Метод обнаружения сетевых скрытых каналов для повышения защищенности сетей пакетной передачи данных

С.В. Айрапетян, К.С. Зайцев

***Аннотация*. Целью настоящей работы является описание разработанного метода обнаружения сетевых скрытых каналов по времени, основанного на алгоритмах машинного обучения. Обнаружение сетевых скрытых каналов является актуальной задачей, так как последние могут быть использованы для утечки конфиденциальной информации. Один из подходов к ее решению является использование алгоритмов машинного обучения. Для применения алгоритмов машинного обучения при решении задачи обнаружения, исследуемый сетевой трафик должен быть предварительно обработан. В статье приведено описание разработанного метода обнаружения, где для обработки сетевого трафика предлагается использовать методы распределенной обработки больших данных, реализованных в Apache Spark. В качестве алгоритма машинного обучения в методе обнаружения предлагается использовать градиентный бустинг над деревьями решений. В работе описана архитектура системы, в которой предлагается реализовать процесс обнаружения скрытых каналов. Исследованы зависимости времени обработки сетевого трафика от различных параметров системы. Предложено использовать новые признаки для обнаружения скрытых каналов. По результатам исследований выявлено, что предложенный метод позволяет эффективно обнаруживать сетевые скрытые каналы, а его особенностью является скорость обнаружения - за счет использования технологий распределенной обработки данных, и повышения точности - за счет добавление новых признаков.**

***Ключевые слова* – скрытый канал, машинное обучение, распределенная обработка, межпакетный временной интервал, статистические характеристики, Apache Spark.**

I. ВВЕДЕНИЕ

Статья получена 16 июня 2022 г.

Айрапетян Сергей Ваграмович, Национальный Исследовательский Ядерный Университет МИФИ, магистрант, sergey.hayrapetyan99@mail.ru

Зайцев Константин Сергеевич, Национальный Исследовательский Ядерный Университет МИФИ, профессор, KSZajtsev@mephi.ru

Как известно, скрытый канал - непредусмотренный разработчиком системы информационных технологий и автоматизированных систем коммуникационный канал, который может быть применен для нарушения политики безопасности [1]. Существование скрытых каналов в системе представляет угрозу информационной безопасности системы, так как скрытый канал не использует законные механизмы передачи информации, а факт передачи информации по скрытому каналу нельзя выявить стандартными методами защиты информации. Из-за широкого распространения IP-сетей было предложено большое количество схем построения скрытых каналов, функционирующих в рамках IP-сетей, и в зависимости от метода сокрытия информации, они разделены на сетевые скрытые каналы по памяти и сетевые скрытые каналы по времени (далее - ССКВ).

В случае сетевого скрытого канала по памяти, отправитель напрямую записывает скрытую информацию в сетевые объекты, такие как поля заголовка протоколов, в то время как получатель может считывать скрытую информацию из тех же сетевых объектов. В случае ССКВ, отправитель встраивает скрытую информацию изменяя временные параметры легитимного трафика, такие как межпакетный временной интервал. Приемник наблюдает за скрытым трафиком, извлекает из него временные параметры и декодирует скрытую информацию.

Сетевые скрытые каналы по памяти ограничены сетевыми протоколами и, следовательно, не могут отклоняться от определенного поведения. Это делает их обнаружение несложным. С другой стороны, ССКВ демонстрируют стохастическое поведение, и следовательно, их обнаружение является более сложной задачей.

Для решения задачи обнаружения ССКВ предложены различные методы [2-11], которые основаны на статистическом анализе или применении алгоритмов машинного обучения. В [12] проведен сравнительный анализ некоторых методов обнаружения ССКВ, в результате которого было выявлено, что один из лучших результатов

показал метод обнаружения на основе алгоритма градиентного бустинга над деревьями решений. В [12] авторы отметили сложность процесса обучения и применения моделей машинного обучения, так как в исследуемых ими методах не был описан сам процесс обработки сетевого трафика.

