АНАЛИЗ PE-ЗАГОЛОВКА ДЛЯ ОБНАРУЖЕНИЯ ВРЕДОНОСНЫХ ПРОГРАММ НА ОСНОВЕ МАШИННОГО ОБУЧЕНИЯ

ФАН ТХИ ХАЙ ХА

Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина)

Аннотация. В данной статье представлены рекомендуемые методы классификации и обнаружения вредоносных программ PE формата для ОС Windows на основе машинного обучения, а также рекомендации по их внедрению.

Ключевые слова: машинное обучение, вредоносные программы (malware), PE (Portable Executable) заголовок, …

Введение

Сегодня компьютерные программы и приложения разрабатываются с большой скоростью. Вредоносные программные обеспечения (вредоносные ПО, агл. *malware*) появились и растут во многих форматах и становятся все более изощренными. Компьютерные преступники используют их в качестве инструмента для проникновения, кражи или подделки информации, нанося огромный ущерб отдельным лицам, предприятиям и даже угрожая национальной безопасности. В то же время машинное обучение также широко используется в качестве мощного инструмента для экспертов по безопасности для выявления вредоносных программ с высокой точностью, когда количество вредоносных достаточно велико, также их варианты стали разнообразными. Одним из основных методов является метод анализа PE-заголовка.

Целью является исследование существующих подходов к статическому анализу и отбор значимых для анализа атрибут PE файлов.

Приведенная в работе информация относится к исследованию вредоносных программ, представляющих собой приложение Windows (PE-.exe файл). Portable Executable (PE, переносимый исполняемый) — формат исполняемых файлов, объектного кода и динамических библиотек, используемый в 32- и 64-разрядных версиях операционной системы Microsoft Windows. Это стандартный двоичный формат для файлов EXE и DLL под Windows.

Структура PE файла представлена на рисунке 1.

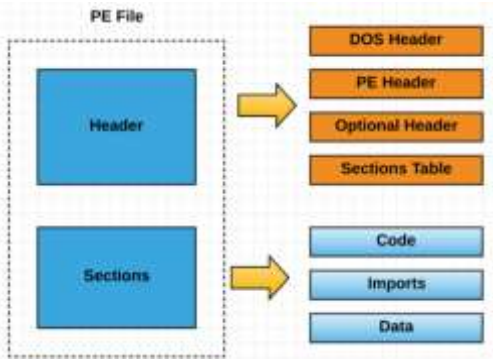


Рис. 1. Структура PE файла

В рамках этой статьи нас интересует роль PE заголовок в выявлении вредоносных программ**.** PE-Заголовок содержит полезную информацию для анализа вредоносного програмытак как он содержит информацию, которая требуется операционная система для запуска исполняемого файла. Понимание общей структуры файлов PE и заголовка PE имеет

важное значение в процессе анализа вредоносных программ, поэтому специалисты могут легко определить, как вредоносный код воздействует на файл. Особенно для вирусов (вредоносный код выполняет инфекционную функцию).

Роль машинного обучения

Методы обнаружения вредоносных программ можно разделить на методы, основанные на сигнатурах и поведении. В настоящее время детекторы вредоносных программ, основанные на сигнатурах, могут хорошо работать с ранее известными вредоносными программами, которые уже были обнаружены некоторыми поставщиками антивирусных программ. Однако он не может обнаружить полиморфное вредоносное ПО, которое может изменять свои сигнатуры, а также новое вредоносное ПО, сигнатуры которого еще не созданы. Одним из решений этой проблемы является использование эвристического анализа в сочетании с методами машинного обучения, которые обеспечивают более высокую эффективность при обнаружении. Машинное обучение базируется на трех ключевых элементах: набор вредоносных данных (malware dataset), признаки (features) и модели/алгоритмы (model/algorithm).

Краткий рабочий процесс нашего обучения классификатора изображен на рисунке 2. Он состоит из классических этапов машинного обучения.

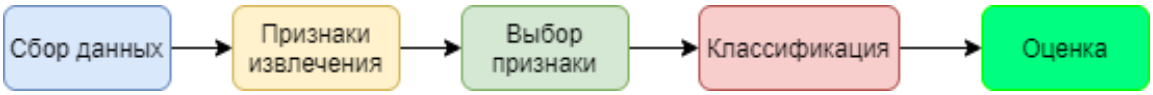


Рис. 2. Предлагаемая схема рабочего процесса простого машинного обучения

Набор данных был получен для проведения эксперимента с помощью сервиса VirusShare, сохранен в формате «data.csv» и состоит из:

- 41 323 бинарных файлов Windows (исполняемые файлы .exe и .dll) в виде допустимых (легитимных) файлов.
- 96 724 вредоносных файла, загруженных с веб-сайта VirusShare.

Тем не менее, архивы содержали все виды вредоносных файлов. Поэтому нам нужно было отфильтровать исполняемые файлы PE.

