ЗАЩИТА ИНФОРМАЦИИ И НАДЕЖНОСТЬ СИСТЕМ
INFORMATION PROTECTION AND SYSTEM RELIABILITY

УДК 004.056.53
https://doi.org/10.37661/1816-0301-2022-19-1-19-31

Оригинальная статья
Original Paper

Детектирование признаков сетевой разведки с использованием модели дерева решений

Н. П. Шараев, С. Н. Петров✉

Белорусский государственный университет информатики и радиоэлектроники,
ул. П. Бровки, 6, Минск, 220013, Беларусь
✉*E-mail: sergpetrov@inbox.ru*

Аннотация

Цели. Своевременное обнаружение сетевой разведки позволяет снизить риски информационной безопасности организаций. Исследование проводилось с целью разработки программного модуля обнаружения признаков сетевой разведки с использованием методов машинного обучения.

Методы. Основными методами детектирования признаков сетевой разведки являлись: анализ открытых датасетов соответствующего назначения; формирование метрик, характерных для сетевой разведки; разработка набора данных разведки на основе определенных метрик. Исследовалась эффективность методов машинного обучения для задачи классификации.

Результаты. Спроектированы топология и тестовый сегмент в корпоративной сети РУП «Белтелеком» для создания датасета. Для детектирования и анализа событий разработано средство мониторинга, результаты работы которого использовались в качестве основы для нового датасета.

Реализация метода дерева принятия решений в виде программного кода позволила увеличить скорость работы модуля приблизительно в два раза (0,147 мс). Практические испытания разработанного модуля показали факт сработки на все типы сканирования сетей с помощью утилит Nmap и Masscan.

Заключение. Анализ датасета методом главных компонент показал наличие пограничной области между событиями легального трафика и трафика сетевой разведки, что положительно сказалось на обучении модели. Изучены и протестированы наиболее перспективные методы машинного обучения с использованием различных гиперпараметров. Наилучшие результаты показал метод дерева принятия решений с параметрами criterion = gini и splitter = random и скоростью работы 0,333 мс.

Ключевые слова: сетевая разведка, аномалии сетевого трафика, машинное обучение, метрики признаков разведки, датасеты

Для цитирования. Шараев, Н. П. Детектирование признаков сетевой разведки с использованием модели дерева решений / Н. П. Шараев, С. Н. Петров // Информатика. – 2022. − Т. 19, № 1. – С. 19–31.
https://doi.org/10.37661/1816-0301-2022-19-1-19-31

Конфликт интересов. Авторы заявляют об отсутствии конфликта интересов.

Поступила в редакцию | Received 12.08.2021
Подписана в печать | Accepted 10.12.2021
Опубликована | Published 29.03.2022

Detection of network intelligence features with the decision tree model

Nikita P. Sharaev, Sergei N. Petrov✉

Belarusian State University
of Informatics and Radioelectronics,
st. P. Brovki, 6, Minsk, 220013, Belarus
✉*E-mail: sergpetrov@inbox.ru*

Abstract
Objectives. Early detection of network intelligence allows to reduce the risks of information security of organizations. The study was carried out to develop software module for detecting the features of network intelligence by machine learning methods.
Methods. Analysis of open datasets of appropriate destination; formation of metrics characteristic of network intelligence; development of a dataset based on certain metrics; study of the effectiveness of machine learning methods for classification task.
Results. The topology was designed and a test segment was created in the corporate network of RUE "Beltelecom" to create a dataset. A monitoring tool has been developed for detecting and analyzing the events, the results of which were used as the basis for a new dataset.
The implementation of the decision tree method in the form of program code allowed to increase the speed of the module by about 2 times (0,147 ms). Practical tests of the developed module have shown the alarm on all types of network scanning using Nmap and Masscan utilities.
Conclusion. The analysis of the dataset by principal component method showed the presence of a border area between the events of legal traffic and network intelligence traffic, which had a positive effect on the training of the model. The most promising machine learning methods have been studied and tested using various hyperparameters. The best results were shown by the decision tree method with the parameters criterion = gini and splitter = random and speed as 0,333 ms.

Keywords: network intelligence, network traffic anomalies, machine learning, intelligence feature metrics, datasets

For citation. Sharaev N. P., Petrov S. N. *Detection of network intelligence features with the decision tree model*. Informatika [*Informatics*], 2022, vol. 19, no. 1, pp. 19–31 (In Russ.).
https://doi.org/10.37661/1816-0301-2022-19-1-19-31

Conflict of interest. The authors declare of no conflict of interest.

Введение. Сетевая разведка является первым этапом целенаправленной атаки на информационные системы, обнаружение которой позволяет снизить риски информационной безопасности. Она представляет собой комплекс мероприятий по получению и обработке данных об информационной системе, функционирующих в ней информационных ресурсах, средствах защиты информации и используемом программном обеспечении (ПО) [1].

Перечислим наиболее частые способы проведения сетевой разведки:

– анализ открытой (доступной в сети Интернет) информации с использованием расширенных запросов веб-браузеров;

– получение информации (например, списка выделенных IP-адресов, наименования владельца домена) от whois-серверов;

– получение информации от DNS-серверов;

– сканирование информационной сети;

– сканирование портов транспортного уровня.

Повлиять на доступ злоумышленника к информации от whois- и DNS-серверов, а также к открытой информации в сети Интернет невозможно. Наиболее эффективный способ обнаружения сетевой разведки – это выявление фактов сканирования информационной сети и портов транспортного уровня.

