НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
ноябрь–декабрь 2018 Том 18 № 6 ISSN 2226-1494 http://ntv.ifmo.ru/
SCIENTIFIC AND TECHNICAL JOURNAL OF INFORMATION TECHNOLOGIES, MECHANICS AND OPTICS
November–December 2018 Vol. 18 No 6 ISSN 2226-1494 http://ntv.ifmo.ru/en

УДК 004.02; 004.85

АЛГОРИТМ ГРАДИЕНТНОГО БУСТИНГА ДЕРЕВЬЕВ РЕШЕНИЙ В ЗАДАЧЕ ИДЕНТИФИКАЦИИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

К.И. Салахутдинова[a], И.С. Лебедев[b], И.Е. Кривцова[a]

[a] Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация
[b] Санкт-Петербургский институт информатики и автоматизации РАН (СПИИРАН), Санкт-Петербург, 199178, Российская Федерация
Адрес для переписки: kainagr@mail.ru

Информация о статье
Поступила в редакцию 18.06.18, принята к печати 22.09.18
doi: 10.17586/2226-1494-2018-18-6-1016-1022
Язык статьи – русский

Ссылка для цитирования: Салахутдинова К.И., Лебедев И.С., Кривцова И.Е. Алгоритм градиентного бустинга деревьев решений в задаче идентификации программного обеспечения // Научно-технический вестник информационных технологий, механики и оптики. 2018. Т. 18. № 6. С. 1016–1022. doi: 10.17586/2226-1494-2018-18-6-1016-1022

Аннотация
Предложен подход к идентификации версий программного обеспечения на основе алгоритма градиентного бустинга деревьев решений. Предложено применять алгоритм CatBoost, разработанный компанией Яндекс, для решения задачи идентификации программного обеспечения операционных систем Linux с целью уменьшения числа уязвимостей системы, возникающих при установке несанкционированного программного обеспечения пользователями автоматизированных систем. Рассмотрен подход к формированию сигнатур программ и дальнейшему обучению модели классификатора CatBoostClassifier. Поставлена задача последующего распознавания идентифицируемых программ, ранее не задействованных в процессе обучения модели. **Метод.** Для реализации алгоритма градиентного бустинга деревьев решений использовано свободное программное обеспечение CatBoost. На его основе создана мультиклассификационная модель CatBoostClassifier. Применение этой модели позволяет идентифицировать elf-файлы тестовой выборки. **Основные результаты.** Выбраны параметры обучения модели классификации. Проведен эксперимент по идентификации исполняемых файлов с использованием десяти различных признаков формирования сигнатур программ. Полученные результаты сравниваются с результатами ранее разработанного авторами метода идентификации, основанного на применении статистического критерия однородности хи-квадрат при уровне значимости 0,01. **Практическая значимость.** Результаты работы могут быть рекомендованы специалистам по информационной безопасности для проведения аудита электронных носителей информации. Разработанный подход позволяет выявить нарушения установленной политики безопасности при обработке конфиденциальной информации.

Ключевые слова
машинное обучение, градиентный бустинг деревьев решений, CatBoost, идентификация исполняемых файлов, elf-файлы, информационная безопасность

Благодарности
Работа выполнена по теме № 0073-2018-0008.

GRADIENT BOOSTING TREES METHOD IN THE TASK OF SOFTWARE IDENTIFICATION

K.I. Salakhutdinova[a], I.S. Lebedev[b], I.E. Krivtsova[a]

[a]ITMO University, Saint Petersburg, 197101, Russian Federation
[b]Saint Petersburg Institute for Informatics and Automation RAS (SPIIRAS), 199178, Russian Federation
Corresponding author: kainagr@mail.ru

Article info
Received 18.06.18, accepted 22.09.18
doi: 10.17586/2226-1494-2018-18-6-1016-1022
Article in Russian

For citation: Salakhutdinova K.I., Lebedev I.S., Krivtsova I.E. Gradient boosting trees method in the task of software identification. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2018, vol. 18, no. 6, pp. 1016–1022 (in Russian). doi: 10.17586/2226-1494-2018-18-6-1016-1022

