УДК 004.056.57 **DOI** 10.15622/ia.23.3.2

Д.В. Смирнов, О.О. Евсютин

МЕТОДИКА СБОРА ДАННЫХ ОБ АКТИВНОСТИ ВРЕДОНОСНОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ПОД ОС WINDOWS НА БАЗЕ MITRE ATT&CK

Смирнов Д.В., Евсютин О.О. **Методика сбора данных об активности вредоносного программного обеспечения под ОС Windows на базе MITRE ATT&CK.**

Аннотация. Цифровизация современной экономики привела к масштабному проникновению информационных технологий в различные сферы человеческой деятельности. Кроме положительных эффектов это крайне обострило проблему противодействия киберугрозам, реализация которых злоумышленниками часто влечет за собой тяжелые последствия. Вредоносное программное обеспечение (ВПО) занимает важное место на современном ландшафте киберугроз, наиболее громкие киберпреступления последних лет связаны с применением ВПО. В связи с этим активно развивается проблемная область противодействия ВПО и одним из перспективных направлений исследований в данной области является создание методов детектирования ВПО на основе машинного обучения. Однако слабым местом многих известных исследований является построение достоверных наборов данных для моделей машинного обучения, когда авторы не раскрывают особенности формирования, предобработки и разметки данных о ВПО, что компрометирует воспроизводимость этих исследований. В данной работе предлагается методика сбора данных об активности ВПО, основанная на матрице MITRE ATT&CK и Sigma-правилах, и рассчитанная на операционные системы семейства Windows. Предлагаемая методика направлена на повышение качества наборов данных, содержащих характеристики поведения ВПО и легитимных процессов, а также на сокращение времени разметки данных экспертным способом. Для апробации методики подготовлен программный стенд и проведены эксперименты, подтвердившие ее адекватность.

Ключевые слова: кибербезопасность, вредоносное программное обеспечение, MITRE ATT&CK, мониторинг активности процессов, машинное обучение.

1. Введение. Новое тысячелетие можно охарактеризовать как начало новой вехи научно-технического прогресса. Автоматизация рутинной деятельности привела к тому, что информация стала одним из самых ценных ресурсов XXI века. Ценность информации стала причиной особого интереса к конфиденциальным данным различных спецслужб, подразделений промышленного шпионажа, а также преступников [1]. По этой причине мероприятия по защите информации перешли в отдельную область знаний – информационную безопасность.

Для описания угроз информационной безопасности стали использовать термин «ландшафт киберугроз» [2], который позволяет оценить весь перечень типов актуальных киберугроз для какой-либо отрасли. К примеру, существует ландшафт киберугроз для финансовых учреждений или промышленных предприятий

с использованием киберфизических систем [3]. Попытку реализации киберугрозы назовем атакой [4] или вторжением [5]. Основные типы вторжений были описаны еще в 1987 году в фундаментальной работе [5]. Автор исследования отдельно выделил попытки эксплуатации уязвимостей, атаки внутреннего нарушителя, утечки конфиденциальных данных, реализацию отказа в обслуживании, а также использование вредоносного программного обеспечения (ВПО). Каждая из этих киберугроз направлена на компрометацию одного или более свойств безопасности информации. Однако последний вид угроз является одним из наиболее опасных. В результате реализации атак с использованием ВПО возможно нарушение конфиденциальности, целостности и доступности информации. Поэтому противодействие распространению ВПО за счет создания новых более эффективных методов обнаружения является актуальной проблемной областью.

Существующие методы и средства борьбы с ВПО основываются на изучении известных образцов ВПО с целью написания сигнатур, а также на эвристических способах детектирования наиболее общих признаков, характерных для ВПО. Эти признаки либо извлекаются из файлов вредоносных программ, либо формируются на основе характеристик поведения вредоносных процессов. В соответствии с этим подходы к обнаружению ВПО делятся на статические и динамические.

Для работы с извлеченными признаками в настоящее время достаточно активно стали применяться различные методы машинного обучения. Например, данному вопросу посвящены недавние обзоры [6, 7, 8, 9]. Однако данное направление исследований содержит немало открытых проблем. Большинство исследователей не акцентирует внимание на вопросах формирования выборки ВПО, предобработки и разметки полученных данных. Во многих работах присутствуют ссылки на готовые наборы данных без уточнения деталей формирования этих наборов данных, что компрометирует практическую ценность этих работ. В некоторых исследованиях [10, 11, 12, 13, 14] описывается создание новых наборов данных, однако основной акцент делается на выделении признаков для создания моделей машинного обучения, а вопросы предобработки и разметки полученных данных опускаются. В патентах [15, 16, 17] отечественных и зарубежных лидеров в области кибербезопасности предлагаются отдельные способы формирования паттернов поведения ВПО в виде цепочек событий, однако в качестве метода разметки

данных используется экспертный метод для дальнейшего создания моделей машинного обучения.

Наша работа направлена на повышение качества наборов данных, содержащих характеристики поведения ВПО и легитимных процессов, а также на сокращение времени разметки данных. Под повышением качества понимается обеспечение соответствия набора данных некоторым заранее заданным критериям качества. Наше исследование опирается на критерии, представленные в работе [18]. Краткое описание данных критериев приведено в подразделе 2.2 настоящей статьи. При этом отметим, что разработка подходов к формированию наборов семейств и образцов ВПО и легитимных программ выходит за рамки нашего исследования.

Основным результатом данного исследования является методика сбора данных об активности ВПО под ОС Windows на базе матрицы MITRE ATT&CK [19], где под активностью ВПО понимаются поведенческие признаки вредоносных процессов ОС Windows, создаваемые в ходе работы ВПО. Новизна методики заключается в формализации этапов проведения эксперимента по сбору данных об активности ВПО и требований к содержанию обязательных полей и значений в событиях процессов ОС, а также применении Sigma-правил [20] для автоматизированной разметки этих событий в соответствии с используемыми ВПО техниками, тактиками и процедурами, соответствующими матрице MITRE ATT&CK. В данной работе предлагается использование Sigma-правил для разметки событий, а не для детектирования вредоносной активности. Использование Sigma-правил для обнаружения атак является нецелесообразным ввиду неприемлемого уровня ложноположительных срабатываний на объемной инфраструктуре для всех общедоступных Sigma-правил, а применение ограниченного набора этих правил существенно снизит эффективность обнаружения ВПО [21, 22]. Таким образом, применение предлагаемой методики позволит повысить применимость и воспроизводимость методов обнаружения ВПО на основе машинного обучения.

Статья организована следующим образом. Во втором разделе представлен обзор проблемной области обнаружения ВПО. Третий раздел описывает матрицу MITRE ATT&CK и подход к сопоставлению техник атаки и Sigma-правил. В четвертом разделе рассмотрены механизмы мониторинга активности процессов ОС Windows [23]. Предлагаемая методика сбора данных об активности процессов ОС Windows приведена в пятом разделе. Результаты вычислительных экспериментов с предлагаемой методикой

представлены в шестом разделе. Заключение подводит итоги настоящего исследования.

2. Обзор подходов к обнаружению ВПО. Задачу обнаружения ВПО в информационной системе можно формализовать следующим образом. Пусть $O = \{o_1, o_2, ..., o_m\}$ – множество объектов-файлов, имеющихся в информационной системе, и $T = \{t_1, t_2, ..., t_n\}$ – множество моментов дискретного времени. Известно, что множество O может быть разделено на два подмножества $O = O_S \cup O_M$, $O_S \cap O_M = \varnothing$, где подмножество O_S содержит легитимное ПО, а подмножество O_M – вредоносное ПО, но неизвестен способ этого разделения.

Множество всевозможных событий, которые могут происходить в информационной системе, назовем алфавитом событий и обозначим $E = \{e_1, e_2, ..., e_k\}$. В алфавит E входят события файловой системы, сетевые события, события запуска процесса и т.д. В каждый момент времени $t \in T$ каждому объекту $o \in O$ может быть поставлена в соответствие последовательность символов $e^{o,t} = \left(e_1^{o,t}, e_2^{o,t}, ..., e_l^{o,t}\right)$, $e_i^{o,t} \in E$, $i = \overline{1,l}$, которую назовем совокупностью событий, ассоциированных с объектом o в момент времени t.

Алфавит событий может быть построен независимо от множества объектов в информационной системе, однако реализация событий, ассоциированных с конкретными объектами, приводит к порождению элементов данных, которые в общем случае различны для различных объектов. Обозначим совокупность элементов данных, порожденных совокупностью событий $e^{o,t}$ как $d^{o,t} = \left(d_1^{o,t}, d_2^{o,t}, ..., d_l^{o,t}\right)$, $d_i^{o,t} \in A^*$, где A^* – множество всевозможных двоичных последовательностей конечной длины. Заметим, что если $e^{o,t} = \varnothing$, то и $d^{o,t} = \varnothing$.

Тогда задачу обнаружения ВПО в информационной системе определим как построение классификатора следующего вида:

$$F : \left(o, e^{o,t}, d^{o,t}\right) \mapsto c \in \{0,1\}, \tag{1}$$

где $c = 0$, если $o \in O_S$, и $c = 1$, если $o \in O_M$.

Данный классификатор извлекает признаковые характеристики анализируемого объекта из совокупности событий, ассоциированных

с данным объектом, и порожденной этими событиями совокупности элементов данных. Очевидно, что при этом время, за которое система обнаружения ВПО выносит решение, должно быть минимизировано.

2.1. Извлечение признаков и методы обнаружения ВПО. На рисунке 1 приведена классификация извлекаемых признаков ВПО, которыми могут оперировать системы обнаружения ВПО [6, 7].

Рис. 1. Классификация извлекаемых признаков ВПО

Первой группой признаков являются статические признаки, которые можно извлечь без запуска файлов. Основной список таких признаков описан в работе [6], содержащей обзор подходов к детектированию исполняемых файлов ВПО с использованием методов интеллектуального анализа данных. Для решения более общей задачи обнаружения ВПО (без привязки к исполняемым файлам) актуализируем способы извлечения основных статических признаков ВПО:

1. Получение уникальных строковых данных [6]. Данный способ извлечения признаков сводится к экспертной оценке аналитиком ВПО уникальности определенных строк, которые являются осмысленными. Например, в файле ВПО присутствует URL-ссылка для загрузки других модулей ВПО с командного центра злоумышленников. Как правило, данная ссылка задается в виде строки без применения методов шифрования или кодирования. С учетом того, что ссылка содержит сетевой адрес сервера злоумышленников и путь к конкретным файлам, маловероятно появление такой строки в файлах, не относящихся к ВПО.

2. Выделение ключевых байт. Данный способ похож на первый, однако в отличие от него вместо уникальных интерпретируемых строк использует байты или их сочетания. Извлечение байтовых признаков происходит от метода общего анализа n-грамм, который широко используется при обработке текстов на

естественных языках [25]. Важной особенностью данного способа является возможность использования в качестве ключевых байт любых данных (в том числе и бинарных данных исполняемого файла).

3. Выделение особенностей опкодов инструкций процессора (Operational Code, OpCode). Данный способ можно назвать частным случаем предыдущего, однако в нем байты представлены в виде исполняемых опкодов. Например, в работе [26] ведется подсчет частоты использования определенных инструкций процессора для легитимных программ и ВПО. Стоит отметить, что этот способ работает только для исполняемых скомпилированных файлов.

4. Разбор структуры файла: извлечение метаданных файла, а также основных структурных признаков в зависимости от типа файла [27]. Анализ структурных признаков файла помогает обнаружить подозрительные типы объектов внутри файлов, в том числе скриптов и неисполняемых документов (например, признаки «AutoOpen» для макросов или OLE-объекты типа «OLE2link»).

5. Изучение особенностей функций исполняемых файлов [28]. Функцией называют последовательность инструкций процессора, которые образуют единую логическую конструкцию с единой точкой входа и одной или несколькими точками выхода. Данная конструкция состоит из базовых блоков (линейных блоков исполняемого кода без инструкций передачи управления) [29]. Особенности функций используют для обнаружения ВПО.

Данный список извлекаемых файловых признаков не является исчерпывающим, но покрывает значительное число основных статических признаков. Стоит отметить, что при извлечении только статических признаков невозможно детектировать определенные виды атак с использованием ВПО: например, бесфайловые атаки [30] или атаки с применением обфусцированных исполняемых образцов [31, 32, 33]. Для решения данной проблемы были разработаны динамические (поведенческие) подходы к детектированию ВПО. Они основаны на извлечении динамических признаков работы процессов ОС, среди которых могут присутствовать процессы ВПО. Поэтому в данном случае для проверки степени вредоносности файла требуется его запуск. К способам извлечения динамических признаков процессов ОС относят [34, 35, 36]:

1. Перехват событий ОС или API-вызовов. Согласно работе [5], к данной категории динамических признаков относятся записи событий аудита. Позже с расширением возможностей по перехвату API-функций ОС и распространением виртуальных машин

для обнаружения ВПО стали использоваться системные вызовы и их аргументы [34].

