DOI: 10.15514/ISPRAS-2021-33(4)-3

Идентификация прозрачных, сжатых и шифрованных данных в сетевом трафике

[1,2]*А.И. Гетьман, ORCID: 0000-0002-6562-9008 <thorin@ispras.ru>*
[1]*М.К. Иконникова, ORCID: 0000-0003-1530-5133 <mikonnikova@ispras.ru>*
[1]*Институт системного программирования им. В.П. Иванникова РАН, 109004, Россия, г. Москва, ул. А. Солженицына, д. 25*
[2]*Национальный исследовательский университет «Высшая школа экономики», 101978, Россия, г. Москва, ул. Мясницкая, д. 20*

Аннотация. В статье рассматривается задача классификации сетевого трафика на три типа, в зависимости от представления данных в нём: прозрачный, сжатый и шифрованный. Описываются существующие методы классификации, служащие для разделения трафика на прозрачный и непрозрачный, сжатый и шифрованный применительно к сетевым данным и документам. На основе них выбираются методы, показавшие лучшие результаты, и производится отбор лучшей их комбинации и вывод единого результата с применением методов машинного обучения (случайный лес). Также исследуется вопрос классификации потоков как единого целого и предлагается новый, отличный от существующих способ. Завершается статья анализом направлений для дальнейших исследований.

Ключевые слова: анализ сетевого трафика; классификация сетевого трафика; машинное обучение**;** шифрованный трафик

Для цитирования: Гетьман А.И., Иконникова М.К. Идентификация прозрачных, сжатых и шифрованных данных в сетевом трафике. Труды ИСП РАН, том 33, вып. 4, 2021 г., стр. 31-48. DOI: 10.15514/ISPRAS–2021–33(4)–3

Identification of transparent, compressed and encrypted data in network traffic

[1,2]*A.I. Getman, ORCID: 0000-0002-6562-9008 <thorin@ispras.ru>*
[1]*M.K. Ikonnikova, ORCID: 0000-0003-1530-5133 <mikonnikova@ispras.ru>*
[1]*Ivannikov Institute for System Programming of the Russian Academy of Sciences, 25, Alexander Solzhenitsyn st., Moscow, 109004, Russia*
[2]*National Research University, Higher School of Economics, 20, Myasnitskaya Ulitsa, Moscow, 101978, Russia*

Abstract. The article is dedicated to the problem of classifying network traffic into three categories: transparent, compressed and opaque, preferably in real-time. It begins with the description of the areas where this problem needs to be solved, then proceeds to the existing solutions with their methods, advantages and limitations. As most of the current research is done either in the area of separating traffic into transparent and opaque or into compressed and encrypted, the need arises to combine a subset of existing methods to unite these two problems into one. As later the main mathematical ideas and suggestions that lie behind the ideas used in the research done by other scientists are described, the list of the best performing of them is composed to be combined together and used as the features for the random forest classificator, which will divide the provided traffic into three classes. The best performing of these features are used, the optimal tree parameters are chosen and, what's more, the initial three class classifier is divided into two sequential ones to save time needed for classifying in case of transparent packets. Then comes the proposition of the new method to classify

the whole network flow as one into one of those three classes, the validity of which is confirmed on several examples of the protocols most specific in this area (SSH, SSL). The article concludes with the directions in which this research is to be continued, mostly optimizing it for real-time classification and obtaining more samples of traffic suitable for experiments and demonstrations.

Keywords: network traffic analysis; network traffic classification; machine learning; encrypted traffic

For citation: Getman A.I., Ikonnikova M.K. Identification of transparent, compressed and encrypted data in network traffic. Trudy ISP RAN/Proc. ISP RAS, vol. 33, issue 4, 2021. pp. 31-48 (in Russian). DOI: 10.15514/ISPRAS–2021–33(4)–3

1. Введение

Данные, передаваемые в сетевых пакетах, неоднородны по своему представлению. Они могут содержать в себе тексты, написанные на естественном языке, передавать данные через строго структурированные, но всё же человекочитаемые протоколы, содержать документы различных форматов. Также, содержимое пакетов может быть сжато для уменьшения размера передаваемых данных и оптимизации использования каналов связи или зашифровано для обеспечения конфиденциальности.

Разные представления передаваемых данных требуют разных видов их обработки и анализа. Например, сигнатурный поиск может помочь определить протокол передачи сообщения и тип приложенных к сообщению документов, но сжатый трафик нужно предварительно разархивировать. Зашифрованный трафик не поддаётся такому способу анализа, поэтому его классификация и обработка требуют особых подходов.

Задача определения представления передаваемых данных имеет разнообразные практические приложения. Такие сведения могут использоваться в системах обнаружения атак. Наличие сжатого или зашифрованного трафика в ситуации, для которой обычно используется только прозрачный трафик, может свидетельствовать, например, о работе сервера команд ботнета [1].

Сведения о шифровании данных внутри пакетов могут применяться для более точного определения протоколов и типов содержимого, что, в свою очередь, служит для повышения качества обслуживания, выделения приоритетного трафика [2] и анализа устройства и эффективности сети в целом. Эти данные могут использоваться как сами по себе, так и в качестве только одного из признаков в глобальной системе принятий решений.

Также, в зависимости от состава передаваемых данных, могут применяться разные требования к их представлению для обеспечения конфиденциальности личных данных пользователей [3]. Так, медицинские данные не могут передаваться в открытом или сжатом виде - тогда их смогут прочитать злоумышленники.

Архиваторы обычно добавляют свои признаки и метки в файл, чтобы можно было легко понять его формат и алгоритм сжатия, но по сети документы передаются по частям, разделённым между множеством пакетов, и нельзя гарантировать, что система анализа сможет получить и извлечь нужные данные. На практике отличить фрагмент сжатых данных от шифрованных – нетривиальная задача, которой посвящено не одно исследование.

В целом, можно выделить три основных класса представления данных:

- нешифрованный (transparent),
- сжатый (compressed),
- шифрованный (encrypted).

В некоторых исследованиях не делается различий между вторым и третьим классом из-за схожести некоторых их свойств. Тогда они объединяются в единый класс непрозрачного (opaque) трафика.

Таким образом, если производить предварительную обработку трафика его разделением на указанные выше классы, это даст нам данные для решения некоторых из поставленных задач

или позволит к каждому из них применять уже целенаправленно свои методы анализа, что сократит время и трудоёмкость обработки получаемых данных (рис. 1).

Рис.1 Схема классификации и обработки данных
Fig.1 Data classification and processing scheme

Такая классификация может проводиться как для отдельных пакетов, так и для потока пакетов в целом, где поток – это подмножество пакетов, определяемое пятёркой значений <IP адрес отправителя, IP адрес получателя, номер порта отправителя, номер порта получателя, транспортный протокол>. В зависимости от задачи может быть предпочтителен тот или иной способ.