Abstract
Subject of Research. The paper proposes an approach to the use of gradient boosted decision trees algorithm. For this

purpose, CatBoost algorithm developed by Yandex is proposed. Its implementation is aimed at the problem solution of OS Linux software identification in order to reduce the number of system vulnerabilities, which occur due to the installation of unauthorized software by automated system users. We consider an approach to the program signatures formation and further training of CatBoostClassifier classifier model. The subsequent recognition task is set for the identified programs that were not previously involved in the model training process. **Method.** Free CatBoost software was used for implementation of the gradient boosted decision trees algorithm. CatBoostClassifier multi-classification model was created on its basis. The use of this model allows identifying test sample elf-files. **Main Results.** The training parameters of the classification model are selected. An experiment is carried out to identify elf-files with the use of ten different features of emerging signature programs. The results obtained in the new approach are compared with the results of the previously developed method of identification based on the application of the statistical criterion of Chi-square homogeneity at the significance level $p = 0.01$. **Practical Relevance.** The results of the study can be recommended to information security specialists for data media audit. The developed approach gives the possibility to identify violations of the established security policy in the processing of confidential information.

Keywords
machine learning, gradient boosting trees, CatBoost, executable files identification, elf-files, information security

Acknowledgements
Work have been conducted with theme № 0073-2018-0008.

Введение

Существует множество областей применения машинного обучения, например, классификация семейств вредоносных программ на основе их поведения [1], обнаружение вредоносных программ [2] или вторжения в компьютерные сети [3], аутентификация пользователя на основе характеристик его поведения [4].

Алгоритмы машинного обучения обеспечивают не прямое решение задач, а их обобщение и изучение в ходе вычислений [5]. Так, в настоящее время набирает все большую популярность метод градиентного бустинга деревьев решений (gradient boosting trees), основанный на решающих деревьях. Он становится в один ряд с такими методами машинного обучения, как построение искусственных нейронных сетей и линейных моделей.

В настоящей работе авторы рассматривают особенности использования алгоритма градиентного бустинга при решении задачи идентификации программного обеспечения (ПО), т.е. отождествления того или иного исполняемого файла с некоторой известной программой.

Действия пользователей автоматизированных систем по несанкционированной установке программного обеспечения могут привести к росту числа уязвимостей системы и повлиять на ее информационную безопасность. Причиной этого является возможное нелегальное использование интеллектуальной собственности, наличие не декларированных возможностей, дефекты ПО, а также намеренное использование специальных программ, направленных на преодоление установленной защиты, либо противоправных действий внутри сети Интранет или Интернет.

Ранее авторами эта задача уже решалась с использованием методов математической статистики и таких критериев, как угловое преобразование Фишера [6], однородности хи-квадрат и Колмогорова–Смирнова [7].

Процесс идентификации включал сравнение двух сигнатур: унифицированной – созданной на основе обучающей выборки и сигнатуры идентифицируемой программы, создаваемой непосредственно перед этапом сравнения. При создании унифицированной сигнатуры достигалась гибкость подхода, которая позволяла успешно идентифицировать различные версии одной и той же программы при помощи статистических критериев, не использованных в ее создании.

Для идентифицируемой сигнатуры формулировалось решение на заданном уровне значимости принять или отклонить основную гипотезу о схожести двух распределений (сигнатур). При этом возникало достаточно большое число ошибок второго рода, заключающихся в принятии основной гипотезы, в то время как она не верна.

При градиентном бустинге деревьев решений используется такая задача машинного обучения, как классификация, в которой выходными данными для идентифицируемой сигнатуры является наиболее вероятная принадлежность к одному из классов (известной программе).

Объектом исследования в настоящей работе является идентификация программного обеспечения формата elf в Linux системах, предметом – применение метода градиентного бустинга CatBoost в задаче идентификации, а целью – подбор таких параметров градиентного бустинга, при которых будет повышаться количество корректных результатов идентификации ПО. Следует отметить, что авторами рассматривается только не вредоносное ПО, процесс идентификации которого крайне мало рассмотрен в современных научных работах [8, 9].

Метод решающих деревьев и градиентный бустинг

Использование деревьев решений позволяет представлять правила в иерархической структуре. Они содержат вершины, в которые записываются проверяемые условия, и листья, в которые

записываются конечные значения, например, один из классов при решении задачи классификации [10].

