ISSN 1816-0301 (Print)
ISSN 2617-6963 (Online)

УДК 004.492.3
https://doi.org/10.37661/1816-0301-2020-17-3-78-86

Поступила в редакцию 26.02.2020
Received 26.02.2020

Принята к публикации 25.05.2020
Accepted 25.05.2020

Текстовый анализ DNS-запросов для защиты компьютерных сетей от эксфильтрации данных

Я. В. Бубнов✉, Н. Н. Иванов

Белорусский государственный университет информатики и радиоэлектроники, Минск, Беларусь
✉E-mail: girokompass@gmai.com

Аннотация. Предлагается эффективный способ защиты компьютерных сетей от эксфильтрации данных через систему доменных имен (англ. Domain Name System, DNS), которая представляет собой способ сокрытия передачи конфиденциальной информации удаленному злоумышленнику путем инкапсуляции данных в запрашиваемое доменное имя. Рассматриваются DNS-запросы, в которых передается украденная информация, с зараженного вредоносной программой узла на внешний узел, управляемый злоумышленником. Описывается подход для обнаружения подобных запросов с помощью текстовой классификации доменных имен сверточной нейронной сетью. Эффективность подхода базируется на предположении, что доменные имена, используемые для эксфильтрации данных, отличаются от доменных имен, сформированных из слов естественного языка. Для классификации запросов в сверточной нейронной сети предлагается использовать символьное встраивание с целью представления строки доменного имени. Производится оценка качества распознавания эксфильтрации данных через DNS с помощью ROC-анализа для обученной нейронной сети.

Демонстрируется архитектура программного обеспечения для развертывания обученной нейронной сети в существующую инфраструктуру DNS с целью практической защиты компьютерных сетей от эксфильтрации данных. Архитектура подразумевает формирование зон с политикой ответов для блокировки отдельных запросов, классифицируемых как вредоносные.

Ключевые слова: система доменных имен, защита компьютерных сетей, эксфильтрация данных, текстовая классификация, сверточная нейронная сеть

Для цитирования. Бубнов, Я. В. Текстовый анализ DNS-запросов для защиты компьютерных сетей от эксфильтрации данных / Я. В. Бубнов, Н. Н. Иванов // Информатика. – 2020. − Т. 17, № 3. – С. 78–86. https://doi.org/10.37661/1816-0301-2020-17-3-78-86

Text analysis of DNS queries for data exfiltration protection of computer networks

Yakov V. Bubnov✉, Nick N. Ivanov

Belarusian State University of Informatics and Radioelectronics, Minsk, Belarus
✉E-mail: girokompass@gmai.com

Abstract. The paper proposes effective method of computer network protection from data exfiltration by the system of domain names. Data exfiltration by Domain Name System (DNS) is an approach to conceal the transfer of confidential data to remote adversary using data encapsulation into the requesting domain name. The DNS requests that transfer stolen information from a host infected by malicious software to an external host controlled by a malefactor are considered. The paper proposes a method of detecting such DNS requests based on text classification of domain names by convolutional neural network. The efficiency of the method is based on assumption that domain names exploited for data exfiltration differ from domain names formed from words of natural language. To classify the requests in convolutional neural network the use of character

embedding for representing the string of a domain name is proposed. Quality evaluation of the trained neural network used for recognition of data exfiltration through domain name system using ROC-analysis is performed.
The paper presents the software architecture used for deployment of trained neural network into existing infrastructure of the domain name system targeting practical computer networks protection from data exfiltration. The architecture implies creation of response policy zones for blocking of individual requests, classified as malicious.

Keywords: domain name system, computer network security, data exfiltration, text classification, convolutional neural network

For citation. Bubnov Ya. V., Ivanov N. N. Text analysis of DNS queries for data exfiltration protection of computer networks. *Informatics*, 2020, vol. 17, no. 3, pp. 78−86 (in Russian). https://doi.org/10.37661/1816-0301-2020-17-3-78-86

Введение. Уязвимости инфраструктуры компьютерной сети могут стать фатальной угрозой для конфиденциальной информации организаций, а следовательно, привести к потере конкурентного преимущества из-за кражи секретных документов, утрате доверия пользователей или даже снижению рыночной капитализации компании. Целевые кибератаки действуют на жертву продолжительное время и эксплуатируют подобные уязвимости как для сокрытия проникновения, так и непосредственно для кражи данных. Одним из способов сокрытия кибератаки служит инкапсуляция протоколов различных уровней модели OSI (Open System Interconnection) в доменное имя DNS-запроса. DNS – это фундаментальная система сети Интернет, которая предоставляет сервис получения информации о доменных именах. Наиболее распространенным вариантом использования DNS является получение сетевого адреса узла в компьютерной сети по его доменному имени. Помимо этого, протокол обмена информацией о доменных именах разработан с учетом возможности будущего расширения. Гибкость протокола DNS позволяет встраивать произвольную информацию в качестве доменного имени, из-за чего возникают трудности в различении вредоносных и безопасных запросов.