Отдельные пакеты потока могут иметь разный тип. Например, для протокола HTTPS можно выделить следующие части потока.

- TCP рукопожатие: пакеты не содержат полезной нагрузки, поэтому формально не относятся ни к одному из классов.
- Обмен ключами: содержимое пакетов нешифрованное.
- Передача сообщений: зашифрованный трафик.

Таким образом, для классификации потока как единого целого нужно установить специальные правила соотношения между встречающимися в нём типами пакетов.

В зависимости от стоящей перед нами практической задачи анализ трафика может производиться как в онлайн-режиме, на лету, так и офлайн, постфактум. Это определяет, какое время мы можем тратить на классификацию отдельного пакета или потока, чтобы достичь компромисса между качеством полученного решения и скоростью его принятия (чтобы принимать достаточно хорошее решение с минимальной задержкой). Также, некоторые потоки могут продолжаться достаточно длительное время или вообще не заканчиваться в обозримом будущем (постоянная поддержка HTTP соединения через Keep-Alive), но их всё равно нужно уметь как-то классифицировать.

На основе всех описанных выше идей рассмотрим задачу классификации трафика по признаку его представления на три класса по потокам и в онлайн-режиме. В разд. 2 будут описаны существующие методы решения близких задач. В разд. 3 будут рассмотрены их особенности, преимущества и недостатки и выделены лучшие из них. Там же будет предложен основанный на изученном материале наш подход к решению данной задачи. В разд. 4 будут описаны эксперименты, проведённые для определения его характеристик. И, наконец, в разд. 5, будут сделаны выводы из полученных результатов.

2. Обзор существующих методов решения

К области классификации зашифрованных данных можно отнести работы, описывающие классификацию трафика, классификацию документов (файлов), а также работы о шифровании и генерации случайных значений.

В [4] предметом классификации являются потоки трафика, которые разделяются на шифрованные и нешифрованные на основе анализа первого значимого пакета (пакета,

содержащего полезную нагрузку). Для этого сначала определяется энтропия (более подробные объяснения применяемых методов будут даны в разд. 3) полезной нагрузки пакета (не менее 16 байтов), а затем проверяется доля печатных символов в первых 96 байтах. Такие ограничения были выбраны для ускорения процесса классификации. Для проверки полученных результатов использовался инструмент SPID [5], который сначала задавал истинные значения протоколов для каждого потока, а затем применялся для определения трафика, классифицированного системой как зашифрованный. Авторы указывают высокие результаты работы своей системы, но сами говорят о её неприменимости к некоторым популярным протоколам (SSH, SSL) из-за особенностей их устройства. Также, система не выделяет сжатые данные в отдельный класс.

В [6] исследователи используют свою систему чтобы отфильтровать трафик, который не может быть анализирован методами DPI. Тем самым они улучшают пропускную способность и качество работы фильтров Snort [7] за счёт предотвращения потерь значимых пакетов и отсутствия временных потерь на анализ зашифрованных пакетов, которые не могут быть анализированы таким способом. В качестве единицы классификации они выбирают пакет сетевого трафика, к которому применяются вероятностные тесты. Для тестирования использовались наборы разных типов файлов, переданные по протоколам поверх TCP на стенде, и выделенные из реальных сетевых трасс, собранных на кампусе, потоки протоколов SSL, SSH, SMTP и HTTP, причём внутри протокола HTTP было также проведено внутреннее деление пакетов на типы в соответствии с полями Content-Type и Content-Encoding. Из представленных результатов можно видеть, что предложенный авторами подход является довольно эффективным, однако, поскольку в нашем случае предпочтительной является классификация не пакетов, а потоков, он нуждается в улучшениях. Также желательно добавить отличия между шифрованным и сжатым трафиком и расширить множество изучаемых протоколов.

Авторы исследования [8] используют комбинацию трёх тестов (энтропия, критерий хи-квадрат, арифметическое среднее значений байтов), применяя их к полезной нагрузке k байтов пакетов (где k меньше размера пакета) и используя методы машинного обучения для определения пороговых значений. Эксперименты на трафике протоколов HTTP, FTP, Telnet, SSH показали лучшие результаты при использовании классификаторов CART [9] и NB [10], также CART показал наилучшее соотношение качества и скорости классификации при использовании только 32 байтов содержимого пакета. Использование сочетания разных тестов позволило разделять трафик на три типа, отделяя сжатый трафик от шифрованного. Также было предложено выбирать байты для анализа не из начала пакета, а случайным образом, что поможет лучше классифицировать разнородные пакеты.

Главной целью исследования [1] является определение блоков данных с высокой энтропией для поиска и профилировки протоколов обмена ключами, что позволит эффективнее искать ботов и потоки их команд в сети. Для этого используется сокращённая энтропия в скользящем окне, пороговые значения выбираются экспериментально. Эксперименты были проведены на выделенные tshark трассах протоколов TLS и некоторых других, отнесённых к прозрачным, а также на наборе данных трафика ботнета. Полученные результаты позволили авторам выдвинуть гипотезу, что созданный метод поможет выделять для дальнейшего исследования нестандартные случаи использования шифрования потоков, которые могут использоваться для передачи командной информации ботам в сети. Отделение шифрованного трафика от сжатого в работе не рассматривалось.

Хотя данное исследование [11] проводилось не на сетевом трафике, а на различных видах файлов (текст, аудио, видео и т.д.), оно интересно тем, что авторы ставят задачу научиться различать случаи сжатия и шифрования данных. Для этого они предлагают применять некоторые методы, которые обычно используются для тестирования генераторов случайных чисел. Из полученных результатов можно видеть, что этот метод успешно применим и что точность классификации растёт с увеличение размера анализируемых файлов (лучшая

получается при анализе 32 или 64 КБ), то есть по мере увеличения количества доступной для тестирования информации.

В [3] авторы исследуют подключённые к интернету вещей медицинские устройства и ищут исходящий от них прозрачный трафик, с целью проверки защиты конфиденциальности данных пользователей (и приходят к неутешительным выводам). Для этой цели они сравнивают использование энтропии, подхода, основанного на символах ASCII, и теста хи-квадрат, который и показывает в их работе лучшие результаты.

[12] является в некотором роде продолжением [3], здесь авторы прибегают к методам машинного обучения для классификации пакетов с высокой энтропией на шифрованные и сжатые (так как сжатие тоже не обеспечивает защиту данных пользователей). В качестве моделей машинного обучения они рассматривают различные виды искусственных нейронных сетей, из которых останавливаются на использовании для своих целей свёрточной нейронной сети, а в качестве признаков выбрана мера хи-квадрат теста, вычисленная на четверти рассматриваемого пакета. Их лучший результат составляет около 70%, что хуже, чем результаты, получаемые в аналогичных тестах другими методами.

В [13] рассматривается классификация потоков на прозрачные и непрозрачные с использованием как можно меньшего числа пакетов. Согласно исследованиям авторов, для этого достаточно найти N последовательных пакетов с высокой энтропией, а затем измерить совместную энтропию следующих M пакетов (то есть рассматривать пакеты как единое целое).