Бустинг является общим методом для повышения производительности любого алгоритма машинного обучения. Суть его заключается в обучении каждой последующей модели с использованием данных об ошибках предыдущих моделей и дальнейшем снижении ошибок. Данный метод теоретически можно использовать для любого слабого алгоритма в целях снижения ошибки обучения [11]. Градиентный бустинг деревьев решений позволяет строить аддитивную функцию в виде суммы деревьев решений итерационно, по аналогии с методом градиентного спуска [12]. Так, на рис. 1 представлены ансамбль из z деревьев и принцип минимизации ошибки Er_z.

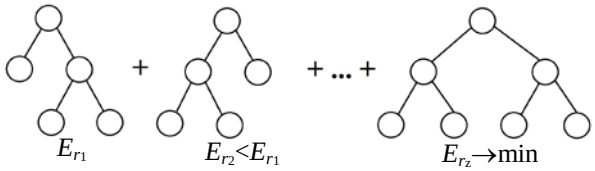

Рис. 1. Минимизация ошибки обучения
при использовании алгоритма градиентного бустинга деревьев решений

Как известно, представленный командой Яндекса в 2017 г. CatBoost – алгоритм машинного обучения, использующий градиентный бустинг, основанный на деревьях решений. Данная разработка является библиотекой с открытым исходным кодом и поддерживает работу из Python, R и командной строки [13]. Особенностью алгоритма является построение симметричных деревьев, возможность работы с категориальными признаками, кроме того, он позволяет обучаться на относительно небольшом количестве неоднородных данных. CatBoost способен решать такие задачи машинного обучения, как регрессия, классификация, мультиклассификация и ранжирование.

Представленные Яндексом результаты теста на производительность [13], проведенного с использованием широко распространенных пакетов данных, взятых с репозиториев UCI, Kaggle, KDD, свидетельствуют о конкурентоспособности предлагаемой реализации алгоритма градиентного бустинга CatBoost среди других реализаций (LightGBM, XGBoost, H2O).

Идентификация программ

В работах [14, 15] описан подход к формированию сигнатур на основе дизассемблированного кода программы и рассмотрено влияние различных ассемблерных команд на результаты идентификации. Таким образом, для настоящей работы в качестве признаков взяты команды: add, and, call, cmp, je, jmp, lea, mov, pop, push.

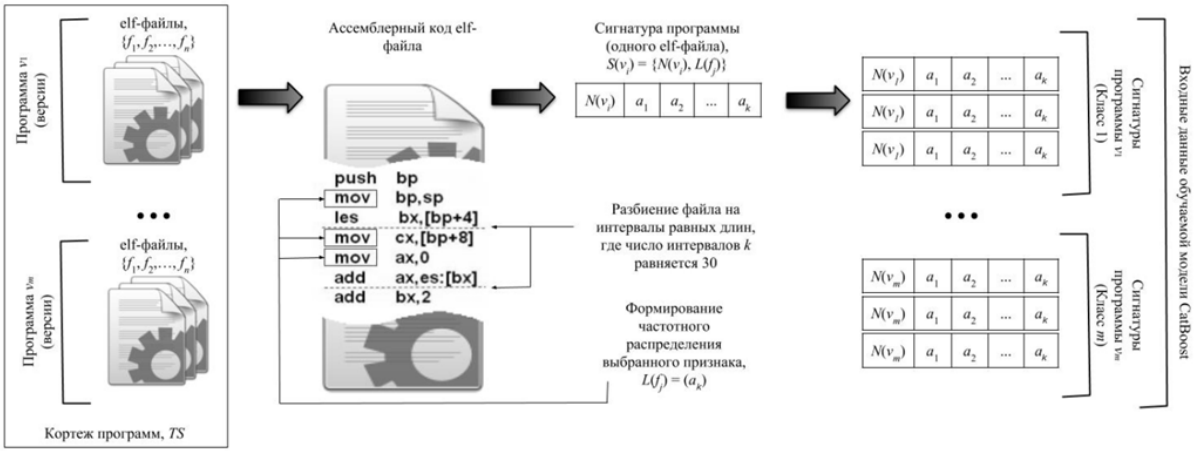

Рис. 2. Процесс создания сигнатур версий программ
и формирование входных данных обучаемой модели CatBoost

Перед тем как использовать CatBoost в качестве средства идентификации программ, необходимо создать модель на основе обучающей выборки, которая в дальнейшем будет решать, к какому классу (программе) относится идентифицируемый файл. Для этого необходимо создать архив сигнатур, формируемых следующим образом.

