МЕТОД ОБНАРУЖЕНИЯ ПРОГРАММ-ВЫМОГАТЕЛЕЙ НА ОСНОВЕ АНАЛИЗА ПОВЕДЕНЧЕСКОГО ОТЧЕТА ИСПОЛНЯЕМОГО ОБЪЕКТА

Стародубов М. И.[1], Артемьева И. Л.[2], Селин Н. А.[3]

DOI: 10.21681/2311-3456-2024-3-85-89

Цель работы: *разработка метода обнаружения программ-вымогателей на основе анализа последовательностей API-вызовов и системных вызовов.*

Метод исследования: *анализ записей в поведенческом отчёте продукта виртуализации с использованием алгоритма глубокого обучения DeBERTa-V3.*

Полученный результат: *несмотря на большое разнообразие семейств и вариаций семейств программ-вымогателей, используемых злоумышленниками в компьютерных атаках, все они оставляют следы своей работы в атакуемой инфраструктуре. Одним из способов выявления вредоносного программного обеспечения и предотвращения заражения является использование технологии «Песочница», в том числе для выявления скрытых возможностей исследуемого объекта и аномалий его поведения. Функционирование любой компьютерной программы можно представить в виде набора записей его действий в отчёте поведения, которые можно рассматривать в качестве признаков объекта. В работе проведен анализ отчётов поведения программ-вымогателей. На сформированном наборе данных с использованием алгоритма глубокого обучения построена модель, позволяющая в дальнейшем выявлять вредоносные объекты, а также описан метод обнаружения программ-вымогателей.*

Практическая ценность *состоит в создании метода обнаружения программ-вымогателей на основе анализа поведенческого отчета исполняемого объекта с использованием алгоритма глубокого обучения DeBERTa-V3.*

Ключевые слова: *вредоносное программное обеспечение, песочница, глубокое обучение, BERT, Ransomware, компьютерные атаки.*

A METHOD FOR DETECTING RANSOMWARE BASED ON THE ANALYSIS OF THE BEHAVIORAL REPORT OF THE EXECUTABLE OBJECT

Starodubov M. I.[4], Artemyeva I. L.[5], Selin N. A.[6]

The aim of the work *is to develop a method for detecting ransomware based on the analysis of sequences of API calls and system calls.*

The research method *is the analysis of records in the behavioral report of the virtualization product using the deep learning algorithm DeBERTa-V3.*

The result obtained: *despite the wide variety of families and variations of the ransomware family used by attackers in computer attacks, they all leave traces of their work in the attacked infrastructure. One of the ways to identify malicious software and prevent infection is to use the Sandbox technology, including to identify the hidden capabilities of the object under study and anomalies of its behavior. The functioning*

1 Стародубов Максим Игоревич, аспирант ФГАОУ ВО «Дальневосточный федеральный университет» (ДВФУ), г. Владивосток, Россия. E-mail: starodubov.mi@dvfu.ru
2 Артемьева Ирина Леонидовна, доктор технических наук, профессор, заместитель директора по науке, профессор департамента программной инженерии и искусственного интеллекта Института математики и компьютерных технологий (Школы) ФГАОУ ВО «Дальневосточный федеральный университет» (ДВФУ), г. Владивосток, Россия. E-mail: artemeva.il@dvfu.ru
3 Селин Никита Александрович, студент департамента Информационной безопасности ФГАОУ ВО «Дальневосточный федеральный университет» (ДВФУ), г. Владивосток, Россия. E-mail: selin.na@dvfu.ru

4 Maxim I. Starodubov, Ph. D. student, Far Eastern Federal University (FEFU), Vladivostok, Russia. E mail: starodubov.mi@dvfu.ru
5 Irina L. Artemyeva, Dr. Sc., Professor, Deputy Director for Scientific Work at the Institute of Mathematics and Computer Technology (School) of the Far Eastern Federal University (FEFU), Vladivostok, Russia. E mail: artemeva.il@dvfu.ru
6 Nikita A. Selin, student of the Information Security Department of the Far Eastern Federal University (FEFU), Vladivostok, Russia. E-mail: selin.na@dvfu.ru

of any computer program can be represented as a set of records of its actions in a behavior report, which can be considered as signs of an object. The paper analyzes reports on the behavior of ransomware programs. Based on the generated data set using a deep learning algorithm, a model is built that allows further detection of malicious objects, and a method for detecting ransomware is described.

