контролируемой системы. В отличие от этого, Йе и др. [25] предложили использовать в качестве признаков последовательность перехватываемых вызовов.

Ланци и др. [10] для построения формализованного описания процесса выполнения контролируемого приложения используют так называемые «поведенческие N-граммы», формируемые методом прохода «плавающего окна» заданной длины N по наблюдаемой последовательности вызовов системных сервисов. Они показали, что наиболее эффективные значения N лежат в интервале от 2 до 4 включительно. Примечательно то, что логика сбора последовательности вызовов опирается на понятие контроля доступа к критическим объектам системы (отдельные файлы, ключи и значения системного реестра).

Рик и др. [18] для описания выполняемых приложений применили символьные цепочки, характеризующие семантику выполняемых системных вызовов и их аргументы. Выделение отдельных признаков тоже производилось с помощью метода «плавающего окна».

Шахзад и др. [21] представили подход к выявлению ВП, опирающийся на контроль состояния определенных полей структур ядра, ассоциированных с анализируемыми процессами. Оценка практической применимости подхода была проведена по отношению к ВП, функционирующим на ОС семейства Linux. Было показано, что для успешного выявления некоторых классов ВП достаточно проведения мониторинга структур ядра на начальных стадиях жизненного цикла процессов. С теоретической точки зрения, подход применим и для ОС Windows в рамках систем класса HIPS (Host Intrusion Prevention System).

Исследование применимости формальных представлений выполнения анализируемых объектов на основе мониторинга выполняемых машинных команд проводилось в работах [2] и [6]. Первый подход ориентирован на обнаружение ВП на основе поиска непрерывных цепочек исполняемых инструкций, свойственных вредоносным программ. Вторая работа расширяет данный подход за счет ориентации его на идентификацию отдельных семейств ВП, вводит формальное определение модели анализируемого объекта и использует собственные критерии, задающие понятие цепочки машинных команд и необходимое время их сбора.

Комбинирование классификаторов. Вопрос применения методов комбинирования методов ИАД для обнаружения ВП исследуется, как правило, в контексте задачи повышения точности принятия решения и улучшения показателей эксплуатационных характеристик. В качестве примера работы, освещающей прикладные аспекты комбинирования, можно рассматривать монографию Кунчевой [9].

Работа Менахема и др. [15] посвящена задаче улучшения показателей точности и времени обнаружения ВП за счет использования ансамбля классификаторов. Для формирования пространства признаков применялись статические признаки, полученные на основе извлечения N-грамм в диапазоне от 3-х до 6-ти. Выделение значимых признаков основано на методах Fisher Score и Information Gain, что позволило изначально сократить количество использованных признаков этого класса до 300 5-грамм и 600 6-грамм. Кроме того, в качестве источника информации применялись структурные характеристики файлов. Использовались данные общих заголовков, секций импорта и экспорта, директории ресурсов. Предложена группа признаков, отображающих количественные и качественные характеристики программных функций, содержащихся в анализируемых файлах и отношения объема кода к общему размеру файла и его отдельных частей. После слияния данных групп признаков и проведения повторной оценки их значимости с помощью метода Information Gain был выделен окончательный набор из 140 признаков, использованных в дальнейшей части работы. В качестве основных методов обучения были реализованы методы Majority Voting, Performance Weighting, Distribution Summation, Bayesian Combination, Naive Bayes Combination, Stacking и предложенный авторами метод комбинирования Troika. Метод Troika основан на четырех-уровневой иерархической системе классификаторов, где в качестве базовых классификаторов

используются методы деревьев решений (Decision Tree, DT), kNN, Voting Features Interval (VFI), One Rule (OneR) и наивный Байесовский классификатор (Naive Bayes, NB), а в качестве классификаторов промежуточного слоя - DT и kNN. Оценивание точности рассмотренных методов комбинирования проводилась как для задачи классификации с двума метками, так и для задачи классификации с количеством классов, равным 8 (7 типов ВП и безопасные программы). Оценка результатов на основе 10-кратной кросс-проверки показала высокую точность метода Troika. Однако, этот метод наряду с методом Stacking, требовал существенного количества времени для обучения. Поддержка вычислений осуществлялась с помощью программного пакета Weka.