– *Сбор программ.* Различные программы и их версии формируются в кортеж $TS = \{v_1, v_2, \ldots, v_m\}$, $i = 1, \ldots, m$, где v_i – выборка различных программ; m – число различных программ; $v_i = \{f_1, f_2, \ldots, f_n\}$, f_j – различные версии i-й программы, n – число файлов в выборке.

– *Выделение характеристики выбранного признака и формирование сигнатур программ.* Каждый файл f_j разбивается на интервалы равной длины, при этом вводится фиксированный множитель для формирования длины шага, корректирующий число получаемых интервалов для файлов разного объема, где за длину интервала принимается количество различных ассемблерных команд на промежутке одного шага. В эксперименте коэффициент подобран таким образом, чтобы число интервалов k равнялось тридцати. Частотное распределение признака для файла f_j записывается как $L(f_j) = (a_k)$, где $j = 1,\ldots, n$.

Таким образом, сигнатура программы примет вид:

$S(v_i) = \{N(v_i), L(f_j)\}$,

где $N(v_i)$ – имя программы v_i. Соответственно сигнатуру идентифицируемого файла можно представить в следующем виде:

$S = \{L\}$,

где $L = (a_k)$ – частотное распределение признака.

В дальнейшем из сигнатур программ $S(v_i)$ формируется массив входных данных обучаемой модели CatBoost (рис. 2), а сигнатуры идентифицируемых файлов S образуют тестовую выборку.

В качестве примера использовано решение задачи классификации при помощи CatBoostClassifier, для которого число классов равно числу различных программ v_i в кортеже TS.

Постановка эксперимента и подбор параметров обучения CatBoostClassifier

Эксперимент проведен на тех же выборках данных, что и в работе [14]. Обучающая выборка состояла из 443 исполняемых файлов операционной системы (ОС) Linux различных версий и разрядностей (32х и 64х), относящихся к 63 различным программам. В тестовую выборку входило 123 файла, относящихся к тем же 63 программам, все они отличались от файлов, используемых в обучающей выборке, и имели разрядность 32х и 64х.

В качестве изменяемых параметров для решения задачи классификации при помощи CatBoostClassifier были выбраны:

– iterations – максимальное число деревьев, которое будет построено при решении задачи машинного обучения;
– learning_rate – скорость обучения, используемая для уменьшения шага градиентного спуска;
– l2_leaf_reg – коэффициент регуляризации $L2$, используемый для расчета значения листов;
– depth – глубина дерева;
– loss_function – метрика (функция потерь), используемая в обучении.

В таблице представлены установленные по умолчанию значения, а также подобранные эмпирическим путем значения выбранных параметров для решения задачи идентификации.

Параметры обучения	Значения по умолчанию	Пользовательские значения
iterations	1000	1000
learning_rate	0,03	0,7
l2_leaf_reg	3	1
depth	6	2
loss_function	Logloss	MultiClass

Параметры обучения CatBoostClassifier

Метрика MultiClass, заданная в параметре loss_function для задачи машинного обучения, связанной с мультиклассификацией, представлена следующим образом:

$$loss\_function = \frac{\sum_{i=1}^{N} w_i \log\left(\frac{e^{a_{it_i}}}{\sum_{i=0}^{M-1} e^{a_{ij}}}\right)}{\sum_{i=1}^{N} w_i}, \quad t \in \{0,\ldots,M-1\},$$

где t_i – метка класса для i-го объекта (из входных данных обучающей выборки); a_i – значение целевой функции для i-го объекта; N – общее число объектов; w_i – вес i-го объекта, который задается в описании набора данных в соответствующих столбцах (если не указано иное) или в параметре sample_weight пакета Python, где значение по умолчанию равно 1 для всех объектов; M – число классов.