***The scientific novelty** consists in the creation of a method for detecting ransomware based on the analysis of the behavioral report of an executable object using the deep learning algorithm DeBERTa-V3.*

***Keywords:** malware, sandbox, deep learning, BERT, Ransomware, computer attacks.*

Введение

Проникновение компьютерных технологий в нашу жизнь приводит к её значительной от них зависимости и вызывает большой интерес злоумышленников. Доля атак[7] с использованием вредоносного программного обеспечения (ВПО, вредоносного ПО) на устройства пользователей превышает 55 %. Наибольший рост модификаций вредоносного ПО наблюдается в классе «программ-вымогателей», в 2023 году раз в 8 дней появлялось новое семейство, а каждые 22 минуты появлялась модификация известных семейств[8]. В связи с этим не теряет своей актуальности проблема обнаружения вредоносного программного обеспечения.

Существует два подхода к анализу программного обеспечения на наличие вредоносной составляющей. Статический анализ изучен хорошо и его эффективность достигает 99.4% [1]. Однако данный вид анализа неэффективен против сложных разновидностей вредоносных программ [2], которые используют методы шифрования [3], скрытия [4], упаковки [5] и полиморфных, олигоморфных и метаморфных преобразований [6]. Обнаружению при помощи динамического анализа уделено не так много внимания [7], хотя этот путь и выглядит более эффективным и более перспективным [8]. Данный вид анализа может помочь получить информацию о последовательности API-вызовов и системных вызовов, которые могут являться индикаторами возможного вредоносного поведения [9]. Однако такие последовательности могут быть очень длинными и сложными для понимания.

В связи с этим остро встаёт вопрос автоматизации их обработки. В контексте обнаружения ВПО для этого используются такие методы, как Word2Vec [10], HMM2Vec [11], BERT [12] и ELMo [13]. BERT показал наибольшую эффективность в системах MalBERT [14] и её усовершенствованной версии MalBERTv2 [15]. Однако, в указанных выше системах используется подход статического анализа. В литературе не было найдено исследований, показывающих связь указанных выше методов с подходом динамического анализа.

Метод обнаружения

Предлагаемый метод обнаружения ВПО основан на анализе и обработке поведенческого отчета исполняемого объекта. Поведенческий отчёт является одной из самых важных частей анализа вредоносных объектов, наряду с исходным кодом, и может дать полное представление о скрытых возможностях исследуемого объекта.

Пусть $Event = \{Event_1,...,Event_k\}$ – множество всех отслеживаемых программой событий. Размер данного множества зависит от средства виртуализации и среды выполнения объекта.

Основной алгоритм:

1. На основе поведенческого отчета исполняемого объекта «обработчик» формирует вектор доступных событий программы в среде выполнения $\tilde{U} = \{Event_1,...,Event_l\}$;
2. Сформированный вектор $\tilde{U}$ подается на вход обученного модуля DeBERTa-V3 [16]. На выходе имеется вектор значимых характеристик SC (significant characteristics);
3. Полученный вектор SC подается на вход заранее обученного классификатора, который на выходе выдает результат $D = \{0,1\}$.

Значение $D = 0$ соответствует тому, что исполняемый объект не является ВПО, а $D = 1$ – исполняемый объект является ВПО.

Схема предлагаемого метода обнаружения ВПО представлена на рисунке:

Рис. 1. Схема метода обнаружения ВПО

Для обучения элементов предлагаемого метода обнаружения ВПО требуется сформировать набор поведенческих отчетов исполняемых объектов из ряда ВПО и не ВПО.

7 Актуальные киберугрозы: IV квартал 2022 года [Электронный ресурс]. – Режим доступа: URL: https://www.ptsecurity.com/ru-ru/research/analytics/cybersecurity-threatscape-2022-q4/ (дата обращения: 02.01.2024).

8 Kaspersky Security Bulletin 2023. Statistics [Электронный ресурс]. – Режим доступа: URL: https://securelist.com/ksb-2023-statistics/111156/ (дата обращения: 02.01.2024).

Формирование набора отчетов поведения для обучения происходит следующим образом:

1. Исполняемый объект отправляется на анализ;
2. На физическом компьютере запускается виртуальная машина с известными параметрами и ей на исполнение передаётся анализируемый объект;
3. Измененные после исполнения объекта параметры передаются обратно на физический компьютер;
4. Формируется поведенческий отчёт объекта.