Анализ сетевой модели стека сетевых протоколов OSI (URL: http://www.williamspublishing.com/PDF/5-8459-0589-3/part.pdf) показал, что эффективное проведение сетевой разведки возможно только на втором, третьем, четвертом и седьмом (канальном, сетевом, транспортном и прикладном) уровнях модели OSI.

Сетевая разведка на втором уровне неэффективна, так как информация на этом уровне предназначена для передачи сообщений в рамках одного широковещательного домена локальной сети и не может выйти за пределы сети организации. При проведении злоумышленником сетевой разведки полученная им информация об инфраструктуре организации либо невалидна, либо относится к сетевой инфраструктуре используемого провайдера.

Проведение сетевой разведки на седьмом уровне модели OSI является сложной задачей из-за большого числа различных прикладных протоколов и служб.

Универсальным способом сетевой разведки остается сканирование третьего и четвертого уровней модели OSI. Доступ к ним злоумышленник может получить как из периметра сети организации, так и извне. Служебная информация, передаваемая на данных уровнях, содержит IP-адреса информационных систем организации, прослушиваемые и открытые транспортные порты, а также информацию о службах, функционирующих на прослушиваемых портах.

Целью сканирования информационной сети является поиск доступных устройств, находящихся в режиме онлайн. Обычно для этого используется сетевой протокол ICMP, предназначенный для передачи сообщений об ошибках и других исключительных ситуациях, возникших при передаче данных. В частности, это происходит, когда запрашиваемая услуга недоступна или хост (маршрутизатор) не отвечает. В подавляющем большинстве ОС (Windows, Linux, Cisco IOS, Huawei VRP и др.) для отправки ICMP-пакетов используются утилиты Ping и Tracert. Указанные утилиты распространены и легализованы производителями, и злоумышленники могут использовать их для изучения топологии корпоративной сети организации. В настоящее время на компьютерах под управлением ОС Windows трафик ICMP по умолчанию блокируют на межсетевом экране. Вместе с тем очень распространенной практикой является настройка исключений в межсетевом экране для данного типа трафика. Это связано с необходимостью быстрой проверки доступности отдельных хостов в корпоративной сети (troubleshooting).

Целью сканирования портов транспортного уровня является определение функционирующих на конкретном узле служб. Для этого проводятся попытки инициализации соединения на транспортных портах, лежащих в диапазоне 1–65535. Согласно спецификации первые 1024 порта зарезервированы под общеизвестные службы ОС, такие как FTP, SSH, SMTP, HTTP, HTTPS, SMB, и не могут быть переназначены. Зарегистрированные порты в диапазоне 1024–49151 могут переопределяться, но обычно не изменяются и назначаются на стандартные службы (RDP, Apache Tomcat, L2TP, MySQL, NFS и т. п.). Поэтому для получения информации о функционирующих и доступных извне службах злоумышленнику необходимо провести либо сканирование общеизвестных и зарегистрированных портов транспортного уровня (1–49151), что может занять ощутимый промежуток времени, либо выполнить частичное сканирование популярных портов. Перечень популярных портов транспортного уровня чаще всего зависит от ПО, которым пользуется злоумышленник. Общий перечень допустимых диапазонов портов транспортного уровня представлен в табл. 1.

Таблица 1
Сетевая разведка различных диапазонов портов транспортного уровня модели OSI

Table 1
Network intelligence of different ranges of ports of the transport layer of the OSI model

Название диапазона *Range name*	Диапазон портов *Port range*	Проведение сетевой разведки *Network intelligence*
Общеизвестные	1–1024	Целесообразно
Зарегистрированные	1024–49151	Целесообразно
Динамические	49152–65535	Нецелесообразно
Общие	1–65535	Нецелесообразно
Популярные	7, 20, 21, 22, 23, 25, 3306, 3389, 5432, 5601, 8080	Целесообразно

Наиболее популярными сканерами являются Nmap (Windows, Linux) и Masscan (Linux), хотя сканирование можно выполнять и с помощью стандартных средств ОС (Powershell) и (или) языков программирования Python, C++ и Ruby. По результатам сканирования нарушитель может построить приблизительную сетевую топологию организации и выявить возможные уязвимости в информационных системах.

В мировой практике задача обнаружения признаков сетевой разведки сопоставима с задачей обнаружения аномалий сетевого трафика, так как в обоих случаях возникает большое количество подозрительного трафика на сетевом и транспортном уровнях сети. Следовательно, признаки аномалий сетевого трафика валидны и могут быть применены для обнаружения сетевой разведки.

Обнаружение аномалии сетевого трафика. В соответствии с работой [2] для выявления аномалии сетевого трафика на сетевом уровне анализируются следующие данные: IP-адреса источника и назначения, дата и время получения IP-пакета, размер IP-пакета. Аномалия сетевого трафика на транспортном уровне определяется исходя из следующих признаков: количества входящих, исходящих либо внутрисетевых IP-, TCP- и UDP-пакетов; количества опросов неразрешенных портов UDP; количества завершенных запросов по протоколу UDP; количества незавершенных запросов по протоколу UDP; превышения продолжительности таймаута ответа узла; количества опросов портов TCP; количества опросов разрешенных портов TCP; количества соединений TCP, находящихся в состоянии установления (SYN); количества соединений TCP, находящихся в открытом состоянии (ESTABLISHED); количества соединений TCP, находящихся в состоянии закрытия (FIN) в единицу времени; отношения количества опросов разрешенных портов протокола TCP к количеству опросов всех портов этого протокола; отношения количества открываемых соединений TCP к общему количеству соединений [3]. Приведенные перечни признаков могут быть неполными, так как в различных источниках выделяют разные признаки.

