УДК 622.276.001

М.Е. Бурлаков

Самарский национальный исследовательский университет им. С.П. Королева, Самара, Россия

ПРИМЕНЕНИЕ МЕТОДА АНАЛИЗА СООТВЕТСТВИЙ ДЛЯ ОПТИМИЗАЦИИ КОМБИНАЦИЙ АТРИБУТОВ У НАБОРОВ ДАННЫХ

На сегодняшний день развитие систем обнаружения вторжений (СОВ) целиком и полностью связано с разработкой как адаптивных, так и неадаптивных алгоритмов. Применение неадаптивных алгоритмов обеспечивает СОВ необходимую скорость работы с полностью отсутствующими ошибками первого и второго рода. С другой стороны, способность СОВ обнаруживать ранее неизвестные угрозы обеспечивается за счет наличия адаптивного компонента. Динамичное развитие адаптивных алгоритмов обеспечивается путем создания новых методов, связанных с искусственными иммунными системами, искусственными нейронными сетями, генетическими алгоритмами и т.д. С целью проверки качества разработанных методов и алгоритмов используются специализированные наборы данных (dataset) – множество запросов, представленных в специальном виде, которые передаются между системами на различных уровнях модели OSI. Примерами подобных наборов данных могут послужить KDD99, NSL-KDD DataSet, ADFA Intrusion Detection Datasets, MACCDC, ISTS, ITOC, DEFCON CTF и т.д. Каждый набор данных содержит в себе два подмножества – тренировочное (для обучения адаптивного алгоритма) и тестовое (для проверки качества его разработки). Данные, описывающие запросы, максимальны с точки зрения их задания, поэтому перед специалистами стоит вопрос оптимизации атрибутного множества этих запросов таким образом, чтобы повысить эффективность и скорость обучения алгоритма со снижением количества либо линейной комбинации атрибутов. В статье рассматривается возможность применения метода анализа соответствий с внедренным механизмом сингулярного разложения матричного представления наборов данных для уменьшения количества линейных комбинаций атрибутов.

Ключевые слова: метод анализа соответствий, сингулярное разложение матриц, наборы данных, линейные комбинации атрибутов.

M.E. Burlakov

Samara National Research University named after S.P. Korolev, Samara, Russian Federation

APPLICATION THE METHOD OF CORRESPONDENCE ANALYSIS TO OPTIMIZE COMBINATIONS OF ATTRIBUTES FROM DATASETS

The development of intrusion detection systems (IDS) is depended on the development both adaptive and non-adaptive algorithms today. On the one hand, the usage of non-adaptive algorithms provides the required speed of operation with the low rates false positive and false negative errors for

the IDS. On the other hand, the ability of IDS to detect unknown threats is provided by the availability of the adaptive components. The intensive development of adaptive algorithms is provided by creating new methods associated with artificial immune systems, artificial neural networks, genetic algorithms, etc. The quality of new developed methods and algorithms are tested through to specialized datasets. The datasets are sets of queries which represented in a special form that are transmitted between systems in different levels of the OSI. There are number of such datasets such as: KDD99, NSL-KDD DataSet, ADFA Intrusion Detection Datasets, MACCDC, ISTS, ITOC, DEFCON CTF, etc. Each dataset contains training and test (work) subsets. The training set is need for the learning the adaptive algorithm. The work subset is need for the testing the quality of developed methods and algorithms. The datasets which described by the queries have full structure in terms of their entities. That's why the specialists in security science are faced with the problem of optimization the attribute set in datasets. This optimization allows to increase the efficiency of algorithm learning and to decrease the number of its linear combination. The article considers the possibility of using the correspondence analysis method with the embedded mechanism of singular decomposition of the matrix representation of datasets to reduce the number of linear combinations its attributes.

Keywords: the method of correspondence analysis, singular value decomposition of matrices, datasets, linear combination of attributes.

Введение. В настоящее время системы обнаружения вторжений (СОВ) широко представлены на рынке противодействия угрозам и уязвимостям, возникающим на всех уровнях семиуровневой модели OSI. Любая СОВ состоит из алгоритмов двух типов: адаптивных и неадаптивных, каждый из которых имеет свои плюсы и минусы при обнаружении соответствующих угроз и аномалий. Если неадаптивные алгоритмы детерминированы и в основном направлены на детектирование ранее встречавшихся угроз с применением сигнатурных методик, то адаптивные направлены на обнаружение ранее не встречавшихся угроз. К адаптивным алгоритмам можно отнести искусственные нейронные сети (ИНС), искусственные иммунные системы (ИМС), генетические алгоритмы (ГА) и т.д. [1].

Работа любого адаптивного алгоритма заключается в реализации двух операций: обучение алгоритма (формирование базы знаний) и этап классификации (режим боевой эксплуатации).

Исходя из большого количества как уже представленных, так и новых разрабатываемых адаптивных и неадаптивных методов в СОВ, перед специалистами по информационной безопасности всегда стоит вопрос, как более грамотно и эффективно осуществить процесс обучения алгоритма и насколько его дальнейшая работа будет эффективна.

Если с неадаптивными методами все понятно и вопрос решается путем регулярного обновления баз данных и пополнения сигнатур и шаблонов аномальных запросов (запросов, реализующих атаку на какую-либо систему), то с адаптивными методами вопрос стоит куда актуальнее.

В любом адаптивном алгоритме есть понятие машинного обучения, которое понимается как «компьютерная программа, которая обучается на опыте *E* относительно некоторой задачи *T* и меры эффективности *P*, если эффективность выполнения ею задачи *T*, будучи измеренной как *P*, повышается с опытом *E*» [2]. Другими словами, системы машинного обучения позволяют разрабатывать алгоритмы, в основе которых лежат уже имеющиеся знания о проблеме, и применять их к новым, ранее неизвестным формам этой проблемы.

1. Наборы данных (Datasets). Проектирование систем машинного обучения требует корректной формализации задачи, а именно представления объектов реального мира векторами признаков в пространстве R^n, где каждый вектор олицетворяет один обучающий пример, а каждый элемент вектора соответствует определенному свойству (признаку) изучаемого класса объектов (запросов, *requests*). Формализация проблемы, выбор технологии (будь то ИНС, ИМС, ГА и т.д.) и алгоритма обучения являются нетривиальными задачами. Универсальных алгоритмов на данный момент не существует, а потому системы машинного обучения представляют интерес для научного исследования.