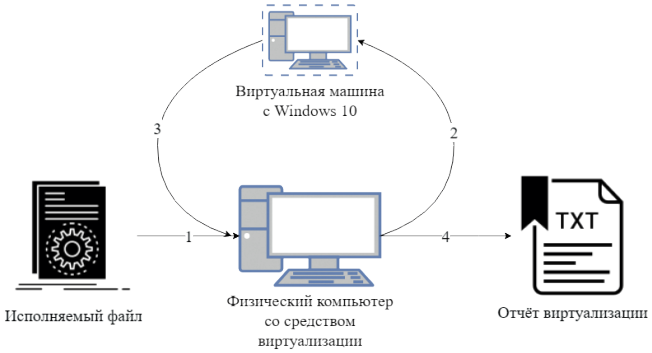

Рис. 2. Получение поведенческого отчёта

Набор отчетов поведения с использованием «обработчика» преобразовывается в массив доступных событий:

$U = \{U_1,...,U_n\}$, где $U_i = \{Event_{i,1},...,Event_{i,l}\}$, n – количество исполняемых объектов для обучения.

Затем с использования функционала $\varphi(U_i)$, обозначающего выполнение программы U_i и приводящий либо к безопасному состоянию системы «0», либо небезопасному состоянию «1», набор $\boldsymbol{U}$ преобразуется в вектор:

$$P = \{P_1,...,P_n\},$$

где $P_i = \varphi(U_i) = \{0,1\}$, отвечающий за принадлежность исполняемого объекта к ВПО.

В основе модуля DeBERTa-V3 лежит метод BERT [17] – метод обработки данных, основанный на трансформерах.

Из-за особенностей DeBERTa-V3 набор $\boldsymbol{U}$ разделяется на $\boldsymbol{B}$ и $\boldsymbol{R}$, где:

$$U_i \in \boldsymbol{B}, \text{ если } P_i == 1;$$
$$U_i \in \boldsymbol{R}, \text{ если } P_i == 0.$$

После обучения набор $\boldsymbol{U}$ обрабатывается DeBERTa-V3 и на выходе получаются значимые характеристики $\boldsymbol{SC}$.

Затем на основе $\boldsymbol{SC}$ и значений $\boldsymbol{P}$ производится обучение «классификатора».

Испытание предлагаемого метода

В этом разделе представлен эксперимент, который является испытанием предложенного метода. Общее представление всего эксперимента на высоком уровне представлено на рисунке 3.

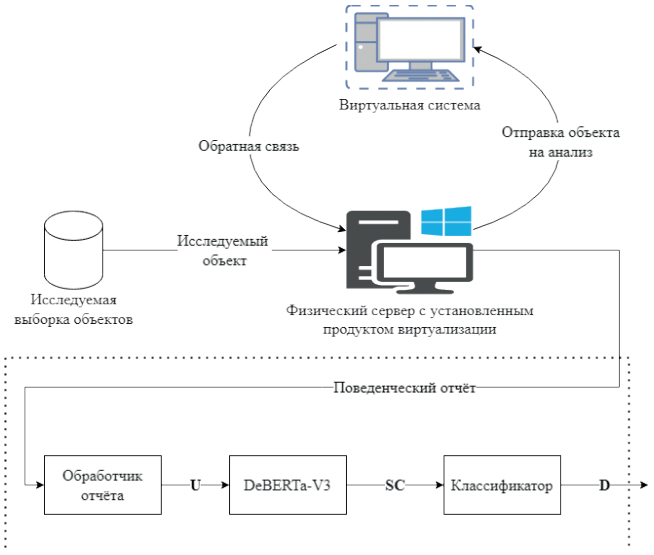

Рис. 3. Схема эксперимента

Процесс состоит из трех основных этапов:

- ✓ создание набора данных;
- ✓ этап предварительной обработки;
- ✓ этап точной настройки.

Набор данных

В качестве исследуемого множества была сформирована выборка их 600 000 файлов, обладающих следующими свойствами:

1. Все файлы являются исполняемыми объектами в системе ОС Windows и имеют формат PE[9];
2. Файлы должны эмулироваться средствами Cuckoo Sandbox;
3. Вредоносные файлы относятся к классу Ransomware.

Набор данных состоит из двух классов:

1) Класс «R» — 300000 вредоносных файлов из класса Ransomware;
2) Класс «B» — 300000 чистых легитимных приложений.

Все объекты выборки были получены из следующих источников: VirusTotal[10], VirusShare[11], Malware.lu[12], MalwareBazaar[13] и GitHib- ytisf/theZoo[14].

9 Формат PE [Электронный ресурс]. – Режим доступа: URL: https://learn.microsoft.com/ru-ru/windows/win32/debug/pe-format (дата обращения: 02.01.2024).

10 VirusTotal – Free Online Virus, Malware and URL Scanner [Электронный ресурс]. – Режим доступа: URL: https://www.virustotal.com/ (дата обращения: 02.01.2024).