Методика встраивания произвольной информации в запросы DNS называется DNS-туннелированием, так как подразумевает создание логического соединения между двумя конечными точками сети. Самый распространенный способ передачи информации через DNS-туннель – кодирование данных с помощью алфавитного кодировщика отправляющей стороной в дочерние домены некоторого делегированного домена. Таким образом корпоративный DNS-сервер с включенным механизмом рекурсивной обработки запроса инициирует запрос авторитативному DNS-серверу, который в случае DNS-туннелирования управляется злоумышленником. Благодаря тому, что данный протокол не блокируется межсетевыми экранами, DNS-туннелирование широко используется как для передачи команд управления зараженному узлу, так и для кражи данных.

В последние годы были обнаружены по меньшей мере два вида сетевых червей: Morto [1] и Federbot [2], применяющих DNS-туннелирование для передачи команд управления зараженным узлам. Вместе с тем специалистами кибербезопасности обнаружено вредоносное программное обеспечение BernhardPOS [3] и FrameworkPOS (URL: https://www.gdatasoftware.com/blog/2014/10/23942-new-frameworkpos-variant-exfiltrates-data-via-dns-requests), использующее DNS-туннелирование для кражи информации о кредитных картах с платежных терминалов.

Таким образом, под термином «эксфильтрация данных» в широком смысле понимается неавторизированная передача произвольной информации, которая рассматривается как форма кражи данных. В статье рассматривается эксфильтрация данных в узком смысле, т. е. злонамеренная кража корпоративных секретов, интеллектуальной собственности и других ценных для владельца данных с помощью техники DNS-туннелирования. Для решения проблемы эксфильтрации данных с помощью протокола DNS предлагается метод, основанный на использовании бинарного классификатора. Бинарная классификация представляет собой метод разделения множества, в данном контексте множества DNS-запросов, на два класса: безопасные и небезопасные DNS-запросы. Класс небезопасных DNS-запросов состоит из запросов, используемых при туннелировании системы доменных имен, все остальные запросы относятся к безопасным.

Предлагаемый классификатор представляет собой сверточную нейронную сеть, обученную на доменных именах из репозитория OpenDNS и доменных именах, используемых при DNS-туннелировании [4]. Эффективность предлагаемого в статье подхода основана на предположении, что безопасные доменные имена представляют подмножество слов естественного языка, в то время как доменные имена, используемые для DNS-туннелирования, таким свойством не обладают.

Существующие методы обнаружения эксфильтрации данных. В работах [5, 6] рассматривается метод обнаружения DNS-туннелей, основанный на частотном анализе биграмм доменного имени, и выдвигается гипотеза, что безопасные доменные имена распределены в соответствии с законом Ципфа, в то время как доменные имена, используемые для туннелирования, соответствуют равномерному распределению. Исходя из этого предположения предлагается оценивать каждое доменное имя по частоте встречаемости биграмм в естественном языке. Для этого применяется предварительно рассчитанная частотная таблица биграмм, на основании которой оценивается каждое доменное имя. Если оценка не превышает некоторый заданный порог, алгоритм относит доменное имя к классу небезопасных имен. Данный подход позволил авторам добиться точности детектирования DNS-туннелей, равной 98,74 %. Несмотря на высокую точность результатов, этот подход обладает рядом недостатков. Во-первых, существует необходимость построения таблицы частот биграмм для каждого естественного языка, что на практике невыполнимо. Во-вторых, анализ производился исключительно на доменных именах, созданных программой туннелирования DNSCat, которая является одной из многих открытых программ для установления DNS-туннелей.