Здесь [14] была предпринята попытка определить тип содержимого (например, текст, изображение, сжатое изображение, зашифрованный текст и т.д.) применением классификатора SVM [15] к вектору энтропии и дополнительному признаку в виде частот последовательностей из 4 битов (для разделения шифрованного и сжатого трафика). Однако, основные эксперименты были проведены на разных видах файлах, а не на сетевом трафике, что оставляет открытым вопрос о полной применимости этого метода в сети.

В [16] описывается использование энтропии для отделения зашифрованных или упакованных PE файлов от обычных.

В [2] потоки VoIP классифицируются на шифрованные и сжатые с использованием набора тестов NIST (англ. The National Institute of Standards and Technology, Национальный институт стандартов и технологий США), разработанных для проверки генераторов псевдослучайных чисел [17]. В некоторых случаях (для некоторых кодеков) из-за особенностей их устройства нельзя выделить явные различия, поэтому предложено дополнительно удалять часть байтов из середины пакетов для искусственного добавления такого различия. Это предложение основано на основных свойствах шифрованного и сжатого трафика: шифрованный трафик должен быть равномерно случайным, в то время как сжатый трафик имеет определённую структуру, и удаление части байтов должно влиять на видимость этой структуры.

В [18] используется NIST для определения сжатых и зашифрованных документов.

В [19] (и расширенной версии статьи [20]) для классификации шифрованного и популярных форматов сжатого трафика используется нейронная сеть со значениями функции плотности вероятности поинтервального распределения значений байтов в фрагментах данных в качестве признаков. По результатам экспериментов на наборе файлов этот метод показывает высокие качество и скорость работы, но для хороших результатов минимальный размер анализируемого фрагмента ограничен снизу примерно 2 КБ.

Таким образом, из описания существующих исследований данной задачи можно видеть, что, хотя многие из подзадач отдельно уже были решены, нет единого решения задачи классификации потоков сетевого трафика на три типа (нешифрованный, сжатый, шифрованный) в онлайн режиме с достаточной скоростью и точностью и на основе небольшого количества данных, анализируемых в пакетах потока.

3. Подходы к решению задачи определения представления трафика

Условно, в решаемой задаче можно выделить следующие подзадачи.

1) Классификация пакетов на прозрачные (нешифрованные) или непрозрачные (сжатые или шифрованные).
2) Классификация непрозрачных пакетов на сжатые и шифрованные.
3) Классификация целого потока на основе входящих в него пакетов или их части.

Первые две подзадачи могут решаться одновременно. Опишем для каждой из подзадач существующие методы её решения и выберем методы, показавшие свою эффективность в существующих исследованиях и соответствующие нашей постановке задачи.

3.1 Классификация пакетов на прозрачные и непрозрачные

Среди методов и метрик, использующихся для решения этой задачи, можно выделить следующие.

3.1.1 Использование особенностей кодировки

В кодировке ASCII печатные символы – это значения 1 байта от 32 до 127. При равномерном распределении вероятностей появления каждого из символов в сообщении, свойственном шифрованному и, чуть в меньшей степени, сжатому трафику, их доля составила бы примерно 37.5%. Однако, для нешифрованного трафика их процент сильно увеличивается, поэтому в качестве порогового значения можно выбрать значительно более высокое значение (например, в [4] авторами было выбрано 75%).

Иногда, для облегчения процесса подсчёта первые 32 байта не выделяются в отдельную категорию и все байты меньше 128 считаются принадлежащими прозрачному (читаемому) трафику. Это может ускорить процесс вычислений, но оказать влияние на точность, например, при возникновении длинных цепочек нулевых символов, не свойственных прозрачному трафику.

3.1.2 Арифметическое среднее

Аналогично предыдущему варианту, можно считать не долю байтов с определёнными значениями, а среднее арифметическое значение всех байтов в последовательности [8]. Для равномерного распределения, это значение будет близко к 127.5, для прозрачных данных, соответственно, меньше.

3.1.3 Энтропия

Эта мера, введённая Шенноном [21] используется для численного выражения значения неопределённости, в частности характеризует непредсказуемость появления какого-либо символа в последовательности. Для структурированных данных и данных на естественном языке такая неопределённость ниже, в то время как сжатые и шифрованные данные по своей природе характеризуются высокой энтропией. Для шифрованного трафика это связано с необходимостью представить результат случайным на вид, чтобы исключить возможность простой расшифровки на основе статистического анализа текста. Для сжатого трафика это связано с максимальным удалением повторяющихся конструкций, чтобы избежать избыточности в данных [2]. Из этого видно, что энтропия может быть эффективна для классификации трафика на прозрачный и непрозрачный, но не для классификации непрозрачного трафика.

Для алфавита $\Sigma = \{0, 1, \dots, m-1\}$ и распределения $p = (p_i)_{i \in \Sigma}$:
$H(p) = -\sum_{i=0}^{m-1} p_i \log p_i$ (формула Шеннона для энтропии, где p_i – вероятности событий из пространства Σ).

Если ω - слово длины N в алфавите Σ , а n_i - количество букв i в слове, то можно вычислить частоту буквы i $f_i = \frac{n_i}{N}$ и энтропию слова как $\widetilde{H}_N^{\mathrm{MLE}}(w) = -\sum_{i=0}^{m-1} f_i \log f_i$, где MLE (Maximum Likelihood Estimator) – метод максимального правдоподобия.

Такое приближение будет сходиться к значению энтропии при $N \to \infty$. Соответственно, это приближённое вычисление энтропии даёт близкое к настоящему значение только при достаточной относительно размера алфавита длине последовательности. В частности, в [1] приводятся расчёты, что для 256 значений байтов потребовались бы примеры длиной около 2000 байтов, что превышает обычное значение MTU.

3.1.4 Сокращённая энтропия

В статьях [22] и [23] рассматриваются теоретические основы энтропии, строятся зависимости вычисленных методом максимального правдоподобия приближенных значений энтропии и реальных значений, описываются корректоры, применяемые для устранения подобных расхождений. Для большего удобства работы с ограниченными размерами сетевых пакетов вводится сокращённая (N-truncated) энтропия, которая определяется как среднее энтропии, посчитанной методом максимального правдоподобия, примеров среди всех слов длины N, выбранных случайным образом в соответствии с вероятностями символов заданного алфавита.

Например, для N=32 (что значительно меньше 256), это значение будет равно 4.87816 с доверительными интервалами $\pm$4×0.081156. Следовательно, для каждых фиксированных N и m (размер алфавита, в данной задаче m=256), можно вычислить такое значение по специальной формуле и сравнивать вычисляемое значение H^{MLE} уже с ним. Если получено значение ниже этой заданной границы, то фрагмент считается прозрачным. Чем больше N, тем меньше доверительный интервал.

