Андрей Н. Кузнецов, Дмитрий А. Вышемирский
Лаборатории ТВП, г. Москва, Нахимовский пр-т, дом 47,117418, Россия
e-mail: kuzz2000@mail.ru, ORCID iD 0000-0002-1782-6584
e-mail: v.dmitry-12@yandex.ru, ORCID iD 0000-0002-9015-7022

ОБ ОДНОМ ПОДХОДЕ К РЕШЕНИЮ ЗАДАЧИ ТОКЕНИЗАЦИИ ПРИ АНАЛИЗЕ БОЛЬШИХ МАССИВОВ ПОЛЬЗОВАТЕЛЬСКИХ ПАРОЛЕЙ

DOI: http://dx.doi.org/10.26583/bit.2017.2.06

Аннотация. Проведен анализ алгоритма выделения в паролях пользователей слов на естественных языках (токенизации), предложенного ранее в работе R. Veras et al. [1] (RV-алгоритм). Выявлены основные недостатки данного подхода. Предложен новый алгоритм токенизации (RGramToken) на основе частотных словарей слов, биграмм и триграмм, позволяющий лучше учесть информацию о вероятности использования слов и словосочетаний в естественном языке. Подготовлены тестовые выборки фраз с учетом возможных искажений типа «вставка» между словами естественного языка. Проведен сравнительный анализ результатов работы на тестовых и реальных выборках алгоритма RGramToken с RV-алгоритмом. Показано более высокое качество токенизации предложенным в работе алгоритмом на искаженных фразах, а также устойчивость к качеству используемых словарей.

Ключевые слова: пароли, утечки, токенизация, разбиение на слова.

Для цитирования. КУЗНЕЦОВ, Андрей Н.; ВЫШЕМИРСКИЙ, Дмитрий А. Об одном подходе к решению задачи токенизации при анализе больших массивов пользовательских паролей. Безопасность информационных технологий, [S.l.], v. 24, n. 2, p. 50-60, june 2017. ISSN 2074-7136. Доступно на: <https://bit.mephi.ru/index.php/bit/article/view/105>. Дата доступа: 23 june 2017. doi:http://dx.doi.org/10.26583/bit.2017.2.06.

Andrey N. Kuznetsov, Dmitry A. Vyshemirsky
TVP laboratories, Moscow, Nakhimovsky PR-t, 47, 117418, Russia
e-mail: kuzz2000@mail.ru, ORCID iD 0000-0002-1782-6584
e-mail: v.dmitry-12@yandex.ru, ORCID iD 0000-0002-9015-7022

One approach to solving tokenization problem for analysis of large-scale collections of user-defined passwords

DOI: http://dx.doi.org/10.26583/bit.2017.2.06

Abstract. This paper performs an analysis of the algorithm of password tokenization introduced by R. Veras et al. [1]. We show main limitations of this approach and propose a new tokenization algorithm - RGramToken, based on frequency dictionaries of English words, bigrams and trigrams. Our approach allows better utilization of information about probabilitiy distribution of words and word combinations in a natural language. The results of comparison analysis of these two algorithms on specially prepared tests with warped phrases demonstrate higher efficiency of RGramToken and its robustness on low quality dictionaries.

Keywords: passwords, leaks, tokenization, words isolation.

For citation. KUZNETSOV, Andrey N.; VYSHEMIRSKY, Dmitry. A. One Approach to Solving Tokenization Problem for Analysis of Large-Scale Collections of User-Defined Passwords. IT Security (Russia), [S.l.], v. 24, n. 2, p. 50-60, june 2017. ISSN 2074-7136. Available at: <https://bit.mephi.ru/index.php/bit/article/view/105>. Date accessed: 23 june 2017. doi:http://dx.doi.org/10.26583/bit.2017.2.06.

Введение

Парольная защита на сегодняшний день является одним из наиболее распространенных способов решения задач аутентификации пользователей и обеспечения конфиденциальности пользовательских данных. Распространенность парольной защиты

как элемента криптографических систем практически не снижается, несмотря на многочисленные компрометации, получившие широкую огласку в последние годы: утечки аутентификационных данных миллионов пользователей в сеть Интернет, а также некоторые результаты анализа таких утечек. На примере многих сайтов показано, что знание всего 10 наиболее популярных паролей позволяет получить доступ к данным 5-10% пользователей из их многомиллионной аудитории [2].

