НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
март–апрель 2018 Том 18 № 2 ISSN 2226-1494 http://ntv.ifmo.ru/
SCIENTIFIC AND TECHNICAL JOURNAL OF INFORMATION TECHNOLOGIES, MECHANICS AND OPTICS
March–April 2018 Vol. 18 No 2 ISSN 2226-1494 http://ntv.ifmo.ru/en

УДК 004.02; 311.2

ПОДХОД К ВЫБОРУ ИНФОРМАТИВНОГО ПРИЗНАКА В ЗАДАЧЕ ИДЕНТИФИКАЦИИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

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

[a] Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация
Адрес для переписки: kainagr@mail.ru

Информация о статье
Поступила в редакцию 18.12.17, принята к печати 13.02.18
doi: 10.17586/2226-1494-2018-18-2-278-285
Язык статьи – русский

Ссылка для цитирования: 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

Аннотация

Постановка задачи. Необходимость снижения роста числа уязвимостей системы, вызываемого установкой несанкционированного программного обеспечения на средства вычислительной техники, требует разработки способа автоматизации процесса аудита носителей информации. В работе предложен подход к выявлению информативности ассемблерных команд. Исследовано влияние выбора признака для формирования унифицированных сигнатур программ на результаты идентификации. **Метод.** Для расчета информативности применен метод Шеннона, позволяющий определить информативность признака для произвольного числа классов объектов, не зависящий от объема выборок наблюдаемых признаков. Идентификация elf-файлов основана на применении статистического критерия однородности хи-квадрат. **Основные результаты.** Получены количественные характеристики информативности для 118 ассемблерных команд. Проведен анализ результатов эксперимента по идентификации исполняемых файлов с использованием 10 различных признаков для формирования сигнатур программ. Сравнение выполнено с помощью критерия однородности хи-квадрат на уровнях значимости $p = 0{,}05$ и $p = 0{,}01$. **Практическая значимость.** Обнаружена важность использования того или иного признака в задаче по формированию сигнатур программ, а также возможность рассмотрения нескольких сигнатур исполняемых файлов в единой связке для создания итоговой оценки принадлежности к известной программе.

Ключевые слова
идентификация исполняемых файлов, elf-файлы, информативность признака, хи-квадрат критерий, информационная безопасность

INFORMATIVE FEATURE SELECTION IN SOFTWARE IDENTIFICATION TASK

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

[a] ITMO University, Saint Petersburg, 197101, Russian Federation
Corresponding author: kainagr@mail.ru

Article info
Received 18.12.17, accepted 13.02.18
doi: 10.17586/2226-1494-2018-18-2-278-285
Article in Russian

For citation: 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

Abstract

Subject of Research. The need for slowdown of the increasing number of vulnerabilities caused by installation of unauthorized software on computer equipment, calls for an approach development to automate the audit of data storage media. The paper proposes an approach for identification of informative assembler commands. We study the effect of a chosen feature used for creation of unified program signature on the identification results. **Methods.** The Shannon method

was used for informativity calculation. It gives the possibility to determine the feature informativity for random number of object classes and is independent of the volume of observed feature samples. Identification of elf-files was based on application of chi-square statistical homogeneity criterion. **Main Results.** Quantitative informativity characteristics for 118 assembler commands are obtained. The analysis of experiment results for executable files identification is carried out with the use of ten different features for creation of program signatures. Comparison is performed by chi-square homogeneity criterion at significance levels $p = 0.05$ and $p = 0.01$. **Practical Relevance.** We have found out the importance of particular feature application in the task of program signatures creation, as well as the possibility of considering several executable file signatures in common to create the final score of belonging to a certain program.

Keywords
identification of executable files, elf-files, feature informativity, chi-square criterion, information security

Введение

Сравнительная легкость доступа к различным страницам, сайтам сети Интернет, открытость программного обеспечения и возможность его модификации – все это ведет к необходимости аудита электронных носителей информации на предмет выявления несанкционированно установленного программного обеспечения (ПО) [1, 2].