3.1.5 Проверка отношений вероятностей

В статье [6] исследуются примеры статистических тестов для проверки гипотез, нулевой H_0 (о том, что распределение равномерно) и альтернативной H_1 (о том, что большинство байтов имеет значения меньше 128, процент байтов, значение которых должно быть меньше 128, параметризуется значением δ). В ходе экспериментов лучшие результаты дало использование последовательной проверки отношения вероятностей (sequential probability ratio test). Этот тест устроен следующим образом:

$\Lambda(\bar{X})$ - это отношение вероятности X при альтернативной гипотезе к вероятности X при нулевой гипотезе.

$\alpha = P(\text{accept } H_1 | H_0)$ – желаемая вероятность false negative результата.

$\beta = P(\text{accept } H_0 | H_1)$ – желаемая вероятность false positive результата.

$g_0(m) = \frac{\beta}{1-\alpha}$ и $g_1(m) = \frac{1-\beta}{\alpha}$ – пороговые значения.

Тогда процедура последовательной проверки отношения вероятностей выглядит следующим образом: на каждой итерации m принимается решение

$$\begin{cases} \text{accept } H_0 \text{ if } \Lambda_m(X_1, X_2, \dots, X_{\mathrm{m}}) \le g_0(m) \\ \text{accept } H_1 \text{ if } \Lambda_m(X_1, X_2, \dots, X_{\mathrm{m}}) \ge g_1(m) \\ \text{continue otherwise} \end{cases}$$

Для того, чтобы гарантировать сходимость метода, рассматривались два способа: ограничить длину анализируемой последовательности байтов или уменьшать пороговые значения в ходе анализа последовательности. Первый способ оказался предпочтительным. Если заданное число байтов проанализировано, а пороговое значение ни в одну из сторон не преодолено, выдаётся тот результат, к которому текущее значение ближе.

В реализации данного алгоритма SPRT использует правдоподобие количества байтов, меньших 128, по распределению Бернулли. Получая пакет на вход, алгоритм сначала пропускает заданное как параметр программы количество байтов $offset$, а затем в каждом шаге анализирует количество байтов $stepsize$, рассчитывая правдоподобность двух описанных выше гипотез по следующей формуле:

$L(\Theta_i | x_j) = C_n^k * \Theta_i^k * (1 - \Theta_i)^{n-k}$, где:

- $n = stepsize$ – количество байтов, проверяемых за один шаг;
- k – количество байтов, меньших 128, в одном шаге;
- Θ_i – доля байтов, меньших 128, при биноминальном распределении, относящемся к гипотезе i;
- x – набор байтов, анализируемых на шаге j.

Затем рассчитывается относительное правдоподобие для двух гипотез

$$\Lambda_j = \frac{L(\Theta_1 | x_j)}{L(\Theta_0 | x_j)}$$

и сумма логарифмов относительных правдоподобий обновляется следующим образом:

$S_j = S_{j-1} + \ln \Lambda_j$ (при $S_0 = 0$)

Для g_0 и g_1 также берутся логарифмы. Если после анализа заданного как параметр $maxBytes$ количества байтов, решение не принято, алгоритм останавливается и принимает одну из гипотез на основании текущего значения S_j.

3.1.6 Критерий хи-квадрат

Критерий хи-квадрат сравнивает наблюдаемую частоту каждого символа f_i с ожидаемой в случае равномерного распределения e_i. Результат вычисляется по формуле

$$\chi^2 = \sum_{i=1}^{n} \frac{(f_i - e_i)^2}{e_i}$$

Чем сильнее реальные частоты символов отличаются от ожидаемых, тем выше значение χ^2. Таким образом, можно установить порог для разделения данных на прозрачные и непрозрачные. Это метод показал лучшие результаты из нескольких методов в [3].

3.2. Классификация непрозрачных пакетов

И сжатые, и шифрованные пакеты показывают более высокие значения энтропии, чем прозрачный трафик, благодаря чему их можно выделить в отдельный класс. Но при этом на первый взгляд оба эти типа представления выглядят как случайный набор значений, и их разделение на классы выглядит нетривиальной задачей. В качестве методов классификации пакетов на сжатые и шифрованные можно выделить следующее.

3.2.1 Использование тестов для генераторов случайных чисел

По определению, цели сжатия и шифрования данных различны. Шифрование нужно для обеспечения конфиденциальности, и хорошая криптосистема должна распределять данные по сообщению равномерно и не давать никаких закономерностей для статистического анализа, который мог бы позволить расшифровать данные третей стороне. Сжатие требуется для уменьшения количества битов, необходимых для передачи информации, путём устранения избыточности, и обычно сжатые данные наоборот имеют определённую структуру. В связи с этим можно считать распределение символов в шифрованных данных более близким к истинно случайным, в отличии от сжатых данных, и использовать для их

разделения тесты, предназначенные для определения качества работы программных генераторов случайных чисел. Одним из таких наборов тестов является NIST [17].

NIST – это пакет статистических тестов, разработанный Лабораторией информационных технологий Национального института стандартов и технологий. В его состав входят 15 тестов для определения меры случайности двоичных последовательностей, которые часто используются для проверки работы генераторов случайных чисел.

Разные тесты NIST имеют разную рекомендуемую длину входной последовательности, в связи с чем не все из них подходят для использования на сетевых пакетах с их ограниченными размерами. Кроме того, некоторые методы не так хороши для различения шифрованного и сжатого трафика согласно результатам экспериментов [11] или слишком вычислительно сложны для применения при классификации пакетов в онлайн режиме.

Из подходящих можно описать следующие тесты.

1) Частотный побитовый тест. Тест определяет, является ли количество нулей и единиц в двоичной последовательности приблизительно одинаковым. Для этого вычисляются:

- S_n – сумма цифр в примере (где 0 заменяется значением -1);
- статистика $s_{\text{obs}} = \frac{|S_n|}{\sqrt{n}}$, где n – длина примера;
- $p - \text{value} = \text{erfc}(\frac{s_{\text{obs}}}{\sqrt{2}})$.

  При $p - \text{value} < 0.01$ последовательность считается неслучайной. Рекомендуется брать минимум 100 битов для анализа.

2) Частотный блочный тест (частота единиц в блоках). Аналогичен предыдущему тесту, но для N блоков фиксированной длины M внутри последовательности. Вычисляются:

- $\pi_i = \frac{\sum_{j=1}^{M} \varepsilon_{(i-1)M+j}}{\text{M}}$ - пропорция единиц в каждом блоке длины M (ε - очередной бит);
- $\chi^2(\text{obs}) = 4\text{M}\sum_{\text{i}=1}^{N}(\pi_\text{i} - 1/2)^2$ - статистика хи-квадрат для наблюдений;
- $\text{p} - \text{value} = \text{igamc}(\frac{\text{N}}{2}, \frac{\chi^2(\text{obs})}{2})$.