Поэтому с целью обучения и дальнейшей проверки эффективности работы систем машинного обучения для того или иного протокола используются соответствующие наборы данных (*Datasets*). Как правило, эти наборы данных имеют следующую структуру:

– множество признаков объектов – конечный набор признаков запросов, обусловленный работой в рамках определенного протокола, например, для протокола TCP это будут атрибуты пакета – элементы заголовка и содержимого, временнЫе параметры;

– обучающая выборка – набор данных, в которых есть как аномальные, так и неаномальные запросы, с помощью которых осуществляется тренировка системы;

– тестовая выборка – как правило, множество, бОльшее обучающего, содержащее в себе аномальные и неаномальные запросы, с помощью которого осуществляется проверка работы обученной системы.

Многие наборы являются по факту стандартом проверки качества работы адаптивных алгоритмов. В рамках работы протоколов *TCP*, *UDP* и *ICMP* можно выделить следующие наиболее востребованные наборы данных [3]: *KDD99, NSL-KDD DataSet, ADFA Intrusion Detection Datasets, MACCDC, ISTS , ITOC, DEFCON CTF, Contagio*

Malware Dump, FIRST 2015, 4SICS, ITOC CDX 2009, Enron Email, Trec spam, Lingspam, PU dataset и т.д.

Каждый из этих наборов применим для тестирования адаптивных компонент, работающих в рамках СОВ для различных протоколов. Каждый из наборов несет свои типы уязвимостей и технологии их эксплуатации. В качестве наборов множеств можно выделить следующие наиболее популярные и зарекомендовавшие себя [4]: *NSL-KDD DataSet* (протокол работы *TCP*,*UDP* и *ICMP*), *CSIC* 2010 (протокол работы *HTTP*/1.1), *Enron Dataset* (протокол работы *SMTP*). Проведем анализ экземпляров аномальных и неаномальных запросов в наборах данных на примере *NSL-KDD DataSet*.

2. Анализ экземпляров аномальных и неаномальных запросов на примере набора данных *NSL-KDD Dataset*. Как было указано выше, все наборы данных состоят из минимум двух подмножеств:

– обучающая выборка (*training dataset*) – на нем осуществляется подбор параметров с целью получения максимального результата в процессе формирование адаптивного алгоритма;

– тестовая выборка (*testing dataset*) – на нем осуществляется проверка качества обучения адаптивного алгоритма.

В каждом из этих множеств метрика объектов одинакова. Рассмотрим формальные модели запросов (метрики) на примере набора данных *NSL-KDD Dataset*. Данный набор является развитием множества *KDD99* – де-факто первый стандарт в области *Dataset* для СОВ, созданный для проведения сравнительного тестирования адаптивных алгоритмов [5, 6].

В силу развития СОВ в реализации *KDD99* было обнаружено большое количество недостатков [7], которые впоследствии были устранены путем реализации *NSL-KDD Dataset*. *NSL-KDD DataSet* содержит в себе ряд преимуществ по сравнению с *KDD99*, среди которых можно выделить такие, как:

– удаление ряда записей с целью устранения влияния частотных характеристик (избыточность, дублирование) на адаптивный механизм;

– более продуманный подход к формированию тестовых и обучающих множеств и т.д.

Состав и описание набора *NSL-KDD Dataset* представлены в табл. 1 и 2.

Таблица 1

Набор данных *NSL-KDD Dataset*

№	Множество	Описание
1	*KDDTrain+*	Обучающая выборка с метками атаки и уровнем сложности
2	*KDDTrain+*20 %	20 % подмножество *KDDTrain+*
3	*KDDTest+*	Проверочное множество с метками атаки и уровнем сложности
4	*KDDTest-21*	Множество из *KDDTest+*. Не включает в себя записи уровней атак, которые выше значения 21

Таблица 2

Распределение записей по множествам в *NSL-KDD Dataset*

Наименование множества	Количество					
	Записей	Нормальных запросов	DoS	Probe	U2R	R2L
KDDTrain+20%	25192	13449	9234	2289	11	209
		53,39 %	36,65 %	9,09 %	0,04 %	0,83 %
KDDTrain+	125973	67343	45927	11656	52	995
		53,46 %	36,46 %	9,25 %	0,04 %	0,79 %
KDDTest+	22544	9711	7458	2421	200	2754
		43,08 %	33,08 %	10,74 %	0,89 %	12,22 %

Объекты в *NSL-KDD Dataset* представляют собой соединения – последовательность (*TCP, UDP, ICMP*)-пакетов, зафиксированную в определенный промежуток времени, в которую заключен поток данных от *IP*-адреса источника к *IP*-адресу назначения в соответствии с некоторым определенным протоколом [8].

Набор данных содержит 4 категории угроз:

– *Denial of Service (dos)*. Набор атак, в которых злоумышленник ограничивает доступ верифицированным пользователям к конкретному сервису через определенный протокол (*Back, Land, Neptune, Pod, Smurf,Teardrop,Apache2, Udpstorm, Processtable, Worm*);

– *Remote to Local (r2l)*. Набор атак, в которых злоумышленник пытается получить доступ извне к локальной машине пользователя (*Guess_Password, Ftp_write, Imap, Phf, Multihop, Warezmaster, Warezclient, Spy, Xlock, Xsnoop, Snmpguess, Snmpgetattack, Httptunnel, Sendmail, Named*);

– *User to Root (u2r)*. Набор атак, в которых злоумышленник, имея доступ к машине жертвы, пытается получить права более привилегированного пользователя (*Buffer_overflow, Loadmodule, Rootkit, Perl, Sqlattack, Xterm, Ps*);

– *Probe*. Набор атак, в которых злоумышленник пытается получить сведения об инфраструктуре пользователя (*Satan, Ipsweep, Nmap, Portsweep, Mscan, Saint*).

Хотя размерность вектора (количество атрибутов в запросе) в *NSL-KDD Dataset* равняется 43, для реальной работы используется 41 атрибут. 42-й атрибут представляет категорию угрозы, а 43-й – сложность реализации атаки (от самого простого к самому сложному). Таким образом, формальная модель запроса для набора данных *NSL-KDD Dataset* представляет из себя вектор размерности 41.

Полный перечень атрибутов запросов *NSL-KDD Dataset* представлен в [5]. Рассчитаем влияние атрибутов в запросах из набора данных на представление его в виде аномального запроса.