Датасеты для обучения нейронных сетей. В настоящее время особой популярностью пользуются алгоритмы, основанные на нейронных сетях и глубоком обучении. Такие алгоритмы применяются для анализа большого количества сложных данных с неопределенными признаками. Анализ задач машинного обучения для определения признаков сетевой разведки показал, что наиболее подходящими являются алгоритмы классификации, регрессии, прогнозирования, кластеризации и сокращения размерности. Их эффективность можно увеличить путем использования групп одинаковых алгоритмов, объединенных для исправления ошибок друг друга. При этом несколько алгоритмов обучения могут показать результат выше, чем каждый из них в отдельности. Обычно объединяют сильно зависящие от входных данных алгоритмы (в частности, регрессию и дерево принятия решений). Это позволяет стабилизировать полученный результат, несмотря на возможное наличие сильных искажений в множестве входных объектов.

Следует отметить два вида совместного использования алгоритмов: бэггинг (bootstrap aggregating) и бустинг (boosting). *Бэггинг* состоит из параллельных однотипных алгоритмов, обученных на разных выборках одного множества входных объектов, и выводит усредненный результат их работы. Основным преимуществом бэггинга является возможность его применения в масштабе реального времени [4]. *Бустинг* состоит из последовательно выполняемых однотипных алгоритмов, обученных на специальных выборках множества входных объектов и выводящих окончательный результат. Каждая специальная выборка включает исходные объекты множества и части данных, на которых алгоритм на предыдущем шаге отработал неправильно [5]. В сравнении с бэггингом бустинг показывает лучший результат, но процесс его выполнения занимает больше времени.

В качестве дополнения к указанным методам исследовалась эффективность многослойного персептрона в связи с возможностью его использования для анализа признаков сетевой разведки в сравнении со сверточными и рекуррентными нейронными сетями. *Многослойный персептрон* (multilayer perceptron, MLP) – простейшая, но эффективная нейронная сеть с несколькими скрытыми слоями, предназначенная для анализа большого количества данных с определяемыми признаками.

Эффективность методов машинного обучения во многом базируется на объеме и качестве данных, используемых для тренировки моделей алгоритмов. Согласно опубликованному отчету консалтинговой компании International Data Corporation «The Digitization of the World from Edge to Core» (URL: https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf) объем накопленных данных для алгоритмов машинного обучения на 2018 г. составляет 33 ЗБ, а к 2025 г. их число может вырасти до 175 ЗБ.

Наборы данных, содержащие большое количество размеченных показателей, пригодных для машинного обучения, называются *датасетами*. Существуют следующие датасеты, связанные с обнаружением аномалий сетевого трафика:

Network Intrusion Detection (URL: https://www.kaggle.com/sampadab17/network-intrusion-detection/);

UNSW_NB15 (URL: https://www.kaggle.com/mrwellsdavid/unsw-nb15/);

2019 Trendmicro CTF Wildcard 400 (URL: https://www.kaggle.com/hawkcurry/2019-trendmicro-ctf-wildcard-400/);

Kitsune Network Attack (URL: https://www.kaggle.com/ymirsky/network-attack-dataset-kitsune/);

NSL-KDD (URL: https://www.unb.ca/cic/datasets/nsl.html);

NTwPSA (Loughborough University Network Traffic with Port Scanning Attack; URL: https://figshare.com/articles/dataset/Loughborough_University_Network_Traffic_with_Port_Scanning_Attack/4630282/3).

Каждый датасет содержит в себе конечное множество метрик, характерных для конкретного события «объект – ответ». Последние три датасета включают данные для обнаружения сетевой разведки. Наиболее популярным датасетом для обнаружения аномалий сетевого трафика и сетевой разведки является NSL-KDD, который представляет собой новую версию устаревшего датасета KDD99, разработанного в 1999 г. для проведения сравнительного тестирования алгоритмов в IPS/IDS, и содержит около 150 тыс. событий. В NSL-KDD устранены отдельные недостатки KDD99 [6], удалены события, влияющие на частотные характеристики (избыточность, дублирование), и создан более продуманный подход к формированию тестовой и обучающей выборок. События NSL-KDD – это последовательности сегментов (дейтаграмм) TCP и UDP, а также ICMP-пакетов, полученные в определенный промежуток времени. Для каждого события проведен расчет 41 метрики, добавлена метка атаки и уровень сложности. Эти датасеты на практике имеют ограниченное применение вследствие малой или, наоборот, избыточной информативности отдельных метрик. Так, метрики датасета NSL-KDD root, file creations, shells и др. обладают избыточной информативностью и не используются при обнаружении признаков сетевой разведки, а также сложны в определении. В расчете метрик датасета NTwPSA mac_src_second и mac_dst_second тоже нет необходимости.

Определение метрик, характерных для сетевой разведки. Для создания эффективного датасета необходимо определить информацию, доступную для извлечения из заголовков протоколов IP, TCP и UDP. Наиболее информативными полями являются: полная длина IP-пакета, IP-адрес источника, IP-адрес назначения, исходящий порт и порт назначения. Для протокола TCP дополнительно внесено поле «Код», описывающее тип запроса при рукопожатии (SYN, ACK, FIN и т. д.). При этом анализ полей должен проводиться не для одного события, а для выборки, что позволяет сформировать связь между событиями и контекст. Предполагается, что наиболее информативными будут метрики, состоящие из отношения количества содержащих отдельный параметр событий к общему количеству событий в выборке. Общее количество событий в выборке сложно определить теоретически, поэтому на начальном этапе используется дифференциация множества событий на совокупности по 30 событий.