2. Выделение атрибутов сетевого трафика (например, IP-адреса, DNS-имена, используемые сетевые порты) или байт содержимого трафика. В 1999 году появился первый промышленный стандарт написания сетевых сигнатур и сетевая система обнаружения вторжений (СОВ) Snort [37], основанная на применении регулярных выражений и сочетаний определенных байтов. В данной системе были реализованы принципы работы СОВ, сформулированные в [5].

3. Вычисление метрик использования ресурсов ОС согласно принципам [5] (например, большое количество событий файловой системы может свидетельствовать о работе ВПО типа TrojanRansom [6]).

Как было отмечено ранее, после извлечения признаков для обнаружения ВПО применяются методы детектирования ВПО. Первым наиболее очевидным методом детектирования ВПО является сигнатурный [24], который сводится к сравнению извлеченных значений признаков с сигнатурами [5]. В настоящее время широкое распространение получил общедоступный сканер файлов и одноименный стандарт написания правил детектирования YARA [38, 39]. Большинство вендоров антивирусного ПО при публикации своих исследований используют данный стандарт для обмена информацией об образцах ВПО. Путем декомпозиции сигнатурного метода на несколько первичных [24] выделим следующие методы детектирования ВПО, основанные на работе с определенными классами признаков:

1. Сравнение найденных уникальных строк со строками, характерными для ВПО и собранными в специальной базе строковых сигнатур.

2. Разбиение байт на n-граммы, построение статистики по n-граммам и проведение частотного анализа n-грамм. Преимущество данного метода состоит в том, что он предоставляет возможность сопоставления функционала ВПО и n-грамм и характеризуется более высокой полнотой обнаружения по сравнению с оценкой частоты встречаемости определенных файловых признаков [25]. Частным случаем данного метода детектирования является метод, основанный на изучении особенностей последовательностей опкодов. Например, в работе [32] проводится выявление определенных последовательностей скомпилированного бинарного кода на основе методов филогенеза, используемых в биологии. Таким образом, в данном подходе к обнаружению ВПО используется сочетание двух способов извлечения статических признаков.

3. Подсчет энтропии разделов файла и их сравнение со значениями, характерными для разделов файлов ВПО. После разбора структуры файла целесообразно рассчитать значение энтропии секции с исполняемым кодом. Если в файле применяется кодирование, то уровень энтропии будет достаточно высоким [31].

4. Графовый анализ. Данный метод применим к сущностям, у которых есть точка входа и точка (или точки) выхода. В случае исполняемых файлов такой сущностью является функция. На основе данного метода строится граф потока управления (Control Flow Graph, CFG) [40], который широко используется для обнаружения исполняемых файлов ВПО, в том числе образцов ВПО с возможностью самомодификации (полиморфное/метаморфное ВПО) [41].

5. Вычисление функций схожести. В данном методе используется сравнение характеристик ВПО с эталонным значением с помощью определенных математических функций. Характерным примером применения данного метода является использование нечетких хешей [33, 41]. Особым направлением исследований стал подход, изложенный в работе [42]. В ней исследователи предложили свести задачу классификации исполняемых файлов к их визуализации в виде изображений, к которым затем применялись методы вычисления функций схожести.

Более подробное описание перечисленных методов детектирования ВПО представлено в обзорных работах [6, 7, 8, 9]. Известно, что сигнатурные методы обнаружения ВПО на основе динамических признаков достаточно широко распространены в системах СОВ, специализированных изолированных средах – песочницах [46], а также решениях класса Endpoint Detection and Response (далее – EDR) [47] на хостах и серверах под управлением ОС Windows. Однако, несмотря на распространенность, указанные методы обладают определенными ограничениями. Необходимость работы в реальном времени предъявляет повышенные требования к производительности работы операционной системы, что сказывается на полноте и точности обнаружения ВПО.

Для решения проблем, которые присущи вышеперечисленным методам детектирования ВПО, стали широко использоваться методы интеллектуального анализа данных. В работе [6] приводится классификация таких методов, используемых в задаче обнаружения ВПО: деревья решений (Decision Tree, DT), байесовские классификаторы (Naive Bayes, NB: наивный байесовский классификатор и мультиномиальный наивный байесовский классификатор). Также за последние 20 лет широкое распространение

получили различные алгоритмы машинного обучения: метод опорных векторов (Support Vector Machine, SVM), метод случайного леса (Random Forest, RF), метод k-ближайших соседей (K-Nearest Neighbor, k-NN), искусственные нейронные сети (Artificial Neural Network, ANN) и глубокое обучение (Deep Learning, DL) [48, 49], различные алгоритмы биоинформатики [50], в том числе и иммунные системы (Artificial Immune System, AIS) [51, 52]. Стоит отдельно выделить методы, в которых совмещены алгоритмы машинного обучения и методы анализа временных рядов [53].

Важным аспектом применения алгоритмов машинного обучения является зависимость их эффективности от качества и объема обучающей и тестовой выборок. Большинство исследователей берут готовые наборы данных для обучения своих моделей. Применительно к задаче обнаружения ВПО такой подход компрометирует практическую ценность соответствующих работ, так как при формировании набора данных по файловым или поведенческим характеристикам ВПО исследователи должны доверять результатам работы других авторов, которые занимались сбором этих данных. Поэтому отдельной задачей для исследователей является создание доверенных наборов данных.

2.2. Наборы данных для обучения моделей детектирования ВПО. Наибольшую ценность для исследователей, использующих готовые выборки, представляет работа [18]. В ней авторы исследовали вопросы планирования экспериментов с ВПО. В частности, им удалось выработать критерии качества наборов данных и требования к прозрачности проведения самого эксперимента. К критериям качества наборов данных исследователи отнесли отсутствие в обучающей выборке легитимных файлов, помеченных как ВПО (A.1), сбалансированность выборки (A.2), правильное разделение выборки на обучающую и тестовую (A.3), преобладание привилегий системы сбора данных над привилегиями ВПО (A.4), соответствие среды сбора данных реальным условиям работы алгоритма классификации (A.5) и наличие легитимной фоновой активности (A.6). Таким образом, при использовании готового набора данных исследователи должны убедиться в его соответствии изложенным в работе [18] требованиям A.1–A.6.

Как показывает обзор литературы, большинство известных наборов данных, используемых для обучения моделей детектирования ВПО, не удовлетворяют приведенным требованиям. Кроме того, многие исследователи опускают детали решений, которыми руководствуются при формировании используемых ими наборов

данных. К примеру, в работе [43] использовался набор данных, именуемый «malimg» [54]. Данный набор данных сформирован на основе системы Anubis [55] – одной из первых облачных систем анализа исполняемых файлов. После обработки данных в системе Anubis исследователи произвели отображение бинарных файлов вредоносных программ в наборы 8-битных целых чисел без знака, представленные в виде матриц. Далее полученные матрицы были визуализированы как изображения в градациях серого. Несмотря на достаточно подробное описание алгоритма получения изображения из исполняемого файла, в [43] не обоснован выбор семейств вредоносных файлов для формирования выборки. Описывая набор данных, исследователи упомянули различные алгоритмы сжатия исполняемого кода (например, UPX), но не сопоставили между собой сходства и отличия исследуемых семейств ВПО. Из общего объема выборки (9342 образца ВПО и 25 рассмотренных семейств ВПО) около половины (4540 исполняемых файлов) приходится на 2 подсемейства: Allaple.A и Allaple.L, что нарушает сбалансированность выборки (A.2).

Другим частым случаем нарушения требований к набору данных является отсутствие преобладания привилегий системы сбора данных над привилегиями ВПО (A.4). В основном данное требование характерно для наборов данных, собранных в песочнице Cuckoo Sandbox – проекте с открытым исходным кодом [56]. Это обусловлено архитектурной особенностью данной песочницы: для мониторинга активности исследуемых файлов используются методы API-перехвата в пользовательском режиме. Одним из примеров набора данных, полученного с использованием песочницы Cuckoo Sandbox, является ransomwaredataset2016 [57]. Этот набор данных использовался в работе [35] по автоматизации анализа ВПО класса криптовымогателей (Ransomware), а также в работе [36] по обнаружению данного класса ВПО с использованием алгоритма роя частиц. Другим примером набора данных, полученным с помощью Cuckoo Sandbox, является MalwareDataset [58]. Этот набор данных исследовался в статье [11] в задаче классификации похожих образцов ВПО на основе n-грамм последовательностей вызовов API-функций.

Таким образом, статические и динамические признаки файлов и процессов ОС имеют свои особенности в решении задачи обнаружения ВПО: статические признаки файлов начали использоваться раньше динамических и их использование незначительно сказывалось на работе ОС Windows, на которой производилась антивирусная проверка. Однако использование только статических признаков не позволяет достичь такой же полноты

обнаружения, как и у динамических признаков ВПО, так как существуют полиморфные семейства ВПО и обфускация. По этой причине необходимо использовать динамические признаки процессов ОС Windows, совмещая их с методами машинного обучения для повышения полноты и точности обнаружения ВПО. Для полноценного применения машинного обучения нужны достоверные наборы данных, которые в открытом доступе отсутствуют.

3. Модели описания цепочки действий злоумышленника. Для классификации ВПО часто используется еще один класс признаков – наличие определенных техник атаки в программном коде (например, сжатие или обфускация исполняемого кода). Для описания техник атаки используются различные модели. Использование этих моделей в работе центра мониторинга кибербезопасности позволяет выстроить процесс выявления активности злоумышленников, а также их атрибуции [59]. Наиболее ранняя модель описания цепочки действий злоумышленника получила название Cyber Kill Chain [60]. Она описывает этапы действий злоумышленника: разведку, подготовку атаки, доставку ВПО, эксплуатацию уязвимости, инсталляцию ВПО, получение управления и реализацию риска. Другой моделью для описания вторжений стала Diamond Model [61]. В ней большое внимание уделялось описанию злоумышленника, его возможностей, используемой инфраструктуры и атакуемой жертвы. Однако все эти модели оказались недостаточными для того, чтобы описать наиболее важные аспекты атаки. Например, модель Cyber Kill Chain не конкретизирует действия, происходящие на определенных стадиях атаки, и используемое ВПО. Diamond Model вообще не подразумевает наличие требований к описанию действий злоумышленника. По этой причине в 2016 году организацией MITRE была создана многомерная модель MITRE ATT&CK [19], основой которой являются техники, тактики и процедуры атакующих (далее – TTP: techniques, tactics and procedures). Данная модель регулярно пополняется на основе открытых источников информации, в которых присутствуют описания инструментов атаки и особенностей их использования злоумышленниками.

3.1. Матрица MITRE ATT&CK. В настоящее время модель MITRE ATT&CK получила наиболее широкое распространение по сравнению с прочими моделями. Первым исследованием, основанным на применении данной модели, была работа [62]. MITRE ATT&CK является матрицей, представляющей почти готовую методику по классификации признаков присутствия злоумышленников в сети организации на основе определенных тактик, техник и процедур,

реализуемых злоумышленниками. Стоит отметить, что матрица MITRE ATT&CK не является матрицей в математическом смысле. Вероятнее всего, матрицей стали называть двухмерную классификацию действий злоумышленника для удобства восприятия. В MITRE ATT&CK по горизонтали первой строки перечислены тактики, которые являются определенными этапами атаки (они пересекаются с этапами атаки в соответствии с Cyber Kill Chain [59]). Вертикальными ячейками матрицы являются техники – любые значимые действия атакующего. Для решения задачи выявления ВПО важно определение того этапа жизненного цикла атаки, на котором происходит обнаружение вредоносных файлов или подозрительной активности (соответствие принципу минимизации времени инцидента).

Рассмотрим пример инцидента с банковским трояном Emotet [63]. Схема атаки в соответствии с MITRE ATT&CK приведена на рисунке 2.

Данное ВПО является достаточно показательным в контексте рассматриваемой задачи. Приведенная схема охватывает не все техники и тактики атаки, предусмотренные MITRE ATT&CK, но наиболее значимые из них. Важно, что ВПО Emotet используется для доставки ВПО других семейств, то есть является загрузчиком. Обычно первыми двумя этапами атаки считаются разведка (Reconnaissance) и подготовка инфраструктуры (Resource Development). Однако они происходят на стороне злоумышленника и по этой причине для детектирования ВПО интереса не представляют, поэтому в отчете об инциденте они не приведены. Первый видимый на стороне атакуемой инфраструктуры этап первоначального доступа (Initial Access) производился с помощью рассылки писем, содержащих вредоносный LNK-файл (техника «Phishing»). Следующие два этапа выполнение (Execution) и закрепление (Persistence) позволяют ВПО запуститься и закрепиться в системе, поэтому первые три этапа на приведенной выше схеме важны для обнаружения и предотвращения дальнейшей работы ВПО. Обычно их старается предотвратить большинство антивирусных программ, перехватывая событие создания нового процесса и сканируя запускаемый образ. Далее идут этапы повышения привилегий (Privilege Escalation) и предотвращение обнаружения (Defense Evasion), на которых злоумышленник повышает свои привилегии и противодействует средствам детектирования. В данном случае была использована техника внедрения кода (Process Injection) в процесс Winlogon, которая заключается в запуске вредоносного кода в контексте другого процесса для обхода детектирования и повышения привилегий для работы от системной учетной записи. Данный пример демонстрирует, что иногда техника позволяет добиться сразу двух

целей: в таком случае эта техника попадет сразу в две колонки, соответствующие тактикам.