Это может быть проблемой, так как для предотвращения утечку информации по скрытым каналом в сети необходимо обрабатывать большие данные в режиме реального времени, объем которых в современных сетях может измеряться в гигабайтах. Однако данная проблема может быть решена применением методов обработки больших данных при обнаружении ССКВ. К основным технологиям обработки больших данных относят нереляционные базы данных, модель обработки информации MapReduce, компоненты кластерной экосистемы Hadoop [13], а также специализированный фреймворк Apache Spark [14].

В отличии от классического обработчика ядра Apache Hadoop с двухуровневой концепцией MapReduce на базе дискового хранилища, Spark использует специализированные примитивы для рекуррентной обработки в оперативной памяти. Благодаря этому многие вычислительные задачи, такие как обработка и анализ сетевого трафика, могут быть реализованы в Spark значительно быстрее.

В настоящей статье предлагается метод обнаружения сетевых скрытых каналов по времени на основе алгоритмов машинного обучения при использовании Apache Spark.

II. ОПИСАНИЕ МЕТОДА

Исследование существующих методов [2-11] показало, что для обнаружения ССКВ больше всего подходят методы, основанные на алгоритмах обучения с учителем, в связи с чем разработанный метод предполагает использование именно алгоритмов этих видов. Далее описаны основные этапы разработанного метода обнаружения.

Этап 1. Формирование обучающей выборки. Первый этап состоит из нескольких шагов. На первом шаге проводится анализ легитимного трафика и выделение из него массива значений длин межпакетных временных интервалов. Данный шаг реализуется посредством передачи легитимного трафика на кластер Apache Spark. При этом выполняется предварительная буферизация данных, о котором подробно описано в разделе IV.

Далее, необходимо сгенерировать скрытый трафик (трафик со скрытно передаваемой информацией) и встает вопрос выбора параметров для скрытых каналов по времени, так как многие схемы построения ССКВ предполагают использование характеристик сети, в которой функционирует скрытый канал. Таким образом, на втором шаге первого этапа предлагаемого метода необходимо установить параметры скрытых каналов, основываясь на анализе легитимного трафика.

После установления параметров скрытых каналов, на третьем шаге производится генерация скрытого трафика и выделяется массив межпакетных временных интервалов. Таким образом, после выполнения 3 шага на выходе получатся 2 набора данных Apache Spark межпакетных временных интервалов — легитимного и скрытого трафиков. Для формирования обучающей выборки, необходимо разбить массивы на образцы.

Для этого, на 4 шаге, на основе анализа возможных пропускных способностей рассматриваемых скрытых каналов необходимо определить длину образцов. На 5 шаге необходимо выполнить разметку данных и определить у образцов статистические характеристики. Выделенные статистические характеристики будут использованы в качестве признаков в моделях машинного обучения. Таким образом, на выходе первого этапа будет получена обучающая выборка.

Этап 2. Обучение модели. На втором этапе разработанного метода производится обучение модели. На данном этапе выполняется необходимая предобработка обучающей выборки, устанавливаются параметры обучаемой модели, проводится кросс-валидация. По результатам 2 этапа будет обучен классификатор сетевых скрытых каналов по времени.

Этап 3. Предобработка проверяемого трафика. На третьем этапе выполняется вся необходимая предобработка проверяемого трафика: выделяются массивы межпакетных временных интервалов, проводится деление на образцы, для каждого образца рассчитывается вектор из выбранных статистических характеристик.

Этап 4. Определение наличия скрытого канала. На четвертом этапе, вектора, определенные на 3 этапе, передаются обучаемому классификатору сетевых скрытых каналов по времени. Наконец, классификатор определяет принадлежность поданного ему на вход вектора к одному из 2 видов трафиков (скрытый или легитимный).

На рис. 1 представлена схема работы метода обнаружения.