Действия пользователей автоматизированных систем, направленные против установленной политики безопасности в организации, могут привести к росту числа уязвимостей системы и повлиять на ее информационную безопасность. Причиной этого являются возможные дефекты ПО, наличие не декларированных возможностей, нелегальное использование интеллектуальной собственности, а также использование специальных программ, направленных на преодоление установленной защиты либо противоправных действий внутри сети Интранет или Интернет. Последнее особенно актуально в области преступлений, связанных с компьютерной информацией [3].

Стоит отметить, что авторы настоящей работы рассматривают обычное пользовательское ПО операционной системы (ОС) Linux (в частности, elf-файлы) и не рассматривают вредоносные программы, методы обнаружения [4–6], идентификации или распознавания которых были представлены во многих научных работах [7, 8]. Научных же работ в рассматриваемой области найдено крайне мало [9, 10].

Применение некоторых стандартных методов анализа ПО, таких как ручной осмотр характерных мест установки программ, сравнение с полной копией данных, сравнение контрольных сумм, контроль CRC (Cyclic Redundancy Check), хэширование, имитовставка, цифровая подпись, не всегда успешно выявляет установленные программы из-за возможного отсутствия эталонного образца (значения хеш-суммы, неизмененной копии файла, цифровой подписи и т.д.), необходимого в перечисленных методах.

Рассматриваемый в работе подход к идентификации ПО, т.е. отождествление того или иного исполняемого файла с некоторой известной программой, направлен на распознавание программы не основываясь на ее целостности. Процесс идентификации сравнивает две сигнатуры: унифицированную, созданную на основе обучающей выборки, и сигнатуру идентифицируемой программы, создаваемой непосредственно перед этапом сравнения. Такая гибкость подхода позволяет успешно идентифицировать различные версии одной и той же программы, даже те, которые ранее не участвовали в создании унифицированной сигнатуры.

Ранее авторами были разработаны новые подходы к созданию сигнатур программ [11, 12], а также методы идентификации на основе этих сигнатур [13, 14]. При этом не уделялось внимание выбору признака, задействованного в формировании сигнатур. В настоящей работе предметом исследования является влияние выбранного признака на точность результата идентификации. Планируется рассмотреть информативность 118 ассемблерных команд и сравнить результаты идентификации по выбранным 10 командам.

Расчет информативности признаков

При рассмотрении исполняемого файла со стороны его дизассемблированного кода было бы крайне полезно выделить те ассемблерные команды, которые наиболее эффективно помогали бы идентифицировать программы. Для решения данной задачи был выбран метод Шеннона, позволяющий определить информативность признака для произвольного числа классов объектов, причем не зависящий от объема выборок наблюдаемых признаков. Этот метод предлагает оценивать информативность как средневзвешенное количество информации (величину устраненной энтропии), свойственное рассматриваемому признаку $x \in X$, где X – пространство признаков.

Для оценки информативности признака x пользуются следующей формулой:

$$I(x) = 1 + \sum_{i=1}^{G} \left(P_i \cdot \sum_{u=1}^{U} P_{i,u} \cdot \log_U P_{i,u} \right), \quad (1)$$

где G – количество градаций признака (в данной работе будут рассмотрены два случая: одна градация – появление выбранной ассемблерной команды; две градации – появление выбранной ассемблерной команды и появление другой, отличной от данной, команды); U – количество классов (это количество частотных распределений ассемблерных команд программ, задействованных при расчете информативности

признаков); P_i – вероятность попадания значения признака в i-ю градацию, которая вычисляется по формуле

$$P_i = \frac{\sum_{u=1}^{U} m_{i,u}}{N},$$

где $m_{i,u}$ – частота появления значения признака в i-ой градации в u-ом классе; N – общее число наблюдений признака; $P_{i,u}$ – вероятность появления i-ой градации признака в k-ом классе, которая вычисляется по формуле

$$P_{i,u} = \frac{m_{i,u}}{\sum_{u=1}^{U} m_{i,u}}.$$

Необходимо отметить, что метод Шеннона дает оценку информативности $I(x)$ в виде нормированной величины, принимающей значения в интервале от нуля до единицы. Считается, что чем ближе значение $I(x)$ к единице, тем выше информативность признака x и, наоборот, чем ближе к нулю, тем ниже информативность признака x.

