Вестник УГАТУ
Уфа: УГАТУ, 2012 Т. 16, № 8 (53). С. 73–79
УПРАВЛЕНИЕ, ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА И ИНФОРМАТИКА

УДК 004.738.5:004.8

**В. Д. Котов**

# ОБНАРУЖЕНИЕ ВРЕДОНОСНЫХ ИНТЕРНЕТ-СТРАНИЦ НА ОСНОВЕ ТЕХНОЛОГИИ НЕЙРОННЫХ СЕТЕЙ

В данной статье представлены результаты исследования, посвященного обнаружению вредоносных страниц на основе машинного обучения. Для определения набора признаков было проанализировано 310 экземпляров вредоносных страниц, появившихся за период 2003-2012 г. В качестве технологии машинного обучения для обнаружения вредоносных страниц была выбрана нейронная сеть. В статье представлены результаты статистических тестов, проведенных с целью проверки эффективности разработанной системы. *Вредоносные URL*; *JavaScript*; *нейронная сеть*; *интернет-угрозы*

## ВВЕДЕНИЕ

В последние годы значительно преобразился ландшафт информационных угроз. В связи с развитием интернет-технологий на первое место среди источников заражения компьютеров вредоносными программами вышли атакующие интернет-страницы. Это связано, в первую очередь, с тем, что современные браузеры представляют собой комплексные модульные приложения, способные обрабатывать большое количество форматов данных (текст, XML, изображения) и поддерживать различные расширения (Adobe Flash, Apple Quick Time и т. п.). Кроме того, подавляющее большинство браузеров имеет встроенный интерпретатор JavaScript. Высокая сложность приложения, как правило, ведет к возникновению в нем ошибок на этапе разработки, которые впоследствии могут быть эксплуатированы злоумышленниками. Согласно обзору интернет-угроз ЗАО «Лаборатория Касперского» за первый квартал 2012 г. [1], доля вредоносных страниц в заражении компьютеров вредоносными программами составляет 84,30%.

В связи с этим перспективным направлением исследования в области компьютерной безопасности является мониторинг интернет-угроз, основанный на поисковых роботах. Целью такого мониторинга является открытие новых вредоносных программ в интернете, их анализ и разработка средств противодействия. В отличие от традиционного подхода, при котором новые средства защиты появляются в ответ на уже произошедшие инциденты, поиск и раннее обнаружение новых типов атак и вредоносных программ позволит значительно сократить масштабы заражения. Важнейшей подзадачей здесь является эффективное обнаружение вредоносных страниц. Об этом и пойдет речь в настоящей статье.

Вклад данной работы в направление исследования можно выразить следующими тезисами:

- представлен анализ экземпляров вредоносных страниц, появившихся в период 2003–2012 гг.;
- выбраны признаки, позволяющие автоматически различать вредоносные и нормальные интернет-страницы;
- выбрана технология машинного обучения для классификации интернет-страниц – нейронная сеть;
- подготовлена обучающая выборка и проведены эксперименты, подтверждающие высокую дискриминационную способность предлагаемого набора признаков.

В статье приводятся краткое введение в проблематику исследования, обзор опубликованных работ по теме исследования, результаты и выводы анализа вредоносных страниц; описан процесс разработки нейросетевого детектора, а также представлены результаты экспериментов.

## ПРОБЛЕМА ВРЕДОНОСНЫХ ИНТЕРНЕТ-СТРАНИЦ

Браузеры являются привлекательным вектором атаки не только в силу свой сложности и, как следствие, подверженности ошибкам, но и потому, что модули браузера и его расширения делят между собой одно адресное пространство. Это обстоятельство позволяет злоумышленникам использовать интерпретатор JavaScript для осуществления атаки. Тот факт, что объекты JavaScript создаются в куче браузера,

Контактная информация: 8(347)273-06-72

позволяет не только использовать их как носители вредоносного кода, но и применять программные структуры языка (циклы, математические операции и операции со строками) для выгодного размещения вредоносных инструкций в памяти и манипулировать их элементами (например, указателем на адрес кучи). Кроме того, сами атакующие сценарии JavaScript могут быть обфусцированы, т. е. упакованы, зашифрованы или как-то иначе усложнены для затруднения анализа. Наличие в JavaScript таких функций как eval (функция, выполняющая код, переданный ей в виде строки в качестве аргумента) и интеграция с DOM-моделью браузера позволяют злоумышленнику скрывать вредоносный код путем различных строковых и математических преобразований с дальнейшей «распаковкой» на лету.