Йе, Ли и др. [24] исследовали вопрос использования ансамбля методов кластеризации для построения системы автоматической категоризации новых, ранее неизвестных файлов формата PE32.

Основным элементом мотивации работы являлась необходимость снижения времени принятия решения о степени опасности каждого поступающего в систему объекта и его семействе. Обучение решающей модели и ее валидация проводились на данных корпорации KingSoft, представляющих наборы ВП, зарегистрированных на протяжении нескольких суток. В качестве основных признаков использовались статические атрибуты, характеризующие частотные характеристики появления машинных инструкций и наличие цепочек инструкций, входящих в функции, хранимые исследуемыми файлами. Идея построения ансамбля методов кластеризации была основана на использовании двух базовых методов. Первым являлся так называемый метод гибридной иерархической кластеризации, объединяющий преимущества метода иерархической кластеризации, как общего подхода к последовательному объединению кластеров на каждом шаге, и метода k-medoids, как метода разделения полученной совокупности кластеров. Данный метод использовался для обработки статических частных показателей наличия инструкций. Вторым являлся метод Weighted Subspace K-Medoids, помимо прочего решающий задачу минимизации пространства значимых признаков наличия тех или иных последовательностей машинных инструкций в рамках отдельного кластера. Работа выполнялась с использованием собственного набора программного инструментария.

Лу и др. [11] предложили подход к решению проблемы точности обнаружения ВП за счет комбинирования как признаков, так и методов. Идея комбинирования признаков заключалась в использовании объединенного множества статических и динамических признаков, где в качестве статических выступали признаки наличия вызовов системных библиотек вида «Имя библиотеки - Имя функции», а динамические признаки представляли набор потенциально подозрительных действий, выполняемых приложениями. Всего для проведения экспериментов использовались данные из ряда открытых источников. Для выделения набора значимых признаков использовался метод Information Gain. Комбинирование методов основано на применении двухуровневой иерархической модели, объединяющей методы опорных векторов (Support Vector Machine, SVM) и Association Rules. Показано, что предложенный авторами подход способен успешно конкурировать с известными методами комбинирования (в плане повышения точности обнаружения ВП) и имеет сравнимое с другими методами время обучения. Для сравнения использовались базовые методы классификации NB, SVM, kNN, DT, OneR и методы комбинирования Voting, Bagging DT, Boosting DT, Stacking и Grading.

Используемые программные средства поддержки вычислений и источники данных. Анализ работ, представленных в данном разделе, показывает следующие особенности при организации экспериментов.

Как правило, исследователи предпочитают использовать существующие свободные программные средства, реализующие методы ИАД. Наиболее популярным является программный пакет WEKA (Waikato Environment for Knowledge Analysis), используемый во многих упомянутых публикациях [5, 7, 8, 15 – 17, 21]. Помимо этого, в ряде работ [21, 23-25] WEKA используется совместно с библиотекой libSVM, реализующей алгоритмы, реализуемые методом опорных векторов (эта библиотека используется сама по себе в работе Алазаба и др. [1]). Сидикки и др. [23] в своих экспериментах используют возможности языка программирования R. Авторы данного обзора в одной из своих последних работ, посвященной данной теме [6], используют программный пакет RapidMiner, позволяющий объединять собственные возможности со всеми указанными выше программными средствами в рамках единой графической независимой от платформы среды. Похожими возможностями обладает программный пакет KEEL (Knowledge Extraction based on Evolutionary Learning), использованный Шахзадом и др. [22] для дополнения возможностей программного пакета WEKA.

Формирование наборов данных, в частности, вредоносных, для проведения экспериментов является сложной задачей в силу достаточно очевидных причин, главной из которых является то, что источник данных должен заслуживать доверия. Предпочтительным источником являются внутренние хранилища данных компаний, специализирующихся на обнаружении ВП.

В качестве примера подобных работ, выполненных с использованием данных антивирусных компаний, можно привести публикации Кинэйбла и Костакиса (F-Secure) [4], Маттика (McAffee) [16], Йе и др. (KingSoft) [25].