Статистики распределения длин паролей, а также структуры используемых в них алфавитов, полученные в результате анализа таких баз реальных паролей, позволяют оценить эффективность тотального опробования. Однако, подсчет теоретико-информационных характеристик парольных массивов, в особенности значения энтропии текста [3], показал, что построение более сложных моделей парольных источников, учитывающих закономерности в выборе паролей и зависимости между символами и группами символов внутри пароля, позволяет существенно повысить эффективность атак, с одной стороны, и сформулировать требования к выбираемым паролям, повышающие их безопасность, с другой [4].

Появились работы, в которых предлагаются различные стохастические подходы к моделированию парольных источников: вероятностные контекстно-независимые грамматики [5-7], марковские модели [8-10], другие оригинальные модели (например, [3]). Практически все известные авторам подходы, включая перечисленные, основаны на статистических зависимостях между символами или группами символов из различных алфавитов. При этом в практике автоматизированного анализа текстов на естественных языках (*Natural Language Processing – NLP*) стандартом де-факто на сегодня являются модели, учитывающие зависимости между словами, которые гораздо лучше отражают реальные свойства текстов.

Для построения модели пароля, в основе которой будут лежать слова естественного языка, необходимо иметь адекватную обучающую выборку, то есть для расчета статистик слов необходимо иметь массив реальных паролей с разметкой по словам. Это, в свою очередь, означает, что необходимо решить так называемую задачу *токенизации* – разделения текста на слова.

Для текстов в традиционном понимании (художественных, публицистических, деловых и т.п.) подходы к решению данной задачи хорошо известны [11, 12] и достаточно эффективны, однако специфика построения паролей – умышленные искажения слов и вставка семантически необусловленных аффиксов – делает эти методы практически неприменимыми. Для массивов паролей необходим новый подход, позволяющий учесть особенности их построения.

Первую известную авторам попытку разработать алгоритм токенизации для паролей предприняли в 2013 году M. Jakobsson и M. Dhiman [13]. В 2014 году команда исследователей из канадского университета Онтарио (R. Veras et al.) предложила модификацию данного алгоритма [1], сочтя первоначальную версию недостаточно эффективной.

Попытка реализовать алгоритм 2014 года на практике и проверить на доступных базах реальных паролей показала неудовлетворительный результат. В настоящей работе делается попытка проанализировать причины низкой эффективности подхода R. Veras и предлагается новый алгоритм выделения слов естественного языка в пользовательских паролях, а также проводится сравнительный анализ эффективности предложенного подхода с решением группы из Онтарио.

Алгоритм R. Veras (RV-алгоритм)

В работе [1] алгоритм токенизации описан достаточно подробно, здесь мы кратко изложим его основные принципы.

В основе предложенного подхода лежит набор словарей имен собственных – имена, фамилии, страны, города и названия месяцев, а также корпус современного английского языка с частотами встречаемости отдельных слов, их биграмм и триграмм (*Contemporary*

Corpus of American English – COCA). Поскольку корпус COCA распространяется на коммерческой основе (бесплатно доступна лишь ограниченная онлайн версия), мы в своих исследованиях использовали свободно распространяемый аналог – корпус *ENCOW* (*English web corpus by COW*) [14].

Пароль представляется как последовательность слов и вставок, не являющихся словами естественного языка. Авторы [1], аналогично подходу, предложенному в [13], строят для каждого пароля все возможные варианты сегментации и вычисляют степень покрытия пароля словарными словами. Вариант с максимальным значением покрытия считается правильным. Если таких вариантов оказывается несколько, для каждого из них вычисляется значение правдоподобия по статистике встречаемости слов и их сочетаний (*N-gram score*). В работе рассматриваются значения $N \in \{1,2,3\}$.

Алгоритм вычисления значения правдоподобия (*score*) изложен авторами в виде псевдокода (рисунок 1).