Эксплуатация уязвимостей в браузерах позволяет злоумышленнику инфицировать компьютеры, с которых посещают вредоносные страницы. Пользователь попадает на скомпрометированный сайт, после чего происходит перенаправление на сервер распространения вредоносных программ. Результатом атаки является загрузка и запуск вредоносной программы (например, троянца или руткита) на компьютер жертвы. Такой тип атаки называется drive-by-download, ее схема изображена на рис. 1.


**Рис. 1.** Схема drive-by-download атаки

Одной из подзадач настоящего исследования является изучение известных вредоносных интернет-страниц и выявления набора признаков, необходимого и достаточного для автоматической классификации. На данном этапе рассматриваются страницы двух классов – нормальные (не содержащие атакующих элементов) и вредоносные.

## СОВРЕМЕННОЕ СОСТОЯНИЕ ПРОБЛЕМЫ

Современные подходы к обнаружению вредоносных страниц, как и других типов вредоносных программ, основаны на поиске сигнатур известных угроз. Так, например, антивирусные сканеры, как правило, регулярно обновляют свои базы, пополняя их все новыми сигнатурами. Данный подход обладает существенными недостатками:

- простота обхода сигнатуры;
- необходимость в регулярном и довольно частом обновлении;
- малая эффективность обнаружения новых типов атак и вредоносных программ.

В дополнение к сигнатурам в современных антивирусных сканерах представлены эвристические алгоритмы. Они способны обнаруживать подозрительные элементы в коде и, таким образом, обнаруживать некоторые новые типы атак, однако дают большое количество ложных срабатываний.

В последнее время набирает популярность подход программ на основе динамического анализа – использование так называемых песочниц. Применительно к проблеме обнаружения drive-by-download атак, песочница может быть реализована в виде виртуальной машины, с установленной на ней полноценной операционной системой (например, Windows XP). В виртуальной среде запускается браузер, который посещает проверяемую страницу. Через некоторое время происходит проверка системы на наличие новых процессов, файлов и ключей реестра. Если таковые обнаруживаются, то это расценивается как факт атаки с проверяемой страницы. Такой подход является весьма эффективным, однако и он обладает рядом недостатков:

- проверка миллионов и десятков миллионов страниц таким способом является затратной по ресурсам и по времени;
- существуют так называемые нерезидентные вредоносные программы, которые не создают новых файлов и процессов, а лишь выполняют ряд вызовов в контексте браузера и самоуничтожаются;
- существуют атаки, ориентированные на определенный браузер, поэтому при таком под-

ходе процесс анализа одной страницы требует ее посещения с нескольких браузеров.

В связи с вышеизложенными проблемами возникает необходимость исследования потенциала алгоритмов машинного обучения в решении задач компьютерной безопасности. В данной работе поднимается проблема статического детектирования вредоносных страниц, т. е. детектирования без эмуляции или сложных вычислений (например, построения абстрактных синтаксических деревьев и т. п.). Преимущество статического анализа состоит в скорости работы, безопасности (поскольку сценарий не выполняется), простоте и масштабируемости.

## ОБЗОР ПУБЛИКАЦИЙ ПО ТЕМЕ ИССЛЕДОВАНИЯ

Рассмотрим некоторые работы, посвященные статическому обнаружению вредоносных интернет-страниц. В [2] авторы уделяют особое внимание опасным функциям JavaScript, таким как eval и document.write. Они анализируют аргументы этих функций, выделяя три параметра: частотное распределение символов, энтропия Шеннона и длина слова. Проанализировав несколько экземпляров вредоносных страниц, авторы [2] делают вывод, что страница является атакующей, если:

- сценарий JavaScript содержит специальные символы;
- энтропия сценария менее 1,2;
- максимальная длина слова в сценарии боле 350 знаков.

Мы также использовали энтропию и частоты некоторых символов, однако наш анализ опровергает тезис 2.