3. Оценка влияния атрибутов на конечный результат работы адаптивного алгоритма. У каждого набора данных имеется свое множество атрибутов, которое участвует в процессе обучения адаптивного алгоритма. Однако имеет смысл поднять вопросы: все ли атрибуты в объекте набора запросов являются критически важными и могут активно влиять на конечный результат при проведении процесса обучения и тренировки адаптивных алгоритмов? Имеется ли основание для снижения количества атрибутов, которое преследует за собой следующие цели:

1) конечное снижение времени обучения адаптивного алгоритма с минимальными потерями в качестве;

2) конечное снижение затрат машинных ресурсов на всех этапах формирования базы знаний адаптивного алгоритма (меньше атрибутов – меньше затраченных машинных ресурсов);

3) конечное увеличение скорости принятия решения за счет уменьшения времени доступа к базе знаний адаптивного алгоритма;

4) нахождение аномальных запросов, объединенных в рамках конечного набора групп по обоснованным признакам (подклассы угроз, локальные классы угроз, кластеризация угроз).

Следовательно, вопрос состоит в том, какие атрибуты оказывают наибольшее влияние (имеют наибольший вес) на процесс обучения и тренировки, а какие дают наименьший вклад.

В качестве метода, используемого для снижения количества атрибутов в объекте запроса, воспользуемся механизмом множественного анализа соответствий в силу большого количества применений в области исследования и оценки характеристических параметров

и объектов [9]. Выбор данного механизма обусловлен наличием характеристических атрибутов, позволяющих рассматривать множество кортежей (векторов) запросов в номинальных шкалах, а также наличием механизма снижения размерности векторов путем преобразования количественных характеристик в качественные.

4. Механизм анализа соответствий. Понятия и определения. Анализ соответствий относится к методам предварительного, или разведочного (*exploritary*) анализа данных. Данный класс методов предназначен для исследования структуры данных, результирующие значения которых относятся к конечной выборке. Они эффективно используются как на предварительном этапе изучения данных, так и для интерпретации результатов моделирования [10]. Разведочные методы призваны порождать гипотезы о распределении и взаимосвязях данных, после чего – на следующем этапе – полученные гипотезы могут тестироваться подтверждающими методами.

Анализ соответствий не предъявляет никаких требований к данным; он может быть применен к любой прямоугольной матрице, составленной из векторов запросов. Единственным ограничением является неотрицательность чисел в ячейках матрицы [11]. Отличительной чертой анализа соответствий является способность преобразования абсолютных значений данных в номинальные с последующим введением метрики. Выделяют два типа анализа соответствий: простой (*simple correspondence analysis, CA*) и множественный (*multiple correspondence analysis, MA*). В настоящей работе в силу наличия более двух переменных используется множественный анализ.

В анализе соответствий существуют три базовых понятия: **профили** (*profiles*), **веса** (*masses*) и **расстояния** (*chi-squared distances*). Здесь и далее в качестве демонстрации работы механизма будут приводиться расчеты на примере аномальных запросов множества *NSL-KDD Dataset.*

4.1. Профили. Под исходным профилем объекта d в анализе соответствий понимается кортеж данных с представлением атрибутов в числовых величинах:

$$d_i = (d_1^i, d_2^i, ..., d_m^i),\ d_j^i = \{1, ..., n_j\},\ n_j \in N. \quad (1)$$

Набор кортежей представляется в виде матрицы $D(n,m)$ размерностью $n \times m$, называемой **матрицей соответствий,** где элементы d^i ее строки, n – количество элементов d, m – количество атрибутов

у элементов *d*. Под **индикаторной матрицей** *Z* понимается представление матрицы *D* в бинарном виде. Процесс бинаризации описывается последовательностью следующих шагов:

1) выделяется строка-кортеж $d^i = (d_1^i, d_2^i, ..., d_m^i)$, $d_j^i = k$, $k \in \{1, ..., n_j\}$ – некоторая числовая величина;

2) d_j^i представляется в виде бинарной строки $z_j^i = (0, ..., 1_j, ..., 0)$, где длина вектора z_j^i равна максимальному значению d_j^i по всем кортежам *d*.

Рассмотрим процесс бинаризации на частном примере.

Пусть дана матрица $D = \{d^1, d^2\}$, состоящая из двух элементов d^1 и d^2, где $d^1 = (1, 2)$, $d^2 = (2, 5)$, здесь первый атрибут (первый элемент в каждом векторе d^i) имеет соответственно минимальное и максимальное значение 1 и 2, второй 1 и 5. Таким образом, после применения процесса бинаризации получим, что все первые элементы каждого вектора d^i будут иметь длину 2 (максимальное значение первого атрибута), вторые элементы, соответственно 5 (максимальное значение второго атрибута). Таким образом:

$$D(2,2) = \begin{pmatrix} d^1 \\ d^2 \end{pmatrix} = \begin{pmatrix} 1_{d_1^1} & 2_{d_2^1} \\ 2_{d_1^2} & 5_{d_2^2} \end{pmatrix} \rightarrow Z(2,7) = \begin{pmatrix} 0100010 \\ 1010000 \end{pmatrix}. \qquad (2)$$

Под **маргинальной суммой** *i*-й строки (M_s) или *j*-го столбца (M_r) понимается сумма элементов строки (3)/столбца (4):

$$M_s^i = \sum_{k=1}^m d_k^i, \qquad (3)$$

$$M_r^j = \sum_{k=1}^n d_j^k. \qquad (4)$$

Для бинарной матрицы *Z* значения маргинальных сумм равно сумме единиц в соответствующих строках/столбцах.

Под **общей маргинальной суммой** ***M*** понимается величина, равная сумме маргинальных сумм по всем строкам и столбцам (5):

$$M = \sum_{i=1}^n \sum_{j=1}^m d_j^i. \qquad (5)$$

Под **стандартным кортежем** в анализе соответствий понимается кортеж данных с нормированными (деленными на общую маргиналь-

ную сумму) значениями. Здесь и далее под кортежем объекта будет пониматься стандартный кортеж объекта.