Невозможность произвести оценку информативности признаков для всего пространства существующих программ является недостатком данного метода.

Для 118 отобранных ассемблерных команд был произведен анализ по выявлению наиболее информативных из них. В эксперименте приняли участие 10 различных программ и отличных версий. В общем, было сформировано 52 частотных распределения признака отражающих частоты встречаемости каждой из 118 команд в дизассемблированном коде программы.

Результаты расчета информативности для $G = 1$ и $G = 2$ представлены в табл. 1 и табл. 2 соответственно, где в целях оптимизации размера таблиц значения информативности $I(x)$ были округлены до четвертого знака после запятой. Все ассемблерные команды (также и те, которые находятся в одной общей ячейке таблицы) расположены в порядке возрастания их информативности.

Ассемблерные команды	**Информативность, $I(x)$**
cmpsb, cmpsw, esc, jc, jcxz, jna, jnae, jnb, jnbe, jnc, jng, jnge, jnl, jnle, jnz, jpe, jpo, jz, lodsb, lodsw, loopnz, loopz, movsb, movsw, repe, repne, retn, sal, scasb, scasw, stosb, stosw, wait	0
rep, jle, dec, **lea**, std, **cmp**, shr, **jmp**, jg, shl, **and**, cld, rol, js, jl, **add**, nop, mul	0,1408–0,1990
jne, **push**, ret, sub, xor, jge, ror, not, test, **mov**, div, jbe, jb, loopne, clc, **je**, jae, xchg, ja, or, repz, cli, adc, pushf, hlt, sar, sti, sbb, neg, jns, idiv, in, **call**, imul, jp, loop, jno, **pop**, repnz, out, iret, loope, jnp, xlat, int, rcl, cmc	0,2001–0,2972
jo, stc, lahf, retf, rcr, sahf, inc, cwd, popf, cbw	0,3026–0,3778
lock	0,4228
daa, les, into, aaa, lds, aam, aas, das, aad	0,6144–0,6785

Таблица 1. Значения информативности для 118 ассемблерных команд, рассчитанные по методу Шеннона с заданным числом градаций признака $G = 1$

Ассемблерные команды	**Информативность, $I(x)$**
cmpsb, cmpsw, esc, jc, jcxz, jna, jnae, jnb, jnbe, jnc, jng, jnge, jnl, jnle, jnz, jpe, jpo, jz, lodsb, lodsw, aaa, les, daa, aas, aam, ja, mul, js, loop, jge, jp, hlt, jl, jns, loopne, iret, das, rep, jae, idiv, loopnz, loopz, movsb, movsw, repe, repne, retn, sal, scasb, scasw, stosb, stosw, wait, cwd, cbw, div, neg, into, lds, aad	0,2024
rcl, std, sbb, lahf, ror, ret, jne, cmc, popf, rcr, adc, jno, in, shl, jbe, jb, pushf, not, clc, rol, int, jle, jg, sub, loope, xlat, cld, sti, shr, jnp, repnz, cli, dec, repz	0,2025
imul, sar, sahf, **jmp**, xor, nop, **and**, jo, **je**, stc, test, **push**, retf	0,2026
out, **cmp**, inc	0,2027
or	0,2028
xchg	0,2029
add, **pop**	0,203
lea	0,2032
call	0,2036
mov	0,2044
lock	0,2095

Таблица 2. Значения информативности для 118 ассемблерных команд, рассчитанные по методу Шеннона с заданным числом градаций признака $G = 2$

Очевидно, что от выбранного значения числа градаций признака зависит и порядок информативности ассемблерных команд. Такое расхождение происходит в результате существенного влияния дополнительной градации (не появление рассматриваемого признака, а появление другой, отличной ассемблерной команды) на рассчитываемые параметры P_i и $P_{i,u}$ в формуле (1). Таким образом, учитывается отношение числа встречаемости рассматриваемой ассемблерной команды к числу появления всех остальных 117 ассемблерных команд, что позволяет рассчитать информативность не только на основе различия частот встречаемости команды в различных классах, но и на основе ее доли по отношению к остальным командам.