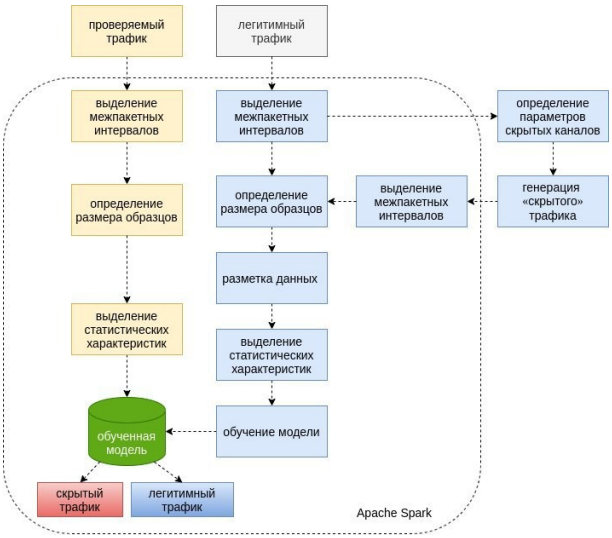

Рис. 1. Схема работы метода обнаружения ССКВ

Как видно из рис. 1, в разработанном методе обнаружения ССКВ для обработки сетевого трафика, обучения и применения модели машинного обучения предлагается использовать контур Apache Spark.

III. ИСПОЛЬЗУЕМЫЕ ПРИЗНАКИ

Ниже приведены статистические характеристики, которые были использованы в существующих методах обнаружения в качестве признаков, и которые предложено использовать в разработанном методе ССКВ.

1. *Количество уникальных значений в последовательности межпакетных временных интервалов.*
2. *Сумма коэффициентов автокорреляции*.
3. *Мультимодальность на основе ядерной оценки плотности.*
4. *Мультимодальность основанная на анализе Парето.*
5. *Тест T_R (Runs test).*
6. *Тест T_s (Sign test).*
7. *Частота моды.*
8. *Энтропия.*
9. *Показатель Херста*.
10. *Колмогоровская сложность K.*

Вышеупомянутые характеристики были выбраны в существующих методах из-за их физических свойств, однако как показали исследования, не все характеристики имеют одинаково высокую значимость при построении модели. Далее описаны еще 5 статистических характеристик, предлагаемые в качестве расширения множества ранее рассмотренных признаков.

11. Эксцесс межпакетного временного интервала. Эксцесс — момент порядка 4 стандартизованного распределения случайной величины.

12. Коэффициент асимметрии. Коэффициент асимметрии характеризует симметричность распределения межпакетных временных интервалов.

13. Т-критерий Уэлча. Т-критерий Уэлча проверяет гипотезу на близость двух последовательностей.

14. Закономерность в дисперсии (Regularity test).

15. Среднее уникальных значений в последовательности межпакетных временных интервалов.

IV. ОБРАБОТКА СЕТЕВОГО ТРАФИКА

Для обеспечения эффективного обнаружения скрытых каналов необходимо обрабатывать сетевой трафик в режиме реального времени, а сам процесс обработки должен быть выполнен с наибольшей возможной скоростью, так как чем раньше будет обработан сетевой трафик, тем быстрее будет обнаружен скрытый канал (при его наличии в системе), и как следствие, будет предотвращена утечка конфиденциальной информации. С учетом этого, в разработанном методе обнаружения ССКВ для обработки сетевого трафика предлагается использовать Apache Spark.

При попытке передать в контур Apache Spark напрямую данные из источников, таких как сетевые адаптеры, маршрутизаторы и т.д., могут возникнуть трудности, так как нет унифицированного интерфейса для взаимодействия с устройствами для передачи трафика. Учитывая данный факт, возникает необходимость предварительной буферизации исследуемого сетевого трафика. В настоящий момент популярным решением для буферизации больших данных является Apache Kafka [15]. Apache Kafka сохраняет текущее и все прежние состояния системы и может использоваться в качестве достоверного источника исторических данных, а основными преимуществами Apache Kafka являются масштабируемость, производительность и расширяемость. Именно поэтому многие анализаторы позволяют взаимодействовать с Apache Kafka, что позволяет загружать в выделенный топик Apache Kafka сетевой трафик для последующей передачи в среду обработки. В этом можно убедиться на примере изучения анализатора NTOP, основанного на протоколе Netflow [16].