В [3] авторами был предложен набор метрик для обнаружения факта обфускации сценария JavaScript:

- длина сценария в символах;
- среднее число символов в строке;
- количество строк в сценарии;
- количество значений строкового типа;
- число символов в кодировке Unicode;
- количество чисел, представленных в восьмиричной или шестнадцатиричной системах счисления;
- читаемость сценария (сценарий является читаемым, если в нем более 70% символов латинского алфавита, число гласных более 20% и менее 60%, слова в среднем не более 15 символов в длину, а также число повторений одного символа в строке не превышает 2);
- процент символов пробела от общего числа символов;
- количество вызываемых методов;
- количество комментариев в сценарии;
- отношение количества комментариев к общему числу строк;
- количество слов в сценарии;
- процент слов, не являющихся комментариями.

Для детектирования вредоносных сценариев были использованы несколько алгоритмов классификации.

В данной работе, в отличие от перечисленных выше, признаки выбираются исходя из анализа конкретных экземпляров drive-by-download атак. Кроме того, мы предлагаем использовать две категории признаков: признаки сценария JavaScript и признаки страницы в целом (включая HTML). Последним не уделяется внимания ни в одной работе.

Существует также ряд работ [4–6], в которых используются динамические свойства вредоносных сценариев JavaScript или контекст их исполнения. Во всех этих работах в качестве алгоритма обучения выступает наивный байесовский классификатор. Мы не рассматриваем их более подробно, поскольку в своей работе ограничиваемся статическим анализом сценариев.

## НАБОР ДАННЫХ И ПРИЗНАКИ ДЛЯ ИХ КЛАССИФИКАЦИИ

Набор исследуемых вредоносных страниц был получен из следующих источников:

- сайт malwaredomainlist.com – ресурс, предоставляющий список серверов распространения вредоносных программ;
- ресурс VX Heavens (vx.netlux.org), на котором хранится более 300 JavaScript эксплойтов разных лет;
- средство тестирования на проникновения Metaspoit (metasploit.org), откуда было взято более 80 эксплойтов для браузеров и их расширений.

Из более чем 500 экземпляров страниц было отобрано 310 наиболее подходящих для исследования. Критерии отбора были следующими:

- экземпляр должен представлять собой HTML страницу, отдельные JavaScript файлы на данном этапе не рассматриваются;

- вредоносным элементом должен быть JavaScript (а не Visual Basic, как в некоторых случаях);
- атака должна эксплуатировать уязвимость, связанную с ошибкой памяти, а не с логикой работы программы.

На основе полученных страниц был создан набор признаков, по которым можно автоматически детектировать атаки.

Признаки, описывающие сценарий JavaScript.

1. Размер в байтах. В рассматриваемой выборке вредоносные сценарии JavaScript, как правило, больше по размеру, чем нормальные. Это обусловлено тем, что они чаще всего состоят из обфусцированных данных (в виде массива или строки) и алгоритма деобфускации. Данные, как правило, «разбавлены» случайными последовательностями символов для усложнения анализа.

2. Энтропия. Информационная энтропия обфусцированных сценариев, как правило, выше, поскольку происходит преобразование осмысленных конструкций языка в псевдослучайную последовательность байт. Этот признак играет важную роль в определении факта обфускации сценария. Следует отметить, что не все обфусцированные сценарии являются вредоносными, но почти все вредоносные сценарии обфусцированы.

3. Частота пробелов. Наличие большого количества пробелов в сценарии говорит о применении примитивной техники обфускации, которая, хотя и редко встречается в поздних экземплярах вредоносных страниц, но все же должна быть учтена.

4. Частота использования специальных символов (%, ‘, “,\ ,/, ., ,). В результате анализа частотного распределения символов в выборке обнаружилось, что высокие частоты перечисленных знаков характерны для вредоносных сценариев.

5. Частота появления цифр. Многие вредоносные сценарии имеют обфусцированную полезную нагрузку в виде массива целых чисел.

6. Количество шестнадцатиричных значений. Часто полезная нагрузка эксплойта представлена в виде шестнадцатиричной последовательности, закодированной различными способами.

7. Количество вызовов функций работы со строками. Алгоритм деобфускации включает в себя различные преобразования строковых данных. Большое количество строковых функций свидетельствует о том, что сценарий обфусцирован.