Кроме метрик serror rate и dst host rate, предложенных в NSL-KDD, а также port_dst_second от NTwPSA требуется рассмотреть метрики, напрямую связанные с признаками сетевой разведки. В качестве таких метрик могут выступать коды, содержащиеся в TCP-заголовке. Из них наиболее часто используются коды FIN, SYN, ACK, MAIMON, XMAS или же NULL. Данные метрики также необходимо представить в виде отношения, что позволит точнее определить принадлежность события к сетевой разведке. Кроме того, требуется рассчитать

отношение других кодов к выборке, что связано с возможностью проведения злоумышленником ранее неизвестных типов атак. Итог анализа и расчета метрик приведен в табл. 2 [7].

Таблица 2
Формулы расчета и описание предлагаемых метрик

Table 2
Calculation formulas and description of the proposed metrics

Метрика *Metric*	Формула расчета *Calculation formula*	Описание *Description*
count	count_ip_events(all, all) / count_all_events	Отношение количества отправленных сегментов (дейтаграмм) с одного IP-адреса к общему количеству полученных сегментов (дейтаграмм) с различных IP-адресов. Позволяет определить «шумные» хосты
udp	count_ip_events(udp, all) / count_ip_events(all, all)	Отношение количества отправленных дейтаграмм с одного IP-адреса к общему количеству отправленных с этого же IP-адреса сегментов (дейтаграмм). Определяет UDP-трафик
tcp	1.0 - udp	Отношение количества отправленных сегментов с одного IP-адреса к общему количеству отправленных с этого же IP-адреса сегментов (дейтаграмм). Определяет TCP-трафик
tcp_syn	count_ip_events(tcp, syn) / count_ip_events(tcp, all)	Отношение количества отправленных с флагом SYN сегментов с одного IP-адреса к общему количеству отправленных с этого же IP-адреса сегментов. Определяет объем сегментов с флагом SYN
tcp_ack	count_ip_events(tcp, ack) / count_ip_events(tcp, all)	Отношение количества отправленных с флагом ACK сегментов с одного IP-адреса к общему количеству отправленных с этого же IP-адреса сегментов. Определяет объем сегментов с флагом ACK
tcp_fin	count_ip_events(tcp, fin) / count_ip_events(tcp, all)	Отношение количества отправленных с флагом FIN сегментов с одного IP-адреса к общему количеству отправленных с этого же IP-адреса сегментов. Определяет объем сегментов с флагом FIN
tcp_null	count_ip_events(tcp, null) / count_ip_events(tcp, all)	Отношение количества отправленных с флагом NULL сегментов с одного IP-адреса к общему количеству отправленных с этого же IP-адреса сегментов. Позволяет определить объем сегментов с флагом NULL
tcp_xmas	count_ip_events(tcp, xmas) / count_ip_events(tcp, all)	Отношение количества отправленных с флагом XMAS сегментов с одного IP-адреса к общему количеству отправленных с этого же IP-адреса сегментов. Определяет объем сегментов с флагом XMAS
tcp_maimon	count_ip_events(tcp, maimon) / count_ip_events(tcp, all)	Отношение количества отправленных с флагом MAIMON сегментов с одного IP-адреса к общему количеству отправленных с этого же IP-адреса сегментов. Определяет объем сегментов с флагом MAIMON
tcp_other	1.0 - tcp_syn - tcp_ack - tcp_fin - tcp_null - tcp_xmas - tcp_maimon	Отношение количества отправленных сегментов с другими флагами с одного IP-адреса к общему количеству отправленных с этого адреса сегментов. Определяет объем сегментов с другими флагами
uniq_ports	count_uniq_ports / count_ip_events(tcp, all)	Отношение количества уникальных портов, на которые были отправлены сегменты с одного IP-адреса, к общему количеству отправленных с этого адреса сегментов. Определяет перебор портов
flag	–	Флагу выставляется значение единица, если обнаружена сетевая разведка, или значение ноль, если трафик нормальный

Методика проведения эксперимента. Для создания датасета, основанного на реальном трафике, была выбрана топология, использованная для разработки датасета NTwPSA. С небольшими модификациями она внедрена на базе тестового сегмента корпоративной сети РУП «Белтелеком», что позволило создать реалистичный фоновый трафик (рис. 1).

Особенностью данной топологии является наличие двух атакующих хостов, проводящих сетевую разведку с различных ОС с использованием разного ПО, а также встроенного средства мониторинга и хранилища на самом тестовом ПК.

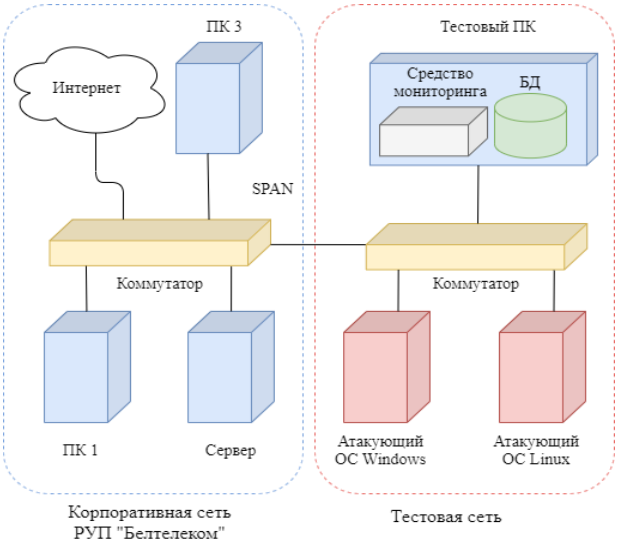

Рис. 1. Топология локальной сети для разработки датасета

Fig. 1. Local network topology for dataset development

В качестве средства мониторинга тестового ПК использовались разработанный для этой цели на языке Python версии 3 программный модуль (URL: https://github.com/ Moon1705/ dissertation) и открытая библиотека sklearn. Модуль разработан с применением свободно распространяемого ПО и находится в открытом доступе под лицензией MIT.