Недостатком установления наиболее информативных команд с использованием числа градаций признака $G = 1$ является невозможность дальнейшего создания сигнатур программ, так как их частота встречаемости в дизассемблированных кодах программ слишком мала для формирования сигнатур с достаточным числом ненулевых значений. Второй подход с использованием числа градаций признака $G = 2$ позволяет устранить данный недостаток.

Имена ассемблерных команд, выделенные жирным начертанием, будут в дальнейшем задействованы в эксперименте по идентификации программ. Основываясь на результатах табл. 1 и табл. 2, примем следующий порядок информативности выбранных 10 ассемблерных команд: mov, call, pop, push, je, lea, add, cmp, and, jmp.

Идентификация программ

Использование в качестве признака одной ассемблерной команды приводит к разработке нового подхода к формированию сигнатур, основанного на разбиении дизассемблированного кода программы на интервалы равной длины и подсчете в них частот встречаемости признака.

Процесс создания унифицированной сигнатуры программы основан на формировании некоторой единой последовательности распределения признака, исходя из схожести нескольких сигнатур различных версий исполняемых файлов, которые относятся к одной и той же программе.

Напомним, что для формирования архива унифицированных сигнатур программ анализируется определенный объем исполняемых файлов, для чего формируется обучающая выборка $TS = \{v_1, v_2,\ldots, v_m\}$, $i = 1,\ldots, m$, где v_i – выборка различных программ; m – количество различных программ; $v_i = \{f_1, f_2,\ldots, f_n\}$, f_j – различные версии i-ой программы, n – количество файлов в выборке.

Каждый файл f_j дизассемблируется и разбивается на интервалы равных длин, при этом вводится фиксированный коэффициент для формирования длины шага, корректирующий количество получаемых интервалов для файлов разного объема. В этом случае за длину интервала принимается количество различных ассемблерных команд на промежутке одного шага. В эксперименте коэффициент был подобран таким образом, чтобы число интервалов равнялось тридцати.

Частотное распределение признака для файла f_j записывается как $L(f_j) = (a_k)$, где a_k – частота признака в k-ом интервале, $j = 1,\ldots, n$, k – количество получаемых интервалов, и зависит от вводимого коэффициента.

Дальнейшее формирование унифицированных сигнатур и сигнатур идентифицируемых файлов было ранее описано авторами в упоминаемых выше работах.

Непосредственно этап сравнения двух сигнатур представляет собой проверку статистической гипотезы об однородности двух выборок, проверяемой с помощью критерия однородности хи-квадрат. Данный критерий позволяет нам сравнивать эмпирические частотные распределения – сигнатуры, функции распределения которых нам не известны.

Если в результате эксперимента получены две независимые выборки объемов n_1 и n_2, причем по рассматриваемому признаку выборки распадаются на k интервалов с частотами $m_1, m_2, \ldots, m_k$ и $m'_1, m'_2, \ldots, m'_k$, то эмпирическое значение хи-квадрат критерия рассчитывают по формуле (2):

$$\chi^2 = n_1 n_2 \sum_{i=1}^{k} \frac{1}{m_i + m'_i}\left(\frac{m_i}{n_1} - \frac{m'_i}{n_2}\right)^2, \qquad (2)$$

где $m_1 + m_2 + \ldots + m_k = n_1$ и $m'_1 + m'_2 + \ldots + m'_k = n_2$.

Доказано, что эта статистика при больших значениях n_1 и n_2 распределена по закону χ^2 с k–1 степенями свободы [15].