  При $p - \text{value} < 0.01$ последовательность считается неслучайной. Рекомендуется брать минимум 100 битов для анализа, $M \geq 20$ и $N < 100$.

3) Тест на последовательность одинаковых битов. Исходно тест заключается в подсчёте числа рядов в исходной последовательности, где ряд представляет собой непрерывную подпоследовательность одинаковых битов. Целью данного теста является вывод о том, действительно ли количество рядов разных длин из 0 и 1 соответствует их количеству в случайной последовательности. Также, можно использовать упрощённый вариант теста, заключающийся просто в вычислении максимального непрерывного количества одинаковых символов подряд.

4) Тест кумулятивных сумм. Для произвольного обхода вычисляется кумулятивная сумма значений битов в подпоследовательности (где 0 заменяется значением -1). Целью является сравнение определяемых сумм с ожидаемым их поведением в абсолютно случайной последовательности.

Пакет считается шифрованным, если он успешно проходит все тесты, пороговые значения в которых были выбраны в [11] на экспериментальной основе.

3.2.2 Использование машинного обучения

Для классификации сжатых и шифрованных пакетов в некоторых работах используются различные методы машинного обучения. Они могут работать как с признаками самого пакета, так и помогать определять пороговые значения для комбинаций других описанных тестов. Для них требуется некоторое количество достоверно размеченных иными способами тренировочных данных.

- **SVM** (англ. Support Vector Machine, метод опорных векторов) [15] – метод машинного обучения, основанный на построении разделяющей гиперплоскости в пространстве признаков объектов. В работе [14] в качестве признаков для этого классификатора используется вектор энтропии всех возможных подпоследовательностей байтов и частоты различных 4-битовых символов. Видно, что даже для частей пакетов в 1024 байта этот способ является вычислительно затратным, а его показанная эффективность не превышает эффективность других методов, поэтому такой вариант классификации далее рассматриваться не будет.
- **CART** (англ. Classification And Regression Tree) – один из алгоритмов обучения дерева решений [9]. В [8] этот метод используется для автоматического определения пороговых значений для трёх статистических методов (энтропия, арифметическое среднее, хи-квадрат) и их объединения, показав лучшие результаты, чем другие из опробованных алгоритмов машинного обучения. Этот метод в исследовании показал хорошие качество и скорость работы. Также используется в [24].
- **Метод k-ближайших соседей** (англ. k-nearest neighbors algorithm, kNN) [25] присваивает объекту тот класс, который наиболее распространён среди k его ближайших по используемой метрике соседей, класс которых уже известен. Для этого метода нужен предварительный этап выделения признаков (в [12] это локальные значения по методу хи-квадрат), а в работе он показал не самые высокие результаты.
- **Нейронные сети.** Также есть исследования по применению искусственных сетей к решаемой задаче. Среди архитектур есть как сети прямого распространения ([12], [19]), так и свёрточные ([12]). В [12] авторам не удалось добиться высоких результатов классификации, хотя лучший из полученных (около 70%) всё-таки статистически превосходит классификацию случайным выбором). В [19] классификаторы, объединяющие признаки, получаемые тестами из NIST, тест хи-квадрат и сам фрагмент, показали хорошие результаты, превосходящие другие методы, при достаточном размере анализируемого фрагмента данных.

3.3 Классификация потоков

Для того, чтобы классифицировать не отдельные пакеты, а весь поток целиком, было предложено следующее.

1) Классифицировать весь поток по первому значимому пакету (не учитывая TCP рукопожатие) [4]. Такой подход работает далеко не во всех случаях. Например, шифрованные протоколы SSH и SSL начинаются с открытого обмена ключами, а протокол HTTP, даже если передаёт сжатые данные, начинается с прозрачных заголовков, поэтому анализа только первого пакета, а тем более его части может не хватить.

2) Склеивать данные полезной нагрузки пакетов и анализировать уже их [2]. Такой подход более перспективен с точки зрения результатов классификации, чем предыдущий, но требует хранения большого количества данных, особенно при одновременном анализе множества потоков. Также, необходимо ограничить количество анализируемых пакетов в потоке, чтобы обеспечить работу в режиме реального времени, и реализовать определение конца потока, если он наступит раньше этой границы.

3) Искать в потоке непрерывные последовательности непрозрачных пакетов [13]. Для разделения потоков на прозрачные и непрозрачные в этом исследовании предлагается найти N (выбрано N=3) последовательных пакетов с высокой энтропией среди 640 первых байтов, а затем измерить совместную энтропию следующих M пакетов. Такое решение позволяет хорошо выделять непрозрачные потоки, однако в случае прозрачных

потоков требуют излишних вычислений (нужно узнать энтропию всех пакетов до конца потока).

3.4. Наш подход

Ни один из описанных выше способов классификации потоков не подходит для нашей постановки задачи или не даёт достаточно хорошие результаты, поэтому мы предложим свой метод, исходя из следующих соображений:

- количество анализируемых пакетов в потоке должно быть ограничено для возможности классификации в онлайн режиме, но должно быть достаточным, чтобы дать представление о потоке и дойти до фазы собственно обмена информацией (после рукопожатия, приветствия, обмена ключами и т.п.);
- поток классифицируется на основе всех анализируемых пакетов, но хранимая до момента принятия решения информация об исследованных пакетах должна быть минимальна для оптимизации работы по памяти;
- все пакеты прозрачного потока прозрачны;
- в случае сжатого или шифрованного потока в начале его могут идти несколько прозрачных пакетов;
- некоторые потоки не содержат полезной нагрузки вообще - их имеет смысл считать прозрачными;
- точность классификации непрозрачных потоков неидеальна, поэтому принимать решение лучше на основе не одного непрозрачного пакета, а нескольких.

На основе приведённых выше решений подзадач, их преимуществ и недостатков относительно поставленных целей, нами было предложены и опробованы следующие тесты для классификации пакетов в различных их сочетаниях.

1) Использование особенностей кодировки (*freq*): вычисление процента байтов со значениями меньше 128 в полезной нагрузке пакета.
2) Энтропия (*entropy*): вычисление энтропии полезной нагрузки пакета.
3) Проверка отношений вероятностей (sprt): в качестве начальных значений выбираются $\alpha = 5, \beta = 5, \text{offset} = 32, \text{stepsize} = 32, \text{theta} = 85$.
4) Критерий хи-квадрат (*chi*): вычисление значения данного теста при условии равномерного распределения значений байтов в качестве нулевой гипотезы.
5) Кумулятивная сумма (адаптация теста из NIST) (*cusum*): вычислить максимальное отклонение байтов (количество таких байтов подряд) от 128 в одну или другую сторону.
6) Наибольшее количество одинаковых символов подряд (адаптация теста из NIST) (runs): в качестве результата возвращается такое значение; в отличие от NIST используются значения байтов, а не битов.
7) Побитовая частота (тест NIST) (*bit freq*).
8) Максимальное количество битовых единиц подряд (адаптация теста NIST) (*bit ones*): как результат теста возвращается такое значение.
9) Частота единиц в блоках (тест NIST) (*bit freq block*).