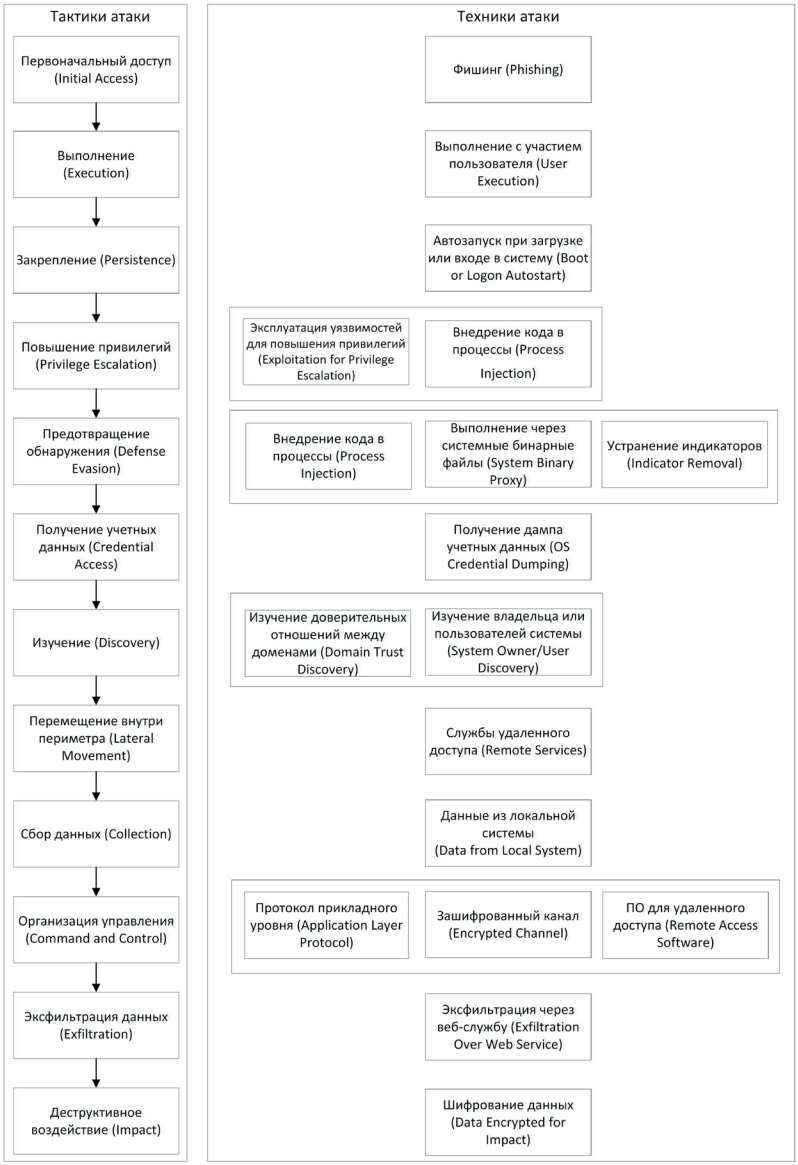

Рис. 2. Пример схемы атаки в соответствии с MITRE ATT&CK

После этих действий обычно наступают дальнейшие изучение (Discovery), захват корпоративной сети (Credential Access, Lateral Movement, Command and Control) и деструктивное воздействие (Impact). На примере инцидента с ВПО Emotet в конечном счете было запущено вымогательское ВПО Quantum. Если злоумышленник смог пройти данные этапы, это значит, что он смог скомпрометировать компьютерную сеть.

Для обнаружения ВПО на определенных этапах атаки требуется механизм сопоставления признаков проявления ВПО с данными из заранее собранной базы. Исследователи из компании Mandiant [64] создали такую базу сигнатур и специальный инструмент *capa* [65], который выводит список используемых в исполняемом коде ВПО техник атаки в соответствии с MITRE ATT&CK. Однако данный подход имеет существенные ограничения ввиду того, что он работает только для исполняемых файлов.

3.2. SIEM-системы и Sigma-правила. Как было отмечено ранее, подходы к детектированию ВПО на основе статических признаков имеют существенные ограничения в виде невозможности обнаружения бесфайлового ВПО и обфусцированных вредоносных программ. В настоящее время в корпоративных сетях используется централизованный подход к получению данных о поведенческих особенностях работы клиентских и серверных систем. На узлах сети помимо антивирусного решения зачастую используется хостовой агент сбора системных событий в виде EDR-решения [47]. Полученные события записываются в журналы, которые собираются и централизованно обрабатываются в SIEM-системе (Security Information and Event Management – управление событиями и информацией о безопасности) [66]. Данная система служит для нормализации, агрегации и корреляции событий с различных источников корпоративной сети (например, антивирусов, межсетевых экранов, средств обнаружения вторжений). Корреляция событий происходит по специальным правилам, которые являются поведенческими сигнатурами [66].

Большинство SIEM-систем имеют свой стандарт описания правил корреляции. В настоящее время существует проект Sigma [20], направленный на унификацию поисковых запросов и соответствующих правил корреляции для популярных SIEM-систем. Синтаксис правил Sigma соответствует логике правил обнаружения вторжений, описанных в [5]. Важным параметром качества правил является степень покрытия матрицы MITRE ATT&CK этими правилами [67]. В рамках текущего исследования сформулирована

следующая гипотеза: сбор данных об активности ВПО можно автоматизировать путем использования средства сбора событий активности процессов ОС, а журналы работы данного средства целесообразно обрабатывать с помощью правил, поддерживающих синтаксис Sigma и привязку к MITRE ATT&CK. Таким образом, формирование набора данных об активности ВПО возможно путем сбора событий активности процессов ОС и последующей их обработки с использованием Sigma-правил, позволяющих построить соответствие «событие ОС → техника атаки».

4. Механизмы мониторинга активности процессов ОС Windows. Динамические признаки позволяют добиться более высокой эффективности обнаружения ВПО по сравнению со статическими. В качестве источников данных в ОС Windows для извлечения динамических признаков используются различные категории событий:

1. События, генерируемые на основе перехвата API-функций (API Hooking) [68]. Данный механизм является наиболее информативным для задач кибербезопасности, однако не относится к штатным: его применение в качестве основного механизма мониторинга активности процессов ОС Windows несет определенные проблемы. Во-первых, такой проблемой является зависимость от версии сборки ОС Windows в случае использования перехвата API-функций в режиме ядра [23]. Во-вторых, данный механизм уязвим к атакам типа «обход защиты» (Defense Evasion согласно MITRE ATT&CK [19]) в случае использования перехвата API-функций в пользовательском режиме [23].

2. События, генерируемые агентом или внутренним механизмом гипервизора песочницы для поведенческого анализа файлов. Наиболее известным общедоступным проектом, в котором реализован данный механизм, является песочница Cuckoo Sandbox [56]. Другим популярным общедоступным решением является drakvuf [69]. Однако такой подход к мониторингу активности процессов ОС Windows возможен только на виртуальных машинах или гипервизорах.

3. События, генерируемые специализированными драйверами с определенными функциями обратного вызова (например, драйверы-минифильтры файловой системы). Широко используются в антивирусных программах, так как данный механизм расширяет возможности по сбору событий и позволяет производить активное реагирование [70]. Имеет существенный недостаток: предоставляет информацию по ограниченному количеству API-функций.

4. События ОС Windows из стандартных журналов System, Security, PowerShell и других [5]. Для использования данных журналов необходимо включить определенные типы событий аудита ОС. Однако эти журналы являются недостаточно информативными для задач кибербезопасности. При этом необходимо отметить, что события ОС Windows в общем случае записываются в журналы благодаря механизму Event Tracing for Windows [71]. Данный механизм является гораздо более гибким, но в то же время сложным и слабо документированным. В настоящее время активно изучается исследователями кибербезопасности, но пока используется недостаточно широко.

Последние два механизма реализованы в бесплатной утилите Sysmon [72] из набора Sysinternals. Данная утилита не является полнофункциональным инструментом аналитики, а лишь предоставляет возможность записи определенных событий в соответствующий журнал. Перечень фиксируемых событий задается специальным конфигурационным файлом перед установкой утилиты в качестве службы ОС Windows. Кроме того, имеется возможность журналировать только те события, которые соответствуют определенным условиям, задаваемым аналитиком.

5. Методика сбора и разметки данных об активности процессов ОС Windows. Несмотря на попытки сбора данных об активности вредоносных программ с использованием утилиты Sysmon и публикации соответствующего общедоступного набора данных [73], исследователи до сих пор не создали в открытом доступе набор размеченных цепочек событий, соответствующих ВПО и легитимным файлам. В данном исследовании предлагается использовать вышеперечисленные механизмы мониторинга активности процессов ОС Windows в составе оригинальной методики формирования воспроизводимого набора данных об активности процессов ОС Windows с возможностью дальнейшей автоматизации разметки данных на основе сигнатур Sigma и матрицы MITRE ATT&CK. Таким образом, решается проблема обеспечения достоверности сбора данных о поведенческих признаках ВПО.

Представим формальное описание решаемой задачи с использованием обозначений, введенных в разделе 2. Пусть $O = \{o_1, o_2, ..., o_m\}$ – множество объектов-файлов, имеющихся в информационной системе, и $T = \{t_1, t_2, ..., t_n\}$ – множество моментов дискретного времени. Рассмотрим множество совокупностей событий, ассоциированных с объектами из множества O в момент времени t,

и множество совокупностей порожденных этими событиями элементов данных: $e^{o_1,t}$, $d^{o_1,t}$, $e^{o_2,t}$, $d^{o_2,t}$, …, $e^{o_m,t}$, $d^{o_m,t}$. Некоторым образом осуществим разбиение данных совокупностей на составляющие $g_{o'=\{o_i,\ldots,o_p\}} = \left(\ldots,\left(e_j^{o_i,t}, d_j^{o_i,t}\right),\ldots,\left(e_q^{o_p,t}, d_q^{o_p,t}\right),\ldots\right)$, включающие в себя события, ассоциированные с разными объектами из множества O, и соответствующие данные. Данные составляющие назовем цепочками событий. Тогда задача формирования набора данных об активности вредоносных программ в информационной системе будет состоять в построении классификатора следующего вида:

$$V : g_{o'=\{o_i,\ldots,o_p\}} = \left(\ldots,\left(e_j^{o_i,t}, d_j^{o_i,t}\right),\ldots,\left(e_q^{o_p,t}, d_q^{o_p,t}\right),\ldots\right) \mapsto c \in \{0,1\}, \qquad (2)$$

где $c = 0$, если $O' \cap O_S = \varnothing$, и $c = 1$ в противном случае.

Необходимо отметить, что отличие от задачи обнаружения ВПО заключается в том, что классификатор V не относит отдельно взятые объекты к вредоносному или легитимному ПО, а определяет наличие или отсутствие вредоносной активности в совокупности событий, ассоциированных с некоторым подмножеством множества объектов O.

Как было отмечено ранее, для сбора данных об активности ВПО целесообразно использовать средства виртуализации для изоляции вредоносного воздействия на реальные вычислительные ресурсы и сетевое окружение. Для этого необходимо подготовить виртуальную машину, в том числе произвести установку типичной программной среды пользователя (редакторы офисных файлов, браузеры, файловые менеджеры, архиваторы), настройку сетевых интерфейсов и снятие снимка состояния [74]. Также необходимо произвести установку и конфигурацию специализированных программных средств для сбора данных об активности процессов ОС Windows.

Таким образом, решение поставленной задачи применительно к ОС Windows может быть представлено в виде методики, включающей четыре основных этапа:

1. Управление состояниями ВМ.
2. Запись журналов активности процессов ОС Windows.
3. Сбор журналов активности процессов ОС Windows.
4. Хранение и обработка журналов.

Приведенные этапы целесообразно выполнять с использованием соответствующих программных средств, как это показано на

рисунке 3. На данной схеме слева показаны этапы, выполняемые на стороне хостовой ОС, а справа – этапы, выполняемые внутри гостевых ОС виртуальных машин.