Изначально происходит импортирование библиотек json и numpy, предназначенных для работы с JSON-объектами и математическими исчислениями соответственно. После этого выполняется импортирование следующих классов из библиотеки sklearn: LabelEncoder для нормализации множества ответов Y; train_test_split для разделения датасета на обучающую и тестовую выборки; LogisticRegression для создания логистической регрессии; QuadraticDiscriminantAnalysis для создания алгоритма квадратичного дискриминантного анализа; SVC для реализации метода опорных векторов; KNeighborsClassifier в качестве метода ближайших соседей; GaussianNB для создания «наивного» байесовского классификатора; DecisionTreeClassifier для разработки дерева принятия решений; MLPClassifier для создания нейронной сети прямого распределения; BaggingClassifier для повышения точности алгоритма путем использования нескольких алгоритмов параллельно; AdaBoostClassifier для повышения точности алгоритма путем последовательного использования нескольких алгоритмов.

Далее происходит загрузка созданного JSON-датасета в оперативную память ПК и конвертирование его в двухмерную матрицу признаков. На данном этапе множество ответов представлено в формате строки (good, bad) и для дальнейшей работы требуется перевести их в бинарный формат (1, 0), что реализуется использованием класса LabelEncoder. Матрица признаков, содержащая перекодированные ответы, с помощью класса train_test_split разделяется на обучающее (80 %, или 800 событий) и тестовое (20 %, или 200 событий) множества с помощью функции псевдослучайных чисел. Анализ созданных множеств показал наличие 201 события сетевой разведки в обучающей выборке (25 %) и 49 событий в тестовом множестве (25 %), что позволяет судить о сбалансированности выборок.

Для генерации более реалистичного трафика на тестовом ПК был поднят статический сайт на основе веб-сервера nginx на 80-м порте. Также с самого компьютера происходили частые выходы в сеть Интернет на различные информационные ресурсы, что позволило создать стандартный трафик на тестируемом компьютере. Сам процесс сетевой разведки осуществлялся с использованием утилит Nmap (Windows) и Masscan (Linux). При этом компьютер с ОС Linux настроен таким образом, что через него можно провести Idle scan (зомби-сканирование). Параметры для представленных утилит приведены в табл. 3.

Таблица 3
Настройки утилит сетевой разведки

Table 3
Settings of network intelligence utilities

Утилита *Utility program*	Тип сканирования *Scan type*	Порты *Ports*	Дополнительные опции *Additional options*
Nmap	SYN (-sS)	1–1024	-Pn
	TCP (-sT)	1024–2048	
	UDP (-sU)	2048–3072	
	Null (-sN)	3072–4096	
	FIN (-sF)	4096–5120	
	Xmas (-sX)	5120–6144	
	ACK (-sA)	6144–7168	
	Windows (-sW)	7168–8192	
	Maimon (-sM)	8192–9216	
	Idle (-sI)	9216–10240	
	SYN (-sS)	10240–11264	-sV -O
	SYN (-sS)	–	-F -sV
Masscan	SYN	1–1024	-rate 1024

Результатом проведения указанных типов сканирования тестового компьютера с включенным средством мониторинга стал текстовый файл, содержащий порядка 500 событий сетевой разведки (исключая легитимный трафик). Приведем фрагмент файла, содержащий описание такого события:

```
[
 {
 "count": 0.4878, "tcp": 0.7919191, "tcp_ack": 0.5619191,
"tcp_fin": 0.0, "tcp_maimon": 0.24, "tcp_null": 0.0, "tcp_other": 0.0,
"tcp_syn": 0.0, "tcp_xmas": 0.0, "udp": 0.2080808, "uniq_ports": 0.94,
 },
 …
].
```

Для удобства применения значения каждой переменной были округлены до сотых, добавлено поле detection: bad (good), указывающее, что данное событие является сетевой разведкой, изменены наименования переменных (добавлено ratio_) и извлечены только уникальные события. В результате было получено 250 событий сетевой разведки и 750 событий легитимного трафика.

Результаты исследования эффективности алгоритмов машинного обучения и разработанного модуля. Все события являются уникальными и нормализованными. Графическое представление созданного датасета с использованием метода главных компонент (principal component analysis, PCA) показано на рис. 2.