4. Эксперименты

Для проведения экспериментального исследования предложенного решения данной задачи был подготовлен набор данных, состоящих из потоков, выделенных из реальных и искусственных сетевых трасс.

4.1 Классификация пакетов

Для экспериментов по сравнению эффективности методов классификации пакетов на три класса использовались выделенные из сетевых трасс пакеты, по 30000 пакетов на класс (20000 для тренировочной выборки, по 5000 для валидационной и тестовой). Эти классы состояли из пакетов протоколов:

- прозрачный трафик: *pop, ftp, smtp, imap, http* (текстовые незашифрованные данные согласно значениям полей *content-type* и *content-encoding*).
- сжатый трафик: *http* (*content-encoding=gzip* или *content-type=png, gif, jpeg*).
- шифрованный трафик: *gquic, ssh, tls*.

Эти пакеты были распределены по указанным классам согласно свойствам протоколов с минимальной ручной валидацией.

Далее были построены три типа классификаторов:

- классификатор на все три класса;
- классификатор для разделения трафика на прозрачный и непрозрачный;
- классификатор для разделения непрозрачного трафика на шифрованный и сжатый.

Для построения классификаторов использовалась модель случайного леса (*RandomForestClassifier*) из библиотеки *scikit-learn*, для которой на валидационной выборке был проведён подбор оптимальный параметров максимальной глубины дерева и количества деревьев (табл. 1). Для каждой комбинации параметров в связи с элементами случайности в процессе обучения производилось несколько обучений модели для выбора лучшей.

Табл. 1. Выбранные для классификаторов на основе модели случайного леса лучшие комбинации параметров
Table 1. Best combinations of parameters chosen for Random Forest classifiers

Тип классификатора	макс. глубина	количество деревьев
1 классификатор	30	25
2 классификатор	25	25
3 классификатор	25	40

Табл. 2. Полученные для указанных параметров результаты (с использованием всех признаков)
Table 2. Results obtained for chosen parameters (using all features)

Тип классификатора	точность	полнота	F1 мера (макро)
1 классификатор	0.9594	0.9604	0.9593
2 классификатор	0.9996	0.9994	0.9995
3 классификатор	0.9395	0.9408	0.9395

Можно видеть (табл. 2), что на задаче разделения трафика на прозрачный и непрозрачный достигается очень хорошее качество, в то время как задача классификации непрозрачного трафика оказывается несколько сложнее (анализ ошибок показывает, что и для классификатора на три класса эта ситуация является основным источником ошибок).

Также, с помощью параметра, характеризующего важность каждого из признаков для процесса классификации, были получены относительные значения важности признаков при классификации согласно критерию Джини. Эти значения представлены в табл. 3, чем больше число, тем важнее признак.

Табл. 3. Важность признаков для классификации согласно критерию Джини
Table 3. Feature importance according to Gini criterion

	freq	***entropy***	***sprt***	***chi***	***cusum***	***runs***	***bit freq***	***bit ones***	***bit freq block***
1	0.1784	0.0711	0.0563	0.1571	0.2764	0.0026	0.0993	0.0640	0.0645
2	0.3244	0.0419	0.1208	0.0016	0.3828	0.0676	0.0145	0.1102	0.0012
3	0.0395	0.1747	0.0064	0.3093	0.0251	0.0329	0.2147	0.0640	0.0766

Для получения общего представления было измерено время работы каждой из функций получения одного из признаков на всех пакетах (Python). Результаты приведены в табл. 4.

Табл. 4. Время получения признака для всех примеров
Table 4. Time to compute the feature for all data samples

	freq	***entropy***	***sprt***	***chi***	***cusum***	***runs***	***bit freq***	***bit ones***	***bit freq block***
время (с)	8.84	10.19	1.15	21.22	9.67	15.86	31.73	128.18	49.95

Можно видеть, что далеко не все признаки одинаковы важны для классификации, кроме того их вычисление занимает разное время, для некоторых большое, относительно других, поэтому имеет смысл использовать только часть из этих признаков для классификации, определив самые важные в каждом из случаев.

Для отбора признаков были выбраны два метода из специализированного раздела библиотеки *scikit-learn*.

- *SequentialFeatureSelector* (последовательный выбор признаков) – формирует подмножество признаков заданного размера жадным способом, на каждом шаге добавляя лучший из оставшихся признаков на основе оценки, полученной кросс-валидацией.
- *RFE* (*Recursive Feature Eliminator* – рекурсивный элиминатор признаков) – используя получаемые обучаемой моделью веса функций, рекурсивно выбирает всё меньшие и меньшие подмножества признаков, пока не достигнет нужного размера подмножества.

Табл. 5. Лучшие признаки для каждого из классификаторов, полученные методами SFS и RFE
Table 5. Best features for each of the classifiers according to the SFS and RFE methods

метод	**классификатор**	**1 признак**	**2 признак**	**3 признак**	**4 признак**	**5 признак**
SFS	1	*freq*	*chi*	*entropy*	*bit_freq*	*bit_ones*
	2	*cusum*	*freq*	*entropy*	*runs*	*bit_ones*
	3	*chi*	*runs*	*sprt*	*entropy*	*bit_freq*
RFE	1	*chi*	*freq*	*cusum*	*entropy*	*bit_ones*
	2	*bit_ones*	*cusum*	*freq*	*sprt*	*entropy*
	3	*entropy*	*chi*	*bit_freq*	*bit_freq_block*	*bit_ones*

Табл. 6. Выбранные для классификаторов на основе модели случайного леса лучшие комбинации параметров (с использованием сокращённого числа параметров)
Table 6. Best combinations of parameters chosen for Random Forest classifiers (using only chosen features)

Тип классификатора	**макс. глубина**	**количество деревьев**
1 классификатор	30	40
2 классификатор	15	20
3 классификатор	30	40

На основе сопоставления результатов, полученных этими двумя методами, и изначальной оценки важности признаков, а также с учётом времени вычисления отдельных признаков,

можно определить самые важные признаки для каждого из классификаторов (табл. 5 и 6), сократив таким образом множество признаков:

1: *freq, chi, entropy, cusum, bit_ones*;
2: *freq, cusum, sprt, entropy, bit_ones*;
3: *bit_freq, chi, entropy.*

Табл. 7. Полученные для указанных параметров результаты (с использованием только выбранных признаков)
Table 7. Results obtained for chosen parameters (using only chosen features)

Тип классификатора	**точность**	**полнота**	**F1 мера (макро)**
1 классификатор	0.9324	0.9341	0.9322
2 классификатор	0.9993	0.9989	0.9991
3 классификатор	0.9147	0.9170	0.9146

Так как классификатор пакетов на прозрачные и непрозрачные показывает хорошее качество и требует меньше признаков, чем классификатор для всех трёх классов, имеет смысл рассмотреть схему классификации в два этапа (выделение непрозрачных пакетов и отдельная их классификация) и сравнить её по качеству и времени с одноэтапным классификатором (табл. 7).

