Инженерный вестник Дона, №3 (2015)
ivdon.ru/ru/magazine/archive/n3y2015/3265

Методика формирования обучающего множества при использовании статических антивирусных методов эвристического анализа

Р.Ю. Демина, И.М. Ажмухамедов

Астраханский государственный университет

Аннотация: В статье рассмотрена проблема антивирусного эвристического анализа. Важной особенностью статических эвристических методик является зависимость показателей обнаружения от состава обучающего множества. Для решения данной проблемы предложено разработать методику формирования обучающей выборки. Были введены понятия меры схожести и матрицы схожести. Мера схожести показывает степень соответствия одного файла другому. Матрица схожести состоит из мер схожести файлов обучающего множества. С использованием введённых определений была разработана методика формирования обучающей выборки. Основная идея методики состоит в том, что бы оставшиеся в обучающем множестве файлы имели как можно меньшую меру схожести с остальными. Проведенная экспериментальная проверка показала эффективность и практическую значимость предложенного решения.
Ключевые слова: информационная безопасность, антивирусная защита, эвристический анализ, машинное обучение, обучающее множество, вредоносное программное обеспечение.

Введение.

Число вирусов постоянно растет и антивирусные компании не всегда успевают своевременно их обнаружить [1] и предложить конечному пользователю соответствующее программное обеспечение (ПО) для нейтрализации угроз.

Традиционно для обнаружения вирусов используются два основных подхода: сигнатурный (детерминированный) и эвристический (вероятностный). На сегодняшний день основным является сигнатурный подход, который предусматривает: идентификацию экспертами антивирусной компании неизвестной программы как вирусной; определение всех ее признаков; выявление модификаций, вносимых данной программой в систему; занесение информации о вирусе в базы данных сигнатур [2]. Антивирусное ПО, использующее только сигнатурный подход, не способно обнаружить вирусы «нулевого дня», информация о которых еще не попала в

базы. К моменту обновления баз пользователю уже может быть причинен значительный ущерб [3].

Для решения данной проблемы дополнительно к сигнатурному подходу используется эвристический анализ [4]. При этом файл с некоторой вероятностью может быть признан «потенциально опасным» исходя из его поведения (динамический подход) или из анализа его структуры (статический подход).

Идея динамического подхода состоит в том, чтобы до того, как программа будет запущена на компьютере пользователя, эмулировать ее запуск в безопасном виртуальном окружении, которое называется также буфером эмуляции, или "песочницей". Динамический эвристический анализатор считывает часть кода приложения в буфер и с помощью специальных программных методов эмулирует его исполнение. Если в процессе этого "псевдоисполнения" обнаруживаются какие-либо подозрительные действия, объект признается потенциально опасным и его запуск на компьютере блокируется до особых распоряжений пользователя.

Динамический анализ, несмотря на его эффективность, требователен к ресурсам ПК, так как приходится использовать безопасное виртуальное пространство. Кроме того, реальный запуск программы на компьютере откладывается на время проверки. Статический эвристический анализ свободен от этих недостатков. Особое место в нем занимает использование алгоритмов классификации характерных для вредоносных программ подозрительных признаков (команд) (деревья решений, «случайный лес», градиентный бустинг и др.) [5, 6].

Этапы статического анализа

Статический анализ в общем случае состоит из двух основных этапов: этапа обучения и этапа использования результатов (обнаружения вирусных программ).

На этапе обучения формируется выборка из зараженных (вирусных) и «чистых» (легитимных) файлов [7, 8]. В структуре файлов выделяются признаки, характеризующие каждый из них как вирусный или легитимный. В результате для каждого файла составляется перечень признаковых характеристик. Далее происходит отбор наиболее значимых (информативных) признаков, а избыточные и нерелевантные признаки отбрасываются. На этапе обнаружения из сканируемого файла извлекаются признаковые характеристики. Они сравниваются с характеристиками, выделенными в процессе обучения; определяется степень их соответствия вредоносному ПО. В случае если степень соответствия выше некоторого порогового значения, файл с некоторой вероятностью признается вирусным.

Постановка задачи формирования обучающей выборки

Одной из важнейших проблем статических методик является отбор файлов для обучающей выборки [9]. Качество построенного классификатора во многом определяется обучающим множеством. При этом большое значение имеют следующие принципы его формирования:

1. Пропорциональность. Необходимо соблюдать определенное соотношение между количеством вирусных и «чистых» файлов, из которых отбираются информативные признаки.

2. Разнообразие. Файлы, составляющие обучающую выборку, должны репрезентативно представлять все реальное многообразие файлов, встречающихся на практике. Количество файлов схожих по своему двоичному представлению и механизмам действия должно быть минимизировано.