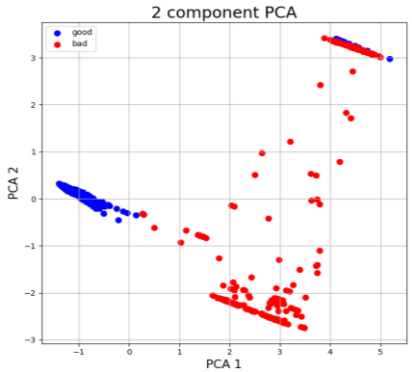

Рис. 2. Графическое представление датасета с помощью метода главных компонент
Fig. 2. Graphical representation of a dataset using the principal component analysis

На рис. 2 можно наблюдать дифференциацию датасета на три множества. Два из них представляют собой разделенный легитимный трафик и трафик сетевой разведки. Третье множество вызывает интерес по причине отсутствия явного разделения между аномалией, созданной злоумышленником, и обычным трафиком пользователей. Предположительно, это UDP-трафик, так как анализ числовых показателей датасета показал незначительное отличие bad-трафика от good-трафика. В общем случае сложности при анализе не возникнут, так как на корпоративных серверах редко допускается UDP-трафик. Тем не менее при разработке модели машинного обучения данный факт необходимо учесть и подобрать метод классификации, дающий минимальное отклонение в представленном множестве [8].

Из приведенных в табл. 4 данных видно, что на 100 % правильный результат дали четыре из семи исследуемых алгоритмов: SVM, K Neighbors, Decision Tree и MLP. При этом частично на их эффективность повлиял выбор отдельных параметров. Так, при использовании линейного и сигмоидного ядер в методе опорных векторов эффективность алгоритма уменьшается.

Таблица 4
Результаты исследования эффективности алгоритмов машинного обучения

Table 4
Results of the study of the effectiveness of machine learning algorithms

Алгоритм *Algorithm*	Параметр *Parameter*	Значение *Value*	Точность, % *Accuracy, %*	Время обучения, мс *Training time, ms*
Logistic Regression	solver	sag	97,0	4,052
		lbfgs	97,0	78,030
		liblinear	97,0	1,704
		saga	97,0	7,107
QDA	reg_param	0,0	95,0	58,084
		0,5	95,0	60,120
SVM	kernel	rbf	100,0	2,112
		poly	100,0	1,830
		linear	98,0	2,326
		sigmoid	95,5	5,304
K Neighbors	n_neighbors	3	100,0	1,458
	n_neighbors	5	100,0	1,409
	n_neighbors	10	100,0	1,351
	n_neighbors weights	3 distance	100,0	1,587
	n_neighbors weights	5 distance	100,0	1,504
	n_neighbors weights	10 distance	100,0	1,380
GaussianNB	var_smoothing	1e-9	97,5	0,892
	var_smoothing	1e-8	97,5	0,868
Decision Tree	criterion	gini	100,0	1,260
	criterion	entropy	100,0	1,086
	criterion splitter	gini random	100,0	0,912
	criterion splitter	entropy random	100,0	0,981
	criterion max_features	gini auto	99,5	0,888
	criterion max_features	entropy auto	99,5	1,653
	criterion splitter max_features	gini random auto	100,0	2,078
	criterion splitter max_features	entropy random auto	100,0	0,953

Окончание таблицы 4

End of table 4

Алгоритм *Algorithm*	Параметр *Parameter*	Значение *Value*	Точность, % *Accuracy, %*	Время обучения, мс *Training time, ms*
MLP	activation	logistic	97,5	922,823
	hidden_layer_sizes	(5, 3)		
	max_iter	500		
	solver	adam		
	activation	logistic	100,0	52,913
	hidden_layer_sizes	(5, 3)		
	max_iter	500		
	solver	lbfgs		
	activation	logistic	98,0	1263,257
	hidden_layer_sizes	(5, 3)		
	max_iter	1000		
	solver	adam		
	activation	logistic	100,0	55,881
	hidden_layer_sizes	(5, 3)		
	max_iter	1000		
	solver	lbfgs		
	activation	relu	97,5	910,756
	hidden_layer_sizes	(5, 3)		
	max_iter	500		
	solver	adam		
	activation	relu	98,0	744,577
	hidden_layer_sizes	(5, 3)		
	max_iter	500		
	solver	sgd		
	activation	relu	100,0	43,871
	hidden_layer_sizes	(5, 3)		
	max_iter	500		
	solver	lbfgs		
	activation	relu	99,0	1812,323
	hidden_layer_sizes	(5, 3)		
	max_iter	1000		
	solver	adam		
	activation	relu	98,0	1026,553
	hidden_layer_sizes	(5, 3)		
	max_iter	500		
	solver	sgd		
	activation	relu	100,0	44,346
	hidden_layer_sizes	(5, 3)		
	max_iter	500		
	solver	lbfgs		
	activation	relu	100,0	45,153
	hidden_layer_sizes	(7,)		
	max_iter	500		
	solver	lbfgs		
	activation	relu	97,0	552,070
	hidden_layer_sizes	(7,)		
	max_iter	1000		
	solver	adam		
	activation	relu	96,5	779,355
	hidden_layer_sizes	(7,)		
	max_iter	1000		
	solver	sgd		
	activation	relu	100,0	21,082
	hidden_layer_sizes	(7,)		
	max_iter	1000		
	solver	lbfgs		

Отдельным важным параметром является скорость работы алгоритма, определяющая возможность работы алгоритма в состоянии потока. Поэтому все алгоритмы с соответствующими параметрами, давшие наилучший результат, дополнительно оцениваются по скорости работы. Наиболее быстрым методом, обучающимся за 0,912 мс, является дерево принятия решения (Decision Tree) с параметрами criterion = gini и splitter = random. Дерево принятия решений было реализовано в виде программного кода на языке Python версии 3 (URL: https://github.com/Moon1705/dissertation). В результате тестирования данного кода значение метрики «точность» составило 100 %. Между тем существуют алгоритмы «наивного» байесовского классификатора (GaussianNB) и дерева принятия решений (Decision Tree) с параметрами criterion = = gini и max_features = auto, время работы которых меньше, чем у дерева принятия решения с параметрами criterion = gini и splitter = random, но с точностью менее 100 %. Было проведено улучшение этих алгоритмов с использованием бэггинга и бустинга, выполнен анализ эффективности и скорости работы после модернизации.