Метрика Logloss не предназначена для работы с задачей мультиклассификации.

Результаты

На рис. 3 представлены усредненные по 10 ассемблерным командам графики кривых ошибок для

значений параметров CatBoostClassifier, установленных по умолчанию и с применением пользовательских настроек. По оси абсцисс отмечено число итераций (построенных деревьев решений), по оси ординат – значение ошибки обучения для данной итерации. Как видно из рисунка, для модели классификации, построенной с установленными по умолчанию параметрами, значение ошибки обучения уменьшается медленнее, чем для модели, построенной на пользовательских значениях параметров.

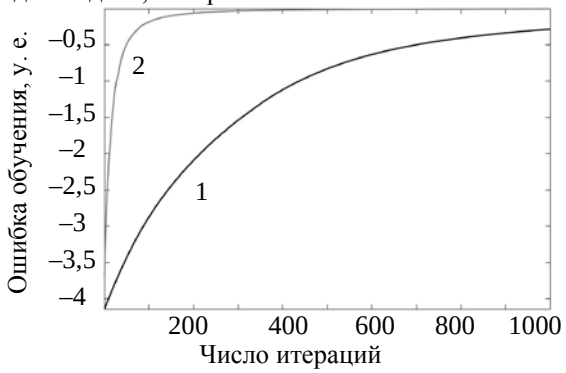

Рис. 3. Усредненные кривые ошибок
(1 – для параметров по умолчанию, 2 – для пользовательских параметров)

Результаты идентификации тестовой выборки по 10 ассемблерным командам приведены на рис. 4. Очевидно, что для всех ассемблерных команд число верно идентифицированных программ выше при установлении пользовательских параметров и достигает максимального значения в 89 % для команды je (110 правильно идентифицированных исполняемых файла из 123).

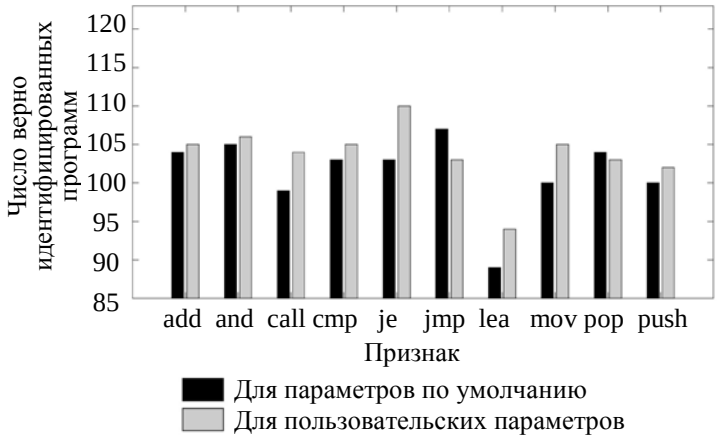

Рис. 4. Число верно идентифицированных исполняемых файлов с различными значениями параметров обучения

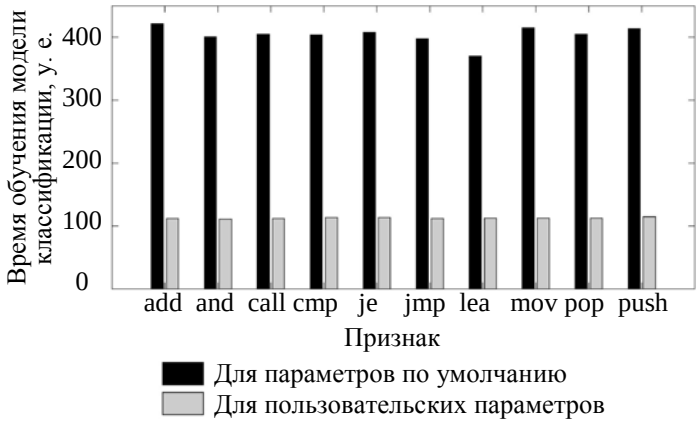

Рис. 5. Время обучения модели классификации

Также из рис. 5 видно значительное преимущество во времени, затрачиваемого на обучение модели классификации с заданными пользовательскими параметрами. Так, путем уменьшения глубины строящихся деревьев достигается повышение скорости обучения примерно в четыре раза при отсутствии потери в качестве классификации.