Algorithm 2 Recursively calculate the N-gram score of a segmentation

```
1: procedure BestNGramScore(C)
2:     score ← 0
3:     l ← Length(C)
4:
5:     if l = 1 then
6:         score ← UnigramProbability(C)
7:     else if l = 2 then
8:         score ← BigramProbability(C)
9:     else if l = 3 then
10:        score ← TrigramProbability(C)
11:    end if
12:
13:    if score = 0 then
14:        for i ← 1, 3 do
15:            a ← BestNGramScore(C[: i])
16:            b ← BestNGramScore(C[i :])
17:            tempScore ← a * b
18:            if tempScore > score then
19:                score ← tempScore
20:            end if
21:        end for
22:    end if
23: end procedure
```

Рис.1 Алгоритм подсчета правдоподобия

По крайней мере два решения в предлагаемом подходе могут приводить к ошибкам в результатах токенизации:

1. выбор кандидатов по максимальному значению покрытия;
2. способ вычисления величины правдоподобия, когда безальтернативно используется вероятность *r*-грамм с максимальным значением *r*.

Разберем эти ситуации на примерах.

Максимизация величины покрытия приводит на практике к тому, что алгоритм предпочтет несколько несвязанных друг с другом низковероятных слов длинному высокочастотному слову, если оно будет хоть на символ короче суммы длин слов в первом варианте.

Проиллюстрировать сказанное можно примером токенизации реального пароля «threeflattenedf», который, очевидно, получен из фразы «three flattened» путем вставки концевого символа «f». Наибольшим покрытием при этом обладает разбиение «three flatten edf» без вставок. Все слова полученной фразы присутствуют как в корпусе COCA, так в корпусе ENCOW. Вероятности *r*-грамм для вычисления вероятности данного и оригинального разбиений по корпусу ENCOW приведены в таблице 1.

Таблица 1. Частоты и вероятности *r*-грамм, входящих в пароль «threeflattenedf»

***r*-грамма**	**Частота**	**Вероятность**

three	$3{,}96*10^{6}$	$5{,}64*10^{-4}$
flattened	$2{,}09*10^{4}$	$2{,}98*10^{-6}$
flatten	$9{,}86*10^{3}$	$1{,}40*10^{-6}$
edf	187	$2{,}66*10^{-8}$
three flattened	12	$2{,}07*10^{-9}$
three flatten	1	$1{,}73*10^{-10}$
flatten edf	0	0
three flatten edf	0	0

Значение правдоподобия разбиения «three flatten edf» составляет p(«three flatten»)*p(«edf») = $4{,}68*10^{-18}$, в то время как для разбиения «three flattened» оно составляет p(«three flattened»)=$2{,}07*10^{-9}$. То есть при вероятности вставки буквы «f» в конце фразы выше $2{,}26*10^{-9}$ вариант разбиения исходной цепочки с вставкой в конце окажется более предпочтительным, но будет отбракован по величине покрытия.

В случае второго замечания если для рассматриваемого варианта разбиения в частотном корпусе находится триграмма (биграмма) с ненулевой вероятностью, то при вычислении правдоподобия всего варианта будет использовано это значение вероятности, даже если с точки зрения максимизации правдоподобия выгоднее рассматривать эту последовательность как набор r-грамм меньшего порядка.

Примером в данном случае может служить цепочка «aalex», при токенизации которой максимальное покрытие достигается для двух вариантов разбиения: «a a lex» и «a alex». Вероятности, необходимые для вычисления правдоподобия полученных разбиений, приведены в таблице 2. В таблице 3 вычислены правдоподобия разбиений при разных комбинациях одограмм, биграмм и триграмм. Все значения условных часот взяты из корпуса ENCOW.