8. Количество вызовов функции «eval». Данная функция встречается и в нормальных сценариях, но частое ее использования (3 и более раз в одном сценарии) является признаком атаки.

9. Количество вызовов функции «write». Вредоносные сценарии часто используют данный метод DOM модели для помещения на страницу фреймов, для перенаправления или создания flash объектов и т. п.

10. Количество вызовов функции «unescape» – данная функция декодирует escape последовательности. Часто полезная нагрузка эксплойта декодируется именно таким образом.

Признаки, описывающие HTML-страницу:

1. Наличие описателя <!doctype> (0|1).
2. Наличие тега HTML (0|1).
3. Наличие тега HEAD (0|1).
4. Наличие тега BODY(0|1).
5. Наличие хотя бы одного тега META (0|1).
6. Количество ссылок типа <a href=…>

Вторая группа признаков была выбрана исходя из наблюдения, что атакующие страницы, как правило, создаются «на скорую руку». Они не содержат метатегов и других спецификаций, не содержат ссылок или содержат очень мало, поскольку их основная задача атаковать, а не имитировать реальный сайт.

Для создания классификатора кроме вредоносных страниц (которые представляют собой положительные примеры), нужны нормальные страницы. Порядка 5000 нормальных страниц было получено следующим образом:

- автоматизированная загрузка 10 тыс. страниц рейтинга Alexa.com Top 1’000’000;
- удаление страниц, не содержащих теги <script>;
- удаление страниц, поврежденных при загрузке.

Полученные страницы представляют собой отрицательные примеры для классификации.

## РАЗРАБОТКА КЛАССИФИКАТОРА

В общем случае алгоритм машинного обучения в оперативном режиме вычисляет значение гипотезы $h$, которое интерпретируется как вероятность отнесения данного экземпляра к положительному классу. Функция $h$ в свою

очередь обладает набором параметров, которые «подстраиваются» в результате обучения.

Чтобы понять, какой алгоритм обучения больше подходит для классификации собранных данных, необходимо провести анализ этих данных в уже преобразованном виде (т.е. когда каждая страница представлена вектором признаков).

Простейшим из алгоритмов классификации является логистическая регрессия. Данный алгоритм вычисляет гипотезу по формуле

$$h_\theta(x) = g(\theta^T x), \quad (1)$$

где g(•) – сигмоидальная сжимающая функция, θ – вектор параметров гипотезы. В простейшем случае логистический классификатор способен успешно решать задачи линейного разделения признаков. Однако собранные данные носят нелинейный характер. Об этом можно судить исходя из результатов эксперимента с логистическим классификатором на исследуемом наборе данных. После обучения классификатора (с использованием алгоритма градиентного спуска) значение гипотезы во всех примерах незначительно варьировалось вокруг 0,005. Есть два способа решения проблемы неэффективного логистического классификатора:

- добавление новых признаков в виде полиномиальных термов от уже существующих, что ведет к экспоненциальному росту размера обучающей выборки;
- использование другого алгоритма обучения.

Одной из технологий, успешно решающей задачи нелинейной классификации, является технология искусственных нейронных сетей. Существует большое количество архитектур и параметров нейронных сетей, однако в данной работе используется классическая нейронная сеть прямого распространения.

Используемая нейронная сеть состоит из нейронов, вычисляющих логистическую функцию по формуле (2) (по сути это формула (1), только здесь раскрыта функция $g(\bullet)$):

$$a_\theta(x) = \frac{1}{1+e^{-\theta^T x}}. \quad (2)$$

Для каждого нейрона определен свой вектор параметров θ (или вектор весовых коэффициентов). Нейронная сеть, соответствующая имеющемуся набору данных, имеет архитектуру, представленную на рис. 2.

Данная нейронная сеть имеет 16 нейронов входного слоя, 20 нейронов скрытого слоя, один выходной нейрон; на вход подаются нормализованные вектора признаков страниц из обучающей выборки.