Следование данным принципам приводит к тому, что любое обновление обучающего множества влечет за собой необходимость переобучения классификатора, поскольку добавление файлов в обучающее множество нарушает принцип пропорциональность, а замена файлов -

нарушает принцип разнообразия. Поэтому задача формирования обучающей выборки является нетривиальной. Формально она может быть сформулирована следующим образом: необходимо разработать методику отбора из множества файлов F обучающей выборки F' с целью повышения эффективности работы классификаторов: увеличение параметров TP и TN. Под TP понимается число верно классифицированных «легитимных» файлов, а под TN – число верно распознанных вирусов.

Решение задачи

Мера схожести файлов

Для решения задачи обеспечения максимального разнообразия обучающей выборки необходимо иметь возможность сравнения файлов. Для этого требуется ввести меру схожести.

При анализе содержимого файл обычно представляется в виде набора n-грамм (обычно 3- или 4-грамм). Под n-граммой понимается любая последовательность расположенных подряд n байт. Количество n-грамм равно количеству байт в файле минус (n-1). Например, файл длиной 1000 байт состоит из 998 3-грамм.

Определим меру схожести файла A с файлом B как отношение числа уникальных 3-грамм файла A, которые встречаются в наборе 3-грамм файла B, к количеству уникальных 3-грамм файла A:

$$\rho(A, B) = \frac{|\bar{A} \cap \bar{B}|}{|\bar{A}|},$$

где $\bar{A}$ –множество n-грамм файла A, $\bar{B}$ –множество n-грамм файла B.

Величина $\rho(A,B) \in [0,1]$ и характеризует количество уникальных n-грамм файла A, входящих во множество уникальных n-грамм файла B. Если

ни одна из n-грамм файла A не входит в файл B, то $\rho(A, B) = 0$. Если все n-граммы файла A входят в файл B, то $\rho(A, B) = 1$.

Следует отметить, что в общем случае $\rho(A, B) \neq \rho(B, A)$. Например, файл A может представлять собой исполняемый модуль, состоящий исключительно из подключенных к нему внешних библиотек, а файл B кроме этих библиотек может содержать и некоторый дополнительный функционал. В этом случае $\rho(A,B) \approx 1$, а $\rho(B,A)$ может существенно отличаться от 1. Файлы A и B считаются взаимно схожими, если $\rho(A, B) \approx \rho(B, A)$.

Введенная таким образом мера позволяет построить матрицу схожести для множества файлов F.

Матрица схожести

Матрицей схожести назовем квадратную матрицу, состоящую из элементов ρ_{ij} – мер схожести i-го файла с j-м. На главной диагонали данной матрицы расположены 1, т.к. каждый файл полностью схож сам с собой. Матрица не обязательно симметрична, поскольку, как было отмечено выше, мера схожести файла A с файлом B не обязательно равна мере схожести файла B с файлом A.

Построенная таким образом матрица схожести может быть положена в основу алгоритма отбора максимально разнообразных файлов F' из множества файлов F ($|F| \geq |F'|$) для использования на этапе обучения классификаторов.

Методика отбора файлов

1. Задать пороговое значение меры схожести k, при превышении которого сравниваемые файлы считаются схожими.

2. Для каждого i-го файла подсчитать параметр L_i – количество j-ых файлов ($i \neq j$), для которых $\rho_{ij} \geq k$; и тем самым выделить группы взаимно схожих файлов. Файлы с $L = 0$ считаются уникальными в рамках данного множества.

3. Из каждой группы произвольно выбрать и оставить только один файл.

4. Включить файлы в итоговую обучающую выборку F' в порядке убывания параметра L.

5. Если в F' недостаточно элементов, то необходимо понизить пороговое значение k и повторить шаги 1-4.

Расчетный пример

Пусть множество F состоит из 10 исполняемых файлов, из которых необходимо отобрать 6 файлов во множество F'.

Допустим, матрица схожести для множества F имеет вид, представленный в таблице № 1.

Таблица № 1

Матрица схожести

Файлы	*L*	1.exe	2.exe	3.exe	4.exe	5.exe	6.exe	7.exe	8.exe	9.exe	10.exe
1.exe	*1*	1	0,713	**0,982**	0,614	0,049	0,049	0,576	0,587	0,008	0,049
2.exe	*5*	**0,945**	1	**0,945**	**0,859**	0,018	0,018	**0,828**	**0,836**	0,003	0,018
3.exe	*1*	**0,982**	0,714	1	0,614	0,048	0,049	0,576	0,587	0,008	0,049
4.exe	0	0,197	0,197	0,197	1	0,012	0,012	0,145	0,152	0,002	0,012
5.exe	*2*	0,301	0,239	0,303	0,042	1	**0,977**	0,009	0,016	0,094	**0,981**
6.exe	*2*	0,301	0,239	0,304	0,042	**0,977**	1	0,009	0,015	0,094	**0,978**
7.exe	0	0,008	0,008	0,008	0,007	0,001	0,001	1	0,015	0,001	0,001
8.exe	0	0,042	0,040	0,042	0,040	0,011	0,011	0,059	1	0,001	0,011
9.exe	0	0,065	0,041	0,065	0,058	0,078	0,078	0,001	0,005	1	0,078
10.exe	*2*	0,301	0,239	0,304	0,042	**0,980**	**0,978**	0,009	0,015	0,094	0

Примем пороговое значение k равным 0,8. В таблице № 1 жирным шрифтом выделены меры схожести, превышающие установленный предел.