Таблица 2. Частоты и вероятности *r*-грамм, входящих в пароль «aalex»

***r*-грамма**	**Частота**	**Вероятность**
a	$1{,}79*10^{8}$	$2{,}5*10^{-2}$
lex	2’789	$3{,}97*10^{-7}$
alex	3’787	$5{,}39*10^{-7}$
a a	22’388	$3{,}86*10^{-6}$
a lex	94	$1{,}62*10^{-8}$
a alex	2	$3{,}45*10^{-10}$
a a lex	0	0

Таблица 3. Правдоподобия всех возможных разбиений цепочки символов «aalex» согласно RV-алгоритму

Разбиение	**Комбинация *r*-грамм**	**Правдоподобие**
a a lex	p(«a a lex»)	0
	p(«a a») * p(«lex»)	$1{,}53*10^{-13}$
	p(«a») * p(«a lex»)	**$4{,}05*10^{-10}$**
a alex	p(«a alex»)	$3{,}45*10^{-10}$
Игнорируемые алгоритмом комбинации *r*-грамм		
a a lex	p(«a») * p(«a») * p(«lex»)	$2{,}48*10^{-10}$
a alex	**p(«a») * p(«alex»)**	**$1{,}35*10^{-8}$**

Из таблицы 3 видно, что исходный алгоритм вычисления правдоподобия результатов токенизации предпочел бы вариант «a a lex», который с мнемонической точки зрения менее привлекателен, чем «a alex». В случае разбиения биграммы (a, alex) на две одограммы, более правдоподобным оказался бы вариант «a alex».

Новый алгоритм токенизации (RGramToken)

Исходя из анализа достоинств и недостатков рассмотренного RV-алгоритма, предлагается новый алгоритм токенизации паролей пользователей. Ниже приводится его формальное описание.

Пусть задан словарь $S = \{s_1, \dots, s_n\}$, $n \in \mathbb{N}$, где s – слово естественного языка, состоящее из символов некоторого алфавита $\boldsymbol{A}$. Пусть заданы также статистики распределения частот r-грамм слов словаря S в некой обучающей выборке $\nu_r = \nu(w_1, \dots, w_r)$ для всех $w \in S$, $r = 1 \div 3$, где $\nu(w_1, \dots, w_r)$ – частота встречаемости r-граммы $w_1, \dots, w_r$.

Будем рассматривать биграммы и триграммы слов с ненулевыми частотами как самостоятельные слова с длиной равной сумме длин входящих в них слов, т.е. построим словарь $S' = S \cup S_2 \cup S_3$, где S_2 и S_3 – словари биграмм и триграмм с ненулевыми частотами соответственно. Тогда мы имеем общий вектор частот $\nu' = \nu(w_i')$, $w_i' \in S'$.

Множество всех возможных несловарных вставок в паролях обозначим через $V = \{\mathrm{v}_i\} \cup \{\emptyset\}$, где v_i – цепочки символов алфавита $\boldsymbol{B}$, в общем случае равного либо включающего $\boldsymbol{A}$, длиной не более максимальной длины пароля l_{max}, таких, что $\mathrm{v}_i \notin S$.

Необходимо представить произвольный пароль $D = (d_1, \dots, d_l)$, $l \in \mathbb{N}$, $d \in \boldsymbol{B}$ в виде последовательности слов словаря S и вставок несловарных последовательностей из множества V оптимальным способом в соответствии с описанным ниже критерием.

На первом этапе строятся все возможные варианты разбиения пароля D. Для всех $i \in [1, l]$, для которых $(d_1, \dots d_{i-1}) = \mathrm{v}_{j_i} \in V$, выбрать все возможные словарные слова $(d_i, \dots, d_{k_i}) = s_{m_i} \in S$, начинающиеся с i-го символа (при $i = 1$ вставка v_{j_i} будет нулевой длины). В результате будут получены цепочки вида $\{\mathrm{v}_{j_i}, s_{m_i}, (d_{k_i+1}, \dots d_l)\}$, для каждой из которых необходимо произвести разбиение нового пароля $D_{k_i} = (d_{k_i+1}, \dots, d_l)$ и конкатенировать результат с уже полученным началом $\{\mathrm{v}_{j_i}, s_{m_i}\}$.

Все полученные последовательности вида $\{\mathrm{v}_1, s_{11}, \dots s_{1k_1}, \mathrm{v}_2, s_{21}, \dots s_{2k_2}, \dots \mathrm{v}_p\}$, где только v_1 и v_p могут быть пустыми, а последовательности слов $s_{i1}, \dots s_{ik_i}$ при любых значениях i не содержат внутри себя цепочек из множества V, являются результатами разбиения пароля D.