Используя для одноэтапного классификатора четыре выбранных выше признаков, мы получаем следующую оценку по времени вычисления признаков: 49,91 с

Для двухэтапного классификатора:

- 4 признака: 29,85 с;
- + 2 признака для части пакетов: 52,95 * <доля пакетов> с, так как эти признаки нужно вычислять уже не для всех пакетов. В нашем случае, при условии, что треть пакетов прозрачная, получается значение 35,3 с => общее время 65,14 с.

Табл. 8. Результаты, полученные для двух типов классификаторов
Table 8. Results obtained for two types of classifiers

классификатор	**точность**	**полнота**	**F1 мера (макро)**
одноэтапный	0.9324	0.9341	0.9322
двухэтапный	0.9423	0.9438	0.9422

Табл. 9. Матрица ошибок для одноэтапного классификатора (сверху истинные значения, слева - полученные)
Table 9. Confusion matrix for one-step classifier

трафик	**прозрачный**	**сжатый**	**шифрованный**
прозрачный	14999	10	0
сжатый	1	14286	299
шифрованный	0	704	14701

Табл. 10. Матрица ошибок для двухэтапного классификатора
Table 10. Confusion matrix for two-step classifier

трафик	**прозрачный**	**сжатый**	**шифрованный**
прозрачный	14998	10	0
сжатый	2	14379	242
шифрованный	0	611	14758

Можно видеть, что оба классификатора показывают примерно одинаковые по качеству результаты, при этом, в зависимости от доли шифрованных пакетов, двухэтапный классификатор может работать быстрее, не производя вычислений некоторых признаков (табл. 8-10).

Также был проведён выбор оптимальных параметров для SPRT (единственный из используемых тестов с переменными параметрами) (табл. 11-13).

Табл. 11. Экспериментально полученные оптимальные значения для теста SPRT
Table 11. Experimentally obtained optimal SPRT test parameters

параметр	значение
отступ от начала (offset)	0
шаг анализа (step)	32
(альтернативная гипотеза) theta	85
(желаемый false negative) alpha	5
(желаемый false positive) beta	5

Табл. 12. Результаты при выборе оптимальных параметров теста SPRT
Table 12. Results for optimal SPRT test parameters

мера	точность	полнота	F1 мера (макро)
результаты	0.9	0.9011	0.8998

Табл. 13. Матрица ошибок при выборе оптимальных параметров теста SPRT
Table 13. Confusion matrix for optimal SPRT test parameters

трафик	прозрачный	сжатый	шифрованный
прозрачный	14997	3	0
сжатый	3	14907	54
шифрованный	0	90	14966

Хотелось бы произвести дальнейшее ускорение процесса принятия решения о пакете для возможности классификации в режиме онлайн. Для этого было предложено не рассматривать пакеты меньше минимального размера и рассматривать не весь пакет целиком, а только его N байтов. Как можно видеть из графика зависимости качества классификации от размера пакета (рис. 2), такая оптимизация возможна.

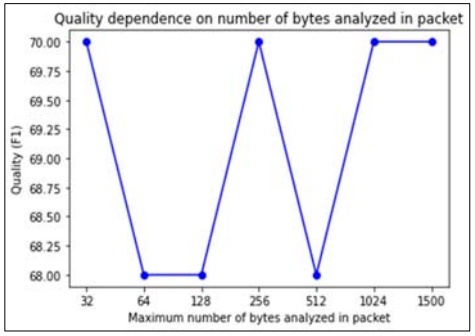

Рис.2. График зависимости качества классификации (в F1 мере) от максимального количества анализируемых байтов в пакете

Fig.2. Dependency of quality of classification (in F1) on maximum number of bytes analized in one packet

На основе экспериментов лучшим значением для ограничения сверху размера анализируемой части пакета стало 1024 байта. Это значение несколько варьировалось для разных типов трафика (прозрачного и непрозрачного) и было выбрано как компромиссное для этих ситуаций. Также, стало возможно ограничить размер анализируемых данных в тесте побитовой частоты (*bit_freq*), разделив его на 8, и сократить длину анализируемой части пакета в тестах проверки отношения вероятностей (*sprt*) и кумулятивных сумм (*cusum*) до 64 байтов без существенных потерь в качестве и с выигрышем по времени.

Для оценки производительности предлагаемые методы были реализованы в виде модулей расширения на языке C++ для разрабатываемой в ИСП РАН системы анализа сетевого трафика Конвеер. Таким образом, обученные в библиотеке *scikit-learn* модели были также переведены на C++ и загружены в систему. При описанных выше ограничениях получились следующие результаты (табл. 14) для файлов трасс, содержащих по 30000 пакетов (тех, на которых обучалась и тестировалась система на Python).

Табл. 14. Результаты и производительность модуля на C++ в составе системы Конвеер
Table 14. Results and performance of C++ module in Konveyer system

трасса	время (нс)*	скорость (Гб/с)	прозрачные	сжатые	шифрованные
transparent	1907	0.7	29952	48	0
compressed	5378	0.24	192	21962	7846
encrypted	5803	0.22	0	918	28868
*время - среднее время классификации одного пакета в нс					

4.2 Классификация потоков

Для классификации потоков предлагается следующая схема: классифицируются первые N пакетов потока, для них возможны варианты: прозрачный (сюда же относятся пакеты, не содержащие полезной нагрузки), сжатый, шифрованный или неопределённый (если задана нижняя граница размера пакета и он меньше неё). Далее для каждого потока рассматриваются N пакетов и применяются следующие правила в указанном порядке:

- если среди них есть шифрованный пакет, то весь поток классифицируется как шифрованный;
- если среди них есть сжатый пакет, то весь поток классифицируется как сжатый;
- если есть неопределённые пакеты, то весь поток классифицируется неопределённым, так как можно было пропустить какую-то важную информацию;
- иначе поток считается прозрачным.

Для некоторых протоколов были проведены эксперименты на выделенных из реального трафика примерах для определения минимального числа пакетов потока для анализа.

Табл. 15. Результаты для разного количества анализируемых пакетов в потоке
Table 15. Results for different number of packets analyzed in network flow

протокол	количество пакетов	всего потоков	прозрачные	сжатые	шифрованные
SSL	10	200	0	75	125
	15	199	0	15	184
	20	194	0	3	191
	25	194	0	2	192
SSH	10	200	0	295	5
	15	200	0	98	102
	20	200	0	1	199
	25	200	0	0	200
GQUIC	10	200	0	0	200
	25	200	0	0	200

Из табл. 15 можно видеть, что в соответствии с устройством протоколов SSL и SSH, где вначале идёт нешифрованная часть, наилучшим количеством пакетов для анализа в потоке можно считать 20: при нём протоколы уже доходят до шифрованной части, что позволяет их

правильно классифицировать, но количество минимально из возможных для более быстрой классификации.