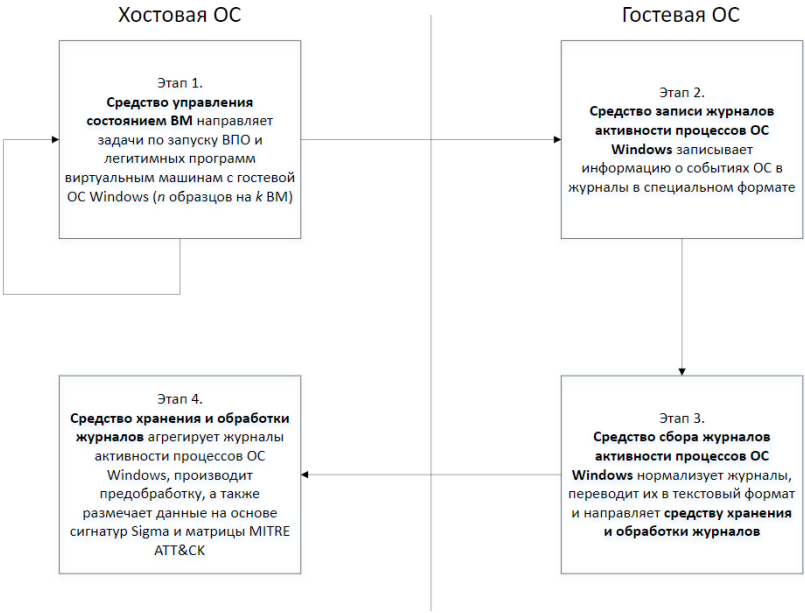

Рис. 3. Методика сбора и разметки данных об активности процессов ОС Windows

Первые три этапа предполагают проведение вычислительного эксперимента по запуску ВПО и легитимных программ для сбора данных об их активности. Последний этап необходим для разметки событий активности исследуемых процессов. Далее рассмотрим этапы предлагаемой методики более подробно.

5.1. Сбор данных об активности ВПО и легитимных программ. На первом этапе сбора данных об активности процессов ОС Windows производится распределение задач по запуску ВПО и легитимных программ по виртуальным машинам. Для этого используется программное средство управления состоянием виртуальных машин и вызова файлов ВПО и легитимных программ (рисунок 4). Поскольку в реальных сценариях реализации киберугроз с использованием ВПО запуск ВПО может произойти в любой момент времени после начала работы системы, то при формировании задач

по запуску ВПО и легитимных программ на виртуальных машинах время запуска должно выбираться случайным образом.

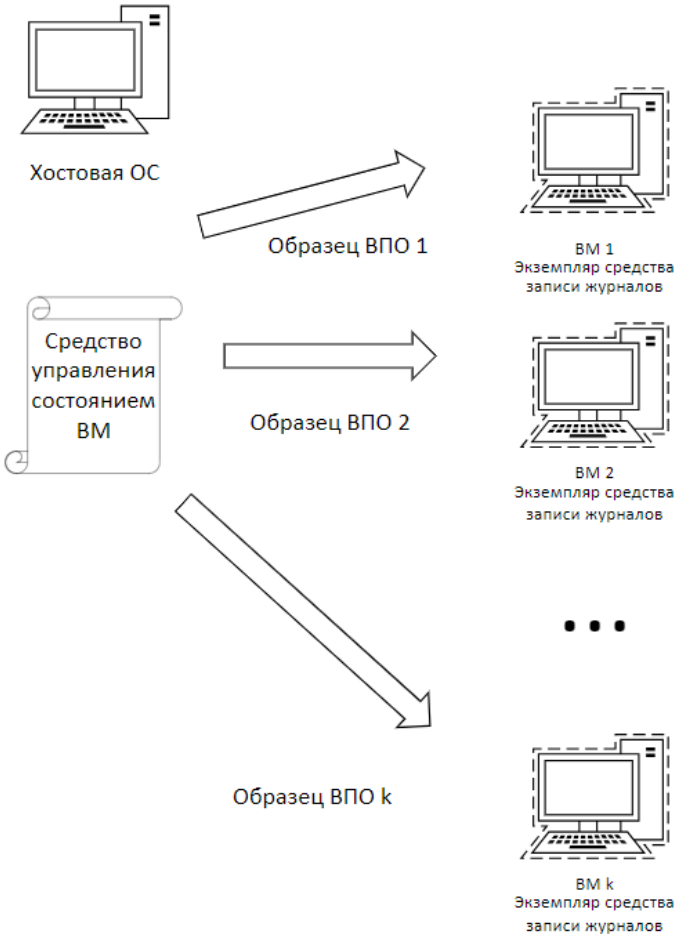

Рис. 4. Распределение задач по анализу образца ВПО и легитимных программ между виртуальными машинами

На втором этапе сбора данных об активности процессов ОС Windows осуществляется запись событий ОС в журналы. Для этого требуется заранее произвести конфигурацию средства записи журналов активности процессов ОС Windows, чтобы обеспечить мониторинг необходимых событий и исключить наименее важные

с точки зрения кибербезопасности события. Необходимые поля событий приведены в разделе 5.2, а пример конфигурации описан в разделе 6.1.

На третьем этапе производятся синтаксический разбор, нормализация и пересылка журналов средству хранения и обработки журналов. По истечении определенного для каждой виртуальной машины таймаута средство управления состоянием виртуальных машин должно остановить выполнение текущей задачи, выполнить откат конкретной ВМ к сделанному на этапе подготовки эксперимента снимку состояния и направить новую задачу. По этой причине средство управления состоянием виртуальных машин должно работать асинхронно для каждой ВМ. Описанные этапы методики схематично изображены на рисунке 5.

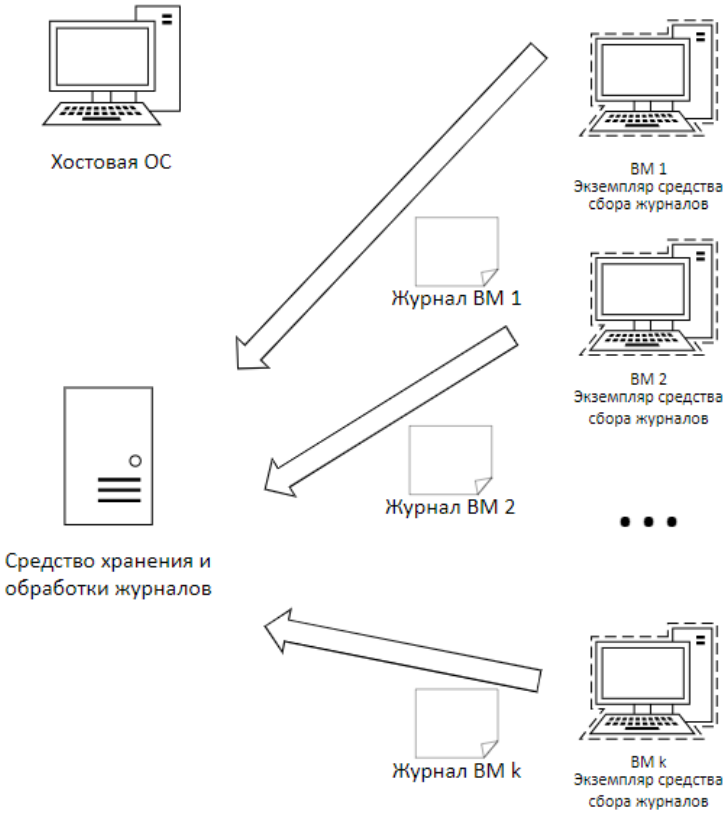

Рис. 5. Сбор и обработка журналов активности процессов ОС Windows

5.2. Разметка событий активности процессов ВПО и легитимных программ. Как было отмечено ранее, четвертый этап работы методики относится к разметке событий активности процессов ВПО и легитимных программ на основе Sigma-правил и матрицы MITRE ATT&CK. До проведения разметки данных необходимо проверить корректность и непротиворечивость проверяемых событий и используемых Sigma-правил, в том числе и на отсутствие синтаксических ошибок. Для проверки записанных журналов активности процессов ОС Windows сигнатурами Sigma необходимо соблюдать соответствие полей записанных событий и полей, используемых в Sigma-правилах. Поля записи событий активности процессов ОС Windows, необходимые для корректной работы Sigma, приведены в таблице 1.

Таблица 1. Поля событий, необходимые для проверки Sigma-правилами

Содержимое поля события	Назначение поля события
Тип и название события	Идентификация действия
Имя и сессия пользователя	Идентификация пользователя, от имени которого работает процесс ОС Windows
Идентификатор процесса, полное имя файла и хеш исполняемого образа процесса	Идентификация экземпляра и образа процесса
Идентификатор родительского процесса и полное имя файла образа родительского процесса	Идентификация экземпляра и образа родительского процесса
Командная строка, с помощью которой был создан данный процесс ОС Windows	Идентификация условий, при которых был запущен процесс
Идентификатор и имя объекта, над которым выполняется действие	Идентификация объекта
Запрашиваемые права для действия над объектом	Идентификация запрашиваемых прав доступа процесса к объекту
Дополнительные поля	Любые дополнительные поля, специфичные для определенных действий (например, адрес памяти старта потока для события CreateRemoteThread)

Приведенная таблица позволяет корректно сконфигурировать средство записи журналов активности процессов ОС Windows перед запуском вычислительных экспериментов и затем произвести

проверку записанных событий Sigma-правилами. После указанной проверки необходимо обогатить события новыми полями, содержащими информацию о результатах проверки событий соответствующими Sigma-правилами и приведенными в таблице 2. По умолчанию данные поля должны быть предзаполнены пустыми значениями, в которые вносятся изменения только в случае положительного результата проверки события хотя бы одним Sigma-правилом.

Таблица 2. Новые поля событий по итогам разметки

Содержимое поля события	Назначение поля события
Булево значение, если исполняемый образ процесса соответствует запускаемому образцу ВПО (для запускаемых легитимных программ равно 0, для образцов ВПО равно 1)	Идентификация реальных техник атаки злоумышленника с использованием ВПО
Идентификаторы техник атаки на основе метаданных Sigma-правил	Идентификация потенциальных техник атаки злоумышленника
Имена Sigma-правил	Идентификация Sigma-правил

Разметка считается успешно выполненной, если для всех событий заполнены все поля из приведенной таблицы.

Данная методика работы в целом соответствует логике работы Cuckoo Sandbox, однако в оригинальной методике предлагается использовать штатные механизмы мониторинга активности процессов ОС Windows и применять унифицированные Sigma-правила для обнаружения ВПО, а не специфичные для функционирования Cuckoo Sandbox скрипты [75]. Таким образом, предлагаемая методика позволяет различным исследователям самостоятельно формировать наборы данных по активности ВПО и использовать собственные или общедоступные Sigma-правила для разметки этих данных. Стоит отметить, что репозиторий Sigma-правил постоянно пополняется новыми правилами, а значит новые техники атаки могут быть обнаружены с помощью данных правил.

6. Проведение экспериментов по сбору данных об активности процессов ОС Windows. Для проведения экспериментов с предлагаемой методикой была развернута и подготовлена виртуальная среда на базе VirtualBox с двумя гостевыми ОС Windows 10. С учетом динамического характера методики

целесообразно учесть определенные ограничения. На результаты экспериментов с запуском ВПО могут влиять особенности изолированной среды: время работы гостевых ОС и степень их загруженности. По этой причине необходимо производить запуск одного файла на нескольких гостевых ОС. Однако с учетом достаточно высоких требований к работе современных ОС Windows на среднем ПК с 8 Гбайт ОЗУ и 4 ядрами процессора возможно запустить не более двух гостевых ОС без существенной потери производительности хостовой ОС. Для снижения влияния временных характеристик на результаты экспериментов были использованы две идентичные гостевые ОС Windows 10, на которые было выделено разное количество ресурсов хоста: на виртуальную машину $ВМ_1$ было выделено 2 ядра процессора, 4 Гбайт ОЗУ и 70 Гбайт ПЗУ, а на виртуальную машину $ВМ_2$ – одно процессорное ядро, 2 Гбайт ОЗУ и 30 Гбайт ПЗУ.

Для управления состояниями виртуальных машин и распределения задач по запуску ВПО было разработано отдельное программное средство на языке Python 3 с использованием библиотеки *virtualbox*. В разработанном программном средстве реализован функционал выполнения команд по запуску файлов ВПО на виртуальных машинах и откат этих виртуальных машин к первоначальному состоянию по таймауту.

6.1. Подготовка виртуальной среды к проведению экспериментов. В качестве средства записи журналов активности процессов ОС Windows целесообразно выбрать инструмент Sysmon, так как он использует штатные механизмы ОС Windows для перехвата системных событий на уровне ядра ОС Windows и отличается наибольшей гибкостью конфигурирования. Для проведения экспериментов конфигурация службы Sysmon настроена следующим образом: включен сбор всех возможных событий, кроме событий с EventID 27 (блокировка создания исполняемых PE-файлов), так как данные события не только отслеживаются, но и блокируются, что неприемлемо для проведения экспериментов. Дополнительно из мониторинга исключаются события, сгенерированные необходимыми для экспериментов утилитами Sysmon и Elastic Agent и процессом виртуальной машины VirtualBox. Конфигурация Sysmon, использованная в настоящем исследовании, опубликована на GitHub [76].

Для хранения и обработки журналов активности процессов ОС Windows рекомендуется использовать любую SIEM-систему или систему обработки больших данных. Бесплатным программным

средством обработки больших данных является ELK-стек, включающий в себя компоненты Elasticsearch, Logstash и Kibana [77]. В рамках экспериментов за основу была взята и адаптирована конфигурация для инсталляции в контейнере docker-elk [78].