Вместе с тем, используются и другие, более доступные, коллективно поддерживаемые источники данных, например набор коллекций VXHeavens (используются в [2, 5-8, 12-14, 21-23]), данные OffensiveComputing (используется Шахзадом [22]) и Malfease (используется Пердиски [17]).

4. ЗАКЛЮЧЕНИЕ

В статье на основании показанного формализованного представления выделены базовые составляющие, используемые для формирования структуры основной части работы. Процесс группировки использованных в работе публикаций осуществлен в соответствии с введенным понятием модели представления объекта.

Показано, что методы интеллектуального анализа данных могут быть использованы не только в качестве средств формирования систем автоматического выявления вредоносных объектов, но и для оценки эффективности применения новых моделей. Дальнейшая работа по этой теме будет связана с исследованиями систем обнаружения современных Интернет–угроз.

Работа выполняется при финансовой поддержке Министерства образования и науки РФ (контракт 11.519.11.4008), РФФИ (проект №13-01-00843-а), программы фундаментальных исследований ОНИТ РАН (проект №2.2), проектов Евросоюза SecFutur и MASSIF.

5. СПИСОК ЛИТЕРАТУРЫ

[1] M. Alazab, R. Layton, S. Venkataraman, P. Watters, Malware Detection Based on Structural and Behavioural Features of API Calls, *2010 International Cyber Resilience Conference*, Perth, Australia, (23-24 August), pp. 1-10.

[2] J. Dai, R. Guha, J. Lee, Efficient Virus Detection Using Dynamic Instruction Sequences, *Journal of Computers*, (4) 5 (2009), pp. 405-414.

[3] J.O. Kephart, G.B. Sorkin, W.C. Arnold, D.M. Chess, G.J. Tesauro, S.R. White, Biologically inspired defenses against computer viruses, *International Joint Conference on Artificial Intelligence*, Montreal, Canada, (20-25 August 1995), pp.985-996.

[4] J. Kinable, O. Kostakis, Malware Classification Based onCall Graph Clustering, *Journal In Computer Virology*, (7) 4 (2011), pp. 233-245.

[5] J.Z. Kolter, M.A. Maloof, Learning to Detect Malicious Executables in the Wild, *2004 International Conference on Knowledge Discovery and Data Mining*, Seattle, WA, USA (22-25 August), pp.470-478.

[6] D.V. Komashinskiy, I.V. Kotenko, Using Low-Level Dynamic Attributes for Malware Detection Based on Data Mining Methods, Lecture Notes in Computer Science, Springer-Verlag, Vol. 7531. *2012 International Conference on Mathematical Methods, Models and Architectures for Computer Network Security*, St.Petersburg, Russia (17-20 October), pp.254-269.

[7] D.V. Komashinskiy, I.V. Kotenko, Malware Detection by Data Mining Techniques Based on Positionally Dependent Features, *2010 Euromicro International Conference on Parallel, Distributed and network-based Processing*. Piza, Italy (17-19 February), pp.617-623.

[8] D.V. Komashinskiy, I.V. Kotenko, Integrated Usage of Data Mining Methods for Malware Detection, *2009 International Workshop "Information Fusion and Geographical Information Systems"*. St.Petersburg, Russia (17-20 May). Lecture Notes in Geoinformation and Cartography, Springer, pp.343-357.

[9] L.I. Kuncheva, *Combining Pattern Classifiers: Methods and Algorithms*. Wiley-Interscience, 2004, 350 p.

[10] A. Lanzi, D. Balzarotti, C. Kruegel, M. Christodorescu, E. Kirda, AccessMiner: Using System-Centric Models for Malware Protection, *2010 ACM conference on Computer and Communication Security*, Chicago, IL, USA (4-8 October), pp.399-412.

[11] Y.-B. Lu, S.-C. Din, C.-F. Zheng, B.-J. Gao, Using Multi-Feature and Classifier Ensembles to Improve Malware Detection, *Journal of Chung Cheng Institute of Technology*, (39) 2 (2010), pp.57-72.

[12] M.M. Masud, L. Khan, B. Thuraisingham, Feature-Based Techniques for Auto-Detection of Novel Email Worms, *2007 Pacific-Asia Conference on Knowledge Discovery and Data*

Mining, Nanjing, China (22-25 May), pp.205-216.