Далее на втором этапе среди множества вариантов необходимо выбрать разбиение, на котором достигается максимум функции правдоподобия. Для этого вычислить величину правдоподобия для каждого из разбиений на основе общего вектора частот $\nu' = \nu(w_i')$, $w_i' \in S'$. Предварительно необходимо вычислить значения относительных частот (вероятностей) по формуле:

$$p(w_i') = \frac{\nu(w_i')}{\sum_{w_i' \in S'} \nu(w_i')}$$

Значение правдоподобия варианта разбиения $\{\mathrm{v}_1, s_{11}, \dots s_{1k_1}, \mathrm{v}_2, s_{21}, \dots s_{2k_2}, \dots \mathrm{v}_p\}$, полученного на первом этапе, вычисляется по формуле

$$L(\mathrm{v}_1, s_{11}, \dots s_{1k_1}, \mathrm{v}_2, s_{21}, \dots s_{2k_2}, \dots \mathrm{v}_p) = L\left(s_{11}, \dots, s_{(p-1)k_{p-1}}\right) \prod_{i=1}^{p} L(\mathrm{v}_i).$$

Для этого цепочка словарных слов $\left(s_{11}, \dots, s_{(p-1)k_{p-1}}\right)$ всеми возможными способами представляется как последовательность $\overline{w'}_j = \left(w'_{i_j}\right)$ элементов множества S'. Далее из множества $\{\overline{w'}_j\}$ возможных последовательностей выбирается вариант с максимальным значением правдоподобия, то есть $L\left(s_{11}, \dots, s_{(p-1)k_{p-1}}\right) = \max_j L(\overline{w'}_j)$, где $L(\overline{w'}_j) = \prod p(w'_{i_j})$.

Правдоподобие вставки для простоты предлагается вычислять по формуле $L(\mathrm{v}_i) = p(\mathrm{v}_0)^{len(\mathrm{v}_i)}$, где $p(\mathrm{v}_0)$ – вероятность вставки одного символа, $len(\mathrm{v}_i)$ – длина вставки v_i в символах. Здесь мы исходим из предположения, что короткие вставки между словами

естественного языка более вероятны, чем длинные. Значение $p(\mathrm{v}_0)$ есть параметр алгоритма и может влиять на результаты токенизации. В дальнейшем можно усложнить алгоритм, введя в него более сложные модели несловарных вставок в паролях. Для пустых вставок $L(\{\emptyset\}) = 1$.

Выбор наилучшего варианта разбиения пароля D производится по критерию максимума правдоподобия, то есть как $argmax\left(L\left(\mathrm{v}_1, s_{11}, \dots s_{1k_1}, \mathrm{v}_2, s_{21}, \dots s_{2k_2}, \dots \mathrm{v}_p\right)\right)$.

Результаты экспериментов

Как было сказано выше, для построения словаря S' и соответствующего массива частот ν' в настоящей работе использовался корпус ENCOW версии мая 2015 года. В качестве слов в данной выборке участвуют не только английские слова, но и различные знаки (кавычки, знаки препинания, апострофы, числа). Слова записаны с учетом регистра.

В ходе предварительного этапа из корпуса были исключены все *r*-граммы, содержащие символы не из английского алфавита, при этом апострофы и кавычки были предварительно удалены, а буквы приведены к нижнему регистру. Совпадающие *r*-граммы объединялись, а их частоты складывались.

Полученный в результате корпус оказался сильно зашумлен искаженными словами с низкими частотами, поэтому из него были исключены слова с частотой ниже выбранного порога v_b. Для оценки влияния значения порога частоты на результат алгоритма были подготовлены корпуса с использованием значений $v_b = 5$ (962 067 слов) и $v_b = 50$ (243 077 слов). Такое усечение словаря исключает из возможных вариантов разбиения редкие фамилии, географические названия и т.п. Этого можно избежать путем добавления в результирующий корпус слов из специализированных словарей имен, названий и т.д., как было сделано в работе [1]. Значение частоты для добавляемых слов должно находиться в интервале $1 \le \nu \le v_b$. В рамках данного исследования такая процедура не проводилась.