В табл. 3 приведен пример представления кортежей объектов в абсолютных шкалах, где в качестве объектов были взяты 3 объекта *TCP* и 3 объекта *UDP* аномальных запросов по 4 атрибута с соответствующими значениями (из тестового множества *NLS-KDD*), где числовыми значениями обозначены номинальные характеристики:

Таблица 3

Запросы в абсолютных шкалах

№ п/п	Флаги				M_r
	access files 1 – меньше 5 2 – от 5 до 10 3 – более 10	root shell 1 – нет 2 – да	srv count 1 – меньше 10 2 – от 10 до 50 3 – свыше 50	service 1 – ftp, 2 – telnet 3 – smtp, 4 – sftp	
1	1	2	2	1	6
2	3	2	1	3	9
3	1	1	1	4	7
4	2	1	3	3	9
5	3	2	3	2	10
6	3	1	2	1	8
M_s	13	7	12	14	$M = 49$

Таблица 4

Запросы в бинарном представлении

№ п/п	Флаги												M_r
	access files 1 – меньше 5 2 – от 5 до 10 3 – более 10			root shell 1 – нет 2 – да		srv count 1 – меньше 10 2 – от 10 до 50 3 – свыше 50			service 1 – ftp, 2 – telnet 3 – smtp, 4 – sftp				
1	0	0	1	1	0	0	1	0	0	0	0	1	4
2	1	0	0	1	0	0	0	1	0	1	0	0	4
3	0	0	1	0	1	0	0	1	1	0	0	0	4
4	0	1	0	0	1	1	0	0	0	1	0	0	4
5	1	0	0	1	0	1	0	0	0	0	1	0	4
6	1	0	0	0	1	0	1	0	0	0	0	1	4
M_s	3	1	2	3	3	2	2	2	1	*2*	*1*	*2*	$M = 24$

Для бинарных значений маргинальные суммы по всем строкам одинаковы.

4.2. Веса. Под **весом стандартного картежа** (строк ω_i /столбцов ω_j) понимается значение маргинальных сумм по строкам/столбцам, деленных на общую маргинальную сумму. В анализе соответствий значения весов определяются следующими соотношениями:

$$\omega_i = \frac{M_s^i}{M},\ \omega_j = \frac{M_r^j}{M}. \tag{6}$$

Под **средним кортежем строки** d^0 понимается кортеж, где атрибуты есть средние значения кортежей по строкам:

$$d^0 = \left\{ \frac{\sum_j d_1^j}{M}, \frac{\sum_j d_2^j}{M}, ..., \frac{\sum_j d_m^j}{M} \right\}. \tag{7}$$

Геометрически средний кортеж – аналог точки, лежащей в центре облака точек, представленных остальными кортежами. Если кортеж сильно отличается от среднего, то соответствующая ему точка будет находиться далеко от центра, и наоборот.

4.3. Метрика. В анализе соответствий в качестве формулы для расчета метрики применяется *взвешенный* аналог евклидова расстояния, где *весом* служит величина, обратная соответствующему элементу среднего кортежа [11]:

$$\rho(d^i, d^{i'}) = \sqrt{\sum_{j=1}^m \frac{(d_j^i - d_j^{i'})^2}{|d_j^0|}}, \tag{8}$$

где $\rho(d^i, d^{i'})$ – взвешенное евклидово расстояние между запросами d^i и $d^{i'}$; d^i, $d^{i'}$ – элементы кортежей; d_j^0 – элементы среднего кортежа строки [12].

Если элементы кортежей запросов имеют бинарный вид (здесь и далее применяется именно это соотношение, так как работа ведется в рамках бинарной матрицы) используется следующая функция расстояния:

$$\rho(d^i, d^{i'})_{bin} = \sum_{j=1}^m |d_j^i - d_j^{i'}|. \tag{9}$$

Для бинарного представления расстояние есть коэффициент несовпадений, представляющий из себя сумму количества позиций, в которых элементы не совпадают (метрика Хаусдорфа). Используя обозначенные понятия и определения, рассмотрим задачу снижения количества комбинаций атрибутов в наборах данных.

5. Снижение количества комбинаций атрибутов запросов. Задача снижения количества комбинаций атрибутов, описанная выше, есть

задача снижения размерности кортежей, которые составляют аномальные запросы. В множественном анализе соответствий задача снижения размерности сводится к поиску гиперплоскости, которая бы наиболее точно отражала расстояния между точками. Фактически эта задача эквивалентна задаче поиска гиперплоскости меньшей размерности, которая была бы в некотором смысле ближе одновременно ко всем точкам. Близость определяется методом взвешенных наименьших квадратов.

В множественном анализе соответствий снижение размерности производится за счет разложения индикаторной матрицы методом сингулярного разложения матриц (*singular value decomposition, SVD*). Сингулярное разложение – декомпозиция вещественной матрицы с целью ее приведения к каноническому виду [13]. Сингулярное разложение позволяет найти ортогональные базисы различных векторных пространств разлагаемой матрицы и рассчитывать ранг текущей матрицы.

Сингулярным разложением матрицы $A_{(m\times n)}$ называется представление, заданное в виде:

$$A_{(m\times n)} = U_{(m\times m)}\mathbf{\Lambda}_{(m\times n)}V^{T}_{(n\times n)}, \tag{10}$$

где для матриц U и V выполняются условие:

$$U^{T}_{(m\times m)}U_{(m\times m)} = U_{(m\times m)}U^{T}_{(m\times m)} = E, \tag{11}$$

$$V^{T}_{(n\times n)}V_{(n\times n)} = V_{(n\times n)}V^{T}_{(n\times n)} = E, \tag{12}$$

где E – единичная матрица.

Матрица $\mathbf{\Lambda}$ – диагональная, с элементами, удовлетворяющими условию:

$$\lambda_1 \geq \lambda_2 \geq ...\lambda_r \geq \lambda_{r+1} = ... = \lambda_n = 0. \tag{13}$$

Существует множество алгоритмов и их программных реализаций сингулярного разложения матриц [14]. В рамках данной работы применены следующие алгоритмы:

– алгоритм, реализованный в библиотеке *Lapack* на языке программирования *Python* (для квадратной невырожденной матрицы, где *rank A = n*, и для случая *rank A(m×n) = r*, где *r = min*(*m,n*). Общее описание алгоритма для матриц размерности *m×n* берется из работы [15];

– алгоритм, реализованный в библиотеке *SVDPACK* на языке программирования *C* (для случая *rank A(m×n) = r,* где *r < min(m,n)*. Общее описание алгоритма для матриц размерности *m×n* берется из работы [16].

6. Алгоритм реализации *SVD* метода. Сингулярное разложение матриц в рамках практической реализации, выполненной на библиотеке

Lapack, язык программирования *Python* (реализовано через простой итерационный алгоритм (или в некоторых реализациях через метод Якоби для собственных значений)), и *SVDPACK*, язык программирования *C* (реализовано через алгоритм Лацоша). Общая идея итерационного алгоритма представлена ниже.