Альтернативный способ обнаружения DNS-туннелирования базируется на гипотезе, что статистические характеристики некоторых численных параметров DNS-запросов при туннелировании отличаются от соответствующих характеристик безопасных запросов. Например, А. Надлер и др. в работе [7] и независимо А. Берг и Д. Форсберг в работе [8] рассматривают энтропию доменного имени, объем трафика к домену, соотношение длины доменного имени к субдомену и иные параметры. Авторы предлагают применять алгоритм изолирующего леса (isolation forest) для поиска аномалий, а также подчеркивают эффективность данного метода обнаружения DNS-туннелирования и высокую точность классификации. Однако проблема такого подхода состоит в его прикладном использовании. Так, в случае применения подобного детектора провайдером DNS, предоставляющим услуги разыменования доменных имен, для вычисления суммарного трафика к вредоносному домену необходим сбор параметров со всех резольверов DNS. Для этого требуется хранение распределенной таблицы частот обращений, что не всегда выполнимо на практике ввиду инженерной сложности задачи и финансовой стоимости обслуживания такой инфраструктуры.

Среди прикладных способов обнаружения эксфильтрации данных через DNS-туннелирование выделяют анализ длины доменного имени [9]. Автор публикации подчеркивает, что длина субдомена в реальности редко превышает 30 символов, а значит, можно установить соответствующие правила для блокировки запросов, превышающих заданный порог. В действительности программы для создания IPv4-туннелей через DNS, например Iodine, позволяют установить значение максимальной единицы передачи (англ. Maximum Transmission Unit, MTU) и тем самым с легкостью обойти подобные ограничения, а учитывая, что DNS-туннель всегда двухточечный, можно полностью избавиться от накладных расходов заголовка IPv4, составляющего 20 байтов, и передавать данные напрямую.

Сверточная нейронная сеть для классификации DNS-запросов. Сформулируем задачу обнаружения DNS-туннеля следующим образом: пусть $\mathbf{X} = \{x_0, x_1, ..., x_n\}$ – запрашиваемое доменное имя, с которым ассоциирована метка $y \in Y$ с вероятностью p_y. Тогда стоит задача нахождения функции

$$y : \mathbf{X} \to Y. \tag{1}$$

Пусть дан алфавит символов $\mathbf{A}$, из которого состоит доменное имя, такое, что $x_i \in \mathbf{A}$, а само доменное имя представляет собой подмножество размещений алфавита $\mathbf{X} \subset \mathbf{A}^k$, где $k = |\mathbf{X}| = 256$ – максимальная длина доменного имени, установленного в спецификации прото-

кола DNS [10]. Таким образом, необходимо решить задачу отнесения слов некоторого заданного алфавита к одному из классов множества Y.

Для поиска функции y воспользуемся методом, предложенным в работе [11], где в качестве классификатора используется сверточная нейронная сеть.

Так как входы нейронной сети представляют собой вектор действительных чисел, требуется определить отображение символьного множества $\mathbf{X}$ в пространство $\mathbf{R}^k$. Для этого применим модифицированный механизм встраивания слов (англ. word embedding), предложенный в статье [12], где вместо словаря слов используется алфавит символов. Пусть дана функция I, которая ставит каждому элементу из $\mathbf{A}$ в соответствие некоторый индекс. Тогда встраивание – это преобразование вектора символов в вектор целых чисел, где каждому символу в соответствие поставлен индекс из алфавита $\mathbf{A}$:

$$I : \mathbf{A} \to \{0, ..., |\mathbf{A}|\}, \ \text{cemb}(\mathbf{X}) = I^{-1}(\mathbf{X}) = \mathbf{E}. \tag{2}$$

Очевидно, что полученный массив $\mathbf{E}$ будет разреженным, так как длина большинства доменных имен редко составляет 256 символов. Чтобы сократить количество нулевых элементов в $\mathbf{E}$, воспользуемся методикой встраивания из работы [13]:

$$E_{k \to k/2} : \mathbf{R}^k \to \mathbf{R}^{k/2}, \ \mathbf{Q} = E_{k \to k/2}(\mathbf{E}). \tag{3}$$

После уменьшения размерности входного вектора применим сверточную фильтрацию

$$\text{conv1d}(\mathbf{Q}; \mathbf{H}, w) = \langle \mathbf{Q}_{i,i+w}, \mathbf{H} \rangle, \ \mathbf{h}_i = \text{conv1d}(\mathbf{Q}; \mathbf{H}, w) + b, \tag{4}$$