Для автоматизированного сбора журналов активности процессов ОС Windows целесообразно использовать готовый программный инструмент Elastic Agent, который имеет API-интеграцию с ELK-стеком [79]. Данное программное средство позволяет производить парсинг журналов событий Sysmon, осуществлять нормализацию данных и пересылать информацию по сети на веб-сервис Elasticsearch. Файл конфигурации для данной утилиты можно сгенерировать из самой веб-панели управления Kibana и затем инсталлировать его на виртуальных машинах для запуска ВПО.

Далее необходимо производить разметку полученных событий Sysmon с помощью Sigma-правил. Для этого была установлена утилита Sigma-cli, позволяющая переводить Sigma-правила в запросы для ELK [80]. Помимо возможности выявления вредоносных событий Sysmon под ОС Windows на базе стека ELK, данная утилита поддерживает возможность перевода Sigma-правил для обнаружения аномальной сетевой активности, подозрительных событий ОС Linux и ОС macOS для систем типа Splunk и большинства распространенных SIEM-решений.

Полный перечень установленного ПО приведен в таблице 3.

Таблица 3. Установленное ПО для проведения экспериментов

Класс ПО	**Локация**	**Название ПО**	**Версия**
Хостовая ОС	Хост	Windows 10	21H2 19044.2486
Средство виртуализации	Хост	VirtualBox	6.1.46 r158378
Средство управления состоянием ВМ	Хост	Python3 virtualbox (библиотека python)	Python 3.10.2 virtualbox 2.1.1
Гостевые ОС	Виртуальная машина (для запуска ВПО)	Windows 10 (для $ВМ_1$ и $ВМ_2$)	21H2 19044.2486
Офисное ПО	Виртуальная машина (для запуска ВПО)	Microsoft Office	2007
ПО для просмотра PDF файлов	Виртуальная машина (для запуска ВПО)	Adobe Reader	8
Браузер	Виртуальная машина (для запуска ВПО)	Internet Explorer	8

Продолжение Таблицы 3

Класс ПО	Локация	Название ПО	Версия
Средство записи системных журналов	Виртуальная машина (для запуска ВПО)	Sysmon Sysinternals	15, версия схемы 4.90
Средство сбора журналов	Виртуальная машина (для запуска ВПО)	Elastic Agent	8.1.1
Гостевая ОС	Виртуальная машина (для развертывания ELK-стека)	Ubuntu	22.04.2 LTS Desktop
Средство контейнеризации	Виртуальная машина (для развертывания ELK-стека)	Docker	24.0.4
Средство хранения и обработки журналов	Виртуальная машина (для развертывания ELK-стека)	ELK стек	8.1.1
Средство преобразования Sigma-правил в запросы к Elasticsearch	Виртуальная машина (для развертывания ELK-стека)	Sigma-cli	0.7.6
Sigma-правила	Виртуальная машина (для развертывания ELK-стека)	Sigma	Merge pull request #4353

6.2. Формирование выборки исследуемого ПО. В рамках проведения экспериментов были выбраны семейства ВПО [81], приведенные в таблице 4.

Критерием отбора было наличие антивирусных исследований, опубликованных в открытых источниках. Из данных семейств были отобраны 100 образцов ВПО, особенности поведения которых хорошо известны, что позволяет достоверно оценивать корректность разметки данных в ходе экспериментов. Также в таблице приведены классы легитимного ПО, которое является достаточно распространенным на компьютерах пользователей и имеет определенные поведенческие сходства с ВПО (например, средства для шифрования файлов пользователя осуществляют такие же действия, как и вымогательское ВПО класса шифровальщиков).

Таблица 4. Исследуемые экземпляры ВПО и легитимных программ

Тип ПО	Класс ПО	Семейства	Кол-во образцов
ВПО	ВПО, используемое APT-группировками (эксплойты, трояны-загрузчики, средства удаленного управления)	Bisonal, MustangPanda, NetTraveler, PittyTiger, Taidoor, Winnti	23
	ВПО, используемое финансово мотивированными группировками (эксплойты, трояны-загрузчики, средства удаленного управления, программы-вымогатели)	CobaltGang, Silence, TA505 (Amadey), RTM, FIN7, Lazarus	25
	Средства удаленного управления	Darkcomet, AsyncRAT, Remcos RAT	5
	Средства кражи паролей и другой конфиденциальной информации	Agent Tesla, Lokibot, IcedID, Formbook	10
	Трояны-загрузчики	Zloader, Emotet, Trickbot, Qbot	6
	Трояны-майнеры	LemonDuck, Monero (XMR) Miner	5
	Программы-вымогатели	WastedLocker, GandCrab, модифицированный WannaCry (была удалена проверка регистрации домена), NotPetya, Egregor, REvil, BlackMatter, Ryuk, DoppelPaymer	14
Хакерские инструменты	Средство извлечения паролей из памяти ОС, фреймворки удаленного управления	Mimikatz, Cobalt Strike, Meterpreter	3
Легитимное ПО	Редакторы (офисное ПО), браузеры, архиваторы, программы шифрования, антивирусные программы	Microsoft Office Word, Microsoft Office Excel, Adobe Reader, Notepad++, Google Chrome, WinRAR, 7zip, VeraCrypt, Антивирус Windows Defender	9

6.3. Проведение серии экспериментов и анализ результатов. Для апробации методики была выполнена серия экспериментов

по запуску перечисленных образцов ВПО и легитимных программ, представленных в таблице 2.

Эксперимент 1 «Выбор таймаута для атомарного исследования активности ПО». В ходе первого эксперимента над одним исследуемым образцом ВПО был выбран таймаут, равный одной минуте в соответствии с временем по умолчанию для проверки одного файла в песочнице Cuckoo Sandbox [82]. Данный эксперимент показал, что работа вредоносной программы завершилась мгновенно (между событиями создания и завершения процесса прошло менее одной секунды) ввиду отсутствия доступа к управляющему серверу ВПО. Даже при успешном соединении с управляющим сервером большинство вредоносных программ отрабатывало достаточно быстро: менее 15 секунд при Интернет-канале 20 Мбит/с. Исключениями были два типа вредоносных программ: загрузчики ВПО и программы-вымогатели. Загрузчики ВПО активировались через какое-то время или переходили в режим ожидания команд от злоумышленников. Программы-вымогатели, в свою очередь, продолжают работать до тех пор, пока не зашифруют все целевые файлы в системе. Таким образом, для всех программ был установлен таймаут *30 секунд*. Данное значение обусловлено тем, что за указанное время вредоносные события с виртуальной машины попадут в ELK, а для ВПО с отложенным запуском достаточно выявлять событие создания запланированной задачи для дальнейшего запуска.

Эксперимент 2 «Запуск исследуемых программ». На обеих виртуальных машинах поочередно запускались образцы ВПО и легитимных программ, представленные в таблице 2. На каждой машине с заданной конфигурацией Sysmon за 30 секунд генерировалось от 2300 до 7800 событий Sysmon, при этом объем выделенных ресурсов на разные виртуальные машины незначительно влиял на количество генерируемых событий. Таким образом, предположение о влиянии количества ресурсов и быстродействия системы на генерацию событий оказалось ошибочным.

Далее был произведен анализ распределения событий Sysmon в зависимости от типов событий. Значительную долю этих событий составляли события взаимодействия с реестром, загрузки исполняемого образа и доступа одного процесса к памяти другого процесса. Для примера приведено распределение типов событий для класса ВПО «Средства удаленного управления» (рисунок 6). Данное распределение является характерным для большинства семейств ВПО и легитимных программ.

Рис. 6. Распределение событий Sysmon в зависимости от типов событий для класса ВПО «Средства удаленного управления»

Совершенно другая круговая диаграмма распределения характерна для программ-вымогателей: особенностью данного класса ВПО является наличие повышенной файловой активности (рисунок 7).

Из похожести приведенных диаграмм распределения событий Sysmon в зависимости от типов событий для большинства семейств ВПО и легитимных программ следует вывод: большинство зафиксированных событий являются характерными для фоновой активности системы. Таким образом, необходимо провести отдельный эксперимент по исследованию фоновой активности ОС Windows.

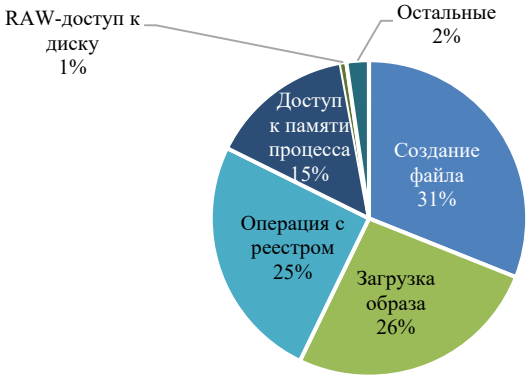

Рис. 7. Распределение событий Sysmon в зависимости от типов событий для класса ВПО «Программы-вымогатели»

Эксперимент 3 «Исследование фоновой активности ОС Windows». Для исследования особенностей фоновой активности был проведен следующий эксперимент, направленный на сравнение количества событий на этапе запуска ОС Windows и во время ее работы. Для этого были проведены запуски обеих виртуальных машин с гостевой ОС Windows 10 и зафиксирована работа в течение 12 минут без выполнения программ из таблицы 2. Распределение количества событий по минутам для BM_1 приведено на рисунке 8.

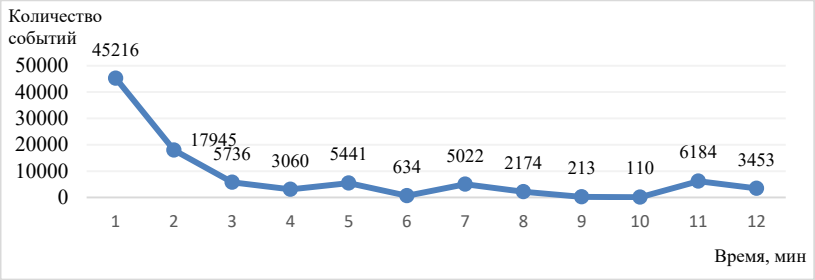

Рис. 8. Количество событий Sysmon после старта ОС Windows 10 в единицу времени

Как видно из данного графика, через 2 минуты после старта системы количество событий становится достаточно равномерным. Разброс значений от минимального 110 до максимального 6184 объясняется особенностью работы самой ОС Windows: возникновение одной фоновой задачи генерирует сразу несколько сотен событий. Таким образом, лавинообразные значения количества событий на графике являются естественными для данной ОС.

Также рисунок 8 подтверждает то, что значительная доля событий Sysmon генерируется самой ОС Windows 10 в фоновом режиме: количество событий Sysmon через 2 минуты после старта ОС Windows входит в интервал количества событий из эксперимента 2. Далее был произведен расчет числа событий Sysmon в зависимости от типов событий для фоновой активности ОС Windows (рисунок 9).

Таким образом, можно модифицировать конфигурационный файл Sysmon, добавив в него значительное количество исключений, характерных для фоновой активности ОС Windows. Модифицированный конфигурационный файл Sysmon доступен в Github авторов данной статьи [76].

Рис. 9. Распределение событий Sysmon в зависимости от типов событий фоновой активности ОС Windows

Эксперимент 4 «Выявление сработавших Sigma-правил». По итогам эксперимента 2 был произведен сбор событий, генерируемых исследуемыми образцами ВПО и легитимных программ из таблицы 2. Для разметки полученного набора данных были запущены утилита Sigma-cli и соответствующий процесс трансляции Sigma-правил в запросы для ELK. По итогу работы данной утилиты был сгенерирован 1641 запрос для ELK. Полученные запросы доступны в Github авторов данной статьи [76].

Далее полученные запросы направлялись в ELK для поиска событий, удовлетворяющих вредоносным шаблонам поведения. В ходе запуска данных запросов было установлено, что 790 из них имеют синтаксические и логические ошибки, связанные с экранированием различных символов: пробелов, двоеточий, слешей и других управляющих символов. Однако с учетом того, что все правила транслируются по единому алгоритму, вероятно, устранение нескольких ошибок в трансляторе сделает все запросы к ELK валидными.

По итогам обработки всех запросов в ELK было установлено, что для 83 из 100 образцов ВПО было обнаружено хотя бы одно вредоносное событие, для 40 образцов – два вредоносных события и для 9 образцов ВПО – три вредоносных события. Для поисковых запросов к ELK и обнаруживаемым ими событиям справедливо утверждение, что одно событие обнаруживается не более чем одним

поисковым запросом. Таким образом, для разметки данных для каждого события были добавлены 5 полей: «sigma_name» и «sigma_id» для идентификации Sigma-правил и поисковых запросов к ELK, с помощью которых было задетектировано то или иное событие; «technique_name» и «technique_id» для идентификации соответствующих техник атаки; а также поле «is_malware» – в случае, если данное событие было сгенерировано запускаемым образцом ВПО. Таким образом, 83% исследованных образцов ВПО были справедливо идентифицированы как ВПО. Среди событий, сгенерированных в ходе работы легитимного ПО, не было обнаружено вредоносных, то есть ошибок первого рода не допущено.