Программный комплекс:

- Jupyter notebook: веб-приложение, которое позволяет структурировать и перезапуск кусков кода в удобной для пользователя форме. Используется на этапах предварительной обработки и классификации данных;
- Модуль Python pefile: модуль Python, используемый для анализа файлов в формате PE. Модуль загружает все атрибуты PE-файла в объект. Эти атрибуты легко доступны через этот объект;
- Библиотека pandas: модуль Python используется для обеспечения функциональности инкапсуляции манипулирования данными, подготовки кадров данных для фазы обучения, предоставляемой scikit-learn. Загрузка вредоносного ПО «data.csv» с использованием библиотеки pandas в Jupyter Notebook представлена на рисунке 3. Результатом, полученным из сценария, является количество образ двух типов форматов, а также количество их важных признак.

```
In [3]: import pandas as pd
        malData = pd.read_csv("data.csv", sep="|")
        legit = malData[0:41323].drop(["legitimate"],axis=1)
        mal = malData[41323::].drop(['legitimate'], axis=1)
        print("Форма легитимного набора данных: %s образцы, %s признаки"%(legit.shape[0],legit.shape[1]))
        print("Форма набора данных вредоносных программ: %s образцы, %s признаки"%(mal.shape[0],mal.shape[1]))

        Форма легитимного набора данных: 41323 образцы, 56 признаки
        Форма набора данных вредоносных программ: 96724 образцы, 56 признаки
```

Рис.3. Загрузка вредоносного ПО «data.csv» с использованием библиотеки pandas

- Библиотека scikit-learn: Python фреймворк для машинного обучения, охватывающие методы выбора функций, алгоритмы классификатора, функции проверки и оценки производительности. Алгоритм выбрал 14 признаки заголовка, которые считаются наиболее важными. На следующих шагах были извлечены только соответствующие заголовки и созданы новые наборы данных для обучения классификатора.

```
LoaderFlags
NumberOfRvaAndSizes
SizeOfHeapCommit
BaseOfCode
SizeOfUninitializedData
SectionMaxRawsize
SizeOfInitializedData
SizeOfImage
ResourcesMaxSize
ResourcesMeanSize
SectionsMeanRawsize
BaseOfData
SizeOfCode
SizeOfStackCommit
SectionsMeanVirtualsize
SizeOfHeapReserve
```

Рис. 4 Самые важные признаки заголовка

Затем необходимо обучить модель на наборе данных вредоносных программ, чтобы он научился различать легитимные и вредоносные образцы. Используются несколько алгоритмов машинного обучения для построения простой модели классификации вредоносных программ: Случайного леса и Градиентного бустинга.

Необходимо протестировать двух разных алгоритма и сравнить их эффективность. Для этого необходимо разделить данные на 4 подмножества: легитимное обучение (legit training), легитимное тестирование (legit test), обучение вредоносных ПО (malware training) и тестирование вредоносного ПО (malware test). Исходный код написан на языке Python, с использованием библиотек, перечисленных выше, простая модель машинного обучения сработала как цель, которую нужно изучить: классифицировать вредоносные программы с относительной точностью высокая (пример исходного кода для алгоритма случайного леса указан на рисунке 5).

Алгоритм случайного леса

```
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_validate
from sklearn.metrics import confusion_matrix
Legit_Train, Legit_Test, Malware_Train, Malware_Test = sklearn.model_selection.train_test_split(Data_new, Target ,test_size=0.2)
clf = sklearn.ensemble.RandomForestClassifier(n_estimators=50)
clf.fit(Legit_Train, Malware_Train)
Result = clf.predict(Legit_Test)
CM = confusion_matrix(Malware_Test,Result)
print("Оценка точности по алгоритму Случайного леса:",clf.score(Legit_Test, Malware_Test)*100)
print("Ложно-положительная ставка:  %f %%" % ((CM[0][1] / float(sum(CM[0])))*100))
print("Ложно-отрицательная ставка: %f %%" % ((CM[1][0] / float(sum(CM[1]))*100)))

Оценка точности по алгоритму Случайного леса: 99.44947482796888
Ложно-положительная ставка:  0.414379 %
Ложно-отрицательная ставка: 0.867052 %
```

Рис. 5. Часть кода для алгоритма случайного леса

Экспериментальное обучение на инструменте Jupyter Notebook, получили следующие результаты о точности каждого алгоритма, ложноположительных и ложноотрицательных срабатываний (таблица 1).

Таблица 1

Результаты выполнения программы и сравнение 2 алгоритмов

№	Алгоритм	Точность, %	Процент ложноположительных срабатываний, %	Процент ложноотрицательных срабатываний, %
1	Случайный лес	99.358927	0.489312	1.000610
2	Градиентный бустинг	99.011227	0.727422	1.619483

Заключение

Экспериментируя с примерно 100 000 файлов, результаты этой статьи показали нам эффективное использование алгоритмов машинного обучения для обнаружения и классификации вредоносных программ (формат PE). Две модели (из двух алгоритмов классификации) машинного обучения, изученные в статье, соответствуют высокой точности, просты в реализации и широко применяются сегодня в областях информационной безопасности.

Список литературы

1. Samuel Kim. PE Header Analysis for Malware Detection [Электронный ресурс]: https://scholarworks.sjsu.edu/cgi/viewcontent.cgi?article=1624&context=etd_projects
2. Cristi Vlad. Machine Learning for Malware Detection [Электронный ресурс]: https://www.youtube.com/watch?v=HMWulRpPpuA&list=PLonlF40eS6nzc7TqDshRo7k-mTM1Tu_j4.
3. Kaspersky Lab. 2016. Kaspersky Security Bulletin 2015. Overall statistics for 2015. [WWW document] - Режим доступа: https://securelist.com/analysis/kasperskysecurity-bulletin/73038/kaspersky-security-bulletin-2015-overall-statistics-for2015, свободный (дата обращения: 20.04.2020)
4. Prateek Lalwani. VirusShare data.csv - Режим доступа: https://raw.githubusercontent.com/prk54/malware-detection-machine-learning-approach/master/data.csv, свободный (дата обращения: 20.04.2020)