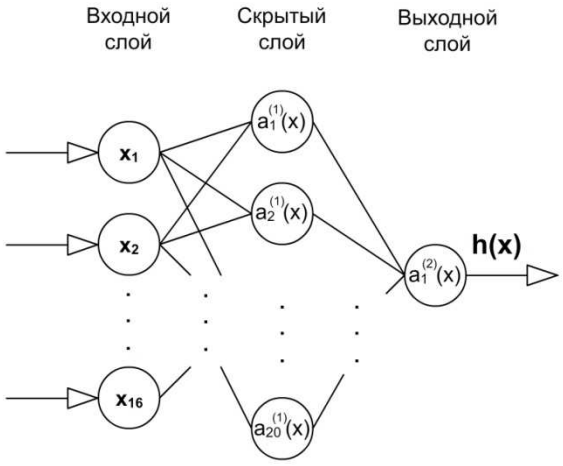


**Рис. 2.** Нейронная сеть для классификации интернет-страниц

Поскольку размер обучающей выборки сравнительно небольшой, для обучения сети предлагается использовать алгоритм Левенберга-Маркуордта. Условием останова алгоритма обучения является значение среднеквадратической ошибки равное 0,001.

## ЭКСПЕРИМЕНТЫ

Прежде чем приступить к описанию экспериментов, необходимо определить показатели эффективности нейросетевого классификатора. В данной работе используется подход к оценке эффективности, описанный в [7]. Введем несколько обозначений:

- событие *A* описывает срабатывание детектора, т. е. подача сигнала тревоги;
- событие *I* описывает появление вредоносной интернет-страницы во множестве анализируемых страниц;
- $N_{TP}$ – количество корректно распознанных вредоносных страниц в ходе эксперимента;
- $N_{FP}$ – количество ложных срабатываний в ходе эксперимента;
- $N_{FN}$ – количество некорректно распознанных вредоносных страниц в ходе эксперимента;
- $N_{TN}$ – количество корректно распознанных нормальных страниц в ходе эксперимента.

Необходимые значения эффективности определяются формулами:

$$TP = \Pr[A \mid \mathrm{I}] = \frac{N_{TP}}{N_{TP} + N_{FN}}; \quad (3)$$

$$FP = \Pr[A \mid \neg\, \mathrm{I}] = \frac{N_{FP}}{N_{FP} + N_{TN}}; \quad (4)$$

$$FN = \Pr[\neg A \mid \mathrm{I}] = \frac{N_{FN}}{N_{FN} + N_{TP}}; \quad (5)$$

$$TN = \Pr[\neg A \mid \neg\, \mathrm{I}] = \frac{N_{TN}}{N_{TN} + N_{FP}}. \quad (6)$$

Особое значение имеют показатели $TP$ – уровень корректно распознанных вредоносных страниц и $FP$ – уровень ложных срабатываний.

Эксперимент проводился с применением техники $k$-ступенчатой кросс-валидации с параметром $k$ = 10. При таком подходе обучающая выборка делится на 10 частей, после чего классификатор обучается на 9 частях и тестируется на 10-й (см. рис 3).


**Рис. 3.** Организация обучающей выборки и данных тестирования

Для получения более достоверных с точки зрения статистики результатов, эксперимент с каждым набором данных был проведен 1000 раз, и в качестве окончательного результата вычислялось среднее арифметическое значение. Поскольку гипотеза представляет собой лишь вероятность, с которой экземпляр может быть отнесен к вредоносным или нормальным страницам, важно определить пороговое значение ε, при котором классификатор детектирует атакующую страницу. В данном случае определение порогового значения производилось экспериментально. Были определены уровни корректного обнаружения и ложных срабатываний для диапазона пороговых значений [0,1, 0,9] с шагом 0,1. Результаты экспериментов представлены в виде графиков на рис. 4.

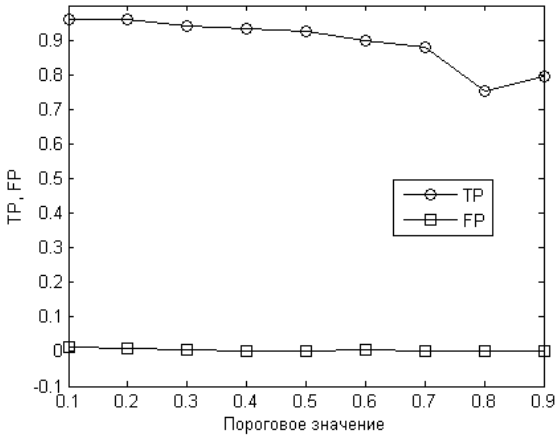


**Рис. 4.** Эффективность классификации при различных значениях ε

Из графика видно, что вероятность ложного срабатывания при всех значениях ε гораздо ниже 0,1. При ε = 0,1, FP = 0,0147, при ε = 0,9, FP = 0,0008. При этом корректность обнаружения TP оказалась самой высокой при ε = 0,1, она составила 97%.