Известно, что критерий однородности хи-квадрат имеет правостороннюю критическую область, поэтому, если при уровне значимости p выполняется неравенство $\chi^2 < \chi^2_p$, то нет оснований отвергнуть гипотезу об однородности распределений.

Постановка эксперимента

В эксперименте было задействовано 443 исполняемых файла обучающей выборки ОС Linux различных версий и разрядностей (32x и 64x), относящихся к 63 различным программам. В тестовую выборку было включено 123 файла, относящихся к тем же 63 программам, все они были отличны от задей-

ствованных файлов, используемых в обучающей выборке, и имели 32х и 64х разрядности. Обучающая выборка формировалась путем скачивания программ с официальных репозиториев ОС Linux для архитектур процессоров x86 и x86-64. Затем, для формирования тестовой выборки, из нее извлекалось по два исполняемых файла различных версий и разрядностей по каждой программе (кроме одной, представленной только в одной разрядности).

Стоит отметить, что архив сигнатур не является фиксированным и запрещенным для добавления новых сигнатур, наоборот, он должен периодически обновляться и иметь актуальные данные для выполнения поставленных исследователем задач.

Критерий однородности в рамках нашей задачи применялся для проверки основной гипотезы H_0 – сигнатуры идентифицируемого файла и программы из архива схожи и относятся к одной и той же программе, при конкурирующей гипотезе H_1 – сигнатуры идентифицируемого файла и программы из архива отличаются значимо и относятся к разным программам.

При помощи программного комплекса STATISTICA полученные частотные распределения с дизассемблированных кодов программ разбивались по классам, где в дальнейшем формировались в единую унифицированную сигнатуру и заносились в архив. Отдельно процесс построения сигнатур происходил для идентифицируемых исполняемых файлов.

В качестве информативных признаков были выбраны следующие ассемблерные команды: mov, call, pop, push, je, lea, add, cmp, and, jmp. Для каждой из них было сформировано частотное распределение.

Вне зависимости от битности идентифицируемого файла, его сигнатура сравнивалась с унифицированной сигнатурой, построенной как для 32-битной программы, так и для 64-битной, при этом принималось решение о принадлежности к рассматриваемой программе, если после применения статистического критерия делался вывод о принятии основной гипотезы.

Целью эксперименты было выявление зависимости результатов идентификации программ от выбора того или иного информативного признака.

Результаты

В процессе сравнения двух сигнатур можно получить следующие результаты:

1. верно принята гипотеза H_0;
2. не верно отвергнута гипотеза H_0;
3. не верно отвергнута гипотеза H_1;
4. верно принята гипотеза H_1.

Первый пункт говорит о том, что для сравниваемых двух сигнатур, относящихся к одной и той же программе, нами была *верно* принята гипотеза об их сходстве. Второй пункт (ошибка первого рода) – для сравниваемых двух сигнатур, относящихся к одной и той же программе, нами была *неверно* отклонена гипотеза об их сходстве. Третий пункт – (ошибка второго рода) для сравниваемых двух сигнатур, относящихся к разным программам, нами была *неверно* принята гипотеза об их сходстве. Четвертый пункт – для сравниваемых двух сигнатур, относящихся к разным программам, нами была *верно* отвергнута гипотеза об их сходстве.

Процент корректных результатов идентификации и ошибок первого и второго рода представлен в табл. 3.

Анализируя данные из табл. 3, можно сделать вывод о том, что для первой половины наиболее информативных ассемблерных команд (mov, call, pop, push, je) в среднем показатель корректных результатов идентификации выше (96,85 при $p = 0{,}05$), а процент числа ошибок второго рода меньше (1,88 при $p = 0{,}05$), чем для второй половины ассемблерных команд (lea, add, cmp, and, jmp), являющихся менее информативными (94,69 и 4,062 при $p = 0{,}05$ соответственно).

Очевидно, что выбор информативного признака существенно влияет на результаты идентификации. Можно подойти к рассмотрению не одного распределения признака, а нескольких, с последующим формированием некоего общего результата по всем распределениям.