Для проведения сравнительного анализа алгоритма RGramToken, предложенного в настоящей работе, и RV-алгоритма, проведен ряд экспериментов на тестовых примерах. В качестве тестовой выборки выступал специальным образом подготовленный словарь фраз, построенный на основе художественного произведения «Crashlander» Ларри Нивена на английском языке. Из текста были удалены все символы, кроме букв английского алфавита и пробела, символы переведены в нижний регистр, после чего удалены слова, не входящие в исходный словарь. Далее весь текст был разделен на *N* фраз так, чтобы каждая *i*-я фраза содержала k_i слов, сумма длин первых $k-1$ слов $l_{k_i-1} < 12$, а всех слов фразы $12 \le l_{k_i} \le 20$ символов без учета пробелов. В итоге размер полученного словаря составил $N = 16646$ фраз.

Далее на основе базового тестового словаря было построено три словаря с искаженными фразами. Каждая фраза искажалась путем вставки случайных несловарных последовательностей символов длины $1 \le m \le 5$, значение m выбиралось случайно и равновероятно. Для оценки устойчивости результатов вставки делались тремя способами: в конец фразы, в середину (в случайную позицию между двумя словами), а также одновременно в середину и конец фразы.

После применения исследуемых алгоритмов к каждой фразе тестовых словарей подсчитывались число успешных разбиений и их доля от общей мощности словаря. Успешным считалось точное восстановление исходного состава фразы с указанием вставок.

Результаты токенизации исходных и искаженных фраз RV-алгоритмом и алгоритмом RGramToken приведены в таблице 4. Эксперименты проводились с использованием корпуса ENCOW с отсечением низкочастотных слов по порогам $v_b = 5$ и $v_b = 50$, алфавиты слов и вставок $|\boldsymbol{A}| = |\boldsymbol{B}| = 26$, вероятность вставки одного символа $p(\mathrm{v}_0) = 1/26$.

Таблица 4. Результаты токенизации на тестовых выборках

Тестовый словарь	**RV-алгоритм**		**RGramToken**	
	$v_b = 5$	$v_b = 50$	$v_b = 5$	$v_b = 50$
Без вставок	16085 (96,6%)	15602 (93,7%)	12284 (73,8%)	12250 (73,6%)
В конец	3974 (23,9%)	5016 (30,1%)	8115 (48,8%)	8116 (48,8%)
В середину	1821 (10,9%)	2885 (17,3%)	11760 (70,7%)	11647 (70,0%)
В конец и в середину	512 (3,1%)	970 (5,8%)	7803 (46,9%)	7759 (46,6%)

Приведенные результаты иллюстрируют неудовлетворительную работу RV-алгоритма на искаженных фразах. Скорее всего, авторы использовали подходы, разработанные для токенизации текстов с опущенными пробельными символами, что не позволило показать хорошие результаты на зашумленных фразах, которые в большей мере соответствуют реальным пользовательским паролям. Особенно хорошо это видно на словаре с двумя искажениями, где RV-алгоритм уступил более чем в 6 раз. Алгоритм RGramToken гораздо успешнее справился с искаженными примерами, хотя и уступил в задаче токенизации неискаженных фраз.

Тот факт, что вставки в середину существенно лучше обрабатываются алгоритмом RGramToken, чем вставки в конец, наглядно демонстрирует выигрыш от более полного использования информации о частотах биграмм и триграмм, что позволяет выбрать более правильные грамматические формы словарных слов.

Эксперименты с различными значениями порога отсечения по частоте встречаемости слов показали, что RV-алгоритм, в отличие от алгоритма RGramToken, чувствителен к качеству используемого словаря, поэтому использование коммерческого корпуса авторами работы [1] представляется оправданным.

В заключение приведем некоторые результаты эксперимента по токенизации на массиве реальных паролей. В качестве источника таких паролей использован известный массив аккаунтов пользователей сайта Rockyou.com, который в 2009 году в результате утечки попал в сеть Интернет [15]. Предварительно все буквы английского алфавита в паролях переведены в нижний регистр. Результаты работы RV-алгоритма и алгоритма RGramToken с порогом отсечения $v_b = 50$ приведены в таблице 5. Вставки обрамлены с помощью пар символов $> слева и <$ справа (т.е. вставка «1» представляется в виде $>1<$), слова и вставки отделены друг от друга пробелами.