где $\mathbf{H}$ – ядро свертки, w – размер ядра свертки, $\mathbf{Q}$ – встраивание символьного представления доменного имени $\mathbf{Q} \in \mathbf{R}^{k/2}$ в пространство действительных чисел размерности $k/2$. Для простоты дальнейшего изложения обозначим функцию активации сверточного слоя i равенством

$$a_{\text{conv1d}}(\mathbf{h}^{(i)}) = \tanh(\mathbf{h}^{(i)}). \tag{5}$$

За каждым сверточным слоем расположим слой субдискретизации, позволяющий существенно снизить пространственный объем числового представления доменного имени путем выбора построчного максимального значения из предыдущего сверточного слоя:

$$a_{\text{max pool}}(\mathbf{z}^{(i)}) = \max(\mathbf{z}^{(i)}), \tag{6}$$

где $\mathbf{z}^{(i)} = \mathbf{W}^{(i)}\mathbf{x}^{(i)} + \mathbf{b}^{(i)}$. Идея заключается в том, что на сверточном слое уже выявлены некоторые признаки, характеризующие доменное имя, поэтому подробная детализация и дальнейшая обработка большого объема данных не представляются необходимыми.

Дополнительно определим слой, используемый для самонормализации сверточной сети [14]:

$$\text{selu}(x; \alpha) = \lambda \begin{cases} x, & x > 0; \\ \alpha e^x - \alpha, & x \le 0, \end{cases} \qquad a_{\text{selu}}(\mathbf{z}^{(i)}) = \text{selu}(\mathbf{z}^{(i)}). \tag{7}$$

Описанный подход позволяет простым способом добиться нормализации выходов сверточного слоя нейронной сети, т. е. сохранения математического ожидания и дисперсии выходов слоя в заданном диапазоне значений.

Определим механизм регуляризации скрытых слоев нейронной сети с помощью оператора прореживания. Пусть R – случайная дискретная величина, принимающая значения из множества $\{0,1\}$, а p – наперед заданная константа, лежащая в диапазоне $[0,1]$. Тогда закон распределения случайной величины R имеет вид

$$F_R = \begin{cases} p, \ R = 1; \\ 1 - p, \ R = 0, \end{cases} \tag{8}$$

а функция прореживания определяется выражением

$$\mathbf{d} = \mathbf{J}_{1,n} * \mathbf{r}_p, \ D_p(\mathbf{Z}) = \mathbf{Z} * \mathbf{d}^T, \tag{9}$$

где $\mathbf{r}_p \in \mathbf{R}^n$ – вектор случайных величин, $\mathbf{J}$ – вектор единиц, $*$ – произведение Адамара.

Так как задача нейронной сети задействована в определении вероятностей отнесения доменного имени к каждому классу из Y, требуется, чтобы сумма всех вероятностей составляла единицу. Данное требование выполняется при применении функции softmax к выходному слою сети:

$$\text{softmax}(\mathbf{x})_i = \frac{e^{x_i}}{\sum_{j=1}^{|\mathbf{x}|} e^{x_j}}, \ a^{(i)}{}_{\text{soft max}} = \text{softmax}(\mathbf{z}^{(i)}). \tag{10}$$

Следовательно, архитектура нейронной сети, используемой для классификации символьной строки доменного имени, представляется следующим образом:

$$\begin{gathered} \mathbf{h}^{(1)} = E_{k \to k/2}(\mathbf{E}), \ \mathbf{h}^{(2)} = \left[a_{\text{conv1d}(\mathbf{Q};\mathbf{H},10)} \circ a_{\max \text{ pool}} \right](\mathbf{h}^{(1)}), \\ \mathbf{h}^{(3)} = \left[a_{\text{conv1d}(\mathbf{Q};\mathbf{H},7)} \circ a_{\max \text{ pool}} \right](\mathbf{h}^{(2)}), \ \mathbf{h}^{(4)} = \left[a_{\text{conv1d}(\mathbf{Q};\mathbf{H},5)} \circ a_{\max \text{ pool}} \right](\mathbf{h}^{(3)}), \\ \mathbf{h}^{(5)} = \left[a_{\text{conv1d}(\mathbf{Q};\mathbf{H},3)} \circ a_{\max \text{ pool}} \right](\mathbf{h}^{(4)}), \ \mathbf{z}^{(6)} = a_{\text{selu}}(\mathbf{h}^{(5)}), \\ \mathbf{z}^{(7)} = D_{0,1}(\mathbf{z}^{(6)}), \ \hat{y}(\mathbf{E}) = a_{\text{soft max}}\left(\mathbf{h}^{(5)} * \mathbf{z}^{(6)} * \mathbf{z}^{(7)} \right). \end{gathered} \tag{11}$$

Отметим, что классификатор $\hat{y}$ определен относительно встраивания символьного представления $\mathbf{E}$ вместо доменного имени $\mathbf{X}$. Приведем конечный вид классификатора:

$$y^* = \hat{y}(\text{cemb}(\mathbf{X})). \tag{12}$$

Для обучения предложенной нейронной сети в качестве функции потерь будем использовать перекрестную энтропию, а в качестве алгоритма оптимизации – adam [15], базирующийся на вычислении градиента, так как сходимость данного алгоритма по сравнению с классическим алгоритмом стохастического градиентного спуска выше.