В качестве базовой процедуры выступает операция поиска наилучшего приближения произвольной матрицы $A = (a_{ij})$ размерностью $m \times n$ матрицей P_1 вида $s \otimes r = (s_i r_j)$, (где s – m-мерный вектор, а r – n-мерный вектор) методом наименьших квадратов:

$$F(s,r) = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}(a_{ij} - s_i r_j)^2 \to \min. \qquad (14)$$

Решение этой задачи дается последовательными итерациями по явным формулам. При фиксированном векторе $r = (r_j)$ значения $s = (s_i)$, доставляющие минимум форме $F(s,r)$, однозначно и явно определяются из равенств $\frac{\partial F}{\partial s_i} = 0$:

$$\frac{\partial F}{\partial s_i} = -\sum_{j=1}^{n}(a_{ij} - s_i r_j) r_j = 0;\ s_i = \frac{\sum_{j=1}^{n} a_{ij} r_j}{\sum_{j=1}^{n} a_j^2}. \qquad (15)$$

Аналогично при фиксированном векторе $s = (\underline{s_i})$ определяются значения $r = (r_j)$:

$$r_j = \frac{\sum_{i=1}^{m} s_i a_{ij}}{\sum_{i=1}^{m} b_j^2}. \qquad (16)$$

В качестве начального приближения вектора r берется случайный вектор единичной длины, вычисляется вектор s, далее для этого вектора s вычисляется вектор r и т.д. Каждый шаг уменьшает значение $F(s, r)$. В качестве критерия остановки используется малость относительного уменьшения значения минимизируемого функционала $F(s, r)$ за шаг итерации $\Delta F/F$ или малость самого значения F.

В результате для матрицы $A = (a_{ij})$ получается наилучшее приближение матрицей P_1 вида $s^1 \otimes r^1 = (s_i^1 r_j^1)$ (здесь верхний индекс – номер итерации). Далее из матрицы A вычитается полученная матрица P_1, и для полученной матрицы $A_1 = A - P_1$ вновь ищется наилучшее приближение P_2 этого же вида и т.д., пока норма A_k не станет достаточно

малой. Таким образом, имеем итерационную процедуру разложения матрицы A в виде суммы матриц ранга 1:

$$A = P_1 + P_2 + ... + P_q \, (P_l = s^l \otimes r^l). \qquad (17)$$

Полагается, $\sigma_l = |s^l||r^l|$ и нормируются векторы s^l, r^l: s^l: = $s^l/|s^l|$; r^l:= $r^l/|r^l|$. После выполнения операции получается аппроксимация сингулярных чисел σ_l и сингулярных векторов (правых – r^l и левых – s^l).

Основная идея применения алгоритма Лацоша описана в работе [17]. Реализация сингулярного разложения разбивается на три непересекающиеся подпрограммы:

1. Когда матрица A квадратная, т.е. *rank* $A = n$ (функция *svd_math_1*);

2. Когда матрица A прямоугольная, *rank* $A(m \times n) = r$, где $r = \min(m, n)$ (функция *svd_math_2*);

3. Когда матрица *rank* $A(m \times n) = r$, где $r < \min(m, n)$ (функция *svd_math_3*).

В качестве функции расчета ранга использовался комплекс *NumPy v.1.13* (язык программирования *Python*). Здесь и далее вызов функции *rank*(*A*) подразумевает задействование данного комплекса. Общая схема вызова методов представлена на рис. 1.

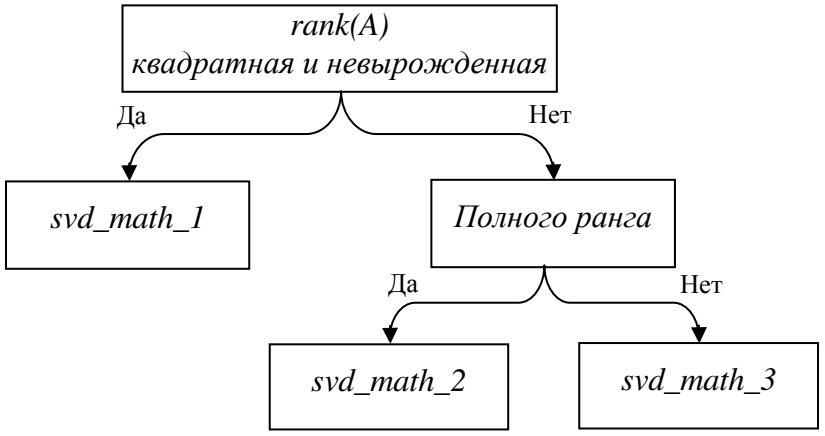

Рис. 1. Алгоритм выбора метода сингулярного разложения матрицы A в зависимости от ее ранга

В первом случае реализация сводится к реализации алгоритма сингулярного разложения для квадратной матрицы размерностью $n \times n$. Псевдокод для этого метода имеет вид:

```
% Математический метод расчета SVD для квадратной матрицы
function [U, S, V] = svd_math_1(A)
%Вход: квадратная матрица A
%Выход: U, V – унитарные матрицы
          S – диагональная матрица
% U, S, V рассчитываются, исходя из дальнейшей выполнимости соот-
ношения A = U*S*V^T
       [m,n] = size(A);    %ранг матрицы
       r = rank (A);%расчета ранга матрицы
       if m ~= n || r ~= n %если матрица неквадратная, то оста-
навливаем алгоритм
              error('Матрица должна быть квадратной')
       end
       B = A' * A ; % вычисляется симметричная положительная оп-
ределенная матрица B
       [V,L] = eig(B);     %рассчитывается V и L=S^2  (простой
итерационный алгоритм)
       S= sqrt(L) ; %корень квадратный из матрицы S^2
       U = A*V/S;   % расчет матрицы U
end;
```