В табл. 5 представлены скорость и точность работы алгоритма до и после применения процедур бустинга и бэггинга, а также скорость работы разработанной функции. Из приведенных данных видно, что использование процедур бустинга и бэггинга не сказалось на эффективности «наивного» байесовского классификатора, это можно объяснить вероятностной природой его решения. Для дерева принятия решений применение процедур бустинга и бэггинга позволило достичь значения точности 100 % при увеличении времени проверки с 0,277 до 0,407 мс (бустинг) и 0,993 мс (бэггинг). Дерево принятия решений с параметрами criterion = gini и splitter = = random также показало точность 100 % при времени проверки 0,333 мс. Указанное время удалось уменьшить с 0,333 до 0,147 мс путем использования программной реализации (вместо обученной модели). Таким образом, метод дерева принятия решений с параметрами criterion = = gini и splitter = random показал лучшие результаты. Дополнительным плюсом использования программной реализации является меньшее количество потребляемых ресурсов.

Таблица 5
Эффективность процедур бустинга и бэггинга

Table 5
Efficiency of boosting and bagging procedures

Алгоритм *Algorithm*	Параметр *Parameter*	Значение *Value*	Метод *Method*	Точность, % *Accuracy, %*	Время обучения, мс *Training time, ms*	Время проверки, мс *Check time, ms*
GaussianNB	var_smoothing	1e-9	normal	97,5	1,245	0,421
			boosting	97,5	4,882	0,766
			bagging	97,5	8,709	1,932
		1e-8	normal	97,5	0,724	,387
			boosting	97,5	6,190	0,916
			bagging	97,5	8,531	1,886
Decision Tree	max_features	auto	normal	99,5	0,721	0,277
	criterion	gini				
	max_features	auto	boosting	100,0	1,606	0,407
	criterion	gini				
	max_features	auto	bagging	100,0	8,114	0,993
	criterion	gini				
	splitter	random	normal	100,0	0,833	0,333
	criterion	gini				
	splitter	random	code	100,0	0	0,147
	criterion	gini				

Результаты анализа практических испытаний разработанного модуля показали факт сработки на всех типах сканирования сетей с помощью утилит Nmap и Masscan. В процессе испытаний были отмечены некоторые особенности. При проведении Idle scan средство мониторинга показывало IP-адрес бота, а не реальный IP-адрес злоумышленника. Тем не менее на практике Idle scan – достаточно редкое явление, так как подобные уязвимости конфигурации отключены

по умолчанию. Замечены ложные сработки (false positive), связанные с широковещательными UDP-запросами DNS- и DHCP-серверов в корпоративной сети, которые приходили на закрытые порты защищаемого сервера. Этот факт стоит принять во внимание при использовании средств защиты в корпоративной сети и добавить серверы, содержащие сетевые службы, которые отправляют широковещательные запросы в список исключенных из анализа IP-адресов (параметр excluded_ips при конфигурации ПО).

Заключение. В результате анализа и отбора наиболее актуальных метрик сформирован датасет, состоящий из 1000 уникальных событий и включающий 250 событий сетевой разведки и 750 легитимных событий. Это выгодно отличает его от датасетов, в которых легитимный трафик преобладает (например, от датасета университета Лафборо) и при использовании которых сетевая разведка будет восприниматься не как аномалия, а как погрешность. Установлено, что для оперативного анализа с применением дерева принятия решений достаточно оперировать примерно половиной определенных ранее метрик. Это позволит ускорить потоковую обработку. Результаты проведенного исследования могут быть полезны при создании актуального и эффективного датасета для выявления признаков сетевой разведки.

Рассмотрены существующие методы классификации, их особенности и способы расчета. Выделены наиболее перспективные методы выявления признаков сетевой разведки, для них проведено обучение и тестирование с использованием различных гиперпара-метров. Наилучшие результаты показал метод дерева принятия решений с параметрами criterion = gini и splitter = = random. Проведена оптимизация отдельных алгоритмов с помощью процедур бэггинга и бустинга для ускорения их работы.

В программном виде реализованы модуль обнаружения признаков сетевой разведки, алгоритм дерева принятия решения и функция для переключения режимов работы модуля (обученное дерево принятия решений или алгоритм с возможностью дообучения модели).

Вклад авторов. *Н. П. Шараев* выполнил сравнительный анализ существующих подходов обнаружения признаков сетевой разведки, сформировал датасет, разработал программный модуль и провел экспериментальные исследования. *С. Н. Петров* определил цели исследования и задачи, которые необходимо было решить для их достижения, принял участие в интерпретации и обобщении полученных результатов.

Список использованных источников

1. Гущин, Р. А. Сетевая разведка / Р. А. Гущин, К. А. Колос ; сост. В. А. Мартинович // Материалы 74-й студ. науч.-техн. конф. – Минск : БНТУ, 2018. – С. 53–54.

2. Караулова, О. А. Оценка аномалий сетевого трафика на основе циклического анализа / О. А. Караулова, Н. В. Киреева // T-comm: телекоммуникации и транспорт. – 2018. – Т. 12, вып. 11. – С. 33.