[13] M.M. Masud, L. Khan, B. Thuraisingham, A Hybrid Model to Detect Malicious Executables, *2007 IEEE International Conference on Communications,* Glasgow, Scotland (24-28 June), pp.1443-1448.

[14] M.M. Masud, L. Khan, B. Thuraisingham, *Data Mining Tools for Malware Detection*. CRC Press Taylor & Francis Group, 2012. 450 p.

[15] E. Menahem, A. Shabtai, L. Rokach, Y. Elovici, Improving Malware Detection by Applying Multi-Inducer Ensemble, *Journal Computational Statistics & Data Analysis* (53) 4 (2009), pp.1483-1494.

[16] I. Muttik, Malware Mining, *2011 Virus Bulletin Conference*, Barcelona, Spain, (5-7 October), pp.46-51.

[17] R. Perdisci, A. Lanzi, W. Lee, McBoost: Boosting scalability in malware collection and analysis using statistical classification of executables, *2008 Computer Security Applications Conference*, Anahem, CA, USA, (8-12 December), pp.301-310.

[18] K. Rieck, T. Holz, C. Willems, P. Dussel, P. Laskov, Learning and Classification of Malware Behavior, *2008 International conference on Detection of Intrusions and Malware and Vulnerability Assessment*, Paris, France (10-11 July), pp.108-125.

[19] I. Santos, Y. Penya, J. Devesa, P. Bringas, N-grams-based File Signatures for Malware Detection, *2009 International Conference on Enterprise Information Systems*, Milan, Italy (6-10 May), pp.317-320.

[20] M. Schultz, E. Eskin, E. Zadok, S. Stolfo, Data Mining Methods for Detection of New Malicious Executables, *2001 IEEE Symposium on Security and Privacy*, Oakland, CA, USA (13-16 May), pp. 38-49.

[21] F. Shahzad, S. Bhatti, M. Shahzad, M. Farooq, In-Execution Malware Detection using Task Structures of Linux Processes, *2011 IEEE International Conference on Communications*, Kyoto, Japan (5-9 June), pp.1-6.

[22] F. Shahzad, M. Farooq, ELF-Miner: Using Structural Knowledge and Data Mining Methods to Detect New (Linux) Malicious Executables, *Journal of Knowledge and Information Systems*, (30) 3 (2012) pp.589-612.

[23] M. Siddiqui, M. Wang, J. Lee, Detecting Internet Worms Using Data Mining Techniques, *Journal of Systemics, Cybernetics and Informatics*, (6) 6 (2008), pp. 48-53.

[24] Y. Ye, T. Li, Y. Chen, Q. Jiang, Automatic Malware Categorization Using Cluster Ensemble, *2010 ACM International Conference on Knowledge discovery and data mining*, Washington, USA (25-28 July), pp.95-104.

[25] Y. Ye, T. Li, K. Huang, Q. Jiang, Y. Chen, Hierarchical associative classifier (HAC) for malware detection from the large and imbalanced gray list, *Journal of Intelligent Information Systems* (35) 1 (2010), pp.1-20.

[26] M. Masud, T. Al - Khateeb, K. Hamlen, L. Khan, J. Han, B. Thuraisingham, Cloud-Based Malware Detection for Evolving Data Streams, *In Journal ACM Transactions on Management Information Systems*, (2) 3 (2011), article 16, 27 p.

***Дмитрий Владимирович Комашинский**, аспирант лаборатории проблем компьютерной безопасности СПИИРАН. В 2000 году окончил Санкт-Петербургское Высшее Военное инженерное училище связи им. Ленсовета. В область научных интересов входят безопасность компьютерных сетей, защита программного обеспечения, обнаружение компьютерных атак, защита от вирусов и сетевых червей, интеллектуальный анализ данных, разработка программного обеспечения.*