Оценка результатов обучения сверточной нейронной сети. Для обучения нейронной сети применялся эталонный набор [5], состоящий как из доменных имен распространенных сервисов сети Интернет, так и из доменных имен, созданных наиболее известными программами для DNS-туннелирования: tuns, iodine и т. д. Каждому доменному имени поставлена в соответствие метка класса, к которому оно относится (безопасное или небезопасное DNS-имя). Эталонный набор разбит на два непересекающихся подмножества: обучающее и валидационное. Обучающий набор используется непосредственно для процесса поиска искомой функции, а валидационный набор – для предотвращения переобучения и обеспечения высокой обобщающей способности нейронной сети. Распределение примеров в каждом наборе по классам неравномерно и находится в соотношении 1:3 (рис. 1).

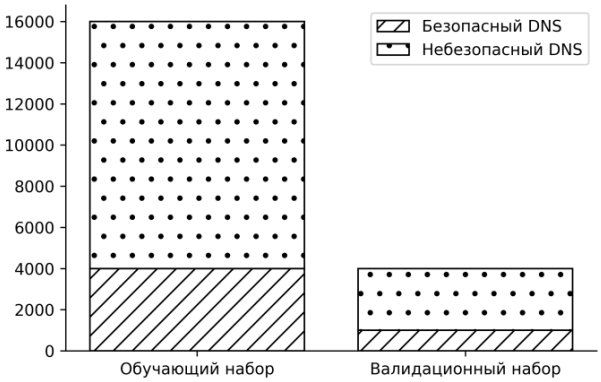


Рис. 1. Распределение классов в обучающем и валидационном наборах

Непосредственно обучение нейронной сети ограничено 10 итерациями, где на каждой итерации производится корректировка весов нейронной сети по всему обучающему набору данных. Ограничение по итерациям также является механизмом регуляризации. График изменения значения функции потерь после каждой итерации представлен на рис. 2. Он демонстрирует необходимость использования валидационного набора для корректировки весов скрытых слоев нейронной сети. Несмотря на то что ошибка обучения уже на второй итерации практически сравнялась с нулем для обучающего набора, ошибка обучения на валидационном наборе возрастает вплоть до третьей итерации. Данные наблюдения свидетельствуют о необходимости использования двух наборов с целью предотвращения переобучения нейронной сети.

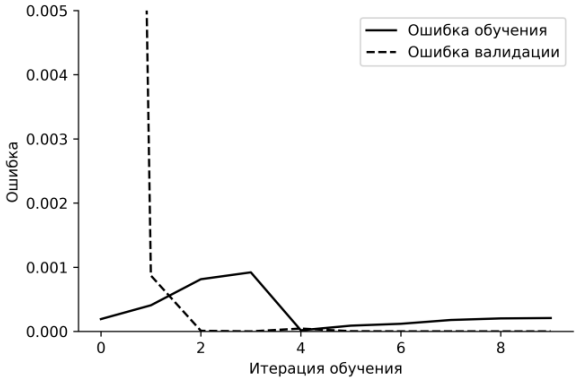
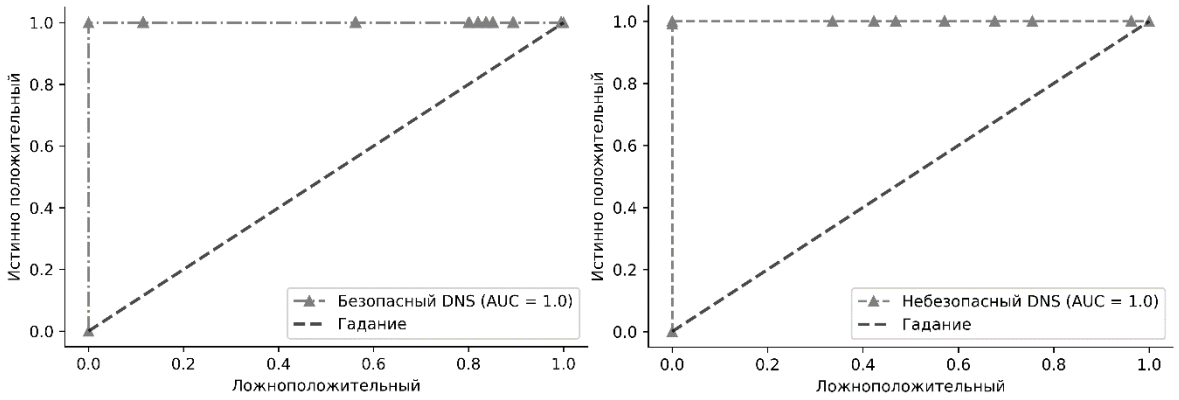