На рис. 1 изображена поверхность, построенная по 10 идентифицируемым сигнатурам (для выбранных ассемблерных команд) файла amarok_2.3.0-0ubuntu4_i386. Здесь по оси абсцисс отмечены интервалы сигнатур (от одного до тридцати); по оси ординат – ассемблерные команды, где первая команда – mov, вторая – call и т.д. по ранее заданному нами порядку информативности, такой же порядок задан и в табл. 3; по оси аппликат – частоты встречаемости ассемблерной команды в файле.

На рис. 2, а, и рис. 2, б, изображены поверхности, построенные по модулю разности между десятью распределениями, построенными для идентифицируемого файла, и десятью распределениями унифицированных сигнатур соответственно. Очевидно, что для одинаковых программ количество возвышений невелико, а для половины ассемблерных команд и вовсе минимально, у различных же программ поверхность имеет большее число возвышений, а их максимальное значение в два раза больше, чем для одинаковых программ.

Отношение между сигнатурой программы из архива и сигнатурой идентифицируемого файла		Сигнатуры относятся к одной и той же программе				Сигнатуры относятся к разным программам			
Гипотеза о сходстве сигнатур		Принимается		Отвергается		Принимается		Отвергается	
Уровень значимости, *p*		0,05	0,01	0,05	0,01	0,05	0,01	0,05	0,01
Ассемблерная команда	mov	0,22	0,24	1,37	1,35	0,69	0,69	97,71	97,71
	call	0,34	0,4	1,25	1,19	2,19	2,19	96,22	96,22
	pop	0,33	0,39	1,26	1,21	2,34	2,34	96,07	96,07
	push	0,26	0,31	1,33	1,28	0,74	0,74	97,67	97,67
	je	0,45	0,47	1,14	1,12	3,43	3,43	94,98	94,98
	lea	0,47	0,53	1,12	1,06	11,44	11,44	86,97	86,97
	add	0,16	0,18	1,44	1,42	0,01	0,01	98,4	98,4
	cmp	0,44	0,48	1,15	1,11	6,34	6,34	92,07	92,07
	and	0,3	0,34	1,29	1,25	1,37	1,37	97,04	97,04
	jmp	0,33	0,39	1,26	1,2	1,15	1,15	97,26	97,26