Для второго случая реализация выглядит иначе:

```
% Математический метод расчета SVD для прямоугольной матрицы в
случае, если rank=min(m,n)
function [U, S, V] = svd_math_2(A)
%Вход: прямоугольная матрица  A, где rank(A)=min(m,n)
%Выход: U, V – унитарные матрицы
          S – диагональная матрица
% U, S, V рассчитываются, исходя из дальнейшей выполнимости соот-
ношения A = U*S*V^T
       [m,n] = size(A);    %ранг матрицы
       r = rank (A);%расчета ранга матрицы
       if r ~= min(m,n) %если не выполняется условие, то останав-
ливаем алгоритм
              error('rank(A) должен равняться min(m,n)')
       end
       % вычисляется симметричная положительная определенная мат-
рица B
       if r == n
              B = A' * A ;
              [V,L] = eig(B);%рассчитывается V и L=S^2 (простой
итерационный алгоритм)
              S= sqrt(L) ; %корень квадратный из матрицы S^2
              U = A*V/S;   % расчет матрицы U
       else   % r == m
              B = A * A' ;
              [U,L] = eig(B);     %рассчитывается U и L=S^2  (про-
стой итерационный алгоритм)
              S= sqrt(L) ; %корень квадратный из матрицы S^2
              V = (S/U' * A)';    %рассчитываем матрицу V
       end
end;
```

Для реализации функции *svd_math_3* двусторонний алгоритм вращения Якоби не работает в силу неприменимости для вырожденных матриц. Для этого применяется алгоритм Ланцоша [18].

Реализация третьего случая принимает вид, эквивалентный алгоритму *svd_math_2*, с той лишь разницей, что для расчета функции *eig*(*B*) используется обозначенный алгоритм [19].

```
% Математический метод расчета SVD для вырожденной матрицы
rank<min(m,n)
function [U, S, V] = svd_math_3(A)
%Вход: прямоугольная матрица  A, где rank(A)<min(m,n)
%Выход: U, V – унитарные матрицы
          S – диагональная матрица
% U, S, V рассчитываются, исходя из дальнейшей выполнимости соот-
ношения A = U*S*V^T
       r = rank (A);%расчета ранга матрицы
       % вычисляется симметричная положительная определенная мат-
рица B
       B = A' * A ;
       %если расчет идет через матрицу V
              [V,L] = eig(B);%рассчитывается V и L=S^2 (алгоритм
Ланцоша)
              S= sqrt(L) ; %корень квадратный из матрицы S^2
              U = A*V/S;    % расчет матрицы U
       %если расчет идет через матрицу U
              [U,L] = eig(B^-1);    %рассчитывается U и L=S^2  (ал-
горитм Ланцоша)
              S= sqrt(L) ; %корень квадратный из матрицы S^2
              V = (S/U' * A)';    %рассчитываем матрицу V
       end
end;
```

Максимальная точность вычисления сингулярных чисел составляет 10^{-14} [20].

Таким образом, если и возникают погрешности, то они незначительны и ими можно пренебречь.

7. Расчет данных на примере набора *NSL-KDD Dataset.* Применяя выбранную методику расчета влияния атрибутов через механизм анализа соответствий для множества *NSL-KDD Dataset*, были получены следующие параметры (табл. 5), а также распределение комбинаций атрибутов (рис. 2 и 3).

Таблица 5

Соотношение параметров в реализации программы и атрибутов множества *NSL-KDD Dataset*

Па-ра-метр	Атрибут	Пример значения	Пара-ра-метр	Атрибут	Пример значе-ния
p1	duration	124	p2	protocol type	icmp
p3	service	ftp_data	p4	flag	SF
p5	source bytes	232	p6	destination bytes	8153
p7	land	0	p8	wrong fragment	1
p9	urgent	1	p10	hot	0
p11	failed logins	1	p12	logged in	1
p13	compromised	0	p14	root shell	0
p15	su attempted	0	p16	root	0
p17	file creations	0	p18	shells	0
p19	access files	1	p20	outbound cmds	0
p21	is hot login	1	p22	is guest login	1
p23	count	123	p24	srv count	32
p25	serror rate	0.20	p26	srv serror rate	0.11
p27	rerror rate	1.00	p28	srv rerror rate	0.00
p29	same srv rate	0.08	p30	diff srv rate	0.15
p31	srv diff host rate	0.43	p32	dst host count	255
p33	dst host srv count	26	p34	dst host same srv rate	0.17
p35	dst host diff srv rate	0.03	p36	dst host same src port rate	0.12
p37	dst host srv diff host rate	0.04	p38	dst host serror rate	0.03
p39	dst host srvb serror rate	1.00	p40	dst host rerror rate	0.01
p41	dst host srv rerror rate	0.57			

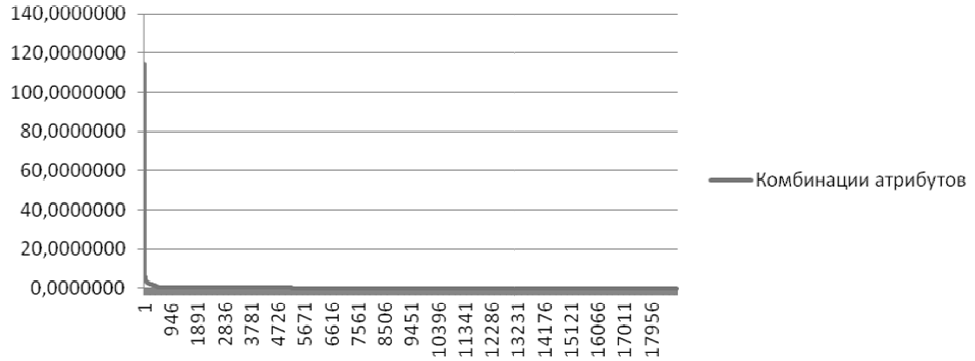

Рис. 2. Распределение значений по конечным атрибутам

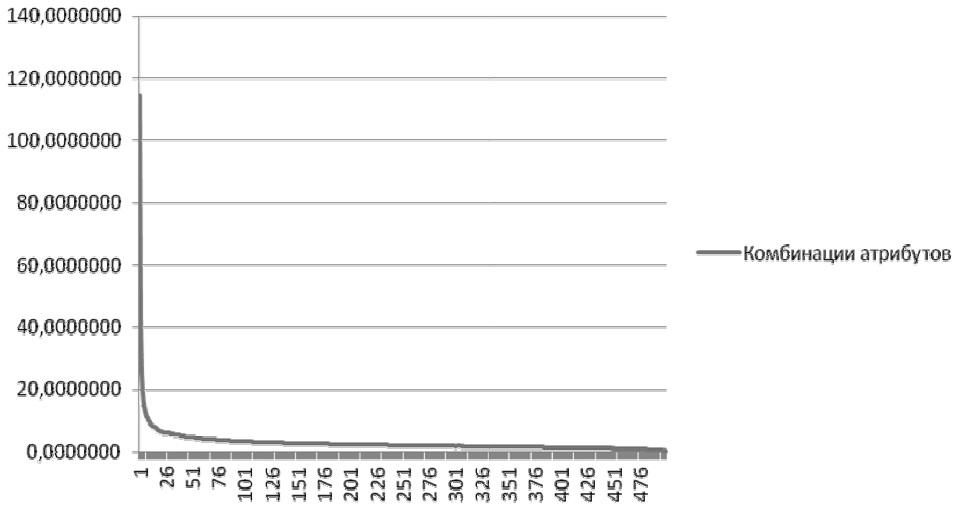

Рис. 3. Распределение значений по конечным атрибутам (сокращенное до 500)

Общая таблица распределений вкладов комбинаций атрибутов в аномальные запросы представлена ниже (первые 30 значений λ_i из соотношения 6).