Рис. 2. Изменение функции потерь на этапах обучения и валидации
в зависимости от количества итераций обучения нейронной сети

Для оценки точности классификации использовались ROC-кривые, показывающие зависимость между ложно- и истинно положительными результатами нейронной сети. На рис. 3 AUC (area under ROC curve) – это площадь под ROC-кривой, представляющая собой количественную характеристику качества классификации модели. Чем выше AUC, тем качественнее классификатор. Также на рис. 3 диагональной штриховой линией изображен график модели, для которой классификация является процессом случайного гадания, т. е. событие отнесения DNS-запроса к одному из классов равновероятно. Близость к этому графику говорит о непригодности метода для классификации. Максимально возможная удаленность ROC-кривой нейросетевой модели от графика гадания демонстрирует высокое качество классификации DNS-запросов.

На рис. 3 видно, что точность модели близка к идеальной. Это подтверждает начальную гипотезу об отличии доменных имен, применяемых в туннелировании, от слов естественного языка, образующих безопасные доменные имена. В противном случае обученная модель не смогла бы различить доменные имена разных классов.



а) *б)*

Рис. 3. ROC-кривые классификации DNS-запросов с помощью обученной нейронной сети: *а)* кривая, соответствующая безопасным DNS-запросам; *б)* небезопасным DNS-запросам

Дополнительно стоит отметить, что полученные высокие результаты могут быть связаны с отсутствием в тестовом наборе доменных имен, используемых в сетях дистрибуции содержимого (content delivery network). Доменные имена в таких сетях генерируются и, как правило, также отличаются от слов естественного языка [16].

Практическое использование для защиты компьютерных сетей. Защита компьютерной сети от эксфильтрации данных состоит в блокировке DNS-запросов, организованных для скрытой передачи информации. Разработанный классификатор позволяет определить, к какому классу относится доменное имя в анализируемом запросе, и, таким образом, принять решение о блокировке DNS-запроса.

На практике блокировка доменных имен осуществляется с помощью зон с политикой ответов (англ. Response Policy Zone, RPZ). Такие зоны либо конфигурируются вручную, если требуется ограничить доступ передаваемых запросов, либо определяются динамически, если список запрещенных зон заранее неизвестен.

Исходя из этих условий, разработан модуль (URL: https://github.com/netrack/dnstun) для CoreDNS – одного из популярных DNS-серверов, используемых как в облачных инфраструктурах (например, Kubernetes), так и в виде самостоятельного сервера. Модуль организовывает процесс разыменования доменного имени с дополнительным анализом на наличие туннелирования (рис. 4).

Рис. 4. Архитектура DNS-сервера CoreDNS с функцией защиты от DNS-туннелирования

На рис. 4 показаны два компонента: dnsTUN, отвечающий за создание встраивания символьного представления доменных имен, и dnsCNN, осуществляющий классификацию представлений. Классификатор использует TensorFlow в качестве среды для выполнения вычислительного графа обученной сверточной нейронной сети.

Заключение. Метод обнаружения DNS-туннелирования с помощью текстового анализа доменных имен обладает преимуществом перед статистическими методами, так как не требует полного анализа содержимого запроса DNS, включая анализ различных временных характеристик, а также времени для накопления исторических данных о конкретной компьютерной сети. Дополнительно текстовый анализ доменных имен демонстрирует преимущество перед анализом биграмм, так как не использует построение таблицы частот для каждого естественного

языка. Описанный метод позволяет осуществлять фоновое обучение детектора и прозрачный перенос в случае изменения конфигурации компьютерной сети.

В настоящей статье сделан только шаг к решению проблемы защиты компьютерных сетей от эксфильтрации данных. Поэтому дальнейшая работа будет направлена на анализ доменных имен из CDN-сетей и поиск эффективных способов практического использования разработанного детектора в реальных сетях.