***Игорь Витальевич Котенко**, доктор технических наук (1999), профессор. Заведующий лабораторией проблем компьютерной безопасности СПИИРАН. Автор более 450 научных работ. Область научных интересов – информационная безопасность, в том числе управление политиками безопасности, разграничение доступа, аутентификация, анализ защищенности, обнаружение компьютерных атак, межсетевые экраны, ложные информационные системы, защита от вирусов и сетевых червей, анализ и верификация протоколов безопасности и систем защиты информации, защита программного обеспечения от взлома и управление цифровыми правами, технологии моделирования и визуализации для противодействия кибер-терроризму; искусственный интеллект, в том числе многоагентные системы, мягкие и эволюционные вычисления, машинное обучение.*

computing@computingonline.net
www.computingonline.net

ISSN 1727-6209
International Journal of Computing

INTELLIGENT DATA ANALYSIS FOR MALWARE DETECTION

Dmitry V. Komashinskiy, Igor V. Kotenko

Laboratory of computer security problems SPIIRAS
39, 14th Liniya, St. Petersburg, 199178, Russia
komashinskiy@comsec.spb.ru, ivkote@comsec.spb.ru,
http://comsec.spb.ru/en

Abstract: *The paper considers a state-of-the-art survey of systems for malware detection and identification based on intelligent data analysis. The SADT methodology was adopted to formalize this process in order to generalize common procedural aspects described in the analyzed papers within the area. The set of basic abstract items specifying the essence of each concrete approach to detect malware is emphasized.*

Keywords: *Intelligent data analysis, malware, detection.*

1. INTRODUCTION

The research is dedicated to issues of using Intelligent Data Analysis (IDA) techniques for designing systems able to detect and identify malicious executable binaries (malware) automatically.

The paper comprises three main structural parts named as follows: “The formal description of using Intelligent Data Analysis techniques”, “Primary ways on analyzing executable binaries” and “State of the art”.

2. FORMAL DESCRIPTION OF USING DATA MINING TECHNIQUES

The first part introduces main results for SADT – based modelling of the learning and functioning processes of systems aimed at automatic malware counteraction (Fig.1).

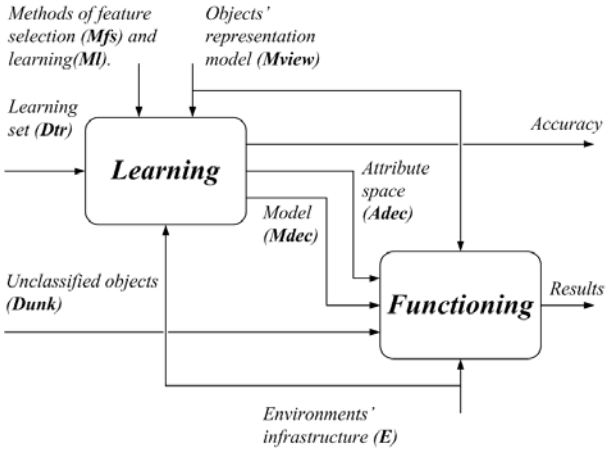

Fig.1 – High level representation of the model

The formal description of such systems is expressed by following statement:

$$S = \langle M_{view}, M_{fs}, M_l, D_{tr}, E \rangle, \qquad (1)$$

where M_{view} – used representation model(s) of analyzed objects,

M_{fs} – method(s) to select valuable features,

M_l – used learning method(s),

D_{tr} – used data for learning and validation and

E – adopted software for preparing necessary environment.

3. ANALYZING EXECUTABLE BINARIES

The second part generalizes essential aspects of modern approaches to executable malware’s detection and identification.

The main static and dynamic groups of malware attributes (features) are distinguished and described.

The main purpose of the section is to emphasize the interconnection of the introduced above representation model’s concept M_{view} with listed in this part groups of attributes which are most valuable for malware detection process.

4. RELATED WORK

The third part of the research is organized as a survey of a number of existing publications [1-14] devoted to the topic of using Intelligent Data

Analysis methods for detecting and identifying executable malware samples.

The survey's structure is based on the formal description posed above (statement 1) and considers the following issues:

(1) static [2-10, 12, 13] and dynamic [1, 11, 14] detection of malicious software; in particular, the practices of combining representation models and learning methods [5, 8, 13],

(2) adopted Intelligent Data Analysis software kits and

(3) used sources of data for performing learning and validation procedures.

The survey discusses different types of attributes and their validity for designing IDA – based automatic malware detection systems.