Результаты всех этапов кросс-валидации и их средние значения TP и FP (обозначено как μ) при ε = 0,1 приведены в табл. 1.

Таблица 1

**Кросс-валидация при ε = 0,1**

| № | TP | FP |
|---|---|---|
| 1 | 0,9935 | 0,0204 |
| 2 | 0,9968 | 0,0358 |
| 3 | 0,9484 | 0,0133 |
| 4 | 0,9871 | 0,0125 |
| 5 | 0,9484 | 0,0282 |
| 6 | 1,0000 | 0,0307 |
| 7 | 1,0000 | 0,0313 |
| 8 | 0,9677 | 0,0170 |
| 9 | 0,9065 | 0,0237 |
| 10 | 0,9710 | 0,0110 |
| μ | 0,9719 | 0,0224 |

## ВЫВОДЫ И ПРОДОЛЖЕНИЕ ИССЛЕДОВАНИЯ

Высокая эффективность предложенного классификатора приводит нас к выводу, что современные вредоносные страницы достаточно просто отличить друг от друга. Действительно, для аналитика вредоносных программ, достаточно одного взгляда, чтобы понять, что перед ним атакующий код. В данной работе проведена

попытка выразить в виде набора числовых характеристик те элементы вредоносных страниц, которые визуально отличают их от нормальных. Ситуация может измениться, если атакующие страницы будут имитировать нормальные сайты, дабы усложнить автоматическое детектирование. В связи с этим направления для дальнейшего исследования проблемы могут быть обозначены следующим образом:

- создание абстрактной модели атакующих страниц с целью прогнозирования формы и содержания более сложных экземпляров;
- исследование применения криптографии в сокрытии вредоносного кода на атакующих страницах;
- определение границ детектируемости вредоносных объектов в интернете.

Разработанный на данном этапе исследования классификатор будет использоваться для мониторинга интернет-сайтов и сбора статистики зараженных страниц.

## СПИСОК ЛИТЕРАТУРЫ

1. **Наместников Ю.** Развитие интернет угроз в первом квартале 2012 г. [Электронный ресурс] (http://www.securelist.com/ru/analysis/208050757/Razvitie_informatsionnykh_ugroz_v_pervom_kvartale_2012_goda).

2. **Younghan Ch.**, **Taeghyoon K.**, **Seokjin Ch.** Automatic detection for java script obfuscation attacks in web pages through string pattern analysis // Proceedings of the International Conference on Future Generation Information Technology. 2009. P. 160–172.

3. **Likarish P., Eunjin J., Insoon J.** Obfuscated malicious javascript detection using classification techniques // Proceedings of the 4th International Conference on the Malicious and Unwanted Software (MALWARE). 2009. P. 47–54.

4. **Cova M., Kruegel C., Vigna C.** Detection and analysis of drive-by-download attacks and malicious JavaScript code // Proceedings of the 19th International Conference on World Wide Web. 2010. P. 281–290.

5. Rozzle: De-Cloaking Internet Malware / C. Kolbitsch [et al.]. [Электронный ресурс] (http://research.microsoft.com/apps/pubs/default.aspx?id=152601).

6. Zozzle: low-overhead mostly static JavaScript malware detection / Ch. Curtsinger [et al.] [Электронный ресурс] (http://research.microsoft.com/apps/pubs/?id=141930).

7. **Hammersland R.** Roc in Assessing IDS Quality [Электронный ресурс] (http://rune.hammersland.net/tekst/roc.pdf).

## ОБ АВТОРЕ

**Котов Вадим Дмитриевич,** асп. каф. вычислительн. техники и защиты информации. Дипл. спец. по защите информации (УГАТУ, 2010). Иссл. в обл. информ. безопасности и интеллектуальных систем.