Эксперименты на других протоколах менее показательны, так как чисто прозрачный трафик, например, гораздо сложнее выделить из реально передаваемых в сети данных. Однако, из имеющихся данных о протоколах, данного количества пакетов будет достаточно для классификации.

5. Выводы

Были изучены существующие подходы к классификации трафика на прозрачный, сжатый и шифрованный и выявлены их недостатки с точки зрения нашей постановки задачи. На основе существующих идей и методов, была предложена реализация, объединившая те из них, которые показали хорошие результаты в проведённых экспериментах.

Наша реализация способна работать с сетевым трафиком, классифицировать как пакеты, так и целые потоки, показывая при этом достаточно хорошие результаты по качеству и приемлемые - по времени. Эта реализация встроена как модуль в более широкую систему анализа трафика и может служить одним из этапов его обработки в ходе решения более сложных задач.

В дальнейшем планируется продолжить сбор и разметку примеров трафика, чтобы обучать и тестировать систему на большем количестве реальных примеров, с учётом их разнообразия и изменчивости. Желательно выделить больше примеров разных типов трафика для разных протоколов. Также, предполагается продолжить поиск оптимальных комбинаций методов и параметров для ускорения получения результатов в условиях работы в реальном времени.

Список литературы / References

[1]. Luo S., Seideman J.D., Dietrich S. Fingerprinting Cryptographic Protocols with Key Exchange using an Entropy Measure. In Proc. of the IEEE Security and Privacy Workshops (SPW), 2018, pp. 170-179.

[2]. Choudhury P., Kumar K.R.P. et al. An empirical approach towards characterization of encrypted and unencrypted VoIP traffic. Multimedia Tools and Applications, vol. 79, issue 1, 2020, pp. 603-631.

[3]. Wood D., Apthorpe N., Feamster N. Cleartext data transmissions in consumer iot medical devices. In Proc. of the 2017 Workshop on Internet of Things Security and Privacy, 2017, pp. 7-12.

[4]. Dorfinger P., Panholzer G., John W. Entropy estimation for real-time encrypted traffic identification. Lecture Notes in Computer Science, vol. 6613, 2011, pp. 164-171.

[5]. Hjelmvik E., John W. Breaking and improving protocol obfuscation. Chalmers University of Technology, Technical Report No. 2010-05, 2010, 34 p.

[6]. White A. M., Krishnan S. et al. Clear and Present Data: Opaque Traffic and its Security Implications for the Future. In Proc. of the 20th Annual Network & Distributed System Security Symposium, 2013, 16 p.

[7]. Roesch M. Snort: Lightweight intrusion detection for networks. In Proc. of the 13th USENIX Conference on System Administration (LISA '99), 1999, pp. 229–238.

[8]. Cha S., Kim H. Detecting encrypted traffic: a machine learning approach. Lecture Notes in Computer Science, vol. 10144, 2016, pp. 54-65.

[9]. Lewis R.J. An introduction to classification and regression tree (CART) analysis. In Proc. of the Annual Meeting of the Society for Academic Emergency Medicine in San Francisco, 2000, 15 p.

[10]. Rish I. An empirical study of the naive Bayes classifier. In Proc. of the Workshop on Empirical Methods in Artificial Intelligence, 2001, pp. 41-46.

[11]. Casino F., Choo K. K. R., Patsakis C. HEDGE: efficient traffic classification of encrypted and compressed packets. IEEE Transactions on Information Forensics and Security, vol. 14, issue 11, 2019, pp. 2916-2926.

[12]. Hahn D., Apthorpe N., Feamster N. Detecting compressed cleartext traffic from consumer internet of things devices. arXiv preprint arXiv:1805.02722, 2018.

[13]. Zhang H., Papadopoulos C. Early detection of high entropy traffic. In Proc. of the IEEE Conference on Communications and Network Security (CNS, 2015, pp. 104-112.

[14]. Wang, Y., Zhang, Z. et al. Using entropy to classify traffic more deeply. In Proc. of the IEEE Sixth International Conference on Networking, Architecture, and Storage, 2011, pp. 45-52.

[15]. Wang L. (ed.). Support vector machines: theory and applications. Springer Science & Business Media, 2005, 412 p.

[16]. Lyda R., Hamrock J. Using entropy analysis to find encrypted and packed malware. IEEE Security & Privacy, vol. 5, issue 2, 2007, pp. 40-45.

[17]. Rukhin A., Soto J. et al. A statistical test suite for random and pseudorandom number generators for cryptographic applications. NIST Special Publication 800-22, 2001, 131 p.

[18]. Sturgill, M., & Simske, S. (2016). Mass Serialization Method for Document Encryption Policy Enforcement. In Proc. of the ACM Symposium on Document Engineering, 2016, pp. 193-196.

[19]. De Gaspari F., Hitaj D. et al. Encod: Distinguishing compressed and encrypted file fragments. Lecture Notes in Computer Science, vol. 12570, 2020, pp. 42-62.

[20]. De Gaspari F., Hitaj D. et al. Reliable Detection of Compressed and Encrypted Data. arXiv preprint arXiv:2103.17059, 2021.

[21]. Shannon C.E. A mathematical theory of communication. The Bell System Technical Journal, vol. 27, no. 3, 1948, pp. 379-423.

[22]. Goubault-Larrecq J., Olivain J. Detecting subverted cryptographic protocols by entropy checking. Research Report LSV-06-13, Laboratoire Spécification et Vérification, ENS Cachan, 2006.

[23]. Goubault-Larrecq J., Olivain J. On the efficiency of mathematics in intrusion detection: the NetEntropy case. In Proc. of the International Symposium on Foundations and Practice of Security, 2013, pp. 3-16.

[24]. Kozachok A. V. et al. Classification of pseudo-random sequences based on the random forest algorithm. In Proc. of the 2020 Ivannikov Memorial Workshop (IVMEM), 2020, pp. 55-58.

[25]. Zahid N., Abouelala O. et al. Fuzzy clustering based on K-nearest-neighbours rule. Fuzzy Sets and Systems, vol 120, issue 2, 2001, pp. 239-247.

Информация об авторах / Information about authors

Александр Игоревич ГЕТЬМАН – старший научный сотрудник, кандидат физико-математических наук. Сфера научных интересов: анализ бинарного кода, восстановление форматов данных, анализ и классификация сетевого трафика.

Aleksandr Igorevich GETMAN – senior researcher, PhD in physical and mathematical sciences. Research interests: binary code analysis, data format recovery, network traffic analysis and classification.

Мария Кирилловна ИКОННИКОВА – аспирант. Научные интересы: анализ сетевого трафика, машинное обучение.

Maria Kirillovna IKONNIKOVA – postgraduate student. Research interests: network traffic analysis, machine learning.