11 VirusShare.com [Электронный ресурс]. – Режим доступа: URL: https://virusshare.com/ (дата обращения: 02.01.2024).

12 Malware.lu/ [Электронный ресурс]. – Режим доступа: URL: https://malware.lu/ (дата обращения: 02.01.2024).

13 MalwareBazaar | Malware sample exchange [Электронный ресурс]. – Режим доступа: URL: https://bazaar.abuse.ch/ (дата обращения: 02.01.2024).

14 A repository of LIVE malwares for your own joy and pleasure [Электронный ресурс]. – Режим доступа: URL: https://github.com[/]ytisf[/]theZoo (дата обращения: 02.01.2024).

Характеристики инструмента для формирования набора отчетов

В качестве среды для исследования поведения объектов была выбрана следующая конфигурация: Cuckoo Sandbox, продукт виртуализации Oracle VM VirtualBox с виртуальной машиной. Операционной системой была выбрана Windows 10 с предустановленными библиотеками, необходимыми для исполняемых файлов.

Данная конфигурация была выбрана по следующим причинам:

1. Возможность подключения к Cuckoo Sandbox дополнительных систем анализа;
2. Cuckoo Sandbox является open source проектом, из-за чего исходный код доступен для изучения и возможно его изменение под конкретную задачу;
3. ОС Windows 10 была выбрана в качестве гостевой операционной системы, так как это первая по числу интернет пользователей система семейства Windows[15].

Исходя из выбранного средства виртуализации и выбранной операционной системы Windows 10 множество Event состоит из 4000 событий.

Характеристики обучения DeBERTa-V3 и классификатора

С момента появления BERT многие исследовательские группы выпустили свои собственные реализации подхода, чаще всего также сопровождаемые предварительно обученными моделями. В данной работе использовалась модель DeBERTa-V3[16], которая состоит из 12 скрытых слоев, размер которых равен 768 нейронам. В качестве бинарного классификатора используется искусственная однослойная нейронная сеть (входной слой 768 нейронов, скрытый слой 768 нейронов и выходной слой размером 1 нейрон). DeBERTa-V3 и однослойная нейронная сеть обучались совместно.

Этап тонкой настройки выполняется на наборе данных, представленном в предыдущем пункте. Обучающие, валидационные и тестовые наборы распределяются как 50%, 20% и 30% соответственно. Обучающие, валидационные и тестовые наборы стратифицированы, что означает, что каждый набор имеет такое же соотношение вредоносного и чистого ПО, как и весь набор данных. Что касается параметров, то модель была точно настроена на 5 эпох с размером пакета (batch size) 32, с использованием в качестве функции оптимизатора «Adam» [18] и со скоростью обучения (learning rate) $3e^{-7}$. Эти параметры были тщательно подобраны, чтобы обеспечить наилучшую производительность модели. Все этапы обучения, тестирования и проверки выполнялись на двух графических процессорах NVIDIA GeForce RTX 3090 Ti с суммарным объёмом видеопамяти 48 ГБ.

Один полный эксперимент занял чуть меньше 13 часов. Кроме того, каждый полный эксперимент выполнялся десять раз с использованием разных исходных данных (т.е. разной последовательности для обучающих/валидационных/тестовых наборов), чтобы получить среднее значение производительности, максимально репрезентативное для модели.

Метрики оценки

В качестве мер оценок были выбраны следующие характеристики:

- ✓ Accuracy (точность) – показатель, оценивающий точность предсказания по всем классам;

$$Accurancy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

- ✓ Precision – показатель, оценивающий отношение числа верно классифицируемых объектов, как положительных, к общему числу положительно распознанных объектов, правильно и неправильно;

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

- ✓ Recall – показатель, оценивающий общее отношение числа верно классифицируемых объектов к общему числу объектов в кластере;

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

- ✓ F1-мера – агрегированный показатель, объединяющий как precision, так и recall;

$$F_1 = 2\frac{precision * recall}{precision + recall} \quad (4)$$

Выбранные меры позволяют объективно оценивать результаты эксперимента.

15 Desktop Windows Version Market Share Worldwide | Statcounter Global Stats [Электронный ресурс]. – Режим доступа: URL: https://gs.statcounter.com/os-version-market-share/windows/desktop/worldwide (дата обращения: 02.01.2024).

16 The implementation of DeBERTa [Электронный ресурс]. – Режим доступа: URL: https://github.com/microsoft/DeBERTa (дата обращения: 02.01.2024).