Результаты идентификации тестовой выборки, проведенной с помощью статистического критерия однородности хи-квадрат на уровне значимости $p = 0{,}01$ и с использованием градиентного бустинга деревьев решений CatBoost, представлены на рис. 6. Из приведенных данных следует, что в среднем для подхода с использованием статистического критерия число верно идентифицированных исполняемых файлов тестовой выборки не превышает 62 %, для подхода с CatBoostClassifier оно достигает 84 %.

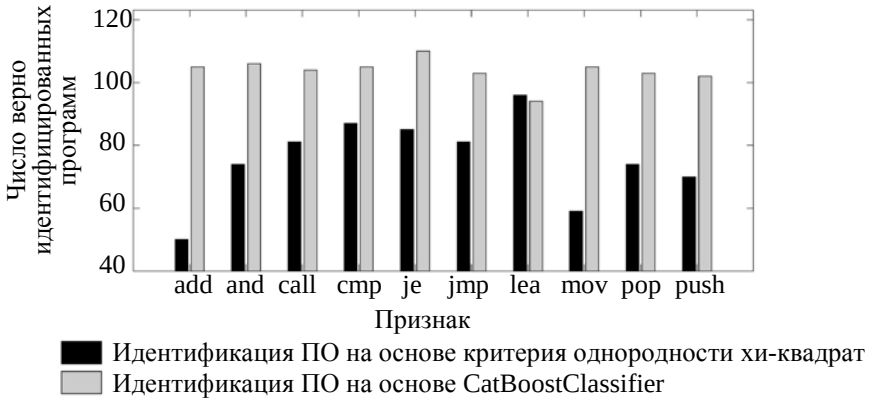

Рис. 6. Количество верно идентифицированных исполняемых файлов по критерию хи-квадрат и с использованием CatBoostClassifier

Сравним вычислительную сложность двух методов, проанализировав некоторые этапы построения сигнатур. Статистический подход с использованием критерия однородности хи-квадрат, в отличие от использования CatBoost, дополняется этапом создания унифицированной сигнатуры [14], происходящим за счет кластеризации, а именно метода *k*-средних. Таким образом, при создании архива сигнатур вычислительная сложность создания унифицированной сигнатуры составляет $O(nkl)$, где n – число объектов, k – число кластеров, l – число итераций. Метод на основе градиентного бустинга деревьев решений не требует создания унифицированных сигнатур, отсюда можно сказать, что вычислительная сложность данного этапа постоянна – $O(1)$.

Далее рассмотрим этап непосредственной идентификации исполняемых файлов. Для метода, основанного на статистическом критерии однородности хи-квадрат, вычислительная сложность попарного сравнения всех сигнатур тестовой выборки с сигнатурами в архиве составляет $O(n')$, где n' – число объектов в архиве сигнатур, получившихся после этапа кластеризации, очевидно, что $n \geq n' \geq m$ (m – число различных программ). Для нового подхода на основе градиентного бустинга деревьев решений – $O(n)$, n остается неизменным, так как этап унификации сигнатур не производился.

Отсюда следует, что для ранее рассматриваемого метода вычислительная сложность составляет – $O(nkl)+O(n')$, тогда как для нового – $O(1)+O(n)$.

Заключение

Для контроля пользователей автоматизированных систем, а также поддержания организационных мер по обеспечению информационной безопасности требуется разработка новых подходов к автоматизированной процедуре идентификации установленного программного обеспечения.

В работе продемонстрирована возможность применения алгоритма градиентного бустинга деревьев решений CatBoost от компании Яндекс для идентификации elf-файлов, а также подобраны параметры классификации.

Таким образом, предложенный подход к обеспечению информационной безопасности путем проведения идентификации программ позволяет достигать 89 % корректных результатов, повышение этого значения возможно при использовании нескольких ассемблерных команд.

Из результатов эксперимента очевидно преимущество нового подхода к задаче идентификации программного обеспечения операционной системы Linux по сравнению с основанным на применении статистического критерия. Недостатком является стохастичность процесса обучения модели, влияющая на корректность результатов идентификации, что обусловливает необходимость выполнения нескольких итераций обучения.