Для каждого файла подсчитаем количество схожих с ним файлов L. В таблице 1 курсивом выделено, для каких файлов $L > 0$.

Определим группы взаимно схожих файлов. Из таблицы видно, что взаимно схожими являются 1-ый и 3-ий, а также 5-ый, 6-ой и 10-ый файлы.

Из каждой группы оставим по одному файлу. Результаты приведены в таблице № 2.

Таблица № 2

Матрица схожести, после исключения схожих файлов

Файлы	*L*	1.exe	2.exe	4.exe	7.exe	8.exe	9.exe	10.exe
1.exe	*0*	1	0,713	0,614	0,576	0,587	0,008	0,049
2.exe	*4*	**0,945**	1	**0,859**	**0,828**	**0,836**	0,003	0,018
4.exe	0	0,197	0,197	1	0,145	0,152	0,002	0,012
7.exe	0	0,008	0,008	0,007	1	0,015	0,001	0,001
8.exe	0	0,042	0,040	0,040	0,059	1	0,001	0,011
9.exe	0	0,065	0,041	0,058	0,001	0,005	1	0,078
10.exe	0	0,301	0,239	0,042	0,009	0,015	0,094	0

Из таблицы 2 видно, что имеется файл (2.exe), для которого количество схожих с ним файлов более 0, хотя взаимно схожих с ним нет.

Исключив данный файл, мы получаем набор из шести максимально различающихся между собой файлов. Они и составят искомое множество F', которое будет использоваться для обучения классификаторов.

Экспериментальная проверка предложенной методики

Для проверки предложенной методики процесс вычисления матрицы схожести был автоматизирован. При проведении экспериментов, состав обучающей выборки был сформирован таким образом, чтобы он максимально отражал реальное состояние предметной области на сегодняшний день:

1. В обучающую выборку в качестве «чистых» файлов были включены исполняемые файлы, выполнение которых требует прав администратора, и действия которых могут быть распознаны как потенциально опасные.

Например, программы удаленного администрирования, очистки диска и другие аналогичные утилиты.

2. В обучающую выборку было включено рекламное ПО. Подобные вирусы в последнее время приобрели широкое распространение и приносят пользователю ряд неудобств из-за своей навязчивости. Подобные вирусы в своем коде не содержат ничего необычного: скачивание дополнительных программ и их запуск – нормальные, часто встречающиеся функции «чистых» программ.

Трудноопределимые файлы в обучающей выборке – это сознательное усложнение работы для классификатора с целью максимально приблизить условия проведения экспериментов к реальным.

Для классификации файлов (для разделения их на «легитимные» и вирусные) использовалась разработанная израильскими учеными методика MAL-ID [10]. Данный классификатор специально создан для решения задач антивирусного анализа. Количество «легитимных» файлов, использованных на этапе обучения, было равно 2228. По мнению авторов методики MAL-ID, соотношение вредоносных и «чистых» файлов должно быть 1 к 3. Исходя из этого, необходимо включить в обучающее множество около 750-800 вирусных файлов (из имеющихся в наличии 1230). Для проверки работы классификатора после обучения было сформировано 6 контрольных наборов «легитимных» и вирусных файлов по 200 файлов в каждом наборе. Проверочные множества не пересекались с обучающими множествами.

Обучение было проведено на двух наборах файлов. Первый набор представлял собой чистые файлы в полном объеме и специально отобранные вирусы. Второй набор состоял из тех же чистых файлов и случайно отобранных вирусных файлов. В таблице № 3 представлены сравнительные результаты распознавания.

Таблица № 3

Результаты распознавания

	TN (%)						TP (%)					
	1	2	3	4	5	6	1	2	3	4	5	6
Случайным образом сформированное обучающее множество	39	56,5	52	46,5	48	66	97,5	86	90	89,5	98,5	83,5
Специальным образом сформированное обучающее множество	42	66	60	52	54,5	71,5	96	84	89	89	98	81,5
Разница показателей между случайно и специально сформированными множествами	3	9,5	8	5,5	6,5	5,5	-1,5	-2	-1	-0,5	-0,5	-2
Среднее значение разницы	6,33						-1,25					

Как видно из таблицы № 3, применение предложенной методики привело к стабильному увеличению параметра TN (повысилось обнаружение вирусов) в среднем на 6,33% во всех контрольных группах. При этом, как и ожидалось, увеличение TN повлекло за собой незначительное (на 1,25%) уменьшение TP (увеличилось количество ложных срабатываний на «чистых» файлах).