Таблица 1

Результаты исследования в сравнении с аналогичными

Исследование	Объём выборки, шт.	Accuracy	Precision	Recall	F1 мера
Наше исследование	600 000	0.97	0.99	0.96	0.98
MalBERT	22 000	0.90	0.79	0.89	0.80
MalBERTv2	22 000	0.93	0.92	0.97	0.89

Результаты

Неудивительно, что BERT очень хорошо справляется с задачами, связанными с текстом, такими как классификация отчётов виртуализации, которые, несмотря на то что являются JSON объектами, содержат в основном текстовые данные. При этом поведенческие отчёты содержат полную информацию о поведении объектов.

Результаты исследования представлены в таблице 1. Разработанный метод имеет лучшие результаты на большем объёме данных.

Заключение

В рамках данной работы разработан метод, позволяющий с эффективностью 97% (F1 мера) распознавать программы-вымогатели на основе анализа поведенческих отчётов.

Как показано в этом исследовании, модели, обученные на поведенческих отчётах, показывают хорошие результаты. Полученные результаты интересны с точки зрения анализа вредоносного ПО на реальных системах, «в живой природе» (англ. «WildList Malware»).

Исследование проведено при финансовой поддержке Минобрнауки России («Грант ИБ МТУСИ») № 40469-25/23-К.

Литература

1. *Ijaz M., Durad M. H., Ismail M. Static and dynamic malware analysis using machine learning //2019 16th International bhurban conference on applied sciences and technology (IBCAST). – IEEE, 2019. – С. 687–691. http://dx.doi.org/10.1109/IBCAST.2019.8667136*
2. *Aboaoja F. A. et al. Malware detection issues, challenges, and future directions: A survey //Applied Sciences. – 2022. – Т. 12. – №. 17. – С. 8482.*
3. *Asghar H. J. et al. Use of cryptography in malware obfuscation //Journal of Computer Virology and Hacking Techniques. – 2024. – Т. 20. – №. 1. – С. 135–152.*
4. *Zhang X. et al. Android application forensics: A survey of obfuscation, obfuscation detection and deobfuscation techniques and their impact on investigations //Forensic Science International: Digital Investigation. – 2021. – Т. 39. – С. 301285.*
5. *Cheng B. et al. {Obfuscation-Resilient} Executable Payload Extraction From Packed Malware //30th USENIX Security Symposium (USENIX Security 21). – 2021. – С. 3451–3468*
6. *Brezinski K. et al. Metamorphic malware and obfuscation: a survey of techniques, variants, and generation kits //Security and Communication Networks. – 2021. – Т. 2023.*
7. *Alsmadi T., Alqudah N. A survey on malware detection techniques //2021 international conference on information technology (ICIT). – IEEE, 2021. – С. 371–376.*
8. *Aslan Ö. A., Samet R. A comprehensive review on malware detection approaches //IEEE access. – 2020. – Т. 8. – С. 6249–6271.*
9. *Maniriho P., Mahmood A. N., Chowdhury M. J. M. API-MalDetect: Automated malware detection framework for windows based on API calls and deep learning techniques //Journal of Network and Computer Applications. – 2023. – Т. 218. – С. 103704.*
10. *Sun J. et al. Categorizing malware via A Word2Vec-based temporal convolutional network scheme //Journal of Cloud Computing. – 2020. – Т. 9. – С. 1–14.*
11. *Chandak A., Lee W., Stamp M. A comparison of word2vec, hmm2vec, and pca2vec for malware classification //Malware analysis using artificial intelligence and deep learning. – 2021. – С. 287–320.*
12. *Yesir S., Soğukpinar İ. Malware detection and classification using fasttext and bert //2021 9th International Symposium on Digital Forensics and Security (ISDFS). – IEEE, 2021. – С. 1–6.*
13. *Pandya V. Contextualized Vector Embeddings for Malware Detection. – 2022. https://doi.org/10.31979/etd.rjra-9c8m*
14. *Rahali A., Akhloufi M. A. Malbert: Malware detection using bidirectional encoder representations from transformers //2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC). – IEEE – 2021. – С. 3226–3231.*
15. *Rahali A., Akhloufi M. A. MalBERTv2: Code Aware BERT-Based Model for Malware Identification //Big Data and Cognitive Computing. – 2023. – Т. 7. – №. 2. – С. 60.*
16. *He P., Gao J., Chen W. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing // arXiv preprint arXiv:2111.09543. – 2021.*
17. *Devlin J. et al. Bert: Pre-training of deep bidirectional transformers for language understanding //arXiv preprint arXiv:1810.04805. – 2018.*
18. *Kingma D. P., Ba J. Adam: A method for stochastic optimization //arXiv preprint arXiv:1412.6980. – 2014.*