После записи сетевыми устройствами в топик Apache Kafka, необходимо извлечь данные из топика и записать их в хранилище данных для последующей обработки.

Применение технологий MapReduce на платформе HDFS при обработке больших данных может быть неэффективным ввиду низкой скорости обработки. С другой стороны, сам HDFS может быть эффективным решением для хранения больших

данных, так как масштабируема и устойчива (благодаря репликации блоков данных).

Особенности HDFS напрямую не позволяют записывать данные, поскольку запись каждого пакета в отдельный файл HDFS приведет к чрезмерной нагрузке на центральный узел имён (англ. name node), хранящего метаданные файловой системы и метаинформацию о распределении блоков. Поэтому процесс перекладки трафика из топика Kafka в файловое хранилище должен быть реализован потоковой системой (которая перед записью данных в хранилище разделит последние на блоки необходимого для HDFS размера), такой как Spark Streaming.

Таким образом, архитектура среды, необходимой для обработки сетевого трафика будет иметь вид, представленный на рис. 2.

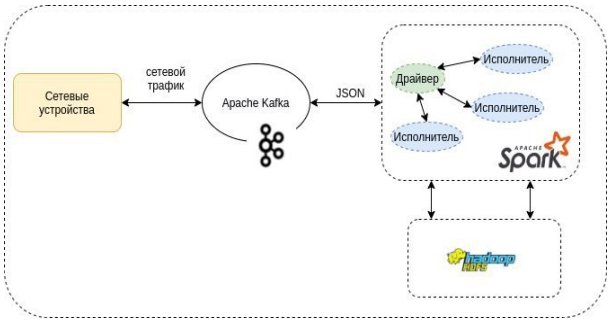

Рис. 2. Архитектура системы для обработки сетевого трафика.

После перекладки сетевого трафика из сетевых устройств в соответствующий топик Apache Kafka, процесс потоковой обработки, инициализированный Spark Streaming, читает данные из топика, преобразует его, и записывает в файловое хранилище HDFS в виде таблицы.

Процесс, инициализированный Spark SQL, читает данные из таблицы и выполняет все необходимые преобразования для выделения последовательностей межпакетных временных интервалов. При этом длина последовательности может задаваться динамически. Процесс выделения статистических характеристик, описанных выше, выполняется при использовании интерфейса расширения классов пользовательскими функциями (англ. User-defined functions, UDF) [14], что позволяет применять функции вычисления статистических характеристик для отдельных последовательностей на распределенных данных.

V. ОБУЧЕНИЕ И ПРИМЕНЕНИЕ МОДЕЛИ

Как уже было отмечено выше, по результатам анализа существующих методов обнаружения выявлено, что один из лучших результатов при обнаружении ССКВ показал метод, основанный на алгоритме градиентного бустинга над деревьями решений. Поэтому в разработанном методе предложено использовать именно этот алгоритм. Для его обучения в контуре Apache Spark предложено использовать проект XGBoost4J-Spark — проект, направленный на интеграцию широко применяемой реализации XGBoost и Apache Spark путем адаптации XGBoost к инфраструктуре MLLib Apache Spark [17]. Процесс обучения модели состоял из 3-х основных этапов, описанных ниже.

Этап 1. Предварительная обработка данных. Модуль Spark MLlib реализован таким образом, что модели могут принимать на вход только одну колонку в виде массива признаков. Вдобавок, признаки в массиве не могут быть категорийными и должны иметь формат DoubleType Spark. На данном этапе проведено преобразование выделенных ранее признаков в необходимый формат.

Этап 2. Настройка гиперпараметров. Для настройки гиперпараметров был выбран метод поиска по сетке.

Этап 3. Скользящий контроль. Для выполнения скользящего контроля был выбран метод перекрестной проверки по K блокам.

Для последующего сравнения модели, обученного в контуре Apache Spark с моделями, обученными на нераспределенных данных, был разработан модуль применения моделей машинного обучения на распределенных данных, который реализует следующие основные шаги:

- загрузка обученной модели машинного обучения в оперативную память;
- преобразование входного массива признаков в наборы RDD Apache Spark [14];
- применение загруженной модели к распределенным данным при использовании метода mapPartitions RDD.

VI. РЕЗУЛЬТАТЫ ИССЛЕДОВАНИЙ

Одной из причин, по которой Apache Spark был использован при формировании выборки признаков, это время обработки данных. Время выполнения разработанных модулей может зависеть от следующих параметров процесса формирования статистических характеристик:

– объема обрабатываемых данных;

– вычислительных ресурсов, выделенных под Spark-сессию для выполнения обработки данных;

– установленной длины последовательностей межпакетных временных интервалов.

1. Исследование зависимости времени вычисления статистических характеристик от объема обрабатываемых данных. В рамках данного исследования было определено время вычисления

статистических характеристик для сетевого трафика объемов в 20 ГБ, 60 ГБ, 100 ГБ, 200 ГБ, 250 ГБ. При этом, инициализированной Spark-сессии было выделено 10 исполнителей по 12 ядер и 10 ГБ оперативной памяти, с длиной последовательностей межпакетных временных интервалов 500. На рис. 3 представлен график зависимости времени вычисления статистических характеристик от объема обрабатываемых данных при фиксированных параметрах.

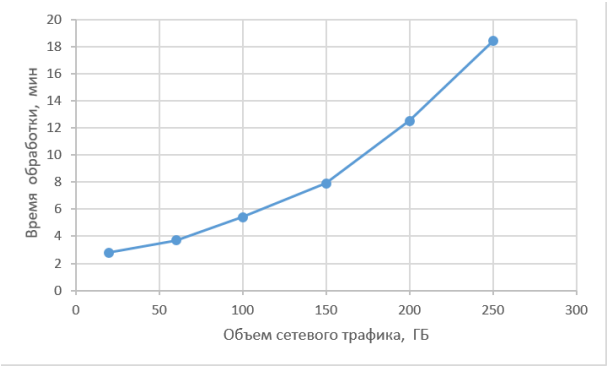

Рис. 3. График зависимости времени обработки сетевого трафика от его объема.

Из графика видно, что зависимость имеет экспоненциальный характер. Это можно объяснить тем, что при увеличении объема обрабатываемой информации увеличивается не только объем задач, который объявлен каждому Spark-исполнителю, а также объем данных, который необходимо передать по сети для обеспечения параллельных вычислений.

2. Исследование зависимости времени вычисления статистических характеристик от выделенных Spark-сессии ресурсов. В рамках данного исследования было подсчитано время выделения статистических характеристик при инициализации Spark-сессий с количеством ядер 80, 100, 120, 140, 160. При этом обрабатывались данные объемом в 100 ГБ, с длиной последовательностей межпакетных временных интервалов 500. На рис. 4 представлен график зависимости времени вычисления статистических характеристик от количества выделенных ядер.

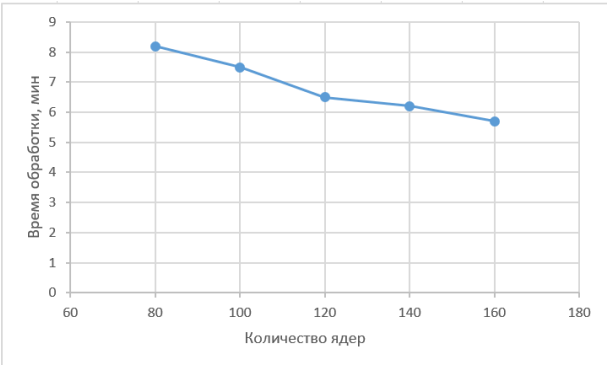

Рис. 4. График зависимости времени обработки сетевого трафика от количества выделенных ядер.

Из графика видно, что зависимость времени обработки сетевого трафика от объема обрабатываемых данных имеет линейный характер.

3. Исследование зависимости времени вычисления статистических характеристик от длины последовательностей межпакетных временных интервалов. В рамках данного исследования было подсчитано время выделения статистических характеристик при различных значениях длины последовательностей межпакетных временных интервалов. При этом обрабатывались данные объемом в 100 ГБ, а инициализированной Spark-сессии было выделено 10 исполнителей по 10 ядер. На рис. 5 представлен график зависимости времени вычисления статистических характеристик от длины последовательностей межпакетных временных интервалов.

Рис. 5. График зависимости времени обработки сетевого трафика от длины последовательностей межпакетных временных интервалов.

Из графика видно, начиная с определенного значения, при увеличении длины последовательностей межпакетных временных интервалов увеличивается время выделения статистических характеристик. Это можно объяснить тем, что при вычислении статистических характеристик используются UDF, а, как известно, оптимизатор Spark Catalyst не всегда позволяет оптимизировать план выполнения запроса, сгенерированного UDF. Как следствие, оптимизатор не может оценить сложность выполнения задач и распределить их по всему кластеру оптимальным способом.

Чтобы проверить качество результата применения обученного классификатора была предложена программа тестирования, которая состояла из следующих основных этапов:

1. Сбор данных.
2. Выделение статистических характеристик из данных.

3. Применение обученной в контуре Apache Spark модели к данным (Модель 1).

5. Обучение модели из [7] (Модель 2).

6. Применение Модели 2 к данным.

7. Выделение показателей точности обоих алгоритмов.

Тестирования проводилось для скрытого канала типа Time Replay Covert Timing Channel, а в качестве показателей качества были выбраны:

- точность (accuracy);
- прецизионность (precision);
- полнота (recall);
- f1-метрика (f1).

Необходимо отметить, что Модель 2 была обучена на первых 10 признаках, в соответствии с методом из [7]. Результаты эксперимента показаны в табл. 1.

Табл. 1. Результаты работы моделей.

Показатель	Модель 1	Модель 2
precision	0,98	0,95
accurancy	0,97	0,92
recall	0,99	0,97
f1	0,97	0,96

Как видно из табл. 2, обучение модели на распределенных данных при добавлении новых признаков по отношению к набору признаков из [7] позволило улучшить точность обнаружения скрытых каналов.

VII. ДИСКУССИЯ ПО ТЕМЕ ИССЛЕДОВАНИЙ

Основными преимуществами разработанного метода обнаружения сетевых скрытых каналов по времени являются малое время и высокая точность обнаружения. Уменьшение времени было достигнуто за счет использование методов распределенной обработки больших данных, реализованных в Apache Spark. По результатам исследований выявлено, что повышение точности обнаружения связана с использованием новых признаков при применении моделей машинного обучения. С другой стороны, необходимо отметить, что тестирование для определения качества обнаружения проводилось для одного канала, и нет данных о точности обнаружения для сетевого трафика с несколькими внедренными скрытыми каналами по времени. Еще одним недостатком предложенного метода обнаружения является необходимость сопровождения кластера с развернутым Apache Spark, где предлагается выполнять обработку сетевого трафика и обнаружение сетевых скрытых каналов

VIII. ЗАКЛЮЧЕНИЕ

Данная работа посвящена разработке метода обнаружения сетевых скрытых каналов по времени на основе алгоритмов машинного обучения при использовании технологий обработки больших данных Apache Spark. В работе описаны основные этапы разработанного метода, выделены статистические характеристики межпакетных временных интервалов, которые используются в качестве признаков в моделях машинного обучения.

В работе описана архитектура системы для обработки сетевого трафика и выделения статистических характеристик в контуре Apache Spark. Приведено описание процессов обучения и применения моделей машинного обучения в контуре Apache Spark. Проведено исследование зависимости времени вычисления статистических характеристик от объема обрабатываемых данных, от выделенных Spark-сессии ресурсов и от длины последовательностей межпакетных временных интервалов.

Выявлено, что в случае обработки 100 ГБ сетевого трафика при выделении 10 исполнителей по 12 ядер и 10 ГБ оперативной памяти процесс выделения статистических характеристик занимает примерно 6 минут, при условии, что длина обрабатываемых последовательностей межпакетных временных интервалов равна 500. Для оценки качества предсказаний обученного классификатора предложена и реализована программа тестирования, по результатам которой выявлено, что обучение модели на распределенных данных при добавлении новых признаков позволило улучшить точность обнаружения скрытых каналов.

БЛАГОДАРНОСТИ

Авторы выражают благодарность Высшей инжиниринговой школе НИЯУ МИФИ за помощь в возможности опубликовать результаты выполненной работы.

БИБЛИОГРАФИЯ

[1] Lampson, B. W. A Note on the Confinement Problem /Communications of the ACM. — 1973. — С. 613-615.

[2] Gianvecchio, S., Wang , H. An Entropy-based approach to detecting covert timing channel // IEEE Transactions on Dependable and Secure Computing.— 2011.— С.785- 797.

[3] F. Iglesias, R. Annessi,T. Zseby DAT detectors: uncovering TCP/IP covert channels by descriptive analytics //Security and Communication Networks. — 2016. — Vol. 9. — No. 15. — С. 3011-3029.

[4] Shrestha, P. A Support Vector Machine-based framework for detection of covert timing

channels//IEEE Transactions on Dependable and Secure Computing.—2016.— С. 274-283.

[5] F. Iglesias, R. Annessi,T. Zseby Are network covert timing channels statistical anomalies? // Proceedings of the 12th International Conference on Availability, Reliability and Security. — 2017. — No. 81.

[6] Iglesias, R. Annessi,T. Zseby Analytic study of features for the detection of covert timing channels in network traffic //Journal of Cyber Security and Mobility. —2020. —Т. 6. — No. 3. — С. 225-270.

[7] Chourib, M. Detecting Selected Network Covert Channels Using Machine Learning /International Conference on High Performance Computing & Simulation (HPCS) —2019. — С. 582– 588.

[8] S. Al-Eidi, O. Darwish, Y. Chen Covert Timing Channel Analysis Either as Cyber Attacks or Confidential Applications/ Sensors (Basel). — 2020. — 20(8).

[9] Yuvaraj, G. Covert Channels Detection with Supported Vector Machine and Hyperbolic Hopfield Neural Network 2019 / S. R. Lingham, J. Rajkamal [Электронный ресурс] https://www.sciencepubco.com/index.php/IJET (дата обращения — 11.03.2022).

[10] Han J., Huang C., Shi F. Covert timing channel detection method based on time interval and payload length analysis/ Computers & Security —2020. — 97.

[11] Al-Eidi S., Darwish O., Chen Y. SnapCatch: Automatic Detection of Covert Timing Channels Using Image Processing and Machine Learning/ IEEE Access —2021. — СС. 177-191. (doi: 10.1109/ACCESS.2020.3046234).

[12] Elsadig M. A., Gafar A. Covert Channel Detection: Machine Learning Approaches / IEEE Access — 2022 (doi:10.1109/ACCESS.2022.3164392.)

[13] Apache Hadoop https://hadoop.apache.org/ дата обращения — 11.04.2022).

[14] Jules S. Damji, Brooke Wenig, Tathagata Das & Denny Lee Learning Spark: Lightning-Fast Big Data Analytics/ изд. O'Reilly Media — 2020 — 399 с.

[15] M.J., Sakr S., Zomaya A. Apache Kafka. Encyclopedia of Big Data Technologies / — Cham — 2018.

[16] NetFlow Traffic Analyzer - NetFlow analyzer and bandwidth monitoring software [Электронный ресурс] —https://www.solarwinds.com/netflow-traffic-analyzer (дата обращения — 19.05.2022).

[17] XGBoost4J-Spark Tutorial [Электронный ресурс] https://xgboost.readthedocs.io/en/stable/jvm/xgboost4j_spark_tutorial.html (дата обращения — 15.05.2022).

Network covert channels detection method for packet data transmission networks security increase

S. V. Hayrapetyan, K.S. Zaytsev

Abstract - The purpose of this paper is to describe the developed method of detecting network covert timing channels, based on machine learning algorithms. Detection of network covert channels is an actual problem, since the latter can be used to leak confidential information. One of the approaches to its solution is the use of machine learning algorithms. In order to apply machine learning algorithms in solving the detection problem, the network traffic under study must be pre-processed. The article describes the developed detection method, where it is proposed to use distributed big data processing methods implemented in Apache Spark to process network traffic. As a machine learning algorithm in the detection method, it is proposed to use gradient boosting over decision trees. The paper describes the architecture of the system in which it is proposed to implement the process of detecting covert channels. The dependences of network traffic processing time on various system parameters are investigated. It is proposed to use new features to detect covert channels. According to the research results, it has been revealed that the proposed method makes it possible to effectively detect network covert channels, and its feature is the speed of detection - through the use of distributed data processing technologies, and increasing accuracy - by adding new features.

Keywords – covert channel, machine learning, distributed processing, inter-batch time interval, statistical characteristics, Apache Spark

REFERENCES

1] Lampson, B. W. A Note on the Confinement Problem /Communications of the ACM. — 1973. — pp. 613-615.

[2] Gianvecchio, S., Wang , H. An Entropy-based approach to detecting covert timing channel // IEEE Transactions on Dependable and Secure Computing.— 2011.— pp.785- 797.

[3] F. Iglesias, R. Annessi,T. Zseby DAT detectors: uncovering TCP/IP covert channels by descriptive analytics //Security and Communication Networks. — 2016. — Vol. 9. — No. 15. — p. 3011-3029.

[4] Shrestha, P. A Support Vector Machine-based framework for detection of covert timing channels//IEEE Transactions on Dependable and Secure Computing.—2016.— pp. 274-283.

[5] F. Iglesias, R. Annessi,T. Zseby Are network covert timing channels statistical anomalies? // Proceedings of the 12th International Conference on Availability, Reliability and Security. — 2017. —No. 81.

[6] Iglesias, R. Annessi,T. Zseby Analytic study of features for the detection of covert timing channels in network traffic //Journal of Cyber Security and Mobility. —2020. —T. 6. — No. 3. — pp. 225-270.

[7] Chourib, M. Detecting Selected Network Covert Channels Using Machine Learning /International Conference on High Performance Computing & Simulation (HPCS) —2019. — pp. 582– 588.

[8] S. Al-Eidi, O. Darwish, Y. Chen Covert Timing Channel Analysis Either as Cyber Attacks or Confidential Applications/ Sensors (Basel). — 2020. — 20(8).

[9] Yuvaraj, G. Covert Channels Detection with Supported Vector Machine and Hyperbolic Hopfield Neural Network 2019 / S. R. Lingham, J. Rajkamal https://www.sciencepubco.com/index.php/IJET (Reviewed — 11.03.2022).

[10] Han J., Huang C., Shi F. Covert timing channel detection method based on time interval and payload length analysis/ Computers & Security —2020. — 97.

[11] Al-Eidi S., Darwish O., Chen Y. SnapCatch: Automatic Detection of Covert Timing Channels Using Image Processing and Machine Learning/ IEEE Access —2021. — pp. 177-191. (doi: 10.1109/ACCESS.2020.3046234).

[12] Elsadig M. A., Gafar A. Covert Channel Detection: Machine Learning Approaches / IEEE Access — 2022 (doi:10.1109/ACCESS.2022.3164392.)

[13] Apache Hadoop https://hadoop.apache.org/ (Reviewed — 11.04.2022).

[14] Jules S. Damji, Brooke Wenig, Tathagata Das & Denny Lee Learning Spark: Lightning-Fast Big Data Analytics/ O'Reilly Media — 2020 — 399 p.

[15] M.J., Sakr S., Zomaya A. Apache Kafka. Encyclopedia of Big Data Technologies / — Cham — 2018.

[16] NetFlow Traffic Analyzer - NetFlow analyzer and bandwidth monitoring software

https://www.solarwinds.com/netflow-traffic-analyzer (Reviewed — 19.05.2022).

[17] XGBoost4J-Spark Tutorial [https://xgboost.readthedocs.io/en/stable/jvm/xgboost4j_spark_tutorial.html (Reviewed — 15.05.2022).