Литература

1. Pektas A., Acarman T. Classification of malware families based on runtime behaviors // Journal of Information Security and Applications. 2017. V. 37. P. 91–100. doi: 10.1016/j.jisa.2017.10.005
2. Nguyen M.H., Nguyen D.L., Nguyen X.M., Quan T.T.

References

1. Pektas A., Acarman T. Classification of malware families based on runtime behaviors. *Journal of Information Security and Applications*, 2017, vol. 37, pp. 91–100. doi: 10.1016/j.jisa.2017.10.005
2. Nguyen M.H., Nguyen D.L., Nguyen X.M., Quan T.T.

Auto-detection of sophisticated malware using lazy-binding control flow graph and deep learning // Computers & Security. 2018. V. 76. P. 128–155. doi: 10.1016/j.cose.2018.02.006
3. Chiba Z., Abghour N., Moussaid K., El Omri A., Rida M. A novel architecture combined with optimal parameters for back propagation neural networks applied to anomaly network intrusion detection // Computers & Security. 2018. V. 75. P. 36–58. doi: 10.1016/j.cose.2018.01.023
4. Горбунов И.В. Особенности использования нечеткого классификатора и алгоритмов машинного обучения для аутентификации по клавиатурному почерку // Электронные средства и системы управления. 2013. № 2. С. 13–18.
5. Gori M. Machine Learning: A Constraint-Based Approach. Morgan Kaufmann, 2017. 580 p.
6. Кривцова И.Е., Салахутдинова К.И., Юрин И.В. Метод идентификации исполняемых файлов по их сигнатурам // Вестник Государственного университета морского и речного флота имени адмирала С.О. Макарова. 2016. № 1(35). С. 215–224.
7. Krivtsova I.E., Lebedev I.S., Salakhutdinova K.I. Identification of executable files on the basis of statistical criteria // Proc. 20th Conference of Open Innovations Association. St. Petersburg, 2017. P. 202–208. doi: 10.23919/FRUCT.2017.8071312
8. Антонов А.Е., Федулов А.С. Идентификация типа файла на основе структурного анализа // Прикладная информатика. 2013. № 2(44). С. 68–77.
9. Казарин О.В. Теория и практика защиты программ. М.: МГУЛ, 2004. 450 с.
10. Кафтанников И.Л., Парасич А.В. Особенности применения деревьев решений в задачах классификации // Вестник ЮУрГУ. Серия: Компьютерные технологии, управление, радиоэлектроника. 2015. № 3(15). С. 26–32.
11. Freund, Y., Schapire R. Experiments with a new boosting algorithm // Proc. 13th Int. Conf. on Machine Learning. Bari, 1996. P. 148–156.
12. Дружков П.Н., Золотых Н.Ю., Половинкин А.Н. Реализация параллельного алгоритма предсказания в методе градиентного бустинга деревьев решений // Вестник ЮУрГУ. 2011. № 37(254). С. 82–89.
13. CatBoost GitHub [Электронный ресурс]. Режим доступа: https://github.com/catboost, свободный. Яз. англ. (дата обращения 29.04.2018).
14. Салахутдинова К.И., Лебедев И.С., Кривцова И.Е. Подход к выбору информативного признака в задаче идентификации программного обеспечения // Научно-технический вестник информационных технологий, механики и оптики. 2018. Т. 18. № 2. С. 278–285. doi: 10.17586/2226-1494-2018-18-2-278-285
15. Druzhinin N.K., Salakhutdinova K.I. Identification of executable file by dint of individual feature // Proc. Int. Conf. on Information Security and Protection of Information Technology, ISPIT-2015. St. Petersburg, Russia, 2015. P. 45–47.