Таблица 5. Примеры разбиения реальных паролей

Пароль	**Разбиение**	
	RV-алгоритм	**RGramToken**
«bedroomeyez»	bed roome yez	bedroom eye $>z<$
«crisine»	chris ine	chris in $>e<$
«girl090pink»	girl $>090<$ pink	girl $>090p<$ in $>k<$
«lovearizona»	love arizona	love arizona
«gopalloveforu»	gopal love foru	go pal love for $>u<$
«martinsac1»	martin sac $>1<$	martins a $>c1<$
«samanthaco»	sama nt hac $>o<$	samantha $>co<$
«silvestrek»	silves trek	silvestre $>k<$
«whitethouse»	white thouse	white $>t<$ house

Объективно определить лучший из двух рассмотренных алгоритмов на множестве реальных паролей не представляется возможным из-за отсутствия исходных фраз, положенных в основу пароля пользователями, поэтому здесь приводятся результаты без оценок качества.

Заключение

Алгоритм RGramToken, предложенный в данной работе для выделения слов естественного языка в реальных пользовательских паролях, позволяет эффективно производить токенизацию на больших массивах парольной информации, доступных для анализа в сети Интернет. Эффективность предложенного метода показана в сравнительных экспериментах с лучшим из известных авторам алгоритмов, описанным в работе R. Veras et al. [1].

Основными преимуществами нового алгоритма является устойчивость к умышленным искажениям исходных фраз путем добавления различных несловарных вставок, что является обычной практикой при выборе пароля. Кроме того, алгоритм RGramToken более эффективно использует информацию о частоте встречаемости слов и словосочетаний, что делает его менее чувствительным к качеству используемого словаря.

Результаты токенизации реальных парольных массивов могут быть использованы в дальнейшем для проведения новых исследований грамматического и семантического состава пользовательских паролей, строить адекватные лингвистические и теоретико-вероятностные модели парольных источников.

Последующие исследования в области выделения слов естественного языка в реальных пользовательских паролях могут быть связаны с изучением природы несловарных вставок и построением их математической модели. Важным для практического применения предложенного алгоритма представляется также формирование рабочего словаря, содержащего качественный частотный корпус и специальные словари имен собственных, известных аббревиатур и сокращений.

СПИСОК ЛИТЕРАТУРЫ:

1. Veras R., Collins C., Thorpe J. On the Semantic Patterns of Passwords and their Security Impact. NDSS. – 2014.

2. Wang D., Jian G., Wang P. Zipf's Law in Passwords. IACR Cryptology ePrint Archive. – 2014. – Т. 2014. – С. 631.

3. Тюрин К.А., Сёмин Р.В. Анализ стойкости парольных фраз на основе информационной энтропии. Известия Южного федерального университета. Технические науки. – 2015. - №5 (166). С. 18-27.

4. Марков Г.А. К вопросу об определении стойкости парольных систем. Сборник трудов Третьей всероссийской НТК «Безопасные информационные технологии». – М.: НИИ РЛ МГТУ им. Н.Э. Баумана, 2012. С. 21-23.

5. Weir M., Aggarwal S., De Medeiros B., Glodek B. Password cracking using probabilistic context-free grammars. 2009 IEEE Symposium on Security and Privacy. – IEEE, 2009. С. 391-405.

6. Houshmad S., Aggarwal S., Flood R. Next Gen PCFG Password Cracking. IEEE, 2015. С. 1776-1791.

7. Yazdi Sh. Probabilistic Context-Free Grammar Based Password Cracking: Attack, Defence and Application. FSU Libraries, 2015.

8. Van Heerden R.P., Vorster J.S. Using Markov Models to crack passwords. The 3rd International Conference on Information Warfare and Security: Peter Kiewit Institute, University of Nebraska, Omaha, USA. – 2008. С. 24-25.

9. Ma J., Yang W., Luo M., Li N. A Study of Probabilistic Password Models. 2014 IEEE Symposium on Security and Privacy. – IEEE, 2014. С. 689-704.