Таблица 3. Результаты идентификации по различным ассемблерным командам, %

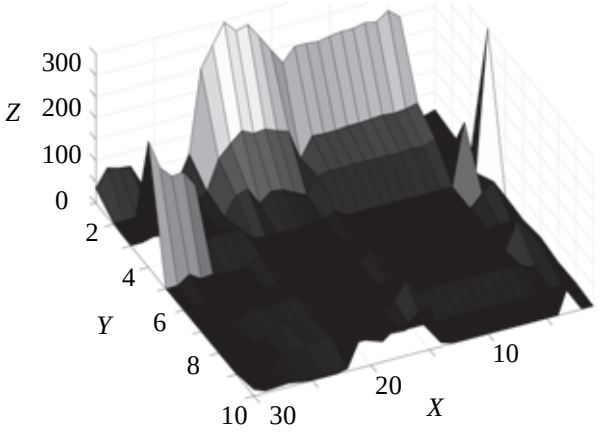

Рис. 1. Сигнатуры 10 ассемблерных команд для amarok_2.3.0-0ubuntu4_i386

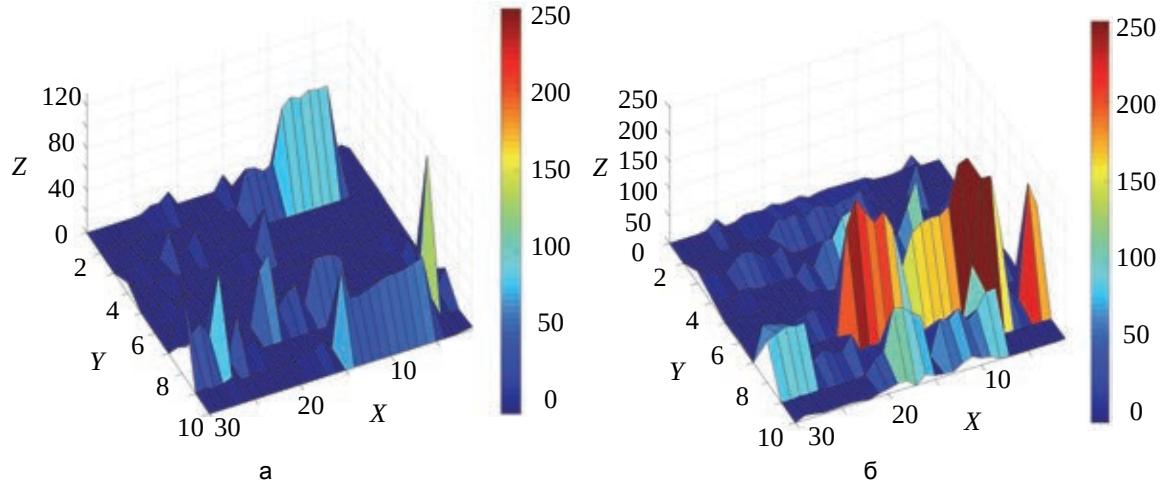

Рис. 2. Абсолютная разница между сигнатурой идентифицируемого файла и унифицированной сигнатурой из архива: для amarok_2.3.0-0ubuntu4_i386 и amarok (а); для amarok_2.3.0-0ubuntu4_i386 и anacorn (б)

Заключение

В работе рассмотрен подход к расчету информативности 118 ассемблерных команд, наиболее информативные из которых впоследствии участвовали в процессе формирования сигнатур для программ обучающей и тестовой выборок.

Результаты эксперимента по идентификации исполняемых файлов показали, что процент корректных результатов идентификации в среднем выше для более информативных команд, выявленных по методу Шеннона. Становится очевидным, что выбор признака является важной частью при использовании разработанного метода по идентификации программ.

В дальнейшем планируется разработка способа формирования итоговой оценки принадлежности идентифицируемого файла к известной программе на основе расчета нескольких частотных распределений ассемблерных команд и применении критерия Фишберна.

Литература

1. Сулейманова Ш.С., Назарова Е.А. Информационные войны: история и современность: Учебное пособие. М.: Этносоциум, 2017. 126 с.
2. Williams S.P., Hardy J.A., Holgate C.A. Information security governance practices in critical infrastructure organizations: a socio-technical and institutional logic perspective // Electronic Markets. 2013. V. 23. N 4. P. 341–351. doi: 10.1007/s12525-013-0137-3
3. Boukhtouta A., Mouheb D., Debbabi M., Alfandi O., Iqbal F., El Barachi M. Graph-theoretic characterization of cyber-threat infrastructures // Digital Investigation. 2015. V. 14. N 1. P. S3–S15. doi: 10.1016/j.diin.2015.05.002
4. Alazab M., Layton R., Venkataraman S., Watters P. Malware detection based on structural and behavioral features of API calls // Proc. International Cyber Resilience Conference (ICR2010). 2010. P. 1–10.
5. Shahzad F., Farooq M. ELF-Miner: Using structural knowledge and data mining methods to detect new (Linux) malicious executables // Knowledge and Information Systems. 2011. V. 30. N 3. P. 589–612. doi: 10.1007/s10115-011-0393-5
6. Li P., Liu L., Gao D., Reiter M.K. On challenges in evaluating malware clustering // Lecture Notes in Computer Science. 2010. V. 6307. P. 238–255. doi: 10.1007/978-3-642-15512-3_13
7. Комашинский Д.В., Котенко И.В. Методы интеллектуального анализа данных для выявления вредоносных программных объектов: обзор современных исследований // Вопросы защиты информации. 2013. № 4(102). С. 21–33.
8. Lai Y.X., Liu Z.H. Unknown malicious identification // Lecture Notes in Electrical Engineering. 2009. V. 39. P. 301–312. doi: 10.1007/978-90-481-2311-7_26
9. Антонов А.Е., Федулов А.С. Идентификация типа файла на основе структурного анализа // Прикладная информатика. 2013. № 2(44). С. 68–77.
10. Казарин О.В. Теория и практика защиты программ. М.: МГУЛ, 2004. 450 с.
11. Кривцова И.Е., Салахутдинова К.И., Кузьмич П.А. Метод построения сигнатур исполняемых файлов с целью их идентификации // Вестник полиции. 2015. Т. 5. № 3(5). С. 97–105. doi: 10.13187/vesp.2015.5.97
12. 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. St. Petersburg, Russia, 2015. P. 45–47.
13. 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.
14. 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
15. Смирнов Н.В., Дунин-Барковский И.В. Курс теории вероятностей и математической статистики. М.: Наука, 1969. 512 с.