Таблица 5. Идентифицированные техники ВПО

Техника атаки, обнаруженная с помощью Sigma-правила	Количество обнаруженных образцов ВПО
Закрепление в системе	71
Обход средств защиты (запуск через легитимный инструмент)	27
Доступ к учетным данным в процессе «lsass.exe»	16
Обход средств защиты (User Account control)	12
Обход средств защиты (Запуск исполняемого файла через «rundll32.exe» не из системной директории)	6

В результате анализа идентифицированных техник была составлена статистика, приведенная в таблице 5. Таким образом, для исследованной выборки образцов ВПО справедливо заключение, что наиболее обнаруживаемой техникой является «Закрепление в системе». Данная техника встречается более, чем в половине исследованных образцов ВПО. Часто техника «Закрепление в системе» выполняется через различные ключи реестра (например, «Run» или «RunOnce») и папку «Startup Folder». Следующей техникой (для менее, чем трети образцов ВПО), которая детектировалась в ходе экспериментов, является «Обход средств защиты» (запуск через легитимный инструмент). Для реализации этой техники ВПО использовало системные утилиты, например, «rundll32.exe» и «mshta.exe». Третьей достаточно распространенной техникой является «доступ к учетным данным в процессе «lsass.exe». Эта техника возможна благодаря тому, что процесс «lsass.exe», необходимый для аутентификации в ОС Windows, кэширует имена пользователей и хеши их учетных данных. Остальные техники

характерны для оставшейся пятой части исследованной выборки образцов ВПО.

Таким образом, предлагаемая методика позволяет исследователям формировать наборы данных об активности ВПО, используя Sigma-правила для автоматизации разметки событий активности процессов ОС Windows. Проведенные экспериментальные исследования подтверждают эффективность данного подхода, который является отличительной особенностью нашего исследования. Описанная методика позволяет любому исследователю воспроизвести научные результаты авторов настоящей статьи и получить набор данных, необходимый для построения модели машинного обучения, осуществляющей обнаружение ВПО. При этом представленная методика априори предполагает соответствие среды сбора данных реальным условиям работы алгоритма классификации, преобладание привилегий системы сбора данных над привилегиями ВПО и наличие легитимной фоновой активности. Тем самым обеспечивается соответствие формируемого набора данных об активности ВПО критериям качества, введенным в работе [18]. Это является главным преимуществом в сравнении с предшествующими исследованиями. Некоторые предшествующие работы, например, [10, 11] включают в себя создание набора данных с последовательностями API-функций ВПО с помощью Cuckoo Sandbox, при этом разметка по исследуемым файлам производится с помощью сторонних сервисов, логика которых неочевидна. В исследовании [12] приведены этапы работы с данными, но разметка данных не раскрыта подробно.

7. Заключение. В данном исследовании затронута актуальная проблема недостаточности полноты и точности обнаружения ВПО существующими методами детектирования ВПО. Для обнаружения ВПО исследователи извлекают статические признаки файлов или динамические признаки процессов ОС Windows (или и те, и другие), к которым затем применяют методы детектирования ВПО. В ходе исследования установлено, что значительная часть авторов использует готовые наборы данных с признаками ВПО. Вопросы формирования этих наборов данных в настоящее время раскрыты недостаточно подробно, что ставит под сомнение применимость используемых методов детектирования ВПО. Также важным аспектом формирования набора данных является обработка этих данных и их разметка. Процесс разметки данных является достаточно трудозатратным. В качестве альтернативы в данной работе предложена собственная методика сбора данных об активности ВПО на основе инструмента Sysmon и поведенческих правил Sigma. Использование правил Sigma

позволяет автоматизировать процесс разметки данных. Для проверки применимости данной методики была проведена серия вычислительных экспериментов. Проведенные эксперименты подтвердили возможность использования правил Sigma для разметки вредоносных событий для дальнейшего создания моделей машинного обучения. Направление дальнейшей работы связано с разработкой инструмента валидации правил Sigma и их применения к журналам событий активности ВПО. Другим направлением исследовательской работы является разработка новых правил Sigma, для которых известны существующие техники атаки, но пока отсутствуют готовые правила детектирования.

Литература

1. Cybercrime Will Cost the World US$6 Trillion by the End of the Year: Study. URL: https://cisomag.eccouncil.org/cybercrime-will-cost-the-world-us6-trillion-by-the-end-of-the-year-study/ (дата обращения: 10.11.2023).
2. Ландшафт угроз. URL: https://encyclopedia.kaspersky.ru/glossary/threat-landscape/ (дата обращения: 08.11.2023).
3. Левшун Д.С., Гайфулина Д.А., Чечулин А.А., Котенко И.В. Проблемные вопросы информационной безопасности киберфизических систем // Информатика и автоматизация. 2020. Т. 19. № 5. С. 1050–1088.
4. ГОСТ Р 51275-2006. Защита информации. Объект информатизации. Факторы, воздействующие на информацию // М.: Госстандарт России. 2006.
5. Denning D. An Intrusion-Detection Model // IEEE Transactions on Software Engineering. 1987. no. 2. pp. 222–232.
6. Abaoaja F., Zainal A., Ghaleb F., Al-rimy B.A.S., Eisa T.A.E., Elnour A.A.H. Malware Detection Issues, Challenges, and Future Directions: A Survey // Applied Sciences. 2022. vol. 12. no. 17. pp. 1–29.
7. Herrera-Silva J., Hernandez-Alvarez M. Dynamic Feature Dataset for Ransomware Detection Using Machine Learning Algorithms // Sensors. 2023. vol. 23. no. 3. pp. 1–24.
8. Ali R., Ali A., Iqbal F., Hussain M., Ullah F. Deep Learning Methods for Malware and Intrusion Detection: A Systematic Literature Review // Security and Communication Networks. 2022. vol. 2022. pp. 1–31.
9. Gibert D., Mateu C., Planes J. The rise of machine learning for detection and classification of malware: Research developments, trends and challenges // Journal of Network and Computer Applications. 2020. vol. 153. pp. 1–22.
10. Kattamuri S., Penmatsa R., Chakravarty S., Madabathula V. Swarm Optimization and Machine Learning Applied to PE Malware Detection towards Cyber Threat Intelligence // Electronics. 2023. vol. 12. no. 2. pp. 1–25.
11. Lu F., Cai Z., Lin Z., Bao Y., Tang M. Research on the Construction of Malware Variant Datasets and Their Detection Method // Applied Sciences. 2022. vol. 12. no. 15. DOI: 10.3390/app12157546.
12. Catak F.O., Yazi A.F., Elezaj O., Ahmed J. Deep learning based Sequential model for malware analysis using Windows exe API Calls // PeerJ Computer Science. 2020. vol. 6. pp. 1–23.
13. Sanchez-Fraga R., Acosta-Bermejo R. Toward a Taxonomy and Multi-label Dataset for Malware Classification // Proceedings of the 10th International Conference in Software Engineering Research and Innovation, CONISOFT. 2022. pp. 150–157.

14. Lee S., Jung W., Lee W., Oh H., Kim E. Android malware dataset construction methodology to minimize bias–variance tradeoff // ICT Express. 2022. vol. 8. no. 3. pp. 444–462.
15. Чистяков А., Лобачева Е., Романенко А. Система и способ машинного обучения модели обнаружения вредоносных файлов // Патент RU2673708C1. 2018.
16. Chistyakov A., Lobacheva E., Romanenko A. System and method for generating a convolution function for training a malware detection model // PATENT USO10922410B2. 2021.
17. Chhetri S., Hewlett W. Execution behavior analysis text-based ensemble malware detector // PATENT EP4044054A1. 2022. pp. 1–26.
18. Rossow C., Dietrich C., Grier C., Kreibich C., Paxson V., Pohlmann N., Bos H., Van Steen M. Prudent Practices for Designing Malware Experiments: Status Quo and Outlook // Proceedings of the 2012 IEEE Symposium on Security and Privacy (S&P 2012). 2012. pp. 65–79.
19. Enterprise Matrix. URL: https://attack.mitre.org/matrices/enterprise/ (дата обращения: 09.11.2023).
20. Sigma. URL: https://github.com/Neo23x0/sigma (дата обращения: 05.11.2023).
21. Ilca L., Lucian O., Balan T. Enhancing Cyber-Resilience for Small and Medium-Sized Organizations with Prescriptive Malware Analysis, Detection and Response // Sensors. 2023. vol. 23. no. 15. pp. 1–33.
22. Lozano M.A., Llopis I.P., Domingo M.E. Threat Hunting Architecture Using a Machine Learning Approach for Critical Infrastructures Protection // Big Data and Cognitive Computing. 2023. vol. 7. no. 2. pp. 1–26.
23. Detecting Process Injection with ETW. URL: https://web.archive.org/web/20221207000139/https://blog.redbluepurple.io/windows-security-research/kernel-tracing-injection-detection (дата обращения: 30.10.2023).
24. Schultz M.G., Eskin E., Zadok E., Stolfo S.J. Data mining methods for detection of new malicious executables // Proceedings of the 2001 IEEE Symposium on Security and Privacy (S&P 2001). 2001. pp. 38–49.
25. Abou-Assaleh T., Cercone N., Keselj V., Sweidan R. N-gram-based detection of new malicious code // Proceedings of the 28th Annual International Computer Software and Applications Conference (COMPSAC 2004). 2004. vol. 02. pp. 41–42.
26. Sai M., Tyagi A., Panda K., Kumar S. Machine learning-based malware detection using stacking of opcodes and bytecode sequences // Proceedings of the 7th International Conference on Parallel, Distributed and Grid Computing (PDGC 2022). 2022. pp. 204–209.
27. Hong J., Jeong D., Kim S. Classifying Malicious Documents on the Basis of Plain-Text Features: Problem, Solution, and Experiences // Applied Sciences. 2022. vol. 12. no. 8. DOI: 10.3390/app12084088.
28. Tian R., Batten L., Versteeg S. Function length as a tool for malware classification // Proceedings of the 3rd International Conference on Malicious and Unwanted Software (MALWARE). 2008. pp. 69–76.
29. Dai J., Guha R., Lee J. Efficient Virus Detection Using Dynamic Instruction Sequences // Proceedings of the International Conference on High Performance Computing and Simulation (HPCS 2008). 2008. pp. 69–76.
30. Khalid O., Ullah S., Ahmad T., Saeed S., Alabbad D., Aslam M., Buriro A., Ahmad R. An Insight into the Machine-Learning-Based Fileless Malware Detection // Sensors. 2023. vol. 23. no. 2. DOI: org/10.3390/s23020612.
31. Baysa D., Low R.M., Stamp M. Structural entropy and metamorphic malware // Journal of Computer Virology and Hacking Techniques. 2013. vol. 9. pp. 179–192.

32. Karim M., Walenstein A., Lakhotia A., Parida L. Malware phylogeny generation using permutations of code // Journal in Computer Virology. 2005. vol. 1. no. 1-2. pp. 13–23.
33. Botacin M., Galhardo Moia V., Ceschin F., Amaral Henriques M., Gregio A. Understanding uses and misuses of similarity hashing functions for malware detection and family clustering in actual scenarios // Forensic Science International: Digital Investigation. 2021. vol. 38. DOI: 10.1016/j.fsidi.2021.301220.
34. Jacob G., Debar H., Filiol E. Behavioral detection of malware: from a survey towards an established taxonomy // Journal in Computer Virology. 2008. vol. 4. pp. 251–266.
35. Sgandurra D., Munoz-Gonzalez L., Mohsen R., Lupu E.C. Automated Dynamic Analysis of Ransomware: Benefits, Limitations and use for Detection // arXiv. 2016.
36. Abbasi M.S., Al-Sahaf H., Mansoori M., Welch I. Behavior-based ransomware classification: A particle swarm optimization wrapper-based approach for feature selection // Applied Soft Computing. 2022. vol. 121. pp. 1–12.
37. Roesch M. Snort-Lightweight Intrusion Detection for Networks // Proceedings of the 13th USENIX Conference on System Administration (LISA '99). 1999. vol. 99. no. 1. pp. 229–238.
38. YARA. The pattern matching swiss knife for malware researchers (and everyone else). URL: https://virustotal.github.io/yara/ (дата обращения: 03.11.2023).
39. signature-base. URL: https://github.com/Neo23x0/signature-base/tree/master/yara (дата обращения: 03.11.2023).
40. Гайворонская С. Исследование методов обнаружения шеллкодов в высокоскоростных каналах передачи данных // М., дис. канд. ф.-м. наук: 05.13.11: защищена 19.09.2014. 2014. 133 с.
41. Bruschi D., Martignoni L., Monga M. Detecting self-mutating malware using control-flow graph matching // Detection of Intrusions and Malware & Vulnerability Assessment: Third International Conference. 2006. pp. 129–143.
42. Naik N., Jenkins P., Savage N., Yang L., Boongoen T., Iam-On N. Fuzzy-import hashing: A static analysis technique for malware detection // Forensic Science International: Digital Investigation. 2021. vol. 37. DOI: 10.1016/j.fsidi.2021.301139.
43. Nataraj L., Karthikeyan S., Jacob G., Manjunath B.S. Malware images: visualization and automatic classification // Proceedings of the 8th International Symposium on Visualization for Cyber Security (VizSec '11). 2011. pp. 1–7.
44. Shabtai A., Moskovitch R., Elovici Y., Glezer C. Detection of malicious code by applying machine learning classifiers on static features, a state-of-the-art survey // Information Security Technical Report. 2009. vol. 14. no. 1. pp. 16–29.
45. Souri A., Hosseini R. A state-of-the-art survey of malware detection approaches using data mining techniques // Human-centric Computing and Information Sciences. 2018. vol. 8. no. 1. pp. 1–22. DOI: 10.1186/s13673-018-0125-x.
46. Wagener G., State R., Dulaunoy A. Malware behaviour analysis // Journal in Computer Virology. 2008. vol. 4. pp. 279–287.
47. Endpoint Detection and Response (EDR). URL: https://encyclopedia.kaspersky.com/glossary/edr-endpoint-detection-response/ (дата обращения: 10.11.2023).
48. Шевалье Я., Фенцль Ф., Коломеец М.В., Рике Р., Чечулин А.А., Краус К. Обнаружение кибератак в транспортных средствах с использованием характеризующих функций, искусственных нейронных сетей и визуального анализа // Информатика и автоматизация. 2021. Т. 20. № 4. С. 845–868.
49. Bostami B., Ahmed M. Deep Learning Meets Malware Detection: An Investigation // Combating Security Challenges in the Age of Big Data: Powered by State-of-the-Art Artificial Intelligence Techniques. 2020. pp. 137–155.