Таблица 6

Таблица элементов λ_i

114,518829	42,444582	25,260859	18,804849	15,231797	13,984953
11,2199130	10,9281900	10,0919070	9,4598370	8,7519740	8,5520310
8,2012420	8,0507690	7,8996570	7,6684320	7,3561290	6,9731140
6,7874970	6,6498500	6,5873480	6,4644990	6,3220350	6,2605470
6,2120660	6,1404730	6,1255670	6,0261060	5,9383530	5,8329650

Первым критерием рассмотрения комбинаций атрибутов выступает параметр «Ненулевое ограничение *SVD*», который означает ранг полученной сингулярной матрицы с учетом погрешности вычислений при реализации [21].

Для определения оптимального числа атрибутов применяется критерий каменистой осыпи, который заключается в поиске точки, где убывание собственных значений замедляется наиболее сильно [22].

Применение метода обеспечивает 97%-процентное покрытие множества комбинаций атрибутов аномальных запросов за счет выделения 48 % наиболее значимых факторов (табл. 7).

Таблица 7

Атрибуты согласно выбранным методам

Метод	Количество комбинаций атрибутов	% от общего числа атрибутов	% общего покрытия
Ненулевое ограничение *SVD*	12 136	64 %	100 %
Метод каменистой осыпи	9 158	48 %	97 %

Выводы. Таким образом, рассмотрена формализация задачи обнаружения аномальных запросов, где определены два класса запросов: класс неаномальных запросов и класс аномальных запросов, а также предложен к рассмотрению наиболее популярный набор данных для протоколов TCP/UDP/ICMP – *NSL-KDD Dataset.* Подробно рассмотрены формальные модели запросов выбранных протоколов, проанализирован метод анализа соответствий применительно к задаче снижения количества аномальных факторов нагрузки в запросах, осуществлен переход от количественных к качественным характеристикам с введением метрики с использованием механизма анализа соответствий. Проведена оптимизация количества атрибутов у наборов данных и дана оценка влияния наборов атрибутов на формирование аномальной характеристики запроса с применением подхода ненулевого ограничения SVD и метода каменистой осыпи.

Библиографический список

1. Burlakov M.E. Research the dynamic of author activities in threats through to public and private sources // Информационные технологии и нанотехнологии: сб. тр. III Междунар. конф. и молодежной школы. – Самара: Новая техника, 2017. – P. 958–961.

2. Saul L.K. Advances in Neural Information Processing Systems // MIT Press. – 2005. – 641 p.

3. Intrusion detection evaluation dataset [Электронный ресурс] // University of New Brunswick. – 2017. – Вып. 1. – URL: http://www.unb.ca/cic/research/datasets/ids.html (дата обращения: 07.08.2017).

4. Al-Hamami A.H. Handbook of Research on Threat Detection and Countermeasures in Network Security // IGI Global. – 2014. – 450 p.

5. Levin I. KDD-99 Classifier Learning Contest // LLSoft's Results Overview. – SIGKDD Explorations. – 2010. – P. 67–75.

6. Lippmann R.P. Evaluating Intrusion Detection Systems: The 1998 DARPA off-line intrusion detection evaluation // DARPA. – 2000. – P. 10–35.

7. Tavallaee M. A Detailed Analysis of the KDD CUP 99 Data Set // Submitted to Second IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA). – 2009. – P. 7–42.

8. Knowledge discovery in databases DARPA archive [Электронный ресурс] // University of California. – 2017. – Вып. 1. – URL: http://www.kdd.ics.uci.edu/databases/kddcup99/task.html (дата обращения: 29.01. 2017).

9. Простой и множественный анализ соответствий как метод разведочного анализа данных [Электронный ресурс] // Высшая школа экономики. – 2013. – Вып. 1. – URL: http://radar-research.ru/wp-content/uploads/2013/10/1.Shaphir_2006_diploma.pdf (дата обращения: 23.08.2017).

10. Адамов С. Система анализа нечисловой информации «САНИ» // Социология: методология, методы, математическое моделирование. – 1992. – № 2. – С. 86–104.

11. Nishisato S. Analysis of categorical data: Dual scaling and its applications // University of Toronto Press. – 1980. – 276 p.

12. Clausen S.-E. Applied correspondence analysis: An introduction. // Sage university papers. Ser. Quantitative applications in the social sciences. – 1998. – Vol. 121. – P. 137–150.

13. Singular Value Decomposition Tutorial [Электронный ресурс] // University of Western Australia. – 2005. – Вып. 1. – URL: https://davetang.org/file/Singular_Value_Decomposition_Tutorial.pdf (дата обращения: 07.12.2017).

14. The SVD Algorithm [Электронный ресурс] // Stanford University. – 2005. – Вып. 1. – URL: https://web.stanford.edu/class/cme335/-lecture6.pdf (дата обращения: 23.09. 2017).

15. SVD-разложение и его практические приложения [Электронный ресурс] // Overleaf. – 2015. – Вып. 1. – URL: https://www.overleaf.com/articles/svd-razlozhieniie-i-iegho-praktichieskiie-prilozhieniia-svd-decomposition-and-its-practical-applications/gkzqbdxdgmry (дата обращения: 28.10.2017).

16. Computing the Sparse Singular Value Decomposition via SVDPACK [Электронный ресурс] // Springer. – 1994. – Вып. 1. – URL: https://link.springer.com/chapter/10.1007/978-1-4613-9353-5_2 (дата обращения: 24.11.2017).