Список использованных источников

1. Zhong, X. Stealthy malware traffic – not as innocent as it looks / X. Zhong, Y. Fu, R. Brooks // Malicious and Unwanted Software (MALWARE) : 10th Intern. Conf., Fajardo, 20–22 Oct. 2015. – Fajardo, 2015. – P. 110–116.

2. On botnets that use DNS for command and control / C. Deitrich [et al.] // Computer Network Defense : 7th European Conf. on Computer Network Defense, Gotheburg, 6–7 Sept. 2011. – Gotheburg, 2011. – P. 9–16.

3. Valenzuela, I. Game changer: identifying and defending against data exfiltration attempts [Electronic resource] // SANS Cyber Security Summit Archive. – 2015. – Mode of access: https://www.sans.org/cyber-security-summit/archives/file/summit-archive-1493840468.pdf. – Date of access: 15.02.2020.

4. Bubnov, Y. DNS tunneling queries for binary classification [Electronic resource] / Y. Bubnov // Mendeley Data. – N. Y., 2019. – Vol. 1. – Mode of access: https://data.mendeley.com/datasets/mzn9hvdcxg/1. – Date of access: 15.02.2020.

5. A bigram based real time DNS tunnel detection approach / C. Qi [et al.] // Procedia Computer Science. – 2013. – Vol. 17. – P. 852–860.

6. Born, K. Detecting DNS tunneling using character frequency analysis / K. Born, D. Gustafson // Proc. of the 9th Annual Security Conf., Las Vegas, 7–8 Apr. 2010. – Las Vegas, 2010. – P. 2–3.

7. Nadler, A. Detection of malicious and low throughput data exfiltration over the DNS protocol / A. Nadler, A. Aminov, A. Shabtai. – 2018. – Mode of access: https://arxiv.org/abs/1709.08395. – Date of access: 15.02.2020.

8. Berg, A. Identifying DNS-tunneled Traffic with Predictive Models [Electronic resource] / A. Berg, D. Forsberg. – 2019. – Mode of access: https://arxiv.org/abs/1906.11246. – Date of access: 12.01.2020.

9. Лукацкий, А. Об утечках через DNS, которые не ловит ни одна DLP [Электронный ресурс] / А. Лукацкий // Бизнес без опасности. – 2018. – Режим доступа: https://www.securitylab.ru/blog/personal/Business_without_danger/343229.php. – Дата доступа: 07.05.2020.

10. Mockapetris, P. Domain names – implementation and specification [Electronic resource] / P. Mockapetris // Internet Standard, ISI. – 1987. – Mode of access: https://tools.ietf.org/html/rfc1035. – Date of access: 15.02.2020.

11. Character-aware Neural Language Models [Electronic resource] / Y. Kim [et al.]. – 2016. – Mode of access: https://arxiv.org/abs/1508.06615. – Date of access: 12.01.2020.

12. Watson, D. Utilizing Character and Word Embedding for Text Normalization with Sequence-to-Sequence Models [Electronic resource] / D. Watson, N. Zalmout, N. Habash. – 2019. – Mode of access: https://arxiv.org/abs/1809.01534. – Date of access: 12.01.2020.

13. Gal, Y. A Theoretically Grounded Application of Dropout in Recurrent Neural Networks [Electronic resource] / Y. Gal, Z. Ghahraamni. – 2016. – Mode of access: https://arxiv.org/abs/1512.05287. – Date of access: 12.01.2020.

14. Self-normalizing Neural Networks [Electronic resource] / G. Klambauer [et al.] – 2017. – Mode of access: https://arxiv.org/abs/1706.02515. – Date of access: 12.01.2020.

15. Kingma, D. Adam: a method for stochastic optimization / D. Kingma, J. Ba // 3rd Intern. Conf. for Learning Representations, San Diego, 7–9 May 2015. – San Diego, 2015. – 15 p.

16. Nygren, E. The Akami network: a platform for high-performance internet applications / E. Nygren, R. Sitaraman, J. Sun // ACM SIGOPS Operating Systems Review. – 2010. – Vol. 44, iss. 3. – P. 2–19.

References

1. Zhong X., Fu Y., Brooks R. Stealthy malware traffic – not as innocent as it looks. *Malicious and Unwanted Software (MALWARE) : 10th International Conference, Fajardo, 20–22 October 2015.* Fajardo, 2015, pp. 110–116.