Авторы

Салахутдинова Ксения Иркиновна – аспирант, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, Scopus ID: 57191362944, ORCID ID: 0000-0001-9254-8652, kainagr@mail.ru

Лебедев Илья Сергеевич – доктор технических наук, профессор, заведующий лабораторией, Санкт-Петербургский институт информатики и автоматизации РАН (СПИИРАН), Санкт-Петербург, 199178, Российская Федерация, Scopus ID: 56321781100, ORCID ID: 0000-0001-6753-2181, isl_box@mail.ru

Кривцова Ирина Евгеньевна – старший преподаватель, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, Scopus ID: 57190307529, ORCID ID: 0000-0003-2483-1637, ikr@cit.ifmo.ru

Auto-detection of sophisticated malware using lazy-binding control flow graph and deep learning. *Computers & Security*, 2018, vol. 76, pp. 128–155. doi: 10.1016/j.cose.2018.02.006
3. Chiba Z., Abghour N., Moussaid K., El Omri A., Rida M. A novel architecture combined with optimal parameters for back propagation neural networks applied to anomaly network intrusion detection. *Computers & Security*, 2018, vol. 75, pp. 36–58. doi: 10.1016/j.cose.2018.01.023
4. Gorbunov I.V. Features of a fuzzy classifier use and machine learning algorithms for authentication using keyboard handwriting. *Elektronnye Sredstva i Sistemy Upravleniya*, 2013, no. 2, pp. 13–18. (in Russian)
5. Gori M. *Machine Learning: A Constraint-Based Approach*. Morgan Kaufmann, 2017, 580 p.
6. Krivtsova I.E., Salakhutdinova K.I., Yurin I.V. Method of executable filts identification by their signatures. *Vestnik Gosudarstvennogo Universiteta Morskogo i Rechnogo Flota Imeni Admirala S.O. Makarova*, 2016, no. 1, pp. 215–224. (in Russian)
7. Krivtsova I.E., Lebedev I.S., Salakhutdinova K.I. Identification of executable files on the basis of statistical criteria. *Proc. 20th Conference of Open Innovations Association*. St. Petersburg, 2017, pp. 202–208. doi: 10.23919/FRUCT.2017.8071312
8. Antonov A.E., Fedulov A.S. File type identification based on structural analysis. *Journal of Applied Informatics*, 2013, no. 2, pp. 68–77. (in Russian)
9. Kazarin O.V. *Theory and Practice of Program Protection*. Moscow, MGUL Publ., 2004, 450 p. (in Russian)
10. Kaftannikov I.L., Parasich A.V. Decision tree's features of application in classification problem. *Bulletin SUSU, Computer Technologies, Automatic Control & Radioelectronics*, 2015, no. 3, pp. 26–32. (in Russian)
11. Freund, Y., Schapire R. Experiments with a new boosting algorithm. *Proc. 13th Int. Conf. on Machine Learning*. Bari, 1996, pp. 148–156.
12. Druzhkov P.N., Zolotykh N.Yu., Polovinkin A.N. Parallel implementation of prediction algorithm in gradient boosting trees method. *Bulletin SUSU*, 2011, no. 37, pp. 82–89. (in Russian)
13. CatBoost GitHub. Available at: https://github.com/catboost (accessed 29.04.2018).
14. Salakhutdinova K.I., Lebedev I.S., Krivtsova I.E. Informative feature selection in software identification task. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2018, vol. 18, no. 2, pp. 278–285 (in Russian). doi: 10.17586/2226-1494-2018-18-2-278-285
15. Druzhinin N.K., Salakhutdinova K.I. Identification of executable file by dint of individual feature. *Proc. Int. Conf. on Information Security and Protection of Information Technology, ISPIT-2015*. St. Petersburg, Russia, 2015, pp. 45–47.

Authors

Kseniya I. Salakhutdinova – postgraduate, ITMO University, Saint Petersburg, 197101, Russian Federation, Scopus ID: 57191362944, ORCID ID: 0000-0001-9254-8652, kainagr@mail.ru

Ilya S. Lebedev – D.Sc., Professor, Laboratory Head, Saint Petersburg Institute for Informatics and Automation RAS (SPIIRAS), 199178, Russian Federation, Scopus ID: 56321781100, ORCID ID: 0000-0001-6753-2181, isl_box@mail.ru

Irina E. Krivtsova – Senior lecturer, ITMO University, Saint Petersburg, 197101, Russian Federation, Scopus ID: 57190307529, ORCID ID: 0000-0003-2483-1637, ikr@cit.ifmo.ru