17. SVDPACK [Электронный ресурс] // Netlib Springer. – 2004. – Вып. 1. – URL: http://www.netlib.org/svdpack/ (дата обращения: 23.12.2017).

18. Боровиков В. Statistica. Искусство анализа данных на компьютере: для профессионалов. – СПб.: Питер, 2003. – 688 с.

19. Farid D.M. Adaptive Intrusion Detection based on Boosting and Naïve Bayesian Classifier // International Journal of Computer Application. – 2011. – URL: http://www.ijcaonline.org/volume24/number3/pxc3873883.pdf (дата обращения: 04.12.2017).

20. Accurate SVDs of Structured Matrices // Netlib. – 2004. – URL: http://www.netlib.org/lapack/lawnspdf/lawn130.pdf (дата обращения: 24.10. 2017).

21. Singular Value Decomposition // NCSU. – 2013. – URL: http://www4.ncsu.edu/~ipsen/REU09/chapter4.pdf (дата обращения: 14.06. 2017).

22. Mukherjee S., Sharma N. Intrusion Detection using Naive Bayes Classifier with Feature Reduction // Science Direct. – 2012. – URL: http://www.sciencedirect.com/science/article/pii/S2212017312002964 (дата обращения: 26.11.2017).

References

1. Burlakov M.E. Research the dynamic of author activities in threats through to public and private sources. *Sbornik trudov III Mezhdunarodnoi konferentsii i molodezhnoi shkoly informatsionnye tekhnologii i nanotekhnologii*. Samara: Novaia tekhnika, 2017, pp. 958-961.

2. Saul L.K. Advances in Neural Information Processing Systems. *MIT Press*, 2005. 641 p.

3. Intrusion detection evaluation dataset. University of New Brunswick, 2017, iss. 1, available at: http://www.unb.ca/cic/research/datasets/ids.html (accessed 07 August 2017).

4. Al-Hamami A.H. Handbook of Research on Threat Detection and Countermeasures in Network Security. *IGI Global*, 2014. 450 p.

5. Levin I. KDD-99 Classifier Learning Contest. LLSoft's Results Overview. SIGKDD Explorations, 2010, pp. 67-75.

6. Lippmann R.P. Evaluating Intrusion Detection Systems: The 1998 DARPA off-line intrusion detection evaluation. *DARPA*, 2000, pp. 10-35.

7. Tavallaee M. A Detailed Analysis of the KDD CUP 99 Data Set. *Submitted to Second IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA)*, 2009, pp. 7-42.

8. Knowledge discovery in databases DARPA archive. University of California, 2017, iss. 1, available at: http://www.kdd.ics.uci.edu/databases/-kddcup99/task.html (accessed 29 January 2017).

9. Prostoi i mnozhestvennyi analiz sootvetstvii kak metod razvedochnogo analiza dannykh [Simple and multiple match analysis as a method of exploratory data analysis]. Vysshaia shkola ekonomiki, 2013, iss. 1, available at: http://radar-research.ru/wp-content/uploads/2013/10/1.Shaphir_2006_diploma.pdf (accessed 23 August 2017).

10. Adamov S. Sistema analiza nechislovoi informatsii "SANI" [Non-numerical information analysis system "SANI"]. *Sotsiologiia metodologiia metody matematicheskoe modelirovanie*, 1992, no. 2, pp. 86-104.

11. Nishisato S. Analysis of categorical data: Dual scaling and its applications. University of Toronto Press, 1980. 276 p.

12. Clausen S.-E. Applied correspondence analysis: An introduction. *Sage university papers. Series: Quantitative applications in the social sciences*, 1998, vol. 121, pp. 137-150.

13. Singular Value Decomposition Tutorial. University of Western Australia, 2005, iss. 1, available at: https://davetang.org/file/Singular_-Value_Decomposition_Tutorial.pdf (accessed 07 December 2017).

14. The SVD Algorithm. Stanford University, 2005, iss. 1, available at: https://web.stanford.edu/class/cme335/lecture6.pdf (accessed 23 September 2017).

15. SVD-razlozhenie i ego prakticheskie prilozheniia [SVD-decomposition and its practical applications]. *Overleaf*, 2015, iss. 1, available at: https://www.overleaf.com/articles/svd-razlozhieniie-i-iegho-praktichieskiie-prilozhieniia-svd-decomposition-and-its-practical-applications/gkzqbdxdgmry (accessed 28 October 2017).

16. Computing the Sparse Singular Value Decomposition via SVDPACK. *Springer*, 1994, iss. 1, available at: https://link.springer.com/-chapter/10.1007/978-1-4613-9353-5_2 (accessed 24 November 2017).

17. SVDPACK. *Netlib Springer*, 2004, iss. 1, available at: http://www.netlib.org/svdpack/ (accessed 23 December 2017).

18. Borovikov V. STATISTICA. Iskusstvo analiza dannykh na komp'iutere dlia professionalov [The Art of Data Analysis on a Computer: For Professionals]. Saint Petersburg: Piter, 2003. 688 p.

19. Farid D.M. Adaptive Intrusion Detection based on Boosting and Naïve Bayesian Classifier // *International Journal of Computer Application*, 2011, available at: http://www.ijcaonline.org/volume24/number3/pxc38-73883.pdf (accessed 04 December 2017).

20. Accurate SVDs of Structured Matrices, Netlib, 2004, available at: http://www.netlib.org/lapack/lawnspdf/lawn130.pdf (accessed 24 October 2017).

21. Singular Value Decomposition. *NCSU*, 2013, available at: http://www4.ncsu.edu/~ipsen/REU09/chapter4.pdf (accessed 14 June 2017).

22. Mukherjee S., Sharma N. Intrusion Detection using Naive Bayes Classifier with Feature Reduction. *Science Direct*, 2012, available at: http://www.sciencedirect.com/science/article/pii/S2212017312002964 (accessed 26 November 2017).

Сведения об авторе

Бурлаков Михаил Евгеньевич (Самара, Россия) – старший преподаватель кафедры «Безопасность информационных систем» Самарского национального исследовательского университета им. акад. С.П. Королева (443086, Самара, Московское шоссе, 34, e-mail: knownwhat@gmail.com).

About the author

Burlakov Mikhail Evgenyevich (Samara, Russian Federation) is a Senior Lecturer in Department of Information Security Systems Samara National Research University named after academician S.P. Korolev (443086, Samara, 34, Moskovskoye Shosse, e-mail: knownwhat@gmail.com).

Получено 25.04.2018