References

1. Suleimanova Sh.S., Nazarova E.A. *Information Wars: History and Present*. Moscow, Etnosotsium Publ., 2017, 126 p. (in Russian)
2. Williams S.P., Hardy J.A., Holgate C.A. Information security governance practices in critical infrastructure organizations: a socio-technical and institutional logic perspective. *Electronic Markets*, 2013, vol. 23, no. 4, pp. 341–351. doi: 10.1007/s12525-013-0137-3
3. Boukhtouta A., Mouheb D., Debbabi M., Alfandi O., Iqbal F., El Barachi M. Graph-theoretic characterization of cyber-threat infrastructures. *Digital Investigation*, 2015, vol. 14, no. 1, pp. S3–S15. doi: 10.1016/j.diin.2015.05.002
4. Alazab M., Layton R., Venkataraman S., Watters P. Malware detection based on structural and behavioral features of API calls. *Proc. International Cyber Resilience Conference, ICR2010*, 2010, pp. 1–10.
5. Shahzad F., Farooq M. ELF-Miner: Using structural knowledge and data mining methods to detect new (Linux) malicious executables. *Knowledge and Information Systems*, 2011, vol. 30, no. 3, pp. 589–612. doi: 10.1007/s10115-011-0393-5
6. Li P., Liu L., Gao D., Reiter M.K. On challenges in evaluating malware clustering. *Lecture Notes in Computer Science*, 2010, vol. 6307, pp. 238–255. doi: 10.1007/978-3-642-15512-3_13
7. Komashinskii D.V., Kotenko I.V. Data mining methods for malware detection: review of state-of-the-art. *Information Security Questions*, 2013, no. 4, pp. 21–33. (in Russian)
8. Lai Y.X., Liu Z.H. Unknown malicious identification. *Lecture Notes in Electrical Engineering*, 2009, vol. 39, pp. 301–312. doi: 10.1007/978-90-481-2311-7_26
9. 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)
10. Kazarin O.V. *Theory and Practice of Program Protection*. Moscow, MGUL Publ., 2004, 450 p. (in Russian)
11. Krivtsova I.E., Salakhutdinova K.I., Kuz'mich P.A Method of construction the signatures of executable files for identification purposes. *Vestnik Policii*, vol. 5, no. 3, pp. 97–105. (in Russian) doi: 10.13187/vesp.2015.5.97
12. 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*. St. Petersburg, Russia, 2015, pp. 45–47.
13. 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)
14. 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
15. Smirnov N.V., Dunin-Barkovskii I.V. *Course of Probability Theory and Mathematical Statistics*. Moscow, Nauka Publ., 1969, 512 p. (in Russian)

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, профессор, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, Scopus ID: 56321781100, ORCID ID: 0000-0001-6753-2181, lebedev@cit.ifmo.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

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., Full Professor, ITMO University, Saint Petersburg, 197101, Russian Federation, Scopus ID: 56321781100, ORCID ID: 0000-0001-6753-2181, lebedev@cit.ifmo.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