50. Зегжда Д.П., Калинин М.О., Крундышев В.М., Лаврова Д.С., Москвин Д.А., Павленко Е.Ю. Применение алгоритмов биоинформатики для обнаружения мутирующих кибератак // Информатика и автоматизация. 2021. Т. 20. № 4. С. 820–844.
51. Jiang J., Zhang F. Detecting Portable Executable Malware by Binary Code Using an Artificial Evolutionary Fuzzy LSTM Immune System // Security and Communication Networks. 2021. vol. 2021. DOI: 10.1155/2021/3578695.
52. Wawryn K., Widulinski P. Detection of anomalies in compiled computer program files inspired by immune mechanisms using a template method // Journal of Computer Virology and Hacking Techniques. 2021. vol. 17. pp. 47–59.
53. Котенко И.В., Саенко И.Б., Лаута О.С., Крибель А.М. Методика обнаружения аномалий и кибератак на основе интеграции методов фрактального анализа и машинного обучения // Информатика и автоматизация. 2022. Т. 21. № 6. С. 1328–1358.
54. malware_images. URL: http://vision.ece.ucsb.edu/~lakshman/malware_images/album/ (дата обращения: 10.11.2023).
55. anubis. URL: http://anubis.iseclab.org/ (дата обращения: 12.11.2023).
56. Cuckoo Sandbox. URL: https://cuckoosandbox.org/ (дата обращения: 10.11.2023).
57. ransomwaredataset2016. URL: https://github.com/rissgrouphub/ransomwaredataset2016 (дата обращения: 10.11.2023).
58. MalwareDataset. URL: https://github.com/WindrunnerMax/MalwareDataset (дата обращения: 13.11.2023).
59. Котенко И., Хмыров С. Анализ моделей и методик, используемых для атрибуции нарушителей кибербезопасности при реализации целевых атак // Вопросы кибербезопасности. 2022. Т. 4. № 50. С. 52–79.
60. The Cyber Kill Chain. URL: https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html (дата обращения: 08.11.2023).
61. The Diamond Model of Intrusion Analysis. URL: https://www.activeresponse.org/wp-content/uploads/2013/07/diamond.pdf (дата обращения: 06.11.2023).
62. Noor U., Anwar Z., Amjad T., Choo K.-K.R. A machine learning-based FinTech cyber threat attribution framework using high-level indicators of compromise // Future Generation Computer Systems. 2019. vol. 96. pp. 227–242.
63. Emotet Strikes Again – LNK File Leads to Domain Wide Ransomware. URL: https://thedfirreport.com/2022/11/28/emotet-strikes-again-lnk-file-leads-to-domain-wide-ransomware/ (дата обращения: 03.11.2023).
64. Mandiant. URL: https://mandiant.com (дата обращения: 12.11.2023).
65. capa. URL: https://github.com/mandiant/capa (дата обращения: 12.11.2023).
66. Levshun D., Kotenko I. A survey on artificial intelligence techniques for security event correlation: models, challenges, and opportunities // Artificial Intelligence Review. 2023. vol. 56. DOI: 10.1007/s10462-022-10381-4.
67. Check out Check Point's coverage of the MITRE ATT&CK enterprise matrix. URL: https://www.checkpoint.com/solutions/mitre-attack/coverage/ (дата обращения: 13.11.2023).
68. Hoglund G., Butler J. Rootkits: Subverting the Windows Kernel: Subverting the Windows Kernel // Addison-Wesley Professional. 2005. 352 p.
69. drakvuf. URL: https://github.com/tklengyel/drakvuf (дата обращения: 09.11.2023).
70. Filter Manager Concepts. URL: https://learn.microsoft.com/en-us/windows-hardware/drivers/ifs/filter-manager-concepts (дата обращения: 15.11.2023).
71. About Event Tracing. URL: https://learn.microsoft.com/en-us/windows/win32/etw/about-event-tracing (дата обращения: 10.11.2023).
72. Sysmon. URL: https://learn.microsoft.com/en-us/sysinternals/downloads/sysmon/ (дата обращения: 09.11.2023).

73. Security Datasets. URL: https://github.com/OTRF/Security-Datasets (дата обращения: 15.11.2023).
74. Sikorski M., Honig A. Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software // No Starch Press, 2012. 800 p.
75. community/modules/signatures/ URL: https://github.com/cuckoosandbox/community/tree/master/modules/signatures/ (дата обращения: 13.11.2023).
76. mongoosecurity. URL: https://github.com/SDanilytics/mongoosecurity (дата обращения: 01.11.2023).
77. What is the ELK Stack? URL: https://www.elastic.co/what-is/elk-stack (дата обращения: 15.11.2023).
78. Elastic stack (ELK) on Docker. URL: https://github.com/deviantony/docker-elk (дата обращения: 25.11.2023).
79. Elastic Agent 8.1.1. URL: https://www.elastic.co/downloads/past-releases/elastic-agent-8-1-1 (дата обращения: 03.11.2023).
80. Sigma Command Line Interface. URL: https://github.com/SigmaHQ/sigma-cli (дата обращения: 01.08.2023).
81. Types of Malware. URL: https://www.kaspersky.com/resource-center/threats/malware-classifications (дата обращения: 09.11.2023).
82. Submission Utility. URL: https://cuckoo.readthedocs.io/en/latest/usage/submit/?highlight=timeout#submission-utility (дата обращения: 01.11.2023).

Смирнов Данил Вадимович — аспирант, старший преподаватель кафедры, кафедра информационной безопасности киберфизических систем Московского института электроники и математики им. А.Н. Тихонова, Национальный исследовательский университет «Высшая школа экономики». Область научных интересов: исследование поведения вредоносных программ и хакерских атак с использованием методов машинного обучения. Число научных публикаций — 2. DVSmirnov@hse.ru; улица Таллинская, 34, 123458, Москва, Россия; р.т.: +7(495)772-9590.

Евсютин Олег Олегович — канд. техн. наук, доцент, заведующий кафедрой, кафедра информационной безопасности киберфизических систем Московского института электроники и математики им. А.Н. Тихонова, Национальный исследовательский университет «Высшая школа экономики»; старший научный сотрудник, лаборатория № 80, ИПУ РАН. Область научных интересов: исследование цифровых водяных знаков и методов их встраивания, цифровая обработка изображений, цифровая стеганография, информационная безопасность. Число научных публикаций — 90. OEvsyutin@hse.ru; улица Таллинская, 34, 123458, Москва, Россия; р.т.: +7(495)772-9590.

DOI 10.15622/ia.23.3.2

D. SMIRNOV, O. EVSUTIN

METHODOLOGY FOR COLLECTING DATA ON THE ACTIVITY OF MALWARE FOR WINDOWS OS BASED ON MITRE ATT&CK

Smirnov D., Evsutin O. **Methodology for Collecting Data on the Activity of Malware for Windows OS Based on MITRE ATT&CK.**

Abstract. The digitalization of the modern economy has led to the emergence of information technologies in various areas of human activity. In addition to positive effects, this has enhanced the problem of countering cyber threats. The implementation of cyber threats often impacts serious consequences, especially when it comes to critical information infrastructure. Malware is an important part of the modern landscape of cyber threats; the most high-profile cybercrimes of recent years are associated with the use of malware. In this regard, the problem area of countering malware is actively developing, and one of the promising areas of research in this area is the creation of methods for detecting malware based on machine learning. However, the weak point of many well-known studies is the construction of reliable data sets for machine learning models, when the authors do not disclose the features of the formation, preprocessing and labeling of data on malware. This fact compromises the reproducibility a lot of studies. This paper proposes a methodology for collecting data on malware activity based on the MITRE ATT&CK matrix and Sigma rules and designed for Windows OS. The proposed methodology is aimed at improving the quality of datasets containing malware and legitimate processes behavior's features, as well as at reducing the time of data label by an expert method. A software stand was prepared and experiments were carried out for testing the methodology. The results of experiments confirmed applicability of our methodology.

Keywords: cybersecurity, malware, MITRE ATT&CK, process activity monitoring, machine learning.

References

1. Cybercrime Will Cost the World US$6 Trillion by the End of the Year: Study. Available at: https://cisomag.eccouncil.org/cybercrime-will-cost-the-world-us6-trillion-by-the-end-of-the-year-study/ (accessed: 10.11.2023).
2. Threat Landscape. Available at: https://encyclopedia.kaspersky.ru/glossary/threat-landscape/ (accessed: 08.11.2023).
3. Levshun D., Gaifulina D., Chechulin A., Kotenko I. Problematic issues of information security of cyber-physical systems. SPIIRAS Proceedings. 2020. vol. 19. no. 5. pp. 1050–1088. (In Russ.).
4. GOST R 51275-2006. [Information Security. Informatization object. Factors affecting information]. M.: Gosstandart Rossii. 2006. (In Russ.).
5. Denning D. An Intrusion-Detection Model. IEEE Transactions on Software Engineering. 1987. no. 2. pp. 222–232.
6. Abaoaja F., Zainal A., Ghaleb F., Al-rimy B.A.S., Eisa T.A.E., Elnour A.A.H. Malware Detection Issues, Challenges, and Future Directions: A Survey. Applied Sciences. 2022. vol. 12. no. 17. pp. 1–29.
7. Herrera-Silva J., Hernandez-Alvarez M. Dynamic Feature Dataset for Ransomware Detection Using Machine Learning Algorithms. Sensors. 2023. vol. 23. no. 3. pp. 1–24.

8. Ali R., Ali A., Iqbal F., Hussain M., Ullah F. Deep Learning Methods for Malware and Intrusion Detection: A Systematic Literature Review. Security and Communication Networks. 2022. vol. 2022. pp. 1–31.
9. Gibert D., Mateu C., Planes J. The rise of machine learning for detection and classification of malware: Research developments, trends and challenges. Journal of Network and Computer Applications. 2020. vol. 153. pp. 1–22.
10. Kattamuri S., Penmatsa R., Chakravarty S., Madabathula V. Swarm Optimization and Machine Learning Applied to PE Malware Detection towards Cyber Threat Intelligence. Electronics. 2023. vol. 12. no. 2. pp. 1–25.
11. Lu F., Cai Z., Lin Z., Bao Y., Tang M. Research on the Construction of Malware Variant Datasets and Their Detection Method. Applied Sciences. 2022. vol. 12. no. 15. DOI: 10.3390/app12157546.
12. Catak F.O., Yazi A.F., Elezaj O., Ahmed J. Deep learning based Sequential model for malware analysis using Windows exe API Calls. PeerJ Computer Science. 2020. vol. 6. pp. 1–23.
13. Sanchez-Fraga R., Acosta-Bermejo R. Toward a Taxonomy and Multi-label Dataset for Malware Classification. Proceedings of the 10th International Conference in Software Engineering Research and Innovation, CONISOFT. 2022. pp. 150–157.
14. Lee S., Jung W., Lee W., Oh H., Kim E. Android malware dataset construction methodology to minimize bias–variance tradeoff. ICT Express. 2022. vol. 8. no. 3. pp. 444–462.
15. Chistyakov A., Lobacheva E., Romanenko A. System and method of machine training model of detecting malicious files. PATENT RU2673708C1. 2018. (In Russ.).
16. Chistyakov A., Lobacheva E., Romanenko A. System and method for generating a convolution function for training a malware detection model. PATENT USO10922410B2. 2021.
17. Chhetri S., Hewlett W. Execution behavior analysis text-based ensemble malware detector. PATENT EP4044054A1. 2022. pp. 1–26.
18. Rossow C., Dietrich C., Grier C., Kreibich C., Paxson V., Pohlmann N., Bos H., Van Steen M. Prudent Practices for Designing Malware Experiments: Status Quo and Outlook. Proceedings of the 2012 IEEE Symposium on Security and Privacy (S&P 2012). 2012. pp. 65–79.
19. Enterprise Matrix. Available at: https://attack.mitre.org/matrices/enterprise/ (accessed: 09.11.2023).
20. Sigma. Available at: https://github.com/Neo23x0/sigma (accessed: 05.11.2023).
21. Ilca L., Lucian O., Balan T. Enhancing Cyber-Resilience for Small and Medium-Sized Organizations with Prescriptive Malware Analysis, Detection and Response. Sensors. 2023. vol. 23. no. 15. pp. 1–33.
22. Lozano M.A., Llopis I.P., Domingo M.E. Threat Hunting Architecture Using a Machine Learning Approach for Critical Infrastructures Protection. Big Data and Cognitive Computing. 2023. vol. 7. no. 2. pp. 1–26.
23. Detecting Process Injection with ETW. Available at: https://web.archive.org/web/20221207000139/https://blog.redbluepurple.io/windows-security-research/kernel-tracing-injection-detection (accessed: 30.10.2023).
24. Schultz M.G., Eskin E., Zadok E., Stolfo S.J. Data mining methods for detection of new malicious executables. Proceedings of the 2001 IEEE Symposium on Security and Privacy (S&P 2001). 2001. pp. 38–49.
25. Abou-Assaleh T., Cercone N., Keselj V., Sweidan R. N-gram-based detection of new malicious code. Proceedings of the 28th Annual International Computer Software and Applications Conference (COMPSAC 2004). 2004. vol. 02. pp. 41–42.
26. Sai M., Tyagi A., Panda K., Kumar S. Machine learning-based malware detection using stacking of opcodes and bytecode sequences. Proceedings of the 7th