It goes through following static attributes types:

- n-grams [2, 10],
- strings [4, 5, 10],
- headers and tables [8-10],
- instructions and their sequences [6, 7, 12, 13],
- static call dependency graphs [3].

The similar approach is used for considering dynamic attributes' feasibility for preparing such automatic decision making systems at levels of machine instructions [1], operating system's calls [14] and structures [11].

The paper also encompasses main sets of Intelligent Data Analysis tools used nowadays by researchers for arranging proper experimental job (WEKA, libSVM, RapidMiner) and touches on the issu of data sources.

This research is being supported by grant of the Russian Foundation of Basic Research (project #13-01-00843-a), Program of fundamental research of the Department for Nanotechnologies and Informational Technologies of the Russian Academy of Sciences (contract #2.2), State contract #11.519.11.4008 and partly funded by the EU as part of the SecFutur and MASSIF projects.

5. REFERENCES

[1] J. Dai, R. Guha, J. Lee, Efficient Virus Detection Using Dynamic Instruction Sequences, *Journal Of Computers*, (4) 5 (2009), pp. 405-414.

[2] J.O. Kephart, G.B. Sorkin, W.C. Arnold, D.M. Chess, G.J. Tesauro, S.R. White, Biologically inspired defenses against computer viruses, *1995 International Joint Conference on Artificial Intelligence*, Montreal, Canada, (20-25 August), pp.985-996.

[3] J. Kinable, O. Kostakis, Malware Classification Based onCall Graph Clustering, *Journal In Computer Virology*, (7) 4 (2011), pp. 233-245.

[4] J.Z. Kolter, M.A. Maloof, Learning to Detect Malicious Executables in the Wild, *2004 International Conference on Knowledge Discovery and Data Mining*, Seattle, WA, USA (22-25 August), pp.470-478.

[5] Y.-B. Lu, S.-C. Din, C.-F. Zheng, B.-J. Gao, Using Multi-Feature and Classifier Ensembles to Improve Malware Detection, *Journal of Chung Cheng Institute of Technology*, (39) 2 (2010), pp.57-72.

[6] M.M. Masud, L. Khan, B. Thuraisingham, Feature-Based Techniques for Auto-Detection of Novel Email Worms, *2007 Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Nanjing, China (22-25 May), pp.205-216.

[7] M.M. Masud, L. Khan, B. Thuraisingham, A Hybrid Model to Detect Malicious Executables, *2007 IEEE International Conference on Communications*, Glasgow, Scotland (24-28 June), pp.1443-1448.

[8] E. Menahem, A. Shabtai, L. Rokach, Y. Elovici, Improving Malware Detection by Applying Multi-Inducer Ensemble, *Journal Computational Statistics & Data Analysis* (53) 4 (2009), pp.1483-1494.

[9] I. Muttik, Malware Mining, *2011 Virus Bulletin Conference*, Barcelona, Spain, (5-7 October), pp.46-51.

[10] M. Schultz, E. Eskin, E. Zadok, S. Stolfo, Data Mining Methods for Detection of New Malicious Executables, *2001 IEEE Symposium on Security and Privacy*, Oakland, CA, USA (13-16 May), pp. 38-49.

[11] F. Shahzad, S. Bhatti, M. Shahzad, M. Farooq, In-Execution Malware Detection using Task Structures of Linux Processes, *2011 IEEE International Conference on Communications*, Kyoto, Japan (5-9 June), pp.1-6.

[12] M. Siddiqui, M. Wang, J. Lee, Detecting Internet Worms Using Data Mining Techniques, *Journal of Systemics, Cybernetics and Informatics*, (6) 6 (2008), pp. 48-53.

[13] Y. Ye, T. Li, Y. Chen, Q. Jiang, Automatic Malware Categorization Using Cluster Ensemble, *2010 ACM International Conference on Knowledge discovery and data mining*, Washington, USA (25-28 July), pp.95-104.

[14] Y. Ye, T. Li, K. Huang, Q. Jiang, Y. Chen, Hierarchical associative classifier (HAC) for malware detection from the large and imbalanced gray list, *Journal of Intelligent Information Systems* (35) 1 (2010), pp.1-20.