10. Duermuth M., Angelstorf F., Castellucia C., Perito D., Chaabane A. OMEN: Faster Password Guessing Using an Ordered Markov Enumerator. International Symposium on Engeneering Secure Software and Systems, 2015. С. 119-132.

11. Grefenstette G., Tapanainen P. What is a word, What is a sentence? Problems of Tokenization. Proceedings of the 3rd Conference on Computational Lexicography and Text Research, COMPLEX'94. – 1994. С. 79-87.

12. Jurish B., Wurzner K.-M. Word and Sentence Tokenization with Hidden Markov Models. JLCL. – 2013. – Т.28, №2. С. 61-83.

13. Jakobsson M., Dhiman M. The benefits of understanding passwords. Mobile Authentication, ser. Springer Briefs in Computer Science. – Springer New York, 2013. С. 5-24.

14. ENCOW14. [Электронный ресурс.] – URL: http://corporafromtheweb.org/encow14/ (дата обращения 30.05.2016).

15. RockYou. [Электронный ресурс.] – URL: http://wiki.skullsecurity.org/Passwords (дата обращения 30.05.2016).

REFERENCES:

[1]Veras R., Collins C., Thorpe J. On the Semantic Patterns of Passwords and their Security Impact. NDSS. – 2014.

[2]Wang D., Jian G., Wang P. Zipf's Law in Passwords. IACR Cryptology ePrint Archive. – 2014. – T. 2014. – p. 631.

[3]Turin K.A., Semin R.V. Analysis of Passphrases Resistance Based on Information Entropy. Izvestiya SFedU. Engeneering Sciences. – 2015. - №5 (166). pp. 18-27. (in Russian)

[4]Markov G.A. About password systems strength counting. Sbornik trudov Tret'ey vserossiyskoy NTK «Bezopasnye informatsionnye tehnologii». – Moscow. Bauman Moscow State Technical University, 2012. pp. 21-23. (in Russian)

[5] Weir M., Aggarwal S., De Medeiros B., Glodek B. Password cracking using probabilistic context-free grammars. 2009 IEEE Symposium on Security and Privacy. – IEEE, 2009. pp. 391-405.

[6] Houshmad S., Aggarwal S., Flood R. Next Gen PCFG Password Cracking. IEEE, 2015. pp. 1776-1791.

[7] Yazdi Sh. Probabilistic Context-Free Grammar Based Password Cracking: Attack, Defence and Application. FSU Libraries, 2015.

[8] Van Heerden R.P., Vorster J.S. Using Markov Models to crack passwords. The 3rd International Conference on Information Warfare and Security: Peter Kiewit Institute, University of Nebraska, Omaha, USA. – 2008. pp. 24-25.

[9] Ma J., Yang W., Luo M., Li N. A Study of Probabilistic Password Models. 2014 IEEE Symposium on Security and Privacy. – IEEE, 2014. pp. 689-704.

[10]Duermuth M., Angelstorf F., Castellucia C., Perito D., Chaabane A. OMEN: Faster Password Guessing Using an Ordered Markov Enumerator. International Symposium on Engeneering Secure Software and Systems, 2015. pp. 119-132.

[11]Grefenstette G., Tapanainen P. What is a word, What is a sentence? Problems of Tokenization. Proceedings of the 3rd Conference on Computational Lexicography and Text Research, COMPLEX'94. – 1994. pp. 79-87.

[12]Jurish B., Wurzner K.-M. Word and Sentence Tokenization with Hidden Markov Models. JLCL. – 2013. – T.28, №2. pp. 61-83.

[13] Jakobsson M., Dhiman M. The benefits of understanding passwords. Mobile Authentication, ser. Springer Briefs in Computer Science. – Springer New York, 2013. pp. 5-24.

[14] ENCOW14. Available at: http://corporafromtheweb.org/encow14/ (accessed 30.05.2016).

[15] RockYou. Available at: http://wiki.skullsecurity.org/Passwords (accessed 30.05.2016).

Поступила в редакцию – 20 февраля июля 2017 г. Окончательный вариант – 21 мая 2017 г.
Received – February 20, 2017. The final version – May 21, 2017.