Выводы

Введенная в работе мера схожести файлов позволила разработать методику формирования обучающего множества для классификаторов, на основе которых реализуются алгоритмы статического поиска вредоносного программного обеспечения. Экспериментальная проверка, проведенная с использованием алгоритма классификации MAL-ID, показала эффективность предложенного подхода. Это позволит в дальнейшем при применении других классификаторов улучшить показатели статического эвристического анализа.

Литература

1. Бурякова Н.А., Чернов А.В. Классификация частично формализованных и формальных моделей и методов верификации

программного обеспечения // Инженерный вестник Дона, 2010, №4 URL: ivdon.ru/ru/magazine/archive/n4y2010/259.

2. Путин, Е.О., Тимофеев А.В. Классификатор для статического обнаружения компьютерных вирусов, основанный на машинном обучении // Information Technologies & Knowledge. 2014. №2. С. 103-112.

3. Ажмухамедов И.М., Марьенков А.Н. Поиск и оценка аномалий сетевого трафика на основе циклического анализа // Инженерный вестник Дона, 2012, №2 URL: ivdon.ru/magazine/archive/n2y2012/742/

4. Рассел С., Норвиг П. Искусственный интеллект: современный подход. 2-е изд. М.: Вильямс, 2007. 1408 с.

5. Marsland S. Machine Learning: An Algorithmic Perspective. 2-nd edition. Chapman and Hall/CRC, 2014. 457 p.

6. Harrington P. Machine Learning in Action. Manning Publications Co, 2012. 382p.

7. Флах П. Машинное обучение. Наука и искусство построения алгоритмов, которые извлекают знания из данных. М.:ДМК Пресс,2015.400 с.

8. Потапов А.С. Распознавание образов и машинное восприятие. СПб.: Политехника, 2007. 552 с.

9. Abou-Assaleh, T., Cercone, N., Keˇselj, V., Sweidan, R. N-gram-based Detection of New Malicious Code Privacy and Security Laboratory // Faculty of Computer Science, Dalhousie University, 2004, №4 URL: vxheaven.org/lib/pdf/N-gram-based%20Detection%20of% 20New% 20Malicious%20Code.pdf

10. Tahan G, Rokach L, Shahar Y Mal-ID: Automatic Malware Detection Using Common Segment Analysis and Meta-Features // Journal of Machine Learning Research. 2012. №13. pp. 949-979.

References

1. Burjakova N.A., Chernov A.V. Inzhenernyj vestnik Dona, 2010. №4. URL: ivdon.ru/ru/magazine/archive/n4y2010/259.

2. Putin, E.O., Timofeev A.V. Information Technologies & Knowledge. 2014. №2. С. 103-112.

3. Azhmuhamedov I.M., Mar'enkov A.N. Inzhenernyj vestnik Dona, 2012. №2. URL: ivdon.ru/magazine/archive/n2y2012/742/

4. Russel S., Norvig P. Iskusstvennyj intellekt: sovremennyj podhod [Artificial Intelligence: A Modern Approach]. 2-nd edition M.: Vil'jams, 2007. 1408 p.

5. Marsland S. Machine Learning: An Algorithmic Perspective. 2-nd edition. Chapman and Hall/CRC, 2014. 457 p.

6. Harrington P. Machine Learning in Action. Manning Publications Co, 2012. 382 p.

7. Flach P. Mashinnoe obuchenie. Nauka i iskusstvo postroenija algoritmov, kotorye izvlekajut znanija iz dannyh [Machine Learning/ The art and Since of Algorithms that Make Sense of Data]. M.: DMK Press, 2015. 400 p.

8. Potapov A.S. Raspoznavanie obrazov i mashinnoe vosprijatie [Recognition of images and machine perception]. SPb: Politehnika, 2007. 552 p.

9. Abou-Assaleh, T., Cercone, N., Keˇselj, V., Sweidan, R. N-gram-based Detection of New Malicious Code Privacy and Security Laboratory. Faculty of Computer Science, Dalhousie University. 2004. №4. URL: vxheaven.org/lib/pdf/N-gram-based%20Detection%20of%20New%20Malicious%20Code.pdf

10. Tahan G., Rokach L., Shahar Y. Mal-ID: Automatic Malware Detection Using Common Segment Analysis and Meta-Features. Journal of Machine Learning Research. 2012. №13. pp. 949-979.