2. Deitrich C., Rossow C., Freiling F., Bos H., van Steen M., Pohlman N. On botnets that use DNS for command and control. *Computer Network Defense : 7th European Conference on Computer Network Defense, Gotheburg, 6–7 September 2011.* Gotheburg, 2011, pp. 9–16.

3. Valenzuela I. Game changer: identifying and defending against data exfiltration attempts. *SANS Cyber Security Summit Archive*, 2015. Available at: https://www.sans.org/cyber-security-summit/archives/file/summit-archive-1493840468.pdf (accessed 15.02.2020).

4. Bubnov Y. DNS tunneling queries for binary classification. *Mendeley Data*. N. Y., 2019, vol. 1. Available at: https://data.mendeley.com/datasets/mzn9hvdcxg/1 (accessed 15.02.2020).

5. Qi C., Chen X., Xu C., Shi J., Liu P. A bigram based real time DNS tunnel detection approach. *Procedia Computer Science*, 2013, vol. 17, pp. 852–860.

6. Born K., Gustafson D. Detecting DNS tunneling using character frequency analysis. *Proceedings of the 9th Annual Security Conference, Las Vegas, 7–8 April 2010*. Las Vegas, 2010, pp. 2–3.

7. Nadler A., Aminov A., Shabtai A. *Detection of Malicious and Low Throughput Data Exfiltration over the DNS Protocol*, 2018. Available at: https://arxiv.org/abs/1709.08395 (accessed 15.02.2020).

8. Berg A., Forsberg D. *Identifying DNS-tunneled Traffic with Predictive Models*, 2019. Available at: https://arxiv.org/abs/1906.11246 (accessed 12.01.2020).

9. Lukatski A. Ob utechkah cherez DNS, kotorye ne lovit ni odna DLP [About leaks through DNS, which are not captured by any DLP]. Biznes bez opasnosti [*Business without Danger*], 2018. Available at: https://www.securitylab.ru/blog/personal/Business_without_danger/343229.php. (accessed 07.05.2020).

10. Mockapetris P. Domain names – implementation and specification. *Internet Standard, ISI*, 1987. Available at: https://tools.ietf.org/html/rfc1035 (accessed 15.02.2020).

11. Kim Y., Jernite Y., Sontag D., Rush A. *Character-aware Neural Language Models*, 2016. Available at: https://arxiv.org/abs/1508.06615 (accessed 12.01.2020).

12. Watson D., Zalmout N., Habash N. *Utilizing Character and Word Embedding for Text Normalization with Sequence-to-Sequence Models*, 2019. Available at: https://arxiv.org/abs/1809.01534 (accessed 12.01.2020).

13. Gal Y., Ghahraamni Z. *A Theoretically Grounded Application of Dropout in Recurrent Neural Networks*, 2016. Available at: https://arxiv.org/abs/1512.05287 (accessed 12.01.2020).

14. Klambauer G., Unterthiner T., Mayr A., Hochreiter S. *Self-normalizing Neural Networks*, 2017. Available at: https://arxiv.org/abs/1706.02515 (accessed 12.01.2020).

15. Kingma D., Ba J. Adam: a method for stochastic optimization. *3rd International Conference for Learning Representations, San Diego, 7–9 May 2015*. San Diego, 2015, 15 p.

16. Nygren E., Sitaraman R., Sun J. The Akami network: a platform for high-performance internet applications. *ACM SIGOPS Operating Systems Review*. 2010, vol. 44, iss. 3, pp. 2–19.

Информация об авторах

Бубнов Яков Васильевич, магистр технических наук, аспирант кафедры электронных вычислительных машин, факультет компьютерных систем и сетей, Белорусский государственный университет информатики и радиоэлектроники, Минск, Беларусь.
E-mail: girokompass@gmail.com

Иванов Николай Николаевич, кандидат физико-математических наук, доцент кафедры электронных вычислительных машин, факультет компьютерных систем и сетей, Белорусский государственный университет информатики и радиоэлектроники, Минск, Беларусь.
E-mail: ivanovnn@gmail.com

Information about the authors

Yakov V. Bubnov, M. Sci. (Eng.), Postgraduate Student of Department of Electronic Computing Machines, Faculty of Computer Systems and Networks, Belarusian State University of Informatics and Radioelectronics, Minsk, Belarus.
E-mail: girokompass@gmail.com

Nick N. Ivanov, Cand. Sci. (Phys.-Math.), Associate Professor of Department of Electronic Computing Machines, Faculty of Computer Systems and Networks, Belarusian State University of Informatics and Radioelectronics, Minsk, Belarus.
E-mail: ivanovnn@gmail.com