International Conference on Parallel, Distributed and Grid Computing (PDGC 2022). 2022. pp. 204–209.
27. Hong J., Jeong D., Kim S. Classifying Malicious Documents on the Basis of Plain-Text Features: Problem, Solution, and Experiences. Applied Sciences. 2022. vol. 12. no. 8. DOI: 10.3390/app12084088.
28. Tian R., Batten L., Versteeg S. Function length as a tool for malware classification. Proceedings of the 3rd International Conference on Malicious and Unwanted Software (MALWARE). 2008. pp. 69–76.
29. Dai J., Guha R., Lee J. Efficient Virus Detection Using Dynamic Instruction Sequences. Proceedings of the International Conference on High Performance Computing and Simulation (HPCS 2008). 2008. pp. 69–76.
30. Khalid O., Ullah S., Ahmad T., Saeed S., Alabbad D., Aslam M., Buriro A., Ahmad R. An Insight into the Machine-Learning-Based Fileless Malware Detection. Sensors. 2023. vol. 23. no. 2. DOI: org/10.3390/s23020612.
31. Baysa D., Low R.M., Stamp M. Structural entropy and metamorphic malware. Journal of Computer Virology and Hacking Techniques. 2013. vol. 9. pp. 179–192.
32. Karim M., Walenstein A., Lakhotia A., Parida L. Malware phylogeny generation using permutations of code. Journal in Computer Virology. 2005. vol. 1. no. 1-2. pp. 13–23.
33. Botacin M., Galhardo Moia V., Ceschin F., Amaral Henriques M., Gregio A. Understanding uses and misuses of similarity hashing functions for malware detection and family clustering in actual scenarios. Forensic Science International: Digital Investigation. 2021. vol. 38. DOI: 10.1016/j.fsidi.2021.301220.
34. Jacob G., Debar H., Filiol E. Behavioral detection of malware: from a survey towards an established taxonomy. Journal in Computer Virology. 2008. vol. 4. pp. 251–266.
35. Sgandurra D., Munoz-Gonzalez L., Mohsen R., Lupu E.C. Automated Dynamic Analysis of Ransomware: Benefits, Limitations and use for Detection. arXiv. 2016.
36. Abbasi M.S., Al-Sahaf H., Mansoori M., Welch I. Behavior-based ransomware classification: A particle swarm optimization wrapper-based approach for feature selection. Applied Soft Computing. 2022. vol. 121. pp. 1–12.
37. Roesch M. Snort-Lightweight Intrusion Detection for Networks. Proceedings of the 13th USENIX Conference on System Administration (LISA '99). 1999. vol. 99. no. 1. pp. 229–238.
38. YARA. The pattern matching swiss knife for malware researchers (and everyone else). Available at: https://virustotal.github.io/yara/ (accessed: 03.11.2023).
39. signature-base. Available at: https://github.com/Neo23x0/signature-base/tree/master/yara (accessed: 03.11.2023).
40. Gaivoronskaya S. Issledovanie metodov obnaruzheniya shellkodov v vysokoskorostnyh kanalah peredachi dannyh [Investigation of shellcode detection methods in high-speed data transmission channels]. PhD thesis. 2014. 133 p. (In Russ.).
41. Bruschi D., Martignoni L., Monga M. Detecting self-mutating malware using control-flow graph matching. Detection of Intrusions and Malware & Vulnerability Assessment: Third International Conference. 2006. pp. 129–143.
42. Naik N., Jenkins P., Savage N., Yang L., Boongoen T., Iam-On N. Fuzzy-import hashing: A static analysis technique for malware detection. Forensic Science International: Digital Investigation. 2021. vol. 37. DOI: 10.1016/j.fsidi.2021.301139.
43. Nataraj L., Karthikeyan S., Jacob G., Manjunath B.S. Malware images: visualization and automatic classification. Proceedings of the 8th International Symposium on Visualization for Cyber Security (VizSec '11). 2011. pp. 1–7.
44. Shabtai A., Moskovitch R., Elovici Y., Glezer C. Detection of malicious code by applying machine learning classifiers on static features, a state-of-the-art survey. Information Security Technical Report. 2009. vol. 14. no. 1. pp. 16–29.

45. Souri A., Hosseini R. A state-of-the-art survey of malware detection approaches using data mining techniques. Human-centric Computing and Information Sciences. 2018. vol. 8. no. 1. pp. 1–22. DOI: 10.1186/s13673-018-0125-x.
46. Wagener G., State R., Dulaunoy A. Malware behaviour analysis. Journal in Computer Virology. 2008. vol. 4. pp. 279–287.
47. Endpoint Detection and Response (EDR). Available at: https://encyclopedia.kaspersky.com/glossary/edr-endpoint-detection-response/ (accessed: 10.11.2023).
48. Chevalier Y., Fenzl S., Kolomeets M., Rieke R., Chechulin A., Krauss C. Cyberattack detection in vehicles using characteristic functions, artificial neural networks and visual analysis. Informatics and Automation. 2021. vol. 20. no. 4. pp. 845–868.
49. Bostami B., Ahmed M. Deep Learning Meets Malware Detection: An Investigation. Combating Security Challenges in the Age of Big Data: Powered by State-of-the-Art Artificial Intelligence Techniques. 2020. pp. 137–155.
50. Zegzhda D., Kalinin M., Krundyshev V., Lavrova D., Moskvin D., Pavlenko E. Application of bioinformatics algorithms for polymorphic cyberattacks detection. Informatics and Automation. 2021. vol. 20. no. 4. pp. 820–844. (In Russ.).
51. Jiang J., Zhang F. Detecting Portable Executable Malware by Binary Code Using an Artificial Evolutionary Fuzzy LSTM Immune System. Security and Communication Networks. 2021. vol. 2021. DOI: 10.1155/2021/3578695.
52. Wawryn K., Widulinski P. Detection of anomalies in compiled computer program files inspired by immune mechanisms using a template method. Journal of Computer Virology and Hacking Techniques. 2021. vol. 17. pp. 47–59.
53. Kotenko I., Saenko I., Lauta O., Kriebel A. Anomaly and cyber attack detection technique based on the integration of fractal analysis and machine learning methods. Informatics and Automation. 2022. vol. 21. no 6. pp. 1328–1358. (In Russ.).
54. malware_images. Available at: http://vision.ece.ucsb.edu/~lakshman/malware_images/album/ (accessed: 10.11.2023).
55. anubis. Available at: http://anubis.iseclab.org/ (accessed: 12.11.2023).
56. Cuckoo Sandbox. Available at: https://cuckoosandbox.org/ (accessed: 10.11.2023).
57. ransomwaredataset2016. Available at: https://github.com/rissgrouphub/ransomwaredataset2016 (accessed: 10.11.2023).
58. MalwareDataset. Available at: https://github.com/WindrunnerMax/MalwareDataset (accessed: 13.11.2023).
59. Kotenko I., Khmyrov S. Analysis of models and techniques used for attribution of cyber security violators in the implementation of targeted attacks. Voprosy kiberbezopasnosti – Cybersecurity issues. 2022. vol. 4. no. 50. pp. 52–79.
60. The Cyber Kill Chain. Available at: https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html (accessed: 08.11.2023).
61. The Diamond Model of Intrusion Analysis. Available at: https://www.activeresponse.org/wp-content/uploads/2013/07/diamond.pdf (accessed: 06.11.2023).
62. Noor U., Anwar Z., Amjad T., Choo K.-K.R. A machine learning-based FinTech cyber threat attribution framework using high-level indicators of compromise. Future Generation Computer Systems. 2019. vol. 96. pp. 227–242.
63. Emotet Strikes Again – LNK File Leads to Domain Wide Ransomware. Available at: https://thedfirreport.com/2022/11/28/emotet-strikes-again-lnk-file-leads-to-domain-wide-ransomware/ (accessed: 03.11.2023).
64. Mandiant. Available at: https://mandiant.com (accessed: 12.11.2023).
65. capa. Available at: https://github.com/mandiant/capa (accessed: 12.11.2023).

66. Levshun D., Kotenko I. A survey on artificial intelligence techniques for security event correlation: models, challenges, and opportunities. Artificial Intelligence Review. 2023. vol. 56. DOI: 10.1007/s10462-022-10381-4.
67. Check out Check Point's coverage of the MITRE ATT&CK enterprise matrix. Available at: https://www.checkpoint.com/solutions/mitre-attack/coverage/ (accessed: 13.11.2023).
68. Hoglund G., Butler J. Rootkits: Subverting the Windows Kernel: Subverting the Windows Kernel. Addison-Wesley Professional. 2005. 352 p.
69. drakvuf. Available at: https://github.com/tklengyel/drakvuf (accessed: 09.11.2023).
70. Filter Manager Concepts. Available at: https://learn.microsoft.com/en-us/windows-hardware/drivers/ifs/filter-manager-concepts (accessed: 15.11.2023).
71. About Event Tracing. Available at: https://learn.microsoft.com/en-us/windows/win32/etw/about-event-tracing (accessed: 10.11.2023).
72. Sysmon. Available at: https://learn.microsoft.com/en-us/sysinternals/downloads/sysmon/ (accessed: 09.11.2023).
73. Security Datasets. Available at: https://github.com/OTRF/Security-Datasets (accessed: 15.11.2023).
74. Sikorski M., Honig A. Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software. No Starch Press, 2012. 800 p.
75. community/modules/signatures/ Available at: https://github.com/cuckoosandbox/community/tree/master/modules/signatures/ (accessed: 13.11.2023).
76. mongoosecurity. Available at: https://github.com/SDanilytics/mongoosecurity (accessed: 01.11.2023).
77. What is the ELK Stack? Available at: https://www.elastic.co/what-is/elk-stack (accessed: 15.11.2023).
78. Elastic stack (ELK) on Docker. Available at: https://github.com/deviantony/docker-elk (accessed: 25.11.2023).
79. Elastic Agent 8.1.1. Available at: https://www.elastic.co/downloads/past-releases/elastic-agent-8-1-1 (accessed: 03.11.2023).
80. Sigma Command Line Interface. Available at: https://github.com/SigmaHQ/sigma-cli (accessed: 01.08.2023).
81. Types of Malware. Available at: https://www.kaspersky.com/resource-center/threats/malware-classifications (accessed: 09.11.2023).
82. Submission Utility. Available at: https://cuckoo.readthedocs.io/en/latest/usage/submit/?highlight=timeout#submission-utility (accessed: 01.11.2023).

Smirnov Danil — Postgraduate student, senior lecturer at the department, Department of information security of cyber-physical systems, Tikhonov Moscow institute of electronics and mathematics, National Research University Higher School of Economics. Research interests: behavior of malware and hacker attacks using machine learning methods. The number of publications — 2. DVSmirnov@hse.ru; 34, Tallinskaya St., 123458, Moscow, Russia; office phone: +7(495)772-9590.

Evsutin Oleg — Ph.D., Associate Professor, Head of the department, Department of information security of cyber-physical systems, Tikhonov Moscow institute of electronics and mathematics, National Research University Higher School of Economics; Senior researcher, Laboratory no. 80, ICS RAS. Research interests: digital watermarks and methods of their embedding, digital image processing, digital steganography, information security. The number of publications — 90. OEvsyutin@hse.ru; 34, Tallinskaya St., 123458, Moscow, Russia; office phone: +7(495)772-9590.