3. Брюхомицкий, Ю. А. Искусственные иммунные системы в информационной безопасности : учеб. пособие / Ю. А. Брюхомицкий. – Ростов н/Д, Таганрог : Изд-во Южн. федер. ун-та, 2019. – 147 с.

4. Кашницкий, Ю. С. Ансамблевый метод машинного обучения, основанный на рекомендации классификаторов / Ю. С. Кашницкий, Д. И. Игнатов // Интеллектуальные системы. Теория и приложения. – 2015. – Т. 19, вып. 4. – С. 37–55.

5. Халкечев, Р. В. Бустинг – еще один способ машинного обучения [Электронный ресурс] / Р. В. Халкечев // Журнал «Яндекс Практикума». – Режим доступа: https://thecode.media/boosting/. – Дата доступа: 12.06.2021.

6. Detailed analysis of the KDD CUP 99 data set / M. Tavallaee [et al.] // 2009 IEEE Symp. on Computational Intelligence for Security and Defense Applications, Ottawa, Canada, 8–10 July 2009. – Ottawa, 2009. – P. 1–6.

7. Шараев, Н. П. Выявление и анализ признаков сетевой разведки методом машинного обучения / Н. П. Шараев, С. Н. Петров // Управление информационными ресурсами : материалы XVII Междунар. науч.-практ. конф., Минск, 12 мар. 2021 г. – Минск : Академия управления при Президенте Республики Беларусь, 2021. – С. 238–240.

8. Шараев, Н. П. Выявление сетевой разведки методами машинного обучения / Н. П. Шараев, С. Н. Петров // Защита информации : сб. материалов 57-й науч. конф. аспирантов, магистрантов и студентов БГУИР, Минск, Беларусь, 19–23 апр. 2021 г. – Минск : БГУИР, 2021. – С. 34–37.

References

1. Gushchin R. A., Kolos K. A. *Network intelligence*. Materialy 74-j studencheskoj nauchno-tekhnicheskoj konferencii [*Materials of the 74th Student Scientific and Technical Conference*], sostavitel' V. A. Martinovich, Minsk, Belorusskij nacional'nyj tekhnicheskij universitet, 2018, pp. 53–54 (In Russ.).

2. Karaulova O. A., Kireeva N. V. *Estimation of network traffic anomalies based on cyclic analysis*. T-comm: telekommunikacii i transport [*T-comm: Telecommunications and Transport*], 2018, vol. 12, no. 11, p. 33 (In Russ.).

3. Bryuhomickij, Yu. A. Iskusstvennye immunnye sistemy v informacionnoj bezopasnosti. *Artificial Immune Systems in Information Security*. Rostov-on-Don, Taganrog, Izdatel'stvo Yuzhnogo federal'nogo universiteta, 2019, 147 p. (In Russ.).

4. Kashnickij Yu. S., Ignatov D. I. *An ensemble method of machine learning based on the recommendations of classifiers*. Intellektual'nye sistemy. Teoriya i prilozheniya [*Intelligent Systems. Theory and Applications*], 2015, vol. 19, no. 4, pp. 37–55 (In Russ.).

5. Halkechev R. V. *Boosting is another way of machine learning*. Zhurnal "Yandeks Praktikuma" [*Yandex Practicum Magazine*] (In Russ.). Available at: https://thecode.media boosting/ (accessed 12.06.2021).

6. Tavallaee M., Bagheri E., Lu W., Ghorbani A. Detailed analysis of the KDD CUP 99 data set. *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, Canada, 8–10 July 2009*. Ottawa, 2009, pp. 1–6.

7. Sharaev N. P., Petrov S. N. *Identification and analysis of signs of network intelligence by machine learning*. Upravlenie informacionnymi resursami : materialy XVII Mezhdunarodnoj nauchno-prakticheskoj konferencii, Minsk, 12 marta 2021 g. [*Information Resource Management: Materials of the* XVII *International Scientific and Practical Conference, Minsk, 12 March 2021*], Minsk, Akademija upravlenija pri Prezidente Respubliki Belarus', 2021, pp. 238–240 (In Russ.).

8. Sharaev N. P., Petrov S. N. *Identification of network intelligence by machine learning methods*. Zashhita informacii : sbornik materialov 57-j nauchnoj konferencii aspirantov, magistrantov i studentov BGUIR, Minsk, Belarus', 19–23 aprelja 2021 g. [*Protection of Information: Collection of Materials of the 57th Scientific Conference of Postgraduates, Undergraduates and Students of BSUIR, Minsk, Belarus, 19–23 April 2021*], Minsk, Belorusskij gosudarstvennyj universitet informatiki i radiojelektroniki, 2021, pp. 34–37 (In Russ.).

Информация об авторах

Шараев Никита Петрович, магистрант кафедры защиты информации, факультет инфокоммуникаций, Белорусский государственный университет информатики и радиоэлектроники.
E-mail: nick.moon.1705@gmail.com

Петров Сергей Николаевич, кандидат технических наук, доцент, доцент кафедры защиты информации, факультет инфокоммуникаций, Белорусский государственный университет информатики и радиоэлектроники.
E-mail: sergpetrov@inbox.ru

Information about the authors

Nikita P. Sharaev, Master Student of the Information Security Department, Faculty of Infocommunications, Belarusian State University of Informatics and Radioelectronics.
E-mail: nick.moon.1705@gmail.com

Sergei N. Petrov, Ph. D. (Eng.), Associate Professor, Associate Professor of the Information Security Department, Faculty of Infocommunications, Belarusian State University of Informatics and Radioelectronics.
E-mail: